Бондаренко О.О, Ковшун М.І.,
Пилипчук О.П., Шестопалов Є.А.

Інформатика

Третій рік – єдиний курс

11 клас

Рівень стандарту
Академічний рівень (половина)

Навчальний посібник
(використовується з «Робочим зошитом»)

Шепетівка
«Аспект»
2011

УДК.004.451 (07)
ББК.32.973.26-018.2я7
Б52

Пробний випуск для апробації
в умовах навчального процесу.

Зауваження та пропозиції щодо змісту навчального матеріалу посібника надсилайте на адресу:
aspekt@aspekt.in.ua

Після випробування в різних регіонах України та обробки зауважень виправлене і доповнене видання посібника буде подано на розгляд експертної комісії МОН України для одержання відповідного грифу.

Бондаренко О.О., Ковшун М.І., Пилипчук О.П., Шестопалов Є.А.

Б52 Інформатика. Третій рік – єдиний курс. 11 клас. Рівень стандарту, Навчальний посібник / – Шепетівка: «Аспект», 2011 – 192 с.

ISBN 978-966-2017-34-2

Навчальний посібник рекомендується для учнів 11-х класів загальноосвітніх навчальних закладів усіх профілів рівня стандарту та академічного рівня (половина), які розпочали вивчення інформатики з 9-го класу.

Навчальний посібник відповідає відповідає вимогам програми «Інформатика. Єдиний базовий курс. 9-11 класи» (автори: Завадський І.О., Дорошенко Ю.О., Пилипчук О.П., Шестопалов Є.А.).

Розрахований на використання з робочим зошитом «Інформатика. 11 клас. Рівень стандарту. Академічний рівень (половина)» /Бондаренко О.О., Ковшун М.І., Пилипчук О.П., Шестопалов Є.А. – «Аспект», 2011.

УДК.004.451 (07)
ББК.32.973.26-018.2я7

ISBN 978-966-2017-34-2

Передмова

Навчальний посібник розрахований на учнів 11-х класів загальноосвітніх навчальних закладів усіх профілів рівня стандарту і частково академічного рівня, які почали вивчати інформатику з 9-го класу.

Поданий у посібнику навчальний матеріал для 11-го класу відповідає чинній програмі «Інформатика. Єдиний базовий курс. 9–11 класи» (автори Завадський І.О., Дорошенко Ю.О., Пилипчук О.П., Шестопалов Є.А.)

Основною структурною особливістю єдиного курсу інформатики є уніфікація змісту навчального матеріалу та вимог до навчальних досягнень рівня стандарту та академічного рівня.

1. Базові поняття програмування – 11 год.;
2. Основи програмування – 16 год.;
3. Растрова комп'ютерна графіка та анімація – 5 год.;
4. Бази даних. Системи керування базами даних – 8 год.;
5. Тривимірна комп'ютерна графіка та анімація – 8 год.;
6. Автоматизоване створення та публікація веб-ресурсів – 7 год.;
7. Мова гіпертекстової розмітки – 4 год.;
8. Колективна розробка проекту – 4 год.

Із наведеного списку навчальних тем академічного рівня, на вивчення яких відводиться 2 год. на тиждень, рівень стандарту (1 год. на тиждень) передбачає вивчення тем 1, 3, 4 і 6.

Для академічного рівня до цих тем додаються теми 2, 5, 7 і 8.

Такий підхід дозволяє застосовувати єдині методики викладання та навчально-методичне забезпечення для шкіл всіх профілів і рівнів.

Кожен параграф посібника розрахований на вивчення протягом одного уроку і має структуру, що відповідає санітарним нормам: теоретичний матеріал – 20 хв.; робота за комп'ютером – 25 хв.

Для поточного закріплення нового матеріалу та вироблення навичок на кожному уроці передбачено виконання практичних робіт за комп'ютером тривалістю до 25 хв. Тематичні роботи виконуються на окремих уроках після формування і закріплення теоретичних знань та практичних навичок, здобутих протягом кількох уроків, які складають розділ.

Практичні та тематичні роботи вміщені в робочий зошит «Інформатика. 11 клас. Рівень стандарту. Академічний рівень (половина)» / Бондаренко О.О., Ковшун М.І., Пилипчук О.П., Шестопалов Є.А. – Шепетівка: «Аспект», 2011. – 52 с.

Зміст

1. Базові поняття програмування

1.1. Принципи роботи у середовищі візуальної розробки програм

В цьому розділі ви познайомитесь з основами проектування та розробки програм для комп'ютера. Програмування – це процес створення комп'ютерних програм за допомогою мов програмування. Програмування поєднує в собі елементи мистецтва, математики й інженерної науки.

Програма та мова програмування

Програма *– впорядкована послідовність команд для комп'ютера, виконання якої реалізує алгоритм розв'язування певної задачі.*

Команди в програмі (програмному коді) записуються мовою програмування.

Мова програмування *– це система позначень, яка використовується для запису алгоритмів для реалізації (виконання) їх за допомогою комп'ютера.*

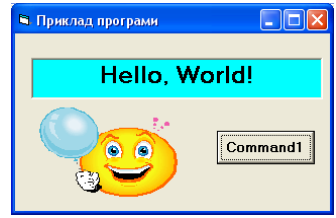
Класифікація мов програмування

Таблиця 1.1

Машинний код	Набір двійкових кодів для роботи центрального процесора певного типу
Мова асемблера	Мова для запису машинних кодів процесора зрозумілими людині символами – мнемоніками
Мова програмування високого рівня	Мова, що максимально наближена до людської мови, використовує для запису команд звичайні слова або їх скорочення (як правило, англійською мовою). Програма мовою високого рівня дозволяє формулювати завдання для комп'ютера у звичній для людини формі
Візуальне програмування	Системи програмування, які використовують графічний інтерфейс. Завдяки цьому програма створюється шляхом проектування макету в графічному вигляді з використанням готових елементів керування

Приклад 1.1. Приклад програми на мові **Visual Basic 6.0**.

```
Privat Sub Command1_Click()
    Label1.Caption = "Hello, World!"
End Sub
```



На малюнку наведено можливий вигляд вікна програми, де після натискання кнопки виконується наведений програмний код.

Середовище програмування Visual Basic

Система програмування *– це комп'ютерна система, призначена для створення програм.*

До складу системи програмування входять: текстовий редактор для введення та редагування тексту програми, компілятор, налагоджувач програми тощо. Призначення цих елементів розглянемо далі.

Інтегроване середовище програмування **Visual Basic IDE** (*Integrated Development Environment*) включає набір меню, панелей і вікон, службові програми, довідкову систему, що у сукупності утворюють «робоче місце» програміста.

IDE *– це «робоче місце» програміста, засобами якого створюються програми.*

Середовище програмування **Visual Basic IDE** є інтегрованим, тому що воно дозволяє виконувати всі дії, необхідні при розробці програмного продукту: проектування і опис складових частин програми, редагування програмного коду, компіляцію усіх елементів програми у виконуваний файл, відлагодження програми.

Запуск Visual Basic

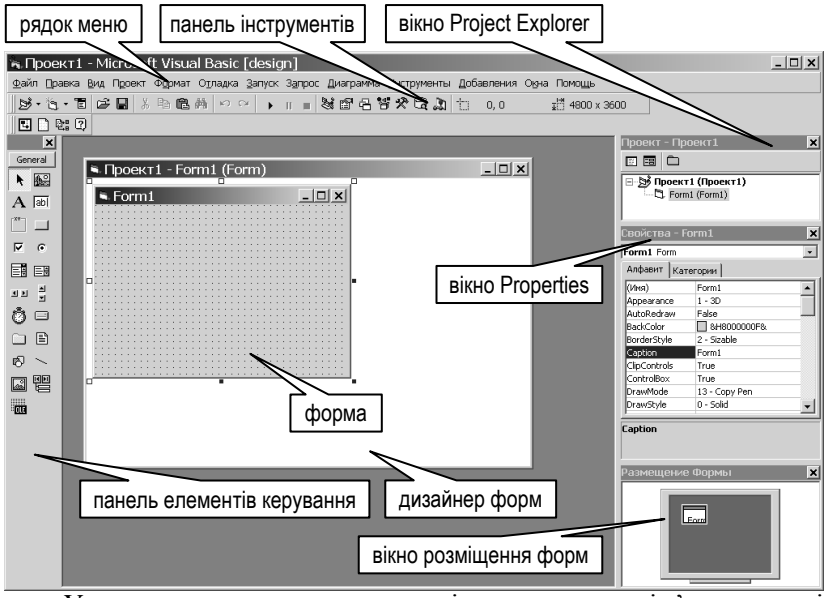
Для запуску **Visual Basic**:

- клацніть на кнопці Пуск на панелі задач Windows;
- в меню, що відкрилося, вкажіть на рядок *Программы*, потім перейдіть на рядок *Microsoft Visual Studio 6.0*;
- клацніть на рядку *Microsoft Visual Basic 6.0*. З'явиться діалогове вікно *New Project* (*Новий проект*), де потрібно вказати тип створюваного проекту. Щоб створити стандартну прикладну Windows-програму (застосунок), яка відкривається і працює у вікні, виберіть *Standard EXE*;
- натисніть кнопку *Open* (*Відкрити*).

Після виконання цих дій ми потрапляємо в середовище розробки **Visual Basic IDE**. При створенні нового проекту на екрані з'являється порожня форма – заготовка вікна майбутньої програми.

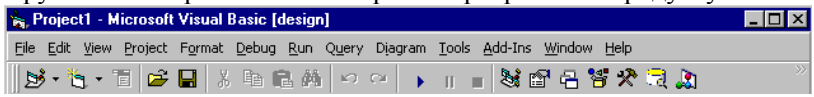
Основні об'єкти середовища IDE

Запустіть Visual Basic. Зверніть увагу на вікна, що з'явилися на екрані. ***Головне вікно*** використовується для керування створенням проекту і запуску розроблюваних програм.



У рядку заголовка головного вікна вказується ім'я проекту і поточний режим роботи Visual Basic: [*design*] (розробка), [*break*] (переривання), [*run*] (виконання).

Під рядком заголовка знаходиться ***Головне меню***, необхідне для доступу до різних функцій Visual Basic. Меню дозволяє керувати всією роботою зі створення програмного продукту.

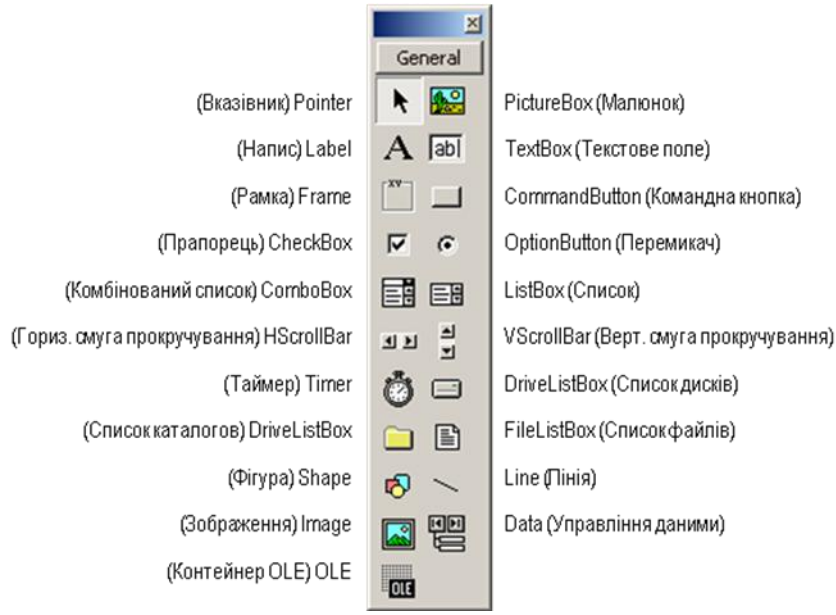


Під головним меню знаходиться ***Панель інструментів*** з кнопками, які призначені для швидкого доступу до деяких команд меню. Якщо ви помістите вказівник на кнопку панелі інструментів, то приблизно через секунду з'явиться спливаюча підказка про призначення даної кнопки.

Дизайнер форм (*Form Designer*). ***Форма*** (*form*) є основним елементом керування при розробці проекту в Visual Basic. Форми – основні будівельні блоки програм на Visual Basic – представлені у вікні дизайнера форм. З вікна *Form* починається розробка вашої програми. Воно призначене для редагування форм, тобто додавання та вилучення з них різних елементів керування.

Кожна відкрита форма має своє вікно дизайнера форм, яке у середовищі розробки зазвичай розташоване у лівому верхньому куті робочого поля головного вікна. Його можна переміщати в межах робочого поля, збільшувати, зменшувати.

Елементи керування (*Controls*), розміщені на формі, забезпечують зручну взаємодію користувача з комп'ютером. Дії



користувача спричиняють ***події*** (*events*) елементів керування, програмна обробка яких дозволяє керувати роботою комп'ютера.

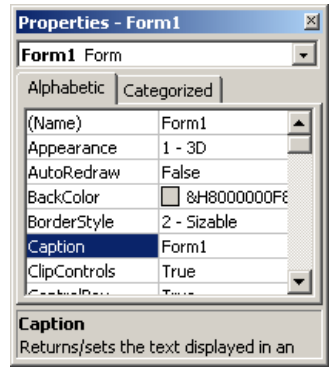
Якщо на екрані не видно вікна дизайнера форм, необхідно вибрати команду меню View / Object.

Панель елементів керування (*Toolbox*). Панель *ToolBox* використовується для додавання на форми проекту різних елементів.

Тут вибирають об’єкти, які потрібно помістити на форму. Наприклад, щоб намалювати лінію, слід клацнути кнопку *Line*. Потім помістити вказівник на те місце форми, де повинна починатися лінія, і, натиснувши ліву кнопку, перетягти вказівник до кінцевої точки лінії, після чого відпустити кнопку.

Інтерфейс користувача *– це сукупність засобів взаємодії програми з користувачем.* ***Компоненти*** *Visual Basic – це готові елементи інтерфейсу, які ви можете застосовувати у своїх програмах.*

У вікні ***Провідник проекту*** (*Project Explorer*) відображаються ті проекти, над якими ви працюєте в даний момент, а також структура кожного з них.



Вікно властивостей (*Properties Window*) призначене для перегляду і встановлення початкових значень деяких параметрів (властивостей) елементів керування. Список, що розкривається, у верхній частині вікна містить назви усіх елементів керування, розташованих на формі, з якою ви працюєте. Перелік властивостей обраного об’єкта (елемента керування) може бути поданий у двох виглядах: *Alphabetic* (За алфавітом) і *Categorized* (За категоріями).

Один з об’єктів завжди є «поточним об’єктом»: саме його властивості перелічені у вікні властивостей. Об’єкт стає поточним, коли на ньому клацнути. Розпізнати це найчастіше можна за обрамленням чи за яким-небудь іншим маркуванням.

Для настроювання властивостей об’єкта призначене вікно властивостей Properties.

Якщо вікно властивостей відсутнє на екрані, то необхідно викликати команду меню *View* ⇨ *Properties Window* або натиснути ***F4***.

Вікно розміщення форм (*Form Layout*) призначене для точного позиціювання форми на екрані при виконанні програми. Щоб задати положення форми, її зображення у вікні розміщення форм за допомогою миші переміщують в потрібне місце.

Поняття проекту

У **Visual Basic** ***проект*** – це сукупність файлів, що створюються в процесі розробки програми (див. таблицю 1.2). Крім того, проектом називають і саму програму у процесі її розробки.

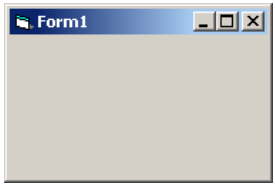
У файлі проекту (наприклад, *Project1.vbp*) міститься інформація про файли програми: список усіх використовуваних програмою файлів, назва проекту, конфігурація *IDE* для роботи над даним проектом і т.п. Проект має модульну структуру. Модулі бувають трьох типів: модуль форми, стандартний модуль та модуль класу. Ми в даному курсі будем створювати проекти, що найчастіше складатимуться з однієї форми, і будуть містити тільки модуль форми. Файл модуля форми (наприклад, *Form1.frm*) містить оголошення змінних, констант, типів даних, процедур обробки подій. Більшість файлів проекту не потребують «ручного» редагування, бо їх вміст генерується автоматично.

Виконання програми

Запустити програму на виконання можна такими способами:

1 спосіб. Відкрити меню *Run*, вибрати команду *Start*.

2 спосіб. Натиснути на панелі інструментів кнопку ▶.



Приклад 1.2. Запустіть програму на виконання − на екрані з'явиться стандартне вікно *Windows*-програми. Вікно порожнє, оскільки ми ще нічого на форму не поміщали, і має заголовок "Form1". Зверніть увагу: проект перейшов у режим виконання [*run*]. Форма дещо змінилась: зникли крапки, вона втратила зв'язок з головним вікном Visual Basic і поводиться, як незалежна програма. Щоб переконатися, згорніть на панель задач головне вікно Visual Basic − вікно проекту залишиться на екрані. Його можна пересувати по всьому екрану, вхопившись за заголовок, і на панелі задач *Windows* з'явиться відповідна кнопка. Вікно належним чином реагує на клацання кнопок керування у правому верхньому куті і дозволяє змінювати розміри.

Зупиніть виконання проекту (кнопка ■). Visual Basic вийде з режиму [*run*] і повернеться в режим [*design*].

Збереження проекту

Щоб зберегти щойно створений проект, або оновити вже існуючий зберігши зроблені доповнення, у меню *File* виберіть команду *Save Project*. Якщо проект зберігається вперше, необхідно виконати такі кроки:

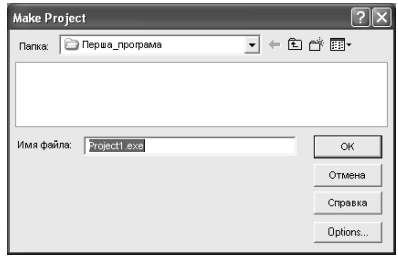
- створити папку для проекту;
- зберегти файл форми в папці проекту;
- зберегти файл проекту в папці проекту.

Збережіть проект, викликавши команду *File* ⇨ *Save Project*. У діалоговому вікні *Save File As*, що відкрилося, відкрийте або створіть загальну папку для VB-проектів, а в ній створіть нову папку для вашого проекту. Збережіть у папку файл форми (*Form1.frm*) і файл проекту (*Project1.vbp*).

Створення EXE-файлу

Компілятор *– програма, яка перекладає програму з мови програмування високого рівня на мову машинних кодів.*

Відкомпільовану програму можна зберегти для подальшого використання. Файл, що містить отриманий при компіляції програмний код, називається виконуваним файлом. Такий файл може мати розширення **.exe* або **.com*.



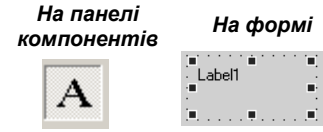
Щоб створити виконуваний файл проекту, необхідно виконати таку послідовність операцій:

- в меню *File* вибрати команду *Make*, поряд з якою вказана назва відкритого проекту з розширенням EXE;
- у діалоговому вікні *Make Project* указати ім'я для створюваного EXE-файлу і папку, в якій його буде збережено;
- якщо є необхідність змінити деякі параметри даного файлу (номер версії, заголовок та значок програми), натиснути кнопку *Options* і внести зміни;
- натиснути *OK*.

Елемент керування Label (Напис)

Label (Напис) – це елемент керування, що застосовується для відображення тексту, недоступного для безпосереднього

редагування користувачем: заголовків, підписів інших елементів керування тощо.



Для створення об'єкту *Напис* потрібно виконати такі дії:

1) вибрати компонент *Label* лівою кнопкою миші на панелі компонентів;
2) навести вказівник миші на форму і перетягуванням створити об'єкт *Напис*.

Розташування об'єкта на формі змінюють, перетягуючи його мишею, а розміри об'єкта можна змінити за допомогою маркерів.

Напис містить текст, який можна прочитати під час роботи програми.

Кожен елемент керування при додаванні на форму отримує унікальне ім'я (властивість ***Name***). **Visual Basic** автоматично надає властивості ***Name*** значення, яке містить назву типу елемента керування з порядковим номером в кінці. Наприклад, для першого елемента *Label* значення властивості *Name* дорівнює *Label1*, для другого – *Label2* і т. д. Ім'я використовується для позначення елемента керування в програмному коді і може бути змінене тільки у вікні *Properties* на етапі розробки.

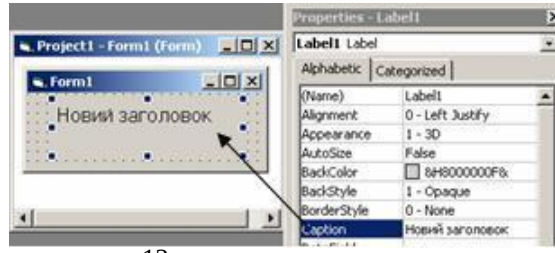
Програмісти, здебільшого, змінюють ім'я так, щоб воно вказувало на призначення елемента. При цьому часто використовують ***префікси***, які позначають належність об'єкта до певного типу.

Якщо елемент управління має змістовне ім'я, то легко визначити його роль у програмі.

Таким чином, ім'я складається з префікса, який ідентифікує тип об'єкта, і зручної назви, що описує призначення об'єкта в програмі:

Ім'я елемента = Префікс + Опис призначення

Для написів застосовують префікс *lbl*. Наприклад, якщо напис містить коментар, то замість імені *Label1* краще взяти *lblComment*.

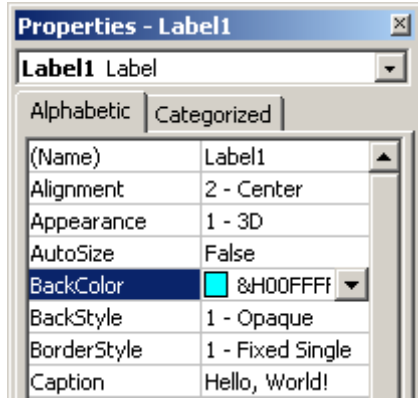


Caption (Заголовок) – це основна властивість елемента керування *Label*,

яка містить текст напису. Значення властивості *Caption* можна змінити як під час розробки вигляду форми у вікні *Properties*, так і в ході виконання проекту, запрограмувавши присвоєння властивості нового значення в програмному коді (детальніше − далі).

Щоб змінити значення властивості у вікні *Properties,* необхідно виконати такі дії:

1) виділити елемент керування на формі;
2) у вікні *Properties* в лівій частині таблиці вибрати властивість *Caption*;
3) клацнути у правій частині таблиці напроти назви властивості і ввести текст.



Приклад 1.3. Властивостям елементу керування *Label1* надані значення:

- властивості *Caption* (Заголовок) надано значення «Hello, World!»;
- для властивості *Alignment* (Вирівнювання) вибрано значення *2-Center*.
- напис виділено рамкою: для властивості *BorderStyle* (Тип межі) вибрано *1 – Fixed Single*.

- параметри шрифту змінено на *Arial, 16, жирний*. Для виклику вікна *Шрифт* потрібно активізувати рядок властивості *Font* і клацнути в ньому кнопку ...;
- для властивості *BackColor* (Колір фону) на вкладці *Palette* вибрано блакитний колір (*FFFF00*&).

Питання для самоконтролю

1. *Що таке мова програмування?*
2. *Чому середовище програмування Visual Basic IDE є інтегрованим?*
3. *Перелічіть складові системи програмування.*
4. *Як запустити систему програмування Visual Basic?*
5. *Що таке форма?*
6. *Як додати елемент керування на форму?*
7. *Як змінити розташування об'єкта на формі і його розміри?*
8. *Як змінити значення певної властивості об'єкта?*
9. *Як запустити проект на виконання?*

10. Опишіть послідовність додавання елементів на форму і зміну властивостей для елементу Напис.
11. Що таке проект?
12. Які файли входять до складу проекту?
13. Опишіть послідовність дій при збереженні проекту.
14. Для чого потрібно створювати EXE-файл для існуючого проекту?
15. Опишіть послідовність операцій, необхідну для створення виконуваного файлу.

1.2. Процедури опрацювання подій

Поняття об'єкта у програмуванні

Технологія роботи у середовищі **Visual Basic** базується на ідеях об'єктно-орієнтованого та візуального проектування інтерфейсу користувача. Програміст створює певну сукупність об'єктів та описує їх взаємодію. Кожен об'єкт має набір властивостей та може виконувати певні дії. Деякі об'єкти є візуальними, тобто зображуються на екрані. Прикладами візуальних об'єктів є відомі вам елементи керування у вікні: кнопки, списки, текстові поля тощо.

*Щоб об'єкт виконав якусь дію, потрібно для нього викликати відповідний фрагмент програми – **метод.***

Отже, структурною одиницею при розробці інтерфейсу користувача в середовищі **Visual Basic** є візуальний об'єкт із певним набором властивостей і методів, який називається ***елементом керування***. Автоматизація розробки інтерфейсу користувача досягається завдяки можливості переносити елемент на форму (у програму) з панелі компонентів і змінювати його властивості, не вносячи вручну змін до програмного коду. Програміст має змогу змінювати значення властивостей даного об'єкта і викликати різні його методи. Також є можливість програмування реакції комп'ютера на різні події, які можуть виникати внаслідок деяких дій користувача. При виконанні програми користувач ініціює певні події і тим самим керує виконанням програми.

Приклад 1.4. Розглянемо взаємодію двох об'єктів: людини та телефонного апарату. Обробляючи подію «надходження сигналу з

телефонної лінії» апарат «вмикає» дзвінок. Дзвінок телефону − це подія, на яку реагує людина: піднімає трубку. Щоб подзвонити комусь, людина застосовує метод «набрати номер». З телефоном при цьому відбувається послідовно декілька подій «натискання кнопки». А властивості характеризують апарат: «колір», «висота», «кількість кнопок» тощо.

Атрибути об'єкта

Будь-яка написана на **Visual Basic** *Windows*-програма – це набір об'єктів (*Objects*). Так само як, скажемо, комп'ютер – набір знімних модулів − плат, приводів і т.д. Об'єкт характеризується певними атрибутами, які поділяються на три категорії:

- події (*Events*), на які об'єкт може реагувати, за умови, що програміст напише програмний код обробки події;
- методи – являють собою окремі дії, які об'єкт здатний виконувати;
- властивості (*Properties*) – характеристики об'єкта.

Кожен об'єкт у Visual Basic, наприклад, форма чи елемент керування, має свій власний набір властивостей, що його описують. Проте деякі властивості притаманні багатьом об'єктам.

Таблиця 1.2

Деякі спільні властивості елементів керування Visual Basic	
Властивість	**Опис**
Height	Висота елемента керування. Вимірюють у спеціальних одиницях − твіпах (див. далі).
Width	Ширина елемента керування (у твіпах)
Left	Відстань (у твіпах) від елемента керування до лівого краю його контейнера (див. далі)
Top	Відстань (у твіпах) від елемента керування до верхнього краю його контейнера
Name	Ім'я, яке використовується для посилань на елемент керування в програмі. Не може змінюватися під час виконання програми
Enabled	Логічна (*True*/*False*, тобто *Так/Ні*) властивість, яка визначає, чи може користувач працювати з цим елементом керування
Visible	Логічна (*True*/*False*, тобто *Так/Ні*) властивість, яка визначає видимість елемента керування під час виконання програми

Подія – це те, що відбувається в програмі або за її межами. Подією є будь-який вплив на елемент керування в активному вікні від миші чи клавіатури. Наприклад, коли користувач тисне на кнопку, відбувається відразу кілька подій: натискаються кнопка миші та командна кнопка *CommandButton*, відпускається кнопка миші. Ці три дії відповідають подіям *MouseDown*, *Click*, *MouseUp*. Події, які відбуваються внаслідок дій користувача (наприклад, при переміщенні миші, натисканні клавіші на клавіатурі, клацання в текстовому полі), існують для багатьох елементів керування.

Методи являють собою фрагменти програмного коду, які вбудовані безпосередньо в елемент керування і виконують ту чи іншу задачу. Хоча різні об'єкти мають різні методи, деякі з методів притаманні багатьом об'єктам. Наприклад, спільними для багатьох елементів керування Visual Basic є методи *Move* (при виклику переміщує об'єкт), *Drag* (обробляє операції на зразок «перетягнути і відпустити»).

Система програмування **Visual Basic**, як і інші подібні системи, дозволяє формувати візуальну складову проекту (інтерфейс) без написання коду, а простими засобами графічного редагування (компонування). Не випадково цей процес називають візуальним конструюванням інтерфейсу користувача.

Але сформована візуальна складова повинна ще певним чином реагувати на ті чи інші події. До таких можна віднести сигнал від миші чи клавіатури, зміну стану елемента керування, одержання повідомлення від іншої *Windows*-програми і т.д.

Елемент керування Command Button (Командна кнопка)

Командна кнопка – один з найрозповсюдженіших елементів керування. Командна кнопка призначена для запуску чи закінчення виконання проектом певних дій, які викликаються натисканням на кнопку.

Вигляд кнопки на панелі *ToolBox* і на формі показано на малюнку.

Помістіть на форму елемент керування *CommandButton*. Даний об'єкт отримає ім'я *Command1*. Виділіть об'єкт *Command1* і перегляньте список його властивостей у вікні *Properties*. При розробці інтерфейсу здебільшого змінюють властивості *Name* (трибуквений префікс для імені командної кнопки – ***cmd***), *Caption*, встановлюють потрібні значення властивостей *Enabled* та *Visible*.

Приклад 1.5. Змініть для об'єкта *Command1* значення властивості *Name* на *cmdHello*. Надайте властивості *Caption* значення «Привітатися». Запустіть проект на виконання.

В режимі [*run*] на кнопку можна натискати мишкою, але при цьому нічого не відбувається, адже ми не написали програму, яка б «пояснила» комп'ютеру, що потрібно робити при натисканні на кнопку.

Подійне програмування

У Visual Basic керування відбувається за допомогою подій, тобто коли в проекті відбувається подія, то виконується процедура опрацювання події (***event procedure***). Значна частина написаного вами програмного коду буде виконуватися у відповідь на дії користувача, тобто при виникненні подій. Такий тип програмування передбачає, що ви повинні знати, коли відбуваються події, і вміти створювати програмний код, який реагує на них.

Для кожного об'єкта існує набір стандартних подій, що можуть виникнути при роботі програми, і для кожної з них може бути написана процедура, яка обробляє подію.

Коли відбувається деяка подія, **Windows** посилає програмі повідомлення. Програма повинна визначити, яка подія стоїть за цим повідомленням, і виконати відповідні дії. Процедури подій повідомляють комп'ютеру, що робити у відповідь на подію. Якщо в програмі немає процедури для даної події, вона ігнорується.

Процедури опрацювання подій вміщують у вікні коду.

Code Window (Вікно коду)

Щоб написати процедуру обробки події для певного об'єкта, перейдіть у вікно редагування коду форми, виконавши одну з дій:

- двічі клацнути мишею на об'єкті;
- вибрати мишею об'єкт і натиснути ***F7***.

В основній частині вікна система підготувала оброблювач цієї події, оформлений у вигляді процедури.

Якщо ж програмний код було створено раніше, то перейти до нього можна, вибравши команду меню *View* ⇨ *Code* або клацнувши у вікні проекту кнопку *View Code* (див. мал.).

У верхній частині вікна коду знаходяться два списки, що розкриваються, – список об'єктів (*Object list*) і список процедур

(*Procedure list*). Список об'єктів містить імена всіх елементів, що є на формі. У списку процедур перелічені всі події, на які може реагувати цей об'єкт.

При виборі зі списків об'єкта і події IDE автоматично створює шаблон процедури обробки цієї події або показує її код, якщо вона створена раніше.

Приклад 1.6. Клацніть на кнопці, що розкриває список

процедур, у верхньому правому полі вікна коду. Розгляньте інші доступні події для елемента керування *CommandButton*.

Структура процедури обробки подій

Оператор *– команда комп'ютеру виконати певну дію, записана за правилами мови програмування.*

Процедура – це набір операторів, представлений у вигляді іменованого блоку коду, який можна викликати з будь-якої частини програми за його ім'ям. Такий код може змінювати властивості об'єктів, отримувати дані від користувача, переміщати об'єкти у формі, обчислювати значення за формулою або записувати дані в базу даних тощо. Тут розглядаються процедури особливого виду – процедури обробки подій.

Загальна структура процедури обробки подій така:

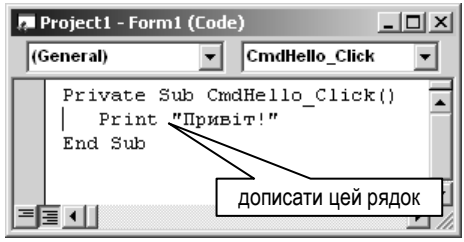
Private Sub НазваЕлемента_НазваПодії()
[програмний код]
End Sub

Рядок заголовка говорить про те, що ми працюємо з ***Private*** (тобто доступною тільки для даної форми), ***Sub****routine* – стандартною підпрограмою – процедурою обробки подій. Ця процедура виконується, коли для елемента керування *НазваЕлемента* відбувається подія *НазваПодії*. Безпосередньо після заголовка записують код процедури обробки подій – послідовність операторів. Комп'ютер виконує спочатку перший оператор, потім

другий і всі наступні оператори один за одним, до оператора завершення процедури *End Sub*.

Якщо двічі клацнути на командній кнопці, створюється процедура обробки події *Click*. Ім'я процедури обробки події складається з імені об'єкта та імені події, зв'язаних символом підкреслення (*Command1_Click*()). Між заголовком і *End Sub* необхідно написати оператори, що будуть виконані при клацанні кнопки.

Приклад 1.7. Продовжимо роботу над проектом з прикладу 1.5. Нехай клацання кнопки повинне призводити до виведення на форму тексту «Привіт!». Цю дію виконує оператор *Print*. Запрограмуйте реакцію на натискання кнопки «Привітатися»:



1) двічі клацніть на ній;
2) у вікні програмного коду внесіть зміни до процедури *Private Sub CmdHello_Click()*, відповідно до малюнка.

Запустіть проект на виконання і проаналізуйте дію кнопки.

Події для форми

У цьому курсі ми розглянемо тільки дві події для форми – *Click* і *Load*.

Подія *Click* відбувається, коли користувач клацає на формі лівою кнопкою миші. Подія *Load* відбувається, коли форма завантажується у пам'ять комп'ютера. Це зручний момент для встановлення початкових значень різних властивостей елементів керування та інших параметрів проекту.

Усі імена процедур обробки подій для форми мають формат: ***Form_****EventName*.

Незалежно від того, яке значення властивості Name ви встановлюєте для форми, ім'я процедури обробки події міститиме слово Form.

Оператор присвоєння

Оператор присвоєння – основний оператор більшості мов програмування, використовується для надання змінній потрібного значення. У мові **Basic** він має вигляд:

Змінна = НовеЗначення, де:

- символ «=» якраз і позначає оператор присвоєння, тобто, дію «присвоїти (надати) значення»;
- *Змінна* – ім'я змінної або властивості елемента керування;
- *НовеЗначення* – константа, змінна або вираз того ж самого типу, що і *Змінна*. Детальніше типи даних будуть розглянуті далі.

Оператор присвоєння діє таким чином: змінній, вказаній в лівій частині оператора присвоєння, присвоюється значення виразу, записаного у правій частині. Ліворуч від оператора присвоєння може бути вказане тільки одне ім'я змінної або властивості елемента керування.

VarName =NewValue		**Приклади**
▪ Ім'я змінної (символьне позначення адреси комірки пам'яті, куди треба помістити результат) ▪ Властивість деякого об'єкта	▪ Константа	A = 6 cmdBlue.Visible = False
	▪ Вираз	Sum = Sum+5
	▪ Ім'я змінної	A = B

Приклад 1.8. Нехай значення змінної *A* дорівнює 3. Тоді після виконання вказівки присвоєння

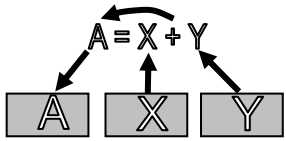
A =A + 1

змінна *A* отримає значення 4.

Приклад 1.9. Нехай *A*=10, *X*=2, *Y*=3. Тоді після виконання вказівки присвоєння

A = X + Y

змінна *A* отримає значення 5.



Попереднє значення змінної A (число 10) буде втрачене.

Приклад 1.10. У вікні коду знаходиться процедура опрацювання події *Click* для елемента керування *Command1* (командна кнопка). Внаслідок виклику процедури (тобто, після клацання на командній кнопці) текст напису *Label1* стає заголовком форми.

Цю дію позначає оператор присвоєння, який властивості *Caption* форми надає значення властивості *Caption* елемента *Label1*.

```
Private Sub Command1_Click()
    Form1.Caption = Label1.Caption
End Sub
```

Більшість властивостей форм і елементів керування є змінними. Їх значення можна задати в режимі розробки і змінювати в ході виконання програми за допомогою оператора присвоєння.

Взаємозв'язок властивостей, методів і подій

Хоча властивості, методи і події – це зовсім різні речі, дуже часто вони виявляються взаємозалежними. Наприклад, якщо елемент керування переміщується за допомогою методу *Move* (зокрема, у відповідь на подію), змінюються властивості *Top* і *Left* елемента керування.

Багатьом властивостям можна присвоювати нові значення. Деякі методи потребують при виклику вхідних параметрів.

Основна відмінність між методами й властивостями полягає в тім, що із властивостями можна працювати як під час розробки проекту, так і під час виконання програми, тоді як методи доступні тільки при виконанні.

Взаємозв'язок властивостей, методів і подій означає, що багато дій можна виконувати декількома способами, використовуючи відповідний програмний код роботи з подіями і методами об'єкта.

Приклад 1.11. Переміщення кнопки *CommandButton*

Переміщення кнопки за допомогою властивостей	**Переміщення кнопки за допомогою методу Move**
Command1.Left = 100 *Command1.Top = 100*	*Command1.Move 100,100*

Приклад 1.12. Відображення і приховування форми на екрані

Робимо форму видимою, змінивши властивість	**Відображаємо форму за допомогою методу**
Form1.Visible = True	*Form1.Show*
Робимо форму невидимою, змінивши властивість	**Приховуємо форму за допомогою методу**
Form1.Visible = False	*Form1.Hide*

Питання для самоконтролю

1. *Що таке властивості елемента керування? Наведіть приклади.*
2. *Опишіть послідовність зміни значень властивостей.*
3. *Що таке методи елемента керування?*
4. *Що таке події? Наведіть приклади подій.*
5. *Опишіть послідовність додавання на форму і зміни властивостей елементу Командна кнопка.*
6. *Як перейти до вікна програмного коду? Опишіть структуру вікна програмного коду.*

7. *Імена яких об'єктів містить список Object list у верхній частині програмного коду?*
8. *Перелічіть та опишіть властивості, спільні для багатьох елементів керування.*
9. *Поясніть структуру процедури обробки події.*
10. *Як внести зміни до програмного коду для програмування реакції на натискання командної кнопки?*
11. *Для чого, на вашу думку, змінюють значення властивості Name елементів керування, що розміщені на формі? Який префікс використовують в іменах командних кнопок (CommandButton)?*
12. *Які події елемента Form вам відомі?*
13. *Поясніть синтаксис і схему дії оператора присвоєння.*
14. *Яким буде результат виконання оператора Label1.Text = Form1.Caption?*

1.3. Етапи розв'язування задач на комп'ютері

Етапи розв'язування задачі на комп'ютері

Найбільш загальними складовими комп'ютерної програми є дані, логіка роботи та інтерфейс користувача.

Дійсно, кожна з програм обробляє певні *дані*. Деякі з них користувач вводить під час виконання програми, користуючись *інтерфейсом користувача*. Обробка даних підпорядкована певному алгоритму, який, по суті, описує *логіку роботи* програми.

Тому, розв'язування прикладної задачі на комп'ютері з використанням програмування включає такі етапи:

1) постановка задачі;
2) проектування інтерфейсу;
3) розробка алгоритму;
4) написання програмного коду;
5) тестування й налагодження програми;
6) аналіз результатів.

І етап. Постановка задачі. Розв'язування практичної задачі починається з її формулювання таким чином, щоб чітко визначити умови задачі:

- Що дано?
- Які дані допустимі?
- Які результати, в якому вигляді повинні бути отримані?

Для чіткого визначення переліку початкових даних і результатів виконання програми зручно використовувати таку форму:

Дано: <Перелік початкових даних>

Потрібно: <Перелік потрібних результатів>

Зв'язок: <Система рівнянь або тверджень, що зв'язують вхідні та шукані дані>

При <Умови допустимості початкових даних>

II етап. Проектування інтерфейсу. Перед розробкою інтерфейсу форми поставте перед собою питання:

- Які елементи керування необхідні для введення інформації?
- Яку інформацію буде обробляти комп'ютер?
- Які елементи керування необхідні для виведення результатів виконання програми?

Розробка інтерфейсу проекту **Visual Basic** включає два кроки:

1) розміщення елементів керування на формі;
2) налаштування властивостей елементів керування.

Дружній інтерфейс користувача *(зручне розташування елементів керування на формі та продумана їх взаємодія), прикладом якого є інтерфейс Windows, буде гарною основою для вашого проекту.*

III етап. Складання алгоритму. Для того, щоб комп'ютер виконав потрібні обчислення і повідомив нам правильний результат, ми повинні скласти для нього чітку інструкцію, задати послідовність дій, яку необхідно виконати для розв'язування задачі. Цю інструкцію називають алгоритмом розв'язування задачі.

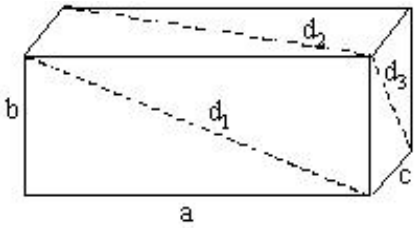
IV етап. Складання програмного коду за розробленим алгоритмом. Програмування (складання програми) – кодування складеного алгоритму однією з мов програмування. Створення програмного коду мовою **Visual Basic** зводиться до написання процедур обробки подій для елементів керування та, за потреби, інших процедур.

V етап. Тестування і налагодження програми. На даному етапі проводиться перевірка правильності роботи програми за допомогою тестів і виправлення виявлених помилок. Тест – це набір спеціально підібраних вихідних даних і результатів роботи програми, очікуваних при цих даних. Процес тестування включає: запуск програми; введення тестового набору даних; порівняння результатів, наведених у тестовому прикладі, з результатами, отриманими після виконання програми.

VI етап. Аналіз результатів. На завершальному етапі програма виконується з даними, що входять до розв'язуваної задачі. Після остаточного виконання програми проводиться аналіз результатів. У випадку невірогідності результатів можлива зміна самого підходу до розв'язання задачі (алгоритму) і повернення до етапу постановки задачі для її корегування та уточнення.

Приклад 1.10. Через круглий ілюмінатор корабля, що затонув, потрібно витягти скриню з коштовностями. Чи вдасться це зробити?

Звичайно, при такому формулюванні задачі відповісти на поставлене питання неможливо. Тому уточнимо постановку задачі, з'ясувавши, які дані потрібні для розв'язування задачі і які значення вважатимемо результатом. Ілюмінатор корабля має форму кола. Припустимо, що скриня має форму паралелепіпеда.



Нехай D – діаметр ілюмінатора,

a, b, c – розміри скрині,

d_1, d_2, d_3 – діагоналі бічних поверхонь скрині.

Скриню можна просовувати через ілюмінатор однією з трьох бічних граней, отже, достатньо, щоб діаметр ілюмінатора виявився більшим хоча б від однієї з трьох діагоналей граней скрині. Для розв'язування задачі необхідно перевірити три умови. Введемо допоміжні змінні P_1, P_2, P_3, які одержують значення 1 або 0 в залежності від виконання відповідної умови.

Дано: R – радіус ілюмінатора,

a, b, c – розміри скрині,

Потрібно: висновок про те, чи є діагональ, менша від діаметра.

Зв'язок: $d_1 = \sqrt{a^2 + b^2}$. Якщо $D > d_1$, то P_1=1.

$d_2 = \sqrt{a^2 + c^2}$. Якщо $D > d_2$, то P_2=1.

$d_3 = \sqrt{c^2 + b^2}$. Якщо $D > d_3$, то P_3=1.

Якщо $P_1 + P_2 + P_3 = 0$, то робимо висновок: коштовності недоступні; в іншому випадку, скриню можна дістати.

При $R>0$; $a>0$; $b>0$; $c>0$.

Постановка задачі може ускладнюватися і доповнюватися. Наприклад, може знадобитися врахувати вагу скрині – чи зможе водолаз підняти її. Якщо передбачається застосування підйомного механізму, треба враховувати товщину тросів, якими обв'яжуть скриню тощо.

Поняття алгоритму

Алгоритм – це скінчена послідовність вказівок, яка однозначно описує процес розв'язування задачі певного типу.

Кожен алгоритм припускає наявність деяких вихідних даних і приводить за обмежений час до певних результатів.

Поняття алгоритму є в інформатиці фундаментальним, як, наприклад, «точка» і «площина» – у геометрії, «простір» і «час» – у фізиці).

Слово «алгоритм» походить від латинської форми написання імені арабського математика Аль-Хорезмі (800-847 рр.), що сформулював правила чотирьох арифметичних дій над числами.

З курсу математики добре відомі алгоритми виконання арифметичних операцій над багатоцифровими числами; алгоритми знаходження коренів лінійних і квадратних рівнянь; алгоритми поділу відрізка на *N* рівних частин, побудови трикутника за заданими його елементами та ін.

Приклад 1.11. Алгоритм побудови графіка функції виду $y = \left| f(|x|) \right|$ може бути поданий у такий спосіб:

1) побудувати графік функції $y = f(x)$ для значень $x \geq 0$;
2) побудовану лінію симетрично відобразити відносно осі *Oy*;
3) частину побудованого графіка, що знаходиться в нижній півплощині, симетрично відобразити відносно осі *Ox*;
4) частину графіка, що знаходиться в нижній півплощині, стерти.

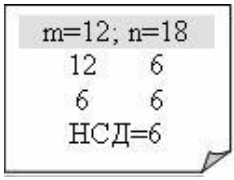
У повсякденному житті людина зустрічається з різними алгоритмами, спрямованими як на обчислення певного значення, так і не зв'язаними з обчислювальними процесами, а такими, що визначають різноманітні послідовності дій.

Приклад 1.12. Числові алгоритми створюються для розв'язування обчислювальних задач.

Алгоритм Евкліда. Алгоритм знаходження найбільшого спільного дільника двох чисел *m* і *n* − *НСД(m,n)* − був описаний в III столітті до н.е. в класичному трактаті «Початки» грецького математика Евкліда.

Розв'язок можна одержати шляхом послідовного віднімання меншого числа від більшого доти, поки $m \neq n$.

- *Крок 1.* Якщо $m=n$, то перейти до кроку 5.
- *Крок 2.* Визначити більше з чисел.
- *Крок 3.* Відняти від більшого числа менше. Отриманою різницею замінити більше число.
- *Крок 4.* Перейти до кроку 1.
- *Крок 5.* *НСД(m,n)=m*.



Приклад 1.13. Алгоритми ігрових задач.

Для багатьох ігор, результат яких залежить не від випадкового збігу обставин, а від кмітливості гравця і попереднього розрахунку, існують алгоритми виграшу (виграшні стратегії).

Гра Баше. Є 11 предметів. За один хід гравець може взяти 1, 2 або 3 предмети. Програє той, хто змушений взяти останній предмет.

Алгоритм виграшу для 1-го гравця:

- *1 хід*: узяти 2 предмети.
- *2 хід і далі*: брати стільки предметів, щоб кількість предметів, узятих разом із суперником за черговий хід, становила 4.

Властивості алгоритму

Розробка алгоритму – найважливіший етап розв'язування задачі. Від якості алгоритму залежать правильність результатів, ефективність використання часу та ресурсів комп'ютера.

Алгоритм повинен мати такі властивості:

Масовість – придатність для обробки великої кількості варіантів вхідних даних. Наприклад, алгоритм обчислення площі трапеції за заданими основами та висотою повинен давати правильний результат при будь-яких коректних вхідних даних, алгоритм знаходження коренів квадратного рівняння повинен бути придатним для розв'язування будь-якого рівняння виду $ax^2+bx+c=0$.

Визначеність (однозначність) – однозначність тлумачення правил виконання дій і порядку їхнього виконання. Наприклад, розпорядження «Порівняти числа *A* і *B*» може бути витлумачене порізному, тому в алгоритмі неприпустиме.

Дискретність означає, що алгоритм повинен складатися з окремих завершених дій. Розпорядження повинні мати форму «виконати», «зробити», а не «виконувати», «робити».

Результативність означає, що виконання послідовності вказівок алгоритму повинно приводити до цілком конкретного

результату. Наприклад, алгоритм розв'язування квадратного рівняння повинен містити перевірку випадку, коли коренів не існує, і передбачати інформування про відсутність коренів.

Формальність означає, що будь-який виконавець, здатний сприймати і виконувати вказівки алгоритму (навіть не розуміючи їх змісту), діючи за алгоритмом, може виконати поставлене завдання. Як відомо, автомати правильно розв'язують багато задач за заданими їм алгоритмами, хоча суті задач, безумовно, автомати розуміти не можуть.

Сукупність всіх команд, які даний виконавець може виконувати, називають його ***системою команд.***

Кожен алгоритм розрахований на певного виконавця, тому повинен містити тільки ті команди, які входять до системи команд виконавця.

Скінченість – потенційна виконуваність алгоритму. Алгоритм повинен складатися зі скінченого числа кроків, кожний з яких вимагає для свого виконання скінченого проміжку часу.

Приклад 1.14. Нехай деякий учень задумав натуральне число *X*. Запропонуємо йому виконати алгоритм:

1) помножити задумане число на три;
2) від добутку відняти одиницю;
3) отриману різницю помножити на 4;
4) від добутку відняти 5;
5) повідомити результат (*Y*).

Якщо ми знаємо результат виконання алгоритму – *Y*, ми легко відгадаємо задумане число *X*, якщо скористаємось алгоритмом відгадування:

1) додайте до *Y* число 9;
2) суму поділіть на 12;
3) повідомте результат (*X*).

Зрозуміло, що дії, виконані над числом *X*, описує формула:

$(3x - 1) \cdot 4 - 5 = y$.

Отже, $x = (y + 9) / 12$.

Алгоритм відгадування числа *X* має властивість *формальності*, тобто якщо інший учень правильно виконає дії алгоритму, він отримає потрібний результат – відгадає число *X*, навіть не знаючи, обчислення за якою формулою цей алгоритм реалізує.

Форми подання алгоритмів

Існують різні форми подання алгоритмів – словесна, формульна, словесно-формульна, графічна, у вигляді програмного коду та інші – залежно від того, на якого виконавця орієнтований алгоритм.

У словесній формі можна подати багато обчислювальних алгоритмів, наприклад, правила виконання арифметичних дій над багатоцифровими числами, обчислення коренів квадратного рівняння, довжини кола, площі трикутника, алгоритм Евкліда для знаходження найбільшого спільного дільника двох чисел та ін.

Обчислювальні алгоритми можна задавати й у вигляді формул. Так, алгоритм обчислення суми членів нескінченно спадної геометричної прогресії можна подати у вигляді формули $S = \frac{b_1}{1-q}$, а алгоритм обчислення площі прямокутного трикутника – у вигляді $S = \frac{a \cdot b}{2}$. Записуючи алгоритми, часто комбінують словесне і формульне подання вказівок.

Для графічного подання вказівок використовують блок-схеми.

Блок-схема алгоритму – графічне зображення алгоритму у вигляді сукупності з'єднаних блоків.

Графічне зображення вказівок

Таблиця 1.4

Назва блоку	Опис дії
Початок (Кінець)	Позначає початок та завершення алгоритму
Процес	Позначає дію, яку потрібно виконати. Прямокутником може бути позначена як вказівка виконати окрему дію (додати два числа, накреслити лінію), так і послідовність логічно пов'язаних дій (виконати розрахунки за заданими формулами, намалювати малюнок, відтворити мелодію), тобто певний процес.
Введення/ виведення	Позначає введення вхідної інформації та виведення проміжної і результуючої інформації.

	Позначає перевірку значення логічного виразу, деякої умови. Логічний вираз може набувати одного з двох значень – *TRUE* (істина, так) або *FALSE* (хибність, ні).

Базові алгоритмічні структури

Різні типи задач вимагають різних підходів до їх розв'язування і, відповідно, описуються різними алгоритмами. Розмаїтість цих алгоритмів велика, але для опису послідовності виконання дій при складанні будь-якого алгоритму використовуються подібні блоки, які називають базовими структурами.

Логічна структура будь-якого алгоритму може бути подана комбінацією трьох базових структур: слідування, розгалуження, повторення (цикл).

Принцип структурного програмування полягає в тому, що на будь-якому етапі проектування алгоритму порядок виконання дій задається однією з базових структур:

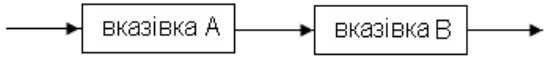
- слідування,
- розгалуження,
- повторення.

Найважливішою особливістю базових структур алгоритмів є те, що кожна з них має єдиний вхід і єдиний вихід.

Кожен прямокутник на блок-схемі алгоритму може бути замінений сукупністю простіших вказівок, що у свою чергу являють собою базові структури алгоритмів чи їх комбінації. При конструюванні алгоритму вихід кожної базової алгоритмічної структури приєднується до входу наступної. Як наслідок, алгоритм в цілому та кожна його частина являє собою ланцюжок базових алгоритмічних структур, що має важливе значення для аналізу алгоритмів, їхнього розуміння, корегування, доказу правильності

Слідування

Базова структура «Слідування» утворюється діями, що виконуються одна за одною, без пропусків або повторень. Алгоритми, у яких використовується тільки структура «Слідування», називаються ***лінійними***.



У структуру «Слідування» можуть бути організовані вказівки ввести початкові значення змінних, обчислити нові значення змінних, вивести результати обчислень тощо.

Приклад 1.15. ***Задача***. Скільки комп'ютерів можна встановити в комп'ютерному класі, довжина якого a м, ширина b м, якщо згідно з санітарними нормами на 1 ПК повинно припадати 6 м^2?

Математична постановка задачі:

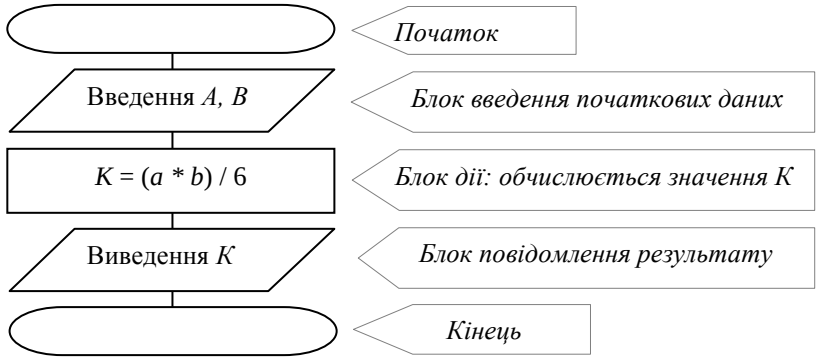
Дано: a – довжина класу (м), b − ширина класу (м).

Потрібно: K – кількість комп'ютерів.

Зв'язок: $K = \dfrac{ab}{6}$.

При $a>0$; $b>0$.

Блок-схема алгоритму:



З базовими структурами «Розгалуження» і «Повторення» ви ознайомитесь далі. Алгоритми розв'язування складних задач, здебільшого, містять у собі базові структури всіх трьох типів.

Питання для самоконтролю

1. *Перерахуйте етапи розв'язування задачі з використанням комп'ютера.*
2. *Опишіть зміст кожного з етапів розв'язування задачі.*
3. *З яких кроків складається етап проектування інтерфейсу?*
4. *Що таке алгоритм? Назвіть основні вимоги до алгоритмів і поясніть суть кожної з них.*
5. *Нехай маємо такий алгоритм:*
   - *прочитати перше число – a1;*
   - *прочитати друге число – a2;*
   - *поділити число a1 на a2;*
   - *записати результат.*

   Чи має даний алгоритм властивості масовості та однозначності?
6. *Деяке задане число, більше одиниці, необхідно зменшити до одиниці шляхом ділення на два:*
   - *поділити число на два;*
   - *за результат прийняти частку (нове число);*
   - *якщо частка не дорівнює 1, перейти на перший крок; інакше – записати результат.*

Чи має даний алгоритм властивість скінченності?

7. *Нехай маємо такий алгоритм:*
   - *взяти 0,5 кг борошна;*
   - *взяти склянку цукру;*
   - *взяти 1 г лимонної кислоти;*
   - *взяти 4 яєчні жовтки;*
   - *взяти склянку кефіру;*
   - *спекти пиріг.*

   Чи має даний алгоритм властивості формальності та однозначності?
8. *Виконайте алгоритм Евкліда для чисел: а) 75 і 120; б) 75 і 84. Які властивості має алгоритм Евкліда?*
9. *Прямокутник, довжини сторін A і B якого задовольняють умову a/b=b/(a-b), називається «золотим». Складіть алгоритм для перевірки, чи є даний прямокутник «золотим». Яким може бути результат виконання складеного вами алгоритму?*

Скласти блок-схеми алгоритмів для розв'язання задач:

10. *Пішохід пройшов S_1 км за час T_1 годин. Яку відстань пройде пішохід за час T_2?*
11. *Ширина шпалер 70 см. Скільки метрів шпалер треба купити для ремонту кімнати довжиною a м, шириною b м та висотою h м? Наявністю вікон та дверей знехтувати.*
12. *Скільки грамів фарби буде потрібно для фарбування стола, якщо на фарбування 1 м^2 потрібно x грамів фарби?*

1.4. Основи програмування мовою Visual Basic

Складові мови програмування

Основними складовими будь-якої мови програмування є алфавіт, синтаксис і семантика.

До ***алфавіту*** мови програмування, як правило, входять: літери латинського алфавіту, цифри, знаки арифметичних операцій, спеціальні символи, ключові (зарезервовані) слова мови.

У мові **Visual Basic** при створенні програм можуть використовуватися такі символи:

- літери латинського алфавіту A..Z, a..z;
- цифри 0..9;
- знаки арифметичних операцій, спеціальні символи: + – * / \^ = < > ( ) . , : ; ‘ « @ $ # & _;
- комбінації символів: <=, >=, <>;
- ключові (зарезервовані) слова, що мають однаковий зміст у будь-якій програмі на **Visual Basic**, наприклад: *Dim, As, If, While* тощо.

Синтаксис мови – сукупність правил побудови команд мови програмування.

Семантика мови – сукупність правил виконання комп'ютером команд, записаних мовою програмування.

З синтаксисом та семантикою команд **Visual Basic** ви будете ознайомлюватися по мірі вивчення мови програмування.

Опис величин

Величина – це об'єкт, який має стале або змінне значення.

Основними характеристиками величини є тип, вид і значення. Деяким величинам надають імена.

Тип величини – це множина допустимих значень величини. Тип визначає обсяг пам'яті, необхідний для збереження величини, та операції, які можна над нею виконувати.

Вид величини визначає спосіб використання величини в алгоритмі. Величина може бути константою або змінною. ***Змінні*** – це іменовані величини, значення яких може змінюватися в ході виконання програми. ***Константи*** – це просто числа чи набори символів. Іноді константам теж надають імена.

Ім'я (ідентифікатор) величини – це назва змінної або константи, що обирається програмістом. Ідентифікатори мають також інші елементи програми: процедури, функції, типи даних.

Обмеження на запис ідентифікаторів:

- допустимі лише латинські літери, цифри та знак підкреслення (зокрема, не допускаються крапки та пропуски);
- імена починаються тільки з літери або знаку підкреслення;
- найбільша довжина імені – 255 символів,
- недопустимі збіги імен з ключовими словами.

У числових константах між цілою і дробовою частинами десяткового дробу ставлять крапку.

Приклад 1.16. Програму набагато легше зрозуміти, якщо всюди, де потрібне число $\pi \approx 3{,}14159$, написано *Pi*, а не цифри 3.14159. Якщо на початку процедури написати: *const Pi = 3.14159*, то при виконанні команди *Print 2 * Pi* комп'ютер замінить ім'я *Pi* вказаним числом і виведе на формі *6.28318*.

Значення змінної може багаторазово змінюватися в процесі виконання програми. Для цього застосовують оператор присвоєння.

Команди	Значення змінних	
	a	b
a=5	5	не визначене
b=8	**5**	**8**
a=a+b	13	8
b=a−b	13	5
a=a−b	**8**	**5**

Приклад 1.17. Розглянемо наведену в таблиці послідовність команд. Як бачимо, використовуючи операції додавання й віднімання та оператор присвоєння вдалося поміняти місцями значення двох змінних.

Робота зі змінними

Змінні можуть зберігати практично будь-які дані: числа, рядки тексту, візуальні об'єкти тощо.

У табл. 1.5 наведено опис найчастіше використовуваних в **Visual Basic** типів даних (крім візуальних об'єктів).

Таблиця 1.5

Типи даних			
Тип даних	**Інформація, що зберігається**	**Обсяг займаної пам'яті (байт)**	**Діапазон значень**
Integer	Цілі числа	2	від –32768 до 32767
Byte	Цілі числа	1	від 0 до 255
Long	Цілі числа	4	від − 2 147 483 648 до 2 147 483 647
Single	Дійсні числа	4	від $-3{,}4\cdot10^{38}$ до $+3{,}4\cdot10^{38}$
Double	Дійсні числа	8	від $-1{,}8\cdot10^{308}$ до $+1{,}8\cdot10^{308}$
String	Текстова інформація	1 на кожен символ	близько 4 млрд. символів; константи записують в лапках
Boolean	Логічні (булеві) значення	2	True (так), False (ні)
Date	Інформація про дату і час	8	від 1.01.100 р. до 31.12.9999 р
Variant	Значення кожного з вищеназваних типів	16+1 на кожен символ	Не визначений

Типи властивостей

Кожна з властивостей об'єктів також належить до певного типу. При налаштуванні об'єктів у вікні властивостей (у режимі

розробки), **Visual Basic** автоматично пропонує відповідний тип даних.

Щоб запрограмувати зміну властивості у процедурі обробки подій з використанням оператора присвоєння, необхідно знати тип властивості, тому що значення властивості можна змінити тільки на значення того ж типу. При цьому в лівій частині оператора присвоєння використовується «запис через крапку»:

Ім'яЕлементу.Ім'яВластивості = Значення

де *Ім'яЕлементу* – це ім'я елементу керування; *Ім'яВластивості* – ім'я властивості, яку слід змінити; ***Значення*** – нове значення, яке присвоюється цій властивості.

Таблиця 1.6

Типи властивостей			
Тип	**Властивості даного типу**	**Приклад використання**	**Пояснення**
Integer	*Top, Left, Height, Width*	*Form1.Width = 4000*	Ширина (*Width*) форми *Form1* стане рівною 4000
Long	*BackColor, ForeColor*	*Form1.BackColor = vbBlue*	Колір фону (*BackColor*) форми, набуде значення константи, що має ім'я *vbBlue* (див. табл. 1.15)
Boolean	*Visible*	*Form1.Visible = False*	Форма стане невидимою
String	*Caption*	*Form1.Caption = "Про програму"*	Заголовок форми з ім'ям *Form1* набуде значення «Про програму»

Оголошення змінних

Після вибору імен і типів змінних треба помістити інформацію по це в проект, тобто потрібно оголосити змінні. Оператор, за допомогою якого оголошується змінна з ім'ям *Ім'яЗмінної* типу *ТипЗмінної*, записується так:

Dim Ім'яЗмінної As ТипЗмінної

де *Dim* – це ключове слово, скорочення від *Dimension* (величина, розмірність), *As* − з англійської «як, у якості».

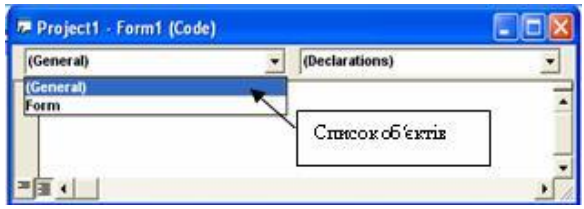
При запуску проекту на виконання компілятор створить всі оголошені змінні, тобто виділить пам'ять для збереження даних в обсязі, який відповідає вказаному типу.

Приклад 1.18. За таким описом:

Dim A As Single, B As Integer

буде виділено 4 байти для змінної *A* та 2 байти для змінної *B*.

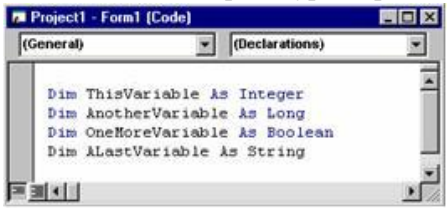
Після виконання оператора *Dim* комірки пам'яті комп'ютера, що виділені під змінні *A* і *B*, є порожніми (*Empty*) − змінні ще не набули значень.



Де ж розміщуються оголошення змінних? Створіть новий проект у **Visual Basic**, перейдіть у вікно коду та відкрийте список об'єктів. Там ви побачите два записи: *(General)* і *Form*. Виберіть *(General) – Загальні*. Вікно коду буде виглядати так:

Так ви відкриєте частину коду, що називається ***розділом загальних оголошень*** (*general declarations*). Якраз тут і розміщують оголошення змінних. Змінні, оголошені в розділі загальних оголошень, є ***глобальними***. Вони можуть використовуватися в будь-якій процедурі обробки подій і зберігають свої значення доти, поки виконується код форми.

Якщо змінна потрібна тільки в коді певної процедури обробки подій, оператор *Dim* вміщують після заголовку даної процедури. Змінні, оголошені в тілі процедури, є ***локальними***. Вони втрачають свої значення при виході з даної процедури.



Математичні оператори

Visual Basic підтримує ряд математичних операцій, які використовують у виразах.

Таблиця 1.7

Математичні операції і відповідні їм символи операторів Visual Basic			
Операція	**Символ оператора**	**Приклад**	**Результат**
Додавання	+	*Res = 15+3*	Res = 18
Віднімання	-	*A = Res – 10*	A = 8
Множення	*	*A = A*2*	A = 16
Ділення	/	*Res = 2.3 / 2*	Res = 1.15
Цілочисельне ділення	\	*Res = 9 \ 2*	Res = 4
Ділення по модулю (обчислення остачі)	*Mod*	*Res = 11 mod 3*	Res = 2

Піднесення до степеня	^	*A = 2^3*	A = 8

Порядок виконання (пріоритет) математичних операцій:

1) піднесення до степеня (^);
2) множення (*) і ділення (/), цілочисельне ділення (\), обчислення остачі від цілочисельного ділення (*Mod*);
3) додавання (+) і віднімання (–).

Якщо операції мають однаковий пріоритет, то вони виконуються зліва направо по черзі.

Приклад 1.22. В результаті виконання оператора присвоєння

*A = 24/2*3*

змінна *A* одержить значення 36.

При необхідності порядок дій регулюється дужками. В результаті виконання оператора присвоєння

*A = 24/(2*3)*

змінна *A* одержить значення 4.

Крім констант та змінних у виразах можуть застосовуватись математичні функції. Наприклад, функцію добування квадратного кореня з аргументу *x* позначають *Sqr(x)*. Детальніше ця та інші функції будуть описані далі.

Логічні оператори

Крім математичних у мові Visual Basic можна обчислювати значення логічних виразів, які мають тип *Boolean*, тобто набувають значення *True* (Істина) або *False* (Хибність). Значення таких виразів можна присвоювати змінним та властивостям логічного типу.

Таблиця 1.8

Логічний оператор	Операція	Логічний вираз	Значення
=	Дорівнює	8=9	False
>	Більше	8>9	False
<	Менше	8<9	True
>=	Більше або дорівнює	5>=5	True
<=	Менше або дорівнює	5<=2	False
<>	Не дорівнює	2<>5	True

Приклад 1.23. Робота з логічними значеннями

```
Dim x As Integer, y As Integer, A As Boolean
    x = 5 : y = 2
    A = x > y   ' змінна A стане рівна True
    A = x < y   ' змінна A стане рівна False
```

Як видно з прикладу, записуючи два оператори в одному рядку, їх відокремлюють двокрапкою

Для побудови складніших логічних виразів використовують стандартні логічні операції, які повертають *True* чи *False* в залежності від значень аргументів (табл. 1.9).

Порядок виконання (пріоритет) логічних операцій:

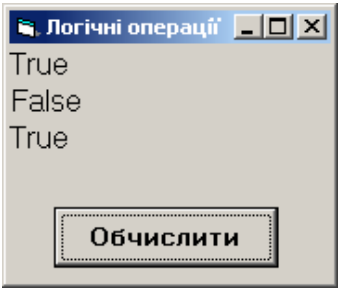
1 – *Not*; 2 – *And*; 3 – *Or*.

Таблиця 1.9

Таблиця істинності для логічних операцій						
A	B	A **AND** B	A **OR** B	A **XOR** B	A	**NOT** A
False	False	False	False	False	False	True
False	True	False	True	True		
True	False	False	True	True	True	False
True	True	True	True	False		
Коментар		*A **AND** B=true, якщо й A, і B істинні*	*A **OR** B=false, якщо й A, і B хибні*	*A **XOR** B=false, якщо A= B*	*Якщо A=True, то NOT A=false*	

Приклад 1.24. Використання логічних операцій

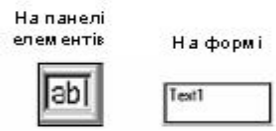
```
Dim x As Integer, y As Integer, z As Integer
Dim A As Boolean
Private Sub Command1_Click()
x = 1:   y = 2:   z = 3
      A = x < y And y < z
      Print A
      A = x > y Or y > z
      Print A
      A = x > y Xor y < z
      Print A
End Sub
```



*Умову, записану з використанням логічних операцій називають **складеною умовою**.*

Елемент керування TextBox

TextBox (текстове поле) являє собою вікно, призначене для введення і виведення тексту під час виконання програми (*run mode*).



Події для елемента керування TextBox. Основна властивість елемента керування *TextBox* – це властивість *Text*. Її значення може змінитись при уведенні в поле тексту з клавіатури або внаслідок присвоєння. З елементом *TextBox* пов'язані дві події, за допомогою яких можна керувати значенням властивості *Text*:

Change (Змінити) – подія відбувається, коли змінюється властивість *Text*;

LostFocus (Втрата фокуса) – подія відбувається, коли користувач закінчує роботу з текстовим полем і викликає подію на іншому елементі керування.

В обробниках цих подій організовують перевірку коректності уведених даних, змінюють стан інших елементів керування тощо.

Властивості текстового поля. Основна властивість − ***Text*** − зберігає рядок тексту, який відображає текстове поле. Щоб текстове поле відразу після запуску проекту було порожнім, у вікні *Properties* слід видалити значення цієї властивості.

Властивість ***Font***, як і в елементу керування *Label*, визначає шрифт, яким відображається текст. Від значення властивості ***Alignment*** залежить горизонтальне вирівнювання тексту: 0 − за лівим краєм, 1 − за правим краєм, 2 − по центру.

Спільна з елементом керування *Label* властивість ***BorderStyle*** визначає тип межі. В елемента керування *Label* початкове значення цієї властивості − *None* (відсутня), а в елемента керування *TextBox* встановлюється значення *Fixed Single* (фіксована одинарна). Кольори залежать від значень властивостей ***BackColor*** (колір фону) і ***ForeColor*** (колір тексту).

Якщо властивості ***MultiLine*** (багаторядковість) надати значення *True*, то текст буде виводитись в декілька рядків. При цьому можуть знадобитись смуги прокручування, наявність та кількість яких залежить від значення властивості ***ScrollBars*** (0 – смуг немає, 1 − горизонтальна, 2 – вертикальна, 3 – обидві).

Як бачимо, елемент керування *TextBox* легко пристосувати для виконання різних задач.

Перетворення типів у Visual Basic

Досить часто у програмах використовують таку логіку роботи. Для **введення** користувачем різних символів, у тому числі і цифр, використовують текстове поле (*TextBox*). Потім комп'ютер виконує математичні **обчислення**, а результат **виводить** за допомогою елемента керування *TextBox* або *Label*.

В цій ситуації виникає проблема: математичні операції можна виконувати тільки з числами (наприклад, значеннями типу *integer*), а властивість *Text* елементу керування *TextBox* має рядковий тип (*string*). Потрібно мати спосіб перетворення рядків у числа і навпаки, чисел у рядки.

Для розв'язання цієї проблеми існують вбудовані функції мови Basic: функція ***Val*** і функція ***Str*** (табл. 1.8).

Функція *– це процедура, яка в результаті виклику повертає деяке значення.*

Завдяки цьому виклики функцій можна використовувати у виразах.

Таблиця 1.10

Функція	Призначення	Формат функції	Приклад використання
Val	повертає числове значення, записане в текстовому рядку	*Val(YourString)*	*YourNumber = Val("23")* Змінна YourNumber отримує числове значення 23.
Str	повертає рядок, що зображає вказане число.	*Str(YourNumber)*	*YourString = Str(23)* Змінна YourString отримує рядкове значення "23"

Приклад 1.25. Додавання чисел *A* і *B*, значення яких вводиться в текстові поля *Text1* і *Text2*. Значення суми *C* виводиться в текстове поле *Text3*.

A = Val (Text1.Text)
B = Val (Text2.Text)
C = A + B
Text3.Text = Str (C)

Питання для самоконтролю

1. *Назвіть основні характеристики величини.*
2. *Які імена недопустимі в якості ідентифікаторів та чому: summa; W1; Priklad 1; (SUM); A-4; Priklad_1; Dim; 9A?*
3. *Що таке тип величини? Назвіть основні типи даних у мові Visual Basic.*
4. *Що таке вид величини? Назвіть і охарактеризуйте види величин.*
5. *Як відбувається оголошення змінних у мові Visual Basic? Які способи існують для оголошення змінних?*
6. *Запишіть оператори оголошення змінних A цілого типу, B логічного типу, C рядкового типу.*
7. *Вкажіть типи констант: а) 3; б) 300; в) 5.4; г) True; д)"парта"; е)"324".*
8. *Який синтаксис має оператор присвоєння? Опишіть схему його виконання.*
9. *Чому дорівнює значення X після виконання послідовності присвоювань:*
   а) y = 2; x = y;
   б) x = 8; x = x+2;
   в) x = 5; x = -x;
   г) x=10; x=x +3?
10. *Запишіть оператори присвоєння, що реалізують такі дії:*
    а) змінній S присвоїти значення суми змінних A і B;

б) обчислити значення добутку змінних A і B і результат присвоїти змінній C;
в) подвоїти значення змінної A;
г) змінити знак змінної t.

11. *Поясніть дію операторів:*
а) Form1.Top = Form1.Top + 300;
б) Form1.Top = 300.
12. *Для чого використовується елемент керування TextBox? Назвіть основні властивості елемента керування TextBox.*
13. *Як виконати перетворення рядкового значення в числове і навпаки? В яких випадках це необхідно?*
14. *Запишіть оператор присвоєння змінній A дійсного типу значення, яке вводиться в текстове поле Text1.*
15. *Опишіть призначення функцій Val та Str.*

1.5. Організація введення і виведення даних

Описану вище логіку роботи комп'ютерної програми можна узагальнити. Більшість програм передбачають введення (***I**nput*) даних користувачем, після чого над ними будуть виконані деякі операції (***P**rocessing*), а результат операцій виведений (***O**utput*) у тому чи іншому вигляді.

Принцип IPO обробки інформації:
введення ⇨ опрацювання ⇨ виведення.

Розглянемо детальніше засоби введення та виведення даних. Нехай змінну *A* оголошено в такий спосіб: *Dim A As Single*. Присвоїти змінній *A* значення можна:

- за допомогою оператора присвоєння;
  A = 3.25
- шляхом введення значення у текстове поле і присвоєння змінній значення властивості *Text*;
  A = Val (Text1.Text)
- за допомогою функції InputBox.
  A = InputBox ("A=?")

Функція InputBox

Для введення даних зручно використовувати функцію *InputBox*. Функція *InputBox* відображає вікно діалогу і очікує, поки користувач введе дані в текстове поле і натисне кнопку. Після цього функція повертає введений текст у вигляді рядка (*String*). Стандартними елементами вікна є кнопки *OK* (підтвердження дії) і *Cancel* (відміна дії). Формат використання функції *InputBox* такий:

Ім'яЗмінної = InputBox* (*підказка*,** [заголовок***]), де:

- *Ім'яЗмінної* – змінна, яка отримує значення, повернуте функцією; перетворення типу повернутого значення до типу змінної відбувається автоматично;
- *підказка* – повідомлення, яке буде виведене у діалоговому вікні функції;
- *заголовок* – заголовок вікна (необов'язковий параметр).

Приклад 1.26. За допомогою функції *InputBox* можна організувати такий діалог з користувачем:

```
Dim D As String
D = InputBox("Як тебе звати?")
Text1.Text ="Привіт, " &  D
```

Тут символом & позначено операцію об'єднання (конкатенації) рядків (детальніше − далі).

Для виведення в попередніх пунктах ми застосовували елементи керування *TextBox*, *Label*. Але не завжди зручно створювати окремий елемент для виведення кожного повідомлення або кожного результату опрацювання даних. **Visual Basic** дозволяє виводити повідомлення в будь-який момент виконання програми за допомогою діалогового вікна ***MsgBox*** та виводити результати роботи програми безпосередньо на форму за допомогою методу ***Print***.

Діалогове вікно MsgBox

Функція *MsgBox* дозволяє вивести на екран діалогове вікно, придатне для відображення будь-якого повідомлення.

Тут і далі у квадратних дужках вказані назви не обов'язкових елементів.

Функція має такий формат:

***MsgBox підказка*, [*кнопки*], [*заголовок*]**, де:

- *підказка* – рядкова константа або змінна для відображення в діалоговому вікні;

- *кнопки* – числовий вираз, який визначає кількість і типи кнопок або малюнок у вікні. Якщо параметр не вказано, то використовується значення 0 (кнопка *ОК*);
- *заголовок* – рядкове значення, яке задає заголовок вікна.

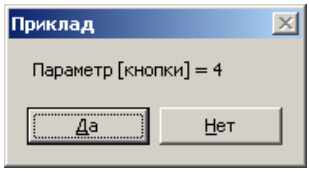
Якщо один з параметрів (крім останнього) відсутній, коми, що відокремлюють його від інших, слід зберігати.

Таблиця 1.11.

Основні константи діалогу *MsgBox*			
Значення	**Опис**	**Значення**	**Опис**
0	Кнопка *ОК*	16	Знак помилки
1	Кнопки *ОК* і *Cancel*	32	Знак питання
3	Кнопки *Yes, No* і *Cancel*	48	Знак оклику
4	Кнопки *Yes* і *No*	64	Знак інформації

Приклад 1.27. При використанні вказаних значень параметрів діалогове вікно має наведений на малюнку вигляд:

MsgBox "Параметр [кнопки] = 4", 4, "Приклад"



Застосування методу Print

Для виведення на формі або на малюнку тексту застосовується метод ***Print***. Синтаксис методу:

[*об'єкт*.]*Print* [*список виведення*],

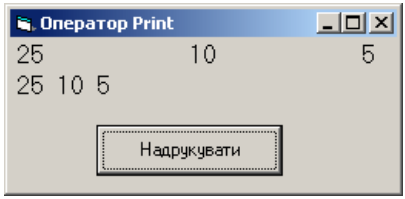
де *об'єкт* – це форма або елемент керування типу *PictureBox* (детальніше про нього − далі), а *список виведення* – перелік текстових рядків або виразів.

Якщо друкування здійснюється на поточній формі, як це було в попередніх прикладах, ім'я об'єкта не вказують.

Якщо в списку виведення декілька значень, їх розділяють комою або крапкою з комою. Значення, відокремлене від попереднього комою, буде надруковане в наступній колонці (ширина колонки 14 символів). Якщо ж числові значення відокремлюються крапкою з комою, то перед і після них додаються пропуски. Рядкові значення в такому разі друкуються без пропусків.

Приклад 1.28.

A = 25 : B = 10 : C = 5
Print A, B, C
Print A; B; C;



Розділювач, поставлений в кінці оператора, впливає на наступні оператори *Print*. Якщо ж розділювача немає, то наступний оператор *Print* виводить дані з нового рядка.

Відокремлювати комою чи крапкою з комою можна і порожні значення.

Як ви вже знаєте, вигляд тексту на формі залежить від установок властивості *Font*. Формат доступу до параметрів шрифту з програмного коду має такий вигляд:

Font.параметр = значення

Приклад 1.29. Наведені оператори друкують рядок «ABC» шрифтом *Arial*, розміром 20 пунктів, накреслення напівжирне:

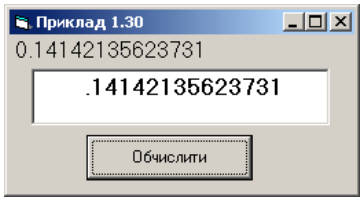
```
Font.Size = 20
Font.Name = "Arial"
Font.Bold = True
Print "ABC"
```

Виведення значень змінних типу Single

При виведенні значень змінних типу *Single* можна одержати значення, яке незручне для читання.

Приклад 1.30. Випробуйте процедуру, в якій обчислюється і виводиться значення $\sqrt{2}/10$:

```
Private Sub Command1_Click()
    a = 2
    Print Sqr(a) / 10
    Text1.Text = Str(Sqr(a) / 10)
End Sub
```



Як бачимо, оператор *Print* виводить забагато десяткових цифр (припустимо, що за умовою задачі така точність обчислень нас не цікавить), а в текстовому полі ще й не показано цифру нуль. Для управління виведенням числових значень існує функція ***Format(),*** яка має такий синтаксис:

Format (***вираз***, ***шаблон***),

де *вираз* – числовий вираз; *шаблон* – рядок-шаблон, який визначає вигляд рядка-результату.

Параметр *шаблон* беруть в лапки, оскільки він являє собою рядок символів.

Для створення шаблона використовуються символи:

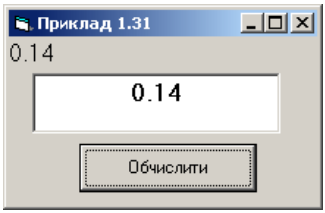
- # – відображення значущих цифр;
- 0 – відображення необхідної кількості цифр. Замість відсутніх значущих цифр будуть виведені нулі.

Приклад 1.31. Якщо у процедурі з прикладу 1.30 додати виведення значення виразу за форматом, отримаємо наведений на малюнку результат:

Print Format(Sqr(a) / 10, "#0.00")

Text1.Text = Format(Sqr(a) / 10, "#0.00")

Зверніть увагу: додаткове перетворення числового значення в рядкове при виведенні в текстове поле не потрібне.



Приклад 1.32. Щоб значення змінної *Res* було виведене з трьома цифрами після коми, використовують виклик функції *Format(Res, "#0.000")*.

Якщо *Res=235.6*, то буде отримано рядок 235.600.

Якщо *Res=0.6789564*, то буде отримано рядок 0.679.

Коментарі

Коментарі – це пояснення до рядків вашого коду, які допомагають зрозуміти, як працює програма. Компілятор їх ігнорує. З коментарями простіше згодом вносити зміни в програму.

Відсутність коментарів – ознака недбалості або невисокої кваліфікації програміста!

Для розміщення коментаря в рядку застосовується апостроф ('): всі символи від апострофа і до кінця рядка вважаються коментарем. Коментар може займати цілий рядок, наприклад:

' Змінити колір форми на червоний

Form1.BackColor = vbRed

або бути поміщеним в одному рядку з оператором:

Form1.BackColor = vbRed ' Змінити колір форми на червоний

Питання для самоконтролю

1. *Як задати значення змінної з використанням функції InputBox?*
2. *Поясніть синтаксис функції MsgBox. Які значення повертає функція MsgBox?*
3. *Наведіть приклади ситуацій, у яких доцільно використовувати діалогове вікно MsgBox.*
4. *Поясніть призначення кожного параметру в операторі*

   MsgBox Str(V) & " грн", , "Загальна сума"

   Чому в списку параметрів стоїть дві коми поспіль?

5. *Поясніть правила використання методу Print.*
6. *Які можливості існують для позиціонування виведення значень на екран при застосуванні методу Print?*
7. *Як змінити параметри шрифту при виведенні тексту?*
8. *Яка функція використовується для керування виведенням числових значень?*
9. *В ході виконання програми змінна A отримала значення 12.567. Яке значення буде виведене в текстове поле Text1 після виконання такого оператора:*
   а) Text1.Text = Format(A, "#0.00");
   б) Text1.Text = Format(A, "#0.0000")?
10. *Для чого у програмний код вміщують коментарі?*
11. *Як додати коментар в рядка програмного коду?*
12. *На початку рядка програмного коду стоїть апостроф. Що це означає?*

1.6. Стандартні функції мови Basic. Таймер

Стандартні математичні функції

У **Visual Basic** є низка функцій для розв'язування математичних задач. Їх можна використовувати безпосередньо при обчисленні значень будь-яких виразів.

У таблиці 1.12 наведено синтаксис і короткий опис математичних функцій, які використовуються найчастіше.

Таблиця 1.12

Запис на Visual Basic	Призначення	Тип результату
Abs(X)	$\|X\|$	збігається з типом аргументу *X*
Atn(X)	*arctg X*	*Double*
Cint(X)	зводить значення *X* до типу *Integer*	*Integer*
Cos(X)	*cos X* (аргумент в радіанах)	*Double*
Exp(X)	e^X	*Double*
Fix(X)	відкидання дробової частини числа *X*	*Integer*
Int(X)	ціла частина числа *X* – найбільше ціле число, що не перевищує *X*	*Integer*
Round(X,a)	округлення числа *X* з точністю до *a* цифр після коми	*Single*
Rnd(X)	генератор випадкових чисел	*Single*
Log(X)	*ln X*	*Double*
Sgn(X)	знакова функція $= \begin{cases} 1 \text{ при } X > 0 \\ 0 \text{ при } X = 0 \\ -1 \text{ при } X < 0 \end{cases}$	*Variant*
Sin(X)	*sin X* (аргумент в радіанах)	*Double*

Sqr(X)	$\sqrt{X}$	*Double*
Tan(X)	*tg X* (аргумент в радіанах)	*Double*

Аргумент в усіх тригонометричних функціях задається в радіанах, а не в градусах. При необхідності переведення значення градусів в радіани слід використовувати формулу:

радіани = градуси $\times \pi$ /180.

Функція *Rnd(X)* – генератор випадкових чисел – повертає довільне число з інтервалу (0;1), при цьому аргумент *X* можна не вказувати. Для отримання при кожному запуску програми різних послідовностей випадкових чисел, необхідно перед першим викликом функції *Rnd* викликати процедуру *Randomize* з параметром *Timer* (запуск генератора випадкових чисел).

Приклади використання випадкових чисел: ігрові програми, в яких потрібне щоразу інше розміщення певних елементів (карт, фішок) або непередбачуваний хід комп'ютера; імітація різних природних процесів, що мають випадковий характер; підпрограми генерування паролів у системах реєстрації користувачів тощо.

Приклад 1.33. Для одержання випадкових цілих чисел з заданого діапазону використовують такий прийом: множать значення *Rnd* на кількість цілих чисел в діапазоні, беруть від отриманого добутку цілу частину і додають перше з чисел діапазону. Наприклад, щоб одержати випадкове ціле число *X*, що належить проміжку від 0 до 99, користуються операторами:

```
Randomize Timer
X = Int (100 * Rnd)
```

А число з проміжку від 15 до 75 отримують операторами:

```
Randomize Timer
X = 15 + Int (61 * Rnd)
```

Приклад 1.34. Дійсне значення перетворюють в ціле за допомогою стандартних функцій *Fix(X)* та *Int(X)*.

Нехай $X = 4.65$. Тоді *Fix (X)* =4; *Int (X)* = 4.

Нехай $X = -4.1$. Тоді *Fix (X)* = −4; *Int (X)* = −5.

Правила запису арифметичних виразів:

1. Вираз повинен бути записаний у вигляді рядка символів, так як записують, наприклад, формули в електронних таблицях.
2. Не можна опускати знак операції множення.
3. Порядок виконання операцій одного пріоритету регулюється дужками.

4. Аргументи функцій записуються в круглих дужках.

Приклад 1.35.

Арифметичний вираз	Запис виразу у програмі
$\frac{2x-5}{3+x}$	*(2*x-5)/(3+x)*
$b+\sqrt{a}$	*b+sqr(a)*
$\cos^2 x$	*Cos(x)^2*
$\cos x^2$	*Cos(x^2)*

Функції для роботи з датою та часом

Приклад 1.36. Присвоєння значень змінним типу *Date*:

BirthDay = #29.10.2008#

EndOfTime = #8:30#

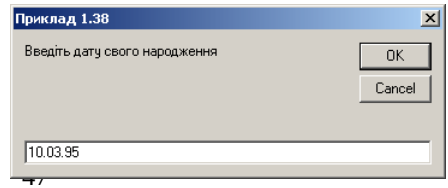
У таблиці 1.13 наведено синтаксис і короткий опис деяких функцій для роботи з датою та часом.

Таблиця 1.13

Запис на Visual Basic	Призначення	Тип результату
Date	Значення поточної дати	*Date*
Time	Значення поточного часу	*Date*
Now	Значення поточної дати і поточного часу	*Date*
Hour (time)	Кількість годин в часі *time*	*Integer*
Minute (time)	Кількість хвилин в часі *time*	*Integer*
Second (time)	Кількість секунд в часі *time*	*Integer*
WeekDay (date)	Номер дня тижня, який указано в аргументі *date*	*Integer*
WeekDayName (n)	Назва дня тижня, який має номер *n*	*String*
DateDiff (S, D1, D2), де *S* – параметр, який може приймати значення: *"d"* – доби; *"h"* – години; *"n"* – хвилини; *"s"* – секунди.	Кількість одиниць часу *S*, що пройшла з часу *D1* до часу *D2*	*Integer*

Приклад 1.37. Відображення поточної дати у текстовому полі:

Text1.Text = Date



Приклад 1.38. В полі *Text1* буде виведено кількість днів, яку прожив користувач.

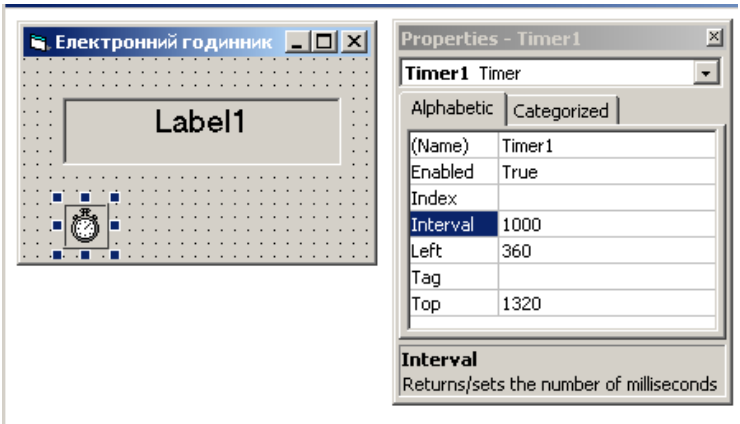
D = InputBox("Введіть дату свого народження")
Text1.Text =DateDiff ("d", D, Now)

При введенні значень змінних типу Date символи «#» писати не треба. Число, місяць та рік відокремлюють крапками.

Елемент керування Timer

Елемент керування *Timer* дозволяє виконувати через певні проміжки часу деякі операції, які не залежать від дій користувача під час роботи програми. Для вмикання таймеру потрібно його властивості *Enabled* надати значення *True*. Час між подіями, які викликає об'єкт *Timer*, визначається властивістю *Interval*, якій можна надавати значення в діапазоні [0; 65535] (мілісекунд), тобто подія *Timer* не може виконуватись рідше, ніж 1 раз у 66 секунд.

Приклад 1.39. Щоб створити на формі «електронний годинник», який відображатиме поточний час, оновлюючись кожну секунду, потрібно помістити на форму елемент Таймер і властивості *Interval* надати значення *1000* (1 секунда). Для



відображення показників часу на формі розмістити елемент *Label.* Далі слід запрограмувати обробку події «Спрацювання таймера». Для цього двічі клацають об'єкт *Timer1* і у процедурі обробки події, що має назву *Timer1_Timer*, вписують рядок:

Label1.Caption = Str(Time)

Призначення даної процедури – змінити значення властивості *Caption* об'єкта *Label1*.

Об'єкт типу Timer не відображається при роботі програми.

Функції для роботи з даними рядкового типу

Велика кількість програм призначені для роботи з текстами: текстові редактори, перекладачі, поштові клієнти, браузери тощо. Для опрацювання текстової інформації у **Visual Basic** існує тип даних *String*. Як ви вже знаєте, значеннями типу *String* є рядки символів, а записуючи рядкові константи, їх беруть у лапки, наприклад: "Добрий ранок", "5a+c" (див. табл. 1.5, 1.6). Найпростіші дії з рядками вже зустрічались вище під час організації виведення на екран.

Рядок є послідовністю символів. Нумерація символів у рядку починається з 1.

Символи в тексті програми, взяті в лапки, сприймаються **Visual Basic** як рядкові значення. Символи, які не взяті в лапки, розглядаються як змінні, ключові слова тощо.

Приклад 1.40. Присвоєння значень рядковим змінним:

Dim A As String, B As String

A = "31 грудня"

B = "75"

Якщо потрібно обмежити довжину рядка конкретною кількістю символів, використовують опис виду:

Dim Ім'яЗмінної As String * довжина

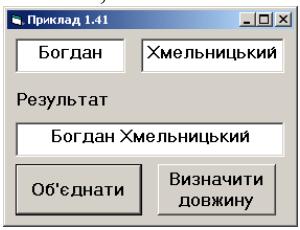
де *довжина* – кількість символів у рядку. Отже, щоб створити рядкову змінну довжиною 10 символів, її оголошують таким чином:

*Dim A As String * 10*

Конкатенація рядків

Конкатенацією двох або більше рядків називають їх об'єднання в один рядок. Цю операцію позначають символом **&**. Операція **&** дозволяє об'єднувати в рядок як змінні, так і константи рядкового типу.

Знак **&** слід відокремлювати пропуском від розміщеного перед ним імені змінної.



Приклад 1.41. Для об'єднання двох рядків, які введені в текстові поля *Text1* і *Text2*, з додаванням пропуску між ними,

в обробник події для кнопки з написом «Об'єднати» (див. мал.) вміщують такі рядки:

A = Text1.Text & " " & Text2.Text

Text3.Text = A

або

Text3.Text = Text1.Text & " " & Text2.Text

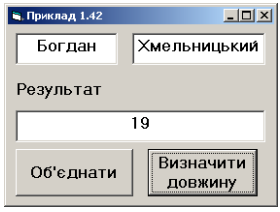
Визначення довжини рядка

Функція ***Len()*** повертає число символів, які містить рядок. Синтаксис функції:

Len (рядок),

де *Len* – ім'я функції; *рядок* – рядкова змінна, константа або вираз, довжину значення якого потрібно визначити.

Приклад 1.42. При натисканні кнопки «Визначити довжину» мають виконатися такі команди:



K = Len(A)

Text3.Text = Str(K)

або

Text3.Text = Str(Len(A))

Одержання підрядка за допомогою функції Mid()

Підрядком називають довільну частину рядка. Функція ***Mid(strS, i, N)*** повертає *N* символів рядка *strS*, починаючи з і-го символу.

Приклад 1.43. Після виконання команд:

A ="Visual Basic"

B =Mid (A, 5, 6)

Змінна B отримає значення *«al Bas»*.

Значення, яке задає початкову позицію, повинне бути не менше 1. Якщо не вказана кількість символів, які слід повернути, функція *Mid()* поверне частину вихідного рядка, починаючи з вказаної початкової позиції і до кінця.

Результат, повернутий функцією Mid(), можна вивести або присвоїти змінній, а початковий рядок залишається без змін.

Приклад 1.44.

Dim A As String, B As String

A = "програмування"

B = Mid (A, 1, 7) & Mid (A, 6, 1)

Питання для самоконтролю

1. *Які математичні функції призначені для перетворення значень дійсного типу в значення цілого типу?*
2. *Поясніть правила запису арифметичних виразів у програмі.*
3. *Як отримати випадкові числа? Наведіть приклади задач, для яких потрібні випадкові числа.*
4. *Запишіть оператор присвоєння, який реалізує таку дію: змінній B типу Single присвоїти значення дробової частини дійсного числа A.*
5. *X=3,567. Чому дорівнює Fix(X), Int(X), Round(X,1)?*
6. *X=–3,567. Чому дорівнює Fix(X), Int(X), Round(X,1)?*
7. *Запишіть на Visual Basic такі формули:*

   а) $\frac{x-5}{2x}$; *б)* x^5; *в)* $\sqrt[3]{1+x}$; *г)* $\cos^2 x$.
8. *Які функції призначені для роботи з датою та часом?*
9. *Які символи використовуються для присвоєння значень змінним типу Date?*
10. *Як виконують об'єднання рядків?*
11. *За допомогою яких засобів можна виконати копіювання підрядка із заданого рядка? Поясніть на прикладі.*
12. *Для чого використовується елемент керування Timer? Опишіть схему його застосування.*

1.7. Графічні елементи керування

Додавання зображень на форму

Для графічного оформлення форми застосовують елементи управління ***Image*** і ***PictureBox***. Елемент управління *Image*  служить для розміщення на формі графічного зображення. Елемент управління *PictureBox*  є свого роду вікном, можливо, з малюнком на фоні, тобто є, перш за все, контейнером для розміщення інших об'єктів.

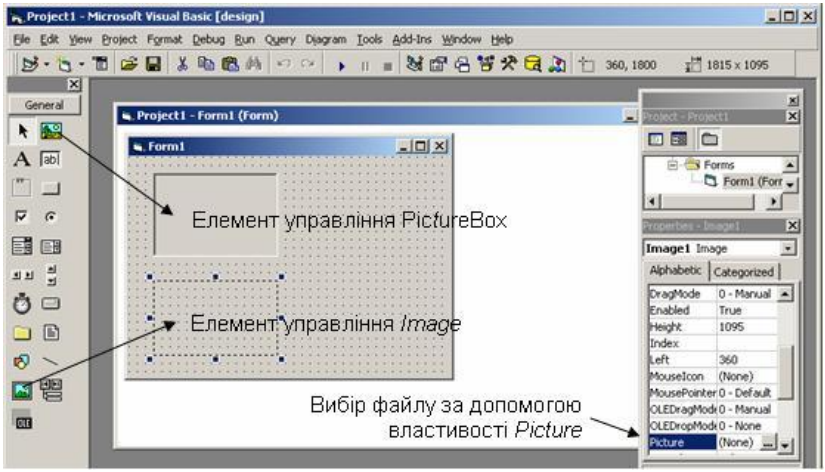
Кожен з елементів додається на форму стандартним способом: необхідно або двічі клацнути потрібний елемент, або, виділивши його піктограму на панелі інструментів, намалювати елемент на формі за допомогою миші.

Відмінність в зовнішньому вигляді елементів після додання на форму викликане тим, що властивість *BorderStyle* для елемента *PictureBox* спочатку має значення 1 – *Fixed Single*, а для елемента *Image* – значення *0 – None*.

Зображення для обох елементів керування визначається властивістю *Picture*, значенням якої є ім'я графічного файлу. Для вибору файлу у вікні *Properties* у рядку *Picture* треба натиснути кнопку  . Після цього з'явиться стандартне вікно для вибору

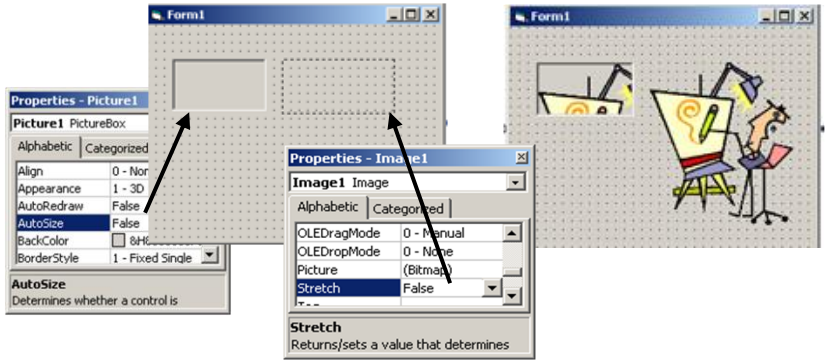
файлу. Підтримуються всі основні формати графічних файлів, такі як *.bmp, .gif, .jpg* та інші.

Елемент управління *PictureBox* має властивість логічного типу *AutoSize*, що дозволяє або забороняє автоматично змінювати розміри елемента управління відповідно до розмірів зображення.



Елемент управління *Image* має властивість логічного типу *Stretch*, яка керує зміною розмірів зображення відповідно до розмірів елемента управління.

Значення *True* властивостям *AutoSize* або *Stretch* необхідно надати перед тим, як завантажувати зображення.



Приклад 1.45. Якщо згадані властивості автомасштабування елементів керування *PictureBox* і *Image* мають значення *False*, а розміри малюнка перевищують розміри елементів, то при

завантаженні зображення отримаємо такий результат: в *PictureBox* відобразиться тільки частина малюнка, а розміри *Image* збільшаться до розмірів зображення (див. мал.).

Завантаження зображення під час виконання програми

Часто виникає потреба змінювати зображення під час виконання програми. Таку можливість забезпечує функція ***LoadPicture()***. Синтаксис функції такий:

VarPicture = LoadPicture (FileName),

де *VarPicture* – змінна або властивість для збереження малюнка; *FileName* – рядкова змінна або константа, яка містить шлях до графічного файлу на диску.

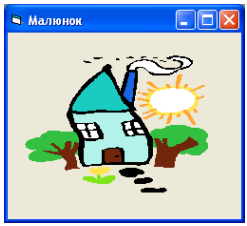
Якщо графічний файл знаходиться в поточному каталозі (у папці проекту), достатньо вказати лише його ім'я.

Проект встановлює зв'язки з файлами в папці проекту в момент відкриття, тому графічні файли потрібно додати до папки проекту до відкриття (або створення нового) проекту.

Приклад 1.46. Наступний оператор ставить у відповідність об'єкту *Image1* малюнок з колекції *Clipart*:

Image1.Picture = LoadPicture ("C:\Program Files\Microsoft Office\Clipart\Pub60cor\ BL00261_.Wmf")

Якщо функція *LoadPicture()* буде викликана без зазначення шляху до файлу, то відбудеться очищення даного елемента від малюнка. Подібного ефекту можна досягти, викликавши для елемента методу *Cls*.



Приклад 1.47. Такі рядки коду реалізують однакові дії – очищують вміст елемента *Image1*:

Image1.Picture = LoadPicture ()

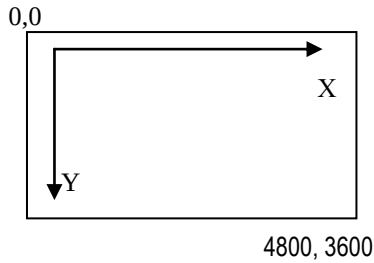
або

Image1.Cls

Система координат

Для визначення положення на формі графічного елемента використовують координати. Стандартна система координат **Visual Basic** передбачає використання вже відомих вам одиниць вимірювання – твіпів. Один ***твіп*** дорівнює 1/20 пункту чи 1/1440 дюйма. Будь-яка точка на формі задається парою чисел *(X, Y)*. Вертикальна координата *Y* зростає з верху до низу, а горизонтальна

X – зліва направо. Поточні розміри форми зазначені у правій частині панелі інструментів *ToolBar*. Початкові розміри форми становлять 4800×3600 твіпів. Один твіп – це дуже маленька відстань: погляньте на форму в режимі [*design*] – вона вся замальована чорними крапками. Відстань від крапки до крапки – 120 твіпів!



Елементи керування Shape і Line

Панель *Toolbox* містить два елементи керування: *Line* (Лінія), що має вигляд відрізка прямої певного кольору, товщини і стилю, та *Shape* (Фігура), що може мати вигляд прямокутника, круга або еліпса. Ці елементи використовуються для створення графічних композицій на формі.

Використання елемента керування Shape

На панелі компонентів виберіть компонент ***Shape*** і намалюйте його на формі перетягуванням. Потім налаштуйте властивості фігури:

1. Виберіть форму фігури зі списку властивості *Shape*:

Допустимі значення властивості *Shape*			
0	Прямокутник	3	Круг
1	Квадрат	4	Округлений прямокутник
2	Овал	5	Округлений квадрат

2. Задайте заповнення фігури. Для властивості ***FillStyle*** (Спосіб заповнення) виберіть зі списку потрібне значення:

Допустимі значення властивості *FillStyle*			
0	Суцільне	4	Діагональ вгору
1	Прозоре	5	Діагональ вниз
2	Горизонтальні лінії	6	Клітинки
3	Вертикальні лінії	7	Діагональні клітинки

Колір зафарбування задається властивістю ***FillColor*** (Колір заповнення). Розкрийте список і виберіть на вкладці *Palette* (Палітра) потрібний колір.

3. Задайте спосіб взаємодії заповнення і тла. Якщо властивість *Backstyle*=0, то простір між лініями заповнення залишиться

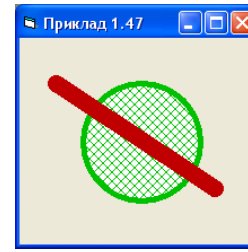
прозорим. При *Backstyle*=1 прозорі ділянки заповняться кольором. вказаним у властивості *BackColor*.

4. Для точного задання розмірів фігури використовуйте властивості *Height* (Висота) і *Width* (Ширина).

5. Для точного розміщення фігури на формі використовуйте властивості *Left* (Відстань від лівого краю форми) і *Top* (Відстань від верхнього краю форми).

Використання елемента керування Line

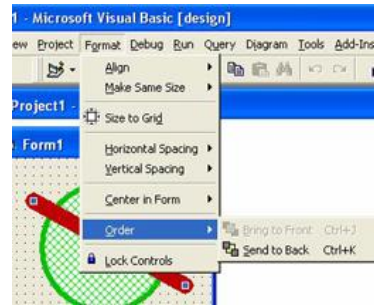
На панелі компонентів виберіть компонент *Line*  . Помістіть вказівник на те місце форми, де повинна починатися лінія, і, натиснувши ліву кнопку, перетягніть вказівник до кінцевої точки лінії. Відпустіть кнопку мишки. На формі з'явиться відрізок прямої. Кінці відрізка можна переміщати за маркери, а увесь відрізок − перетягуючи за будь-яку іншу точку.



Приклад 1.48. Для даних об'єктів *Shape* і *Line* в режимі [*design*] у вікні *Properties* встановлені такі значення властивостей:

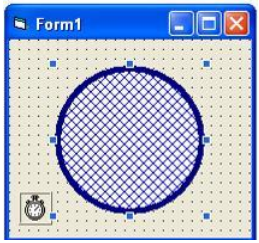
Властивість	Об'єкт *Shape*	Об'єкт *Line*
Shape	3 – Circle	–
BorderWidth	5	15
BorderColor	&H0000C000&	&H000000C0&
X1, Y1	–	480, 600
X2, Y2	–	2600, 2000
BackStyle	1 – Opaque	–
FillStyle	7 – Diagonal Cross	–
FillColor	&H0000FF00&	–

Щоб змінити порядок розташування об'єктів на формі, необхідно виділити об'єкт, який треба перемістити, і вибрати команду *Format* ⇨ *Order* ⇨ *Send To Back* (Перемістити назад) або *Bring To Front* (Перемістити вперед).



Значення властивостей об'єктів *Shape* і *Line* можна змінювати програмним способом.

Приклад 1.49. При спрацьовуванні таймера, випадковим чином змінюються значення властивостей об'єкта *Shape*. Діапазон для генерації випадкового числа обирається в залежності від діапазону допустимих значень тієї чи іншої властивості (наприклад, тип фігури визначається значенням від 0 до 5, отже, інтервал для випадкового числа обмежується числом 6):




```
Private Sub Timer1_Timer()
    Randomize Timer
    Shape1.Shape = Int(6 * Rnd)
    Shape1.FillStyle = Int(8 * Rnd)
    Shape1.Width = Int(2000 * Rnd)
End Sub
```

Задання кольорів

Як ви вже знаєте, при виведенні зображення на екран (у телевізорах, комп'ютерних моніторах тощо) кожен колір – це «суміш» у певній пропорції 3-х базових кольорів – червоного, зеленого і синього.

У програмному коді колір малювання задають за допомогою функції *RGB*, в аргументах якої окремо вказується ступінь насиченості кожної складової (у межах від 0 до 255).

Таблиця 1.14

Основні кольори							
Колір	**R(ed)**	**G(reen)**	**B(lue)**	**Колір**	**R**	**G**	**B**
чорний	0	0	0	жовтий	255	255	0
білий	255	255	255	фіолетовий	255	0	255
червоний	255	0	0	коричневий	205	155	135
зелений	0	255	0	жовтогарячий	255	128	0
синій	0	0	255	сірий	128	128	128

Приклад 1.50. Контур фігури *Shape1* стає коричневим:
Shape1.BorderColor = RGB(205, 155, 135)

Фігура *Shape1* отримує випадковий колір тла:
*Shape1.BackColor = RGB(Int(256*Rnd), Int(256*Rnd), Int(256*Rnd))*

Поряд з використанням функції *RGB* можна, як вже було згадано, застосовувати колірні константи, описані в табл.1.15 (див. приклад в табл.1.6).

Таблиця 1.15

Колірні константи					
Константа	**Значення**	**Опис**	**Константа**	**Значення**	**Опис**
vbBlack	0	чорний	vbMagenta	&HFF00FF	яскраво-червоний
vbBlue	&HFF0000	синій	vbRed	&HFF	червоний
vbCyan	&HFFFF00	блакитний	vbWhite	&HFFFFFF	білий
vbGreen	&HFF00	зелений	vbYellow	&HFFFF	жовтий

Питання для самоконтролю

1. *Які елементи управління використовують для графічного оформлення форми?*
2. *Які дії потрібно виконати, щоб помістити на форму зображення, яке зберігається на диску в графічному файлі?*
3. *Поясніть порядок використання елементів управління Image і PictureBox.*
4. *Назвіть три відмінності між елементами Image і PictureBox.*
5. *Як змінити зображення для елемента Image програмним способом?*
6. *Як змінити розміри зображення програмним способом?*
7. *Запишіть оператор, який надає висоті елемента Image1 значення 200 твіпів.*
8. *Запишіть оператор, який реалізує збільшення відстані від лівого краю до елемента Image1 на 200 твіпів.*
9. *Запишіть оператор, який реалізує додавання зображення до елемента Picture1 з файлу Малюнок1.bmp, який знаходиться на диску C: в папці TEMP.*
10. *Опишіть порядок застосування елементів управління Line та Shape.*
11. *Які властивості притаманні фігурі (Shape)?*
12. *Запишіть оператори, які встановлять суцільне заповнення червоним кольором для фігури Figura1.*

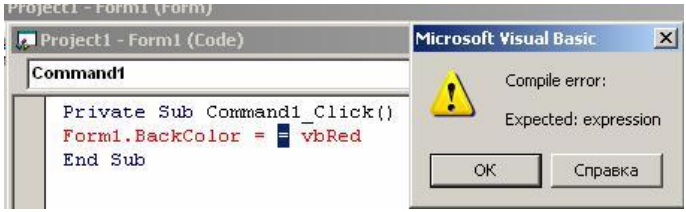
1.8. Налагодження програмного коду

Синтаксичні помилки в програмному коді

Як би ретельно ви не писали текст програми, ви не уникнете помилок в програмному коді. Помилки в написанні або розміщенні ключових слів називаються синтаксичними. Більшість синтаксичних помилок середовище **Visual Basic** дозволяє виправити при наборі програмного коду. Коли ви переводите курсор з рядка, **Visual Basic** вважає, що ви закінчили роботу з рядком, перевіряє його на наявність синтаксичних помилок, і, якщо їх не знаходить, дещо змінює вигляд рядка, приводячи його до стандартного.

Приклад 1.51. Якщо ви робите спробу перевести курсор з рядка, в якому є помилка, підозріле місце буде виділене. Помилку в операторі

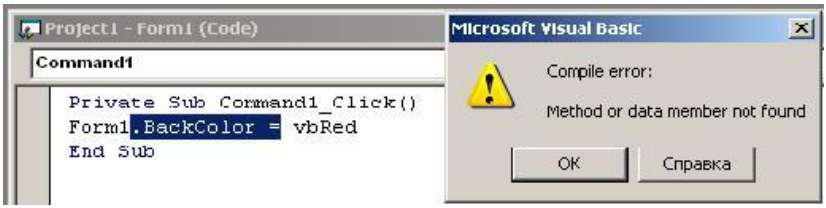
Form1.BackColor = = vbRed



Visual Basic помітить ще в режимі розробки (див. мал.). Натисніть *OK* і виправте помилку.

*Процес усунення помилок називається **налагодженням** програмного коду.*
*Програма, призначена для налагодження програмного коду, називається **налагоджувачем**.*

Приклад 1.52. Налагоджувач **Visual Basic** не завжди правильно виявляє помилки. В даному випадку в імені властивості *BackColor* літера «В» є українською, але система повідомляє, що



для форми даного метода або властивості не існує.

Умовно до синтаксичних можна віднести помилки, які виявляються при компіляції програми (*compile error*). Ці помилки найчастіше виникають, якщо програміст порушує структуру запису складних операторів, або припускає неповний запис програмних блоків операторів.

Приклад 1.53. В даному фрагменті в запису оператора *If Then…Else…End If* відсутній рядок *End If.*

При виправленні помилки, для оперативного одержання довідкової інформації стосовно певного об'єкта, стане в нагоді довідкова система **Visual Basic**.

Інтерактивна довідка

Інтерактивна – тобто діалогова, оперативна, здатна вести діалог і давати підказки в типових ситуаціях.

Щоб звернутись до довідкової системи **Visual Basic** клацніть на слові *Help* у головному меню, потім – на рядку *Contents* (Зміст).

Головною особливістю інтерактивної довідкової системи **Visual Basic** є те, що вона контекстно-залежна.

Приклад 1.54. Виділіть форму і перейдіть у вікно властивостей. Прокручуванням знайдіть властивість *BackColor* і клацніть на ній. Властивість буде виділено. Натисніть клавішу ***F1*** – з'явиться довідкова інформація саме про властивість *BackColor*. Довідкова система розпізнала ваш запит.

Довідка надається за словом, виділеним у вікні властивостей, або за тим, на якому знаходиться курсор у вікні коду. Якщо ж запропонована автоматично інформація не допомогла вирішити проблему, користувач може продовжити роботу з довідковою системою у режимі, подібному до довідкових систем інших програм (**Word**, **PowerPoint** тощо).

Виявлення неоголошених змінних (Option Explicit)

Приклад 1.55. Припустимо, що програміст помилився, набираючи програмний код для надання значень змінним *Wctil* і *Hctil*. В операторі присвоєння він написав *Hcnil* (замість *Hctil*). Система сприймає це як введення нової змінної, значення якої не визначене. Тому в результаті виконання даного коду буде надруковано:

```
Private Sub Command1_Click()
    Wctil = 5:   Hctil = 6
    Pl = Wctil * Hcnil
    Print "Площа = "; Pl
End Sub
```

Площа = 0

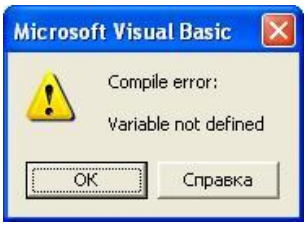
На відміну від інших мов, **Visual Basic** не вимагає оголошувати кожну змінну перед її використанням. Якщо змінна не оголошена, використовується тип даних *Variant* (довільний). Це призводить до зайвої витрати ресурсів пам'яті та непередбачуваних результатів, в зв'язку з використанням значень, заданих за мовчазною згодою.

Щоб уникнути помилкових ситуацій під час роботи зі змінними в програмі, налагодьте **Visual Basic IDE** на необхідність явного оголошення змінних. Для цього:

1) викличте команду меню *Tools / Options*;
2) відкрийте вкладку *Editor* і клацніть на прапорці *Require Variable Declaration* (Вимагати оголошення змінної);
3) натисніть *OK*.

З цього моменту для кожного створюваного проекту у кожен створюваний модуль, у розділ *Declarations,* буде додаватися оператор *Option Explicit*, який свідчить про обов'язковість оголошення всіх змінних в даному модулі (ця опція набере сили з моменту відкриття наступного проекту).

Створіть новий проект і спробуйте ввести з клавіатури оголошення змінних у вікні коду. Зверніть увагу, що у випадку використання в тексті програми неоголошеної змінної при виконанні проекту буде генеруватися помилка. Розглянутий спосіб оголошення змінних називається явним.



Приклад 1.56. Режим явного оголошення змінних допоможе вам запобігти випадкових помилок при наборі. Якщо ви при відсутності оператора *Option Explicit* напишете такий оператор:

*Form1.BackColor = vbRed**d***

то **Visual Basic** не помітить цієї помилки і зафарбує форму в чорний колір, оскільки буде створено змінну *vbRed**d*** типу *Variant* з числовим значенням 0. А в режимі обов'язкового оголошення змінних *vbRed**d*** буде сприйнято як ім'я неоголошеної змінної, і при виконанні проекту з'явиться повідомлення про помилку.

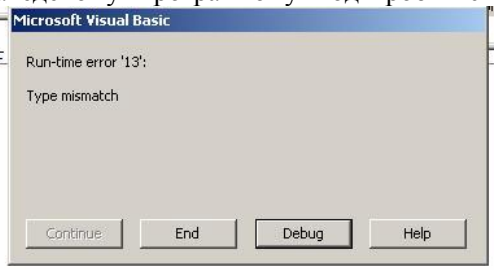
Помилки виконання

Але не всі помилки в програмі є синтаксичними. Деякі з них виявляються вже при виконанні програми, коли робиться спроба опрацювати неприпустимі дані. Такі помилки називають помилками виконання (*run-time error*).

Приклад 1.57. У наведеному програмному коді робиться

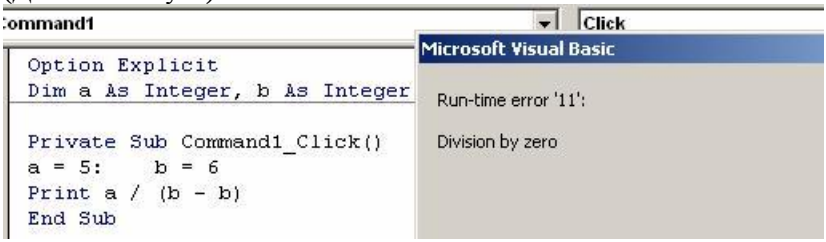
```
Option Explicit
Dim a As Integer, b As Integer

Private Sub Command1_Click()
a = "один":    b = 6
Print "Площа =", a * b
End Sub
```



спроба присвоїти змінній, яку оголошену як цілу, значення типу *String*. Тому виконання програми переривається і виводиться повідомлення про невідповідність типів (*Type mismatch*). Якщо натиснути *Debug*, то рядок, в якому виявлено цю помилку, буде позначений жовтим кольором.

Приклад 1.58. В даному випадку відбувається спроба виконати операцію з неприпустимими даними. Виконання програми переривається, виводиться повідомлення *Division by zero* (Ділення на нуль).



Приклад 1.59. На стадії виконання **Visual Basic** помічає такі помилки, як *Sqr(–25)*, і реагує повідомленням про «Неправильний виклик процедури чи неправильний параметр процедури». У нашому випадку це, звичайно, неприпустиме значення параметра (–25). Після натискання на кнопку *Debug* система переходить у режим переривання і підсвічує рядок з помилкою.

Логічні помилки

Найскладніше виправити помилки, викликані неправильною логікою програми або помилками, які припущені при розробці алгоритму. Такі помилки називаються логічними. Програма може бути скомпільована успішно, але результати її виконання виявляються хибними. Так, якщо ви, бажаючи збільшити число *a* на 1, замість *a*=*a*+1 пишете *a*=*a*+2, то, зрозуміло, ні при компіляції, ні при запуску це виявлено не буде. Найпростіший спосіб виявити логічну помилку – це запустити програму в покроковому режимі, і проаналізувати значення змінних на кожному кроці.

Приклад 1.60. Проект «Виконання програми в покроковому режимі»

1. Створіть новий проект. Помістіть на форму командну кнопку, а в вікні програмного коду наберіть таку програму:

Private Sub Command1_Click()
*a = 2 * 3 + 4*

```
    b = a
    y = a + b + 1
    Print a, b, y, b + y
End Sub
```

2. Запустіть програму. На формі оператором *Print* буде надрукований рядок чисел:

10 10 21 31

3. Виберіть команду *View* ⇨ *Immediate Window.* Вікно *Immediate* зручно використовувати для перевірки значень змінних в процесі виконання програми. Для виведення значень у вікно *Immediate* призначений оператор ***Debug.Print.***

4. Змініть 5-й рядок програмного коду таким чином:

Debug.Print a, b, y, b + y

Виконайте програму. 4 числа з'явилися у вікні *Immediate*. Тобто ми можемо відстежувати проміжні значення змінних без перевантаження форми текстовими полями або результатами роботи операторів *Print.*

5. Комп'ютер виконує програму дуже швидко. Це добре, якщо результати обчислень нас влаштовують. Але, якщо програма видає хибні результати, бажано відстежити, як після виконання кожного оператора змінюються значення змінних в пам'яті – це допомагає знайти помилку в логіці програми. Для цього і призначений покроковий режим виконання програми.

Запустіть проект на виконання не кнопкою *Start*, як ми звикли, а клавішею *F8* на клавіатурі. Це «гаряча клавіша» для команди *Debug* ⇨ *Step Into.* Проект починає виконуватись. Натисніть кнопку *Command1*. Замість того, щоб повністю виконатися і показати результат, програма зупиняється на першому рядку процедури, а саме, на *Private Sub Command1_Click(),* на ознаку чого цей рядок підсвічено жовтим кольором.

Поцікавтесь, чому під час зупинки дорівнюють значення змінних у пам'яті комп'ютера? Для того, щоб дізнатися про це, наводьте курсор миші на позначення змінної в тексті процедури у вікні коду. Як і слід очікувати, з'являється підказка на зразок «*a=Empty*» − змінні ще не набули значень.

*При натисканні на **F8** налагоджувач виконує черговий оператор програми і підсвічує той оператор, який буде виконаний наступним.*

Натисніть **F8**. Жовта смуга переходить на рядок a = 2 * 3 + 4.

F8. Виконується оператор a = 2 * 3 + 4, підсвічується наступний оператор. Перевірте, чому зараз дорівнюють a, b, y.

F8. Виконується оператор b = a, підсвічується наступний рядок. Перевірте, яких значень набули a, b, y.

Зверніть увагу: режим роботи змінився. У заголовку головного вікна **Visual Basic** ви бачите слово [*break*]. Це означає, що система зараз перебуває у режимі переривання, тобто режимі роботи з зупинками. Натискаючи на *F8*, ви наказуєте **Visual Basic** зупинятися після виконання кожного оператора. Тому такий режим зветься *покроковим режимом* виконання програми.

F8. Виконується оператор y = a+b+1, підсвічується наступний рядок.

F8. Виконується оператор Debug.Print a, b, y, b + y, підсвічується рядок End Sub. У вікні *Immediate* з'явилися 4 числа.

F8. Жовта смуга зникає, тому що процедуру виконано. Можна знову натискати на кнопку *Command1*.

6. В будь-який момент покрокового виконання програми ви маєте змогу замість ***F8*** натиснути *Start*, яка в такому випадку має напис *Continue*, і виконання продовжиться в звичайному режимі.

Питання для самоконтролю

1. *Що буде надруковано при виконанні таких фрагментів програм:*
   *а) a=100 : a=10*a+1 : Debug.Print a;*
   б) a=100 : a= -a : Debug.Print a;
   в) a=10 : b=25 : a=b-a : b=a-b : Debug.Print a, b;
   *г) a = (2^2+1) * (20- (2^2)^2) - 11 : b=11 \ (a-4) : Debug.Print a^2 + b – 1.*
2. *Які помилки називаються синтаксичними? Як система сигналізує про синтаксичні помилки?*
3. *Які помилки належать до помилок виконання?*
4. *Які помилки вважаються логічними?*
5. *Як змінити конфігурацію IDE для автоматичного включення режиму явного оголошення змінних?*
6. *Як здійснити друк значень у вікні Immediate?*
7. *Поясніть схему виконання програми в покроковому режимі. В чому полягають помилки в наведених фрагментах програмних кодів? До якого виду належить кожна з помилок?*
8. *Dim a As Integer*
   a = "25 kg" : Print a
   Form1.BackColor = Red
9. *Dim a As Single, x As Single*
   *a = Sqr(Abs(x^2 – 1) / (1+2*x)*
10. *Dim a As Single, b As Single*
    a = 4
    Print a / b
11. *Dim a As Single, b As Single*

```
    Print Sqr(a -  b)
    a = 4 :  b = a*2
12. Dim a As Single, b As Single, c As Single
    a = 3 :  b = 4
    c = Sqr(a^2 -  b^2) 'обчислення гіпотенузи прямокутного трикутника
```

1.9. Вказівка розгалуження

Розгалуження

Розгалуження – це така форма організації дій, при якій, в залежності від виконання або невиконання певної умови, виконується одна з двох послідовностей (серій) дій.

Можна навести багато прикладів вибору дії в залежності від результату виконання умови зі звичайного життя і науки. Наприклад:

- якщо день робочий, то йдемо в школу, інакше будемо відпочивати;
- якщо в рівностороннього чотирикутника кути прямі, то його називають квадратом, інакше його називають ромбом;
- якщо удар пружний, то повна механічна енергія зберігається, інакше − змінюється.

Таку структуру називають повним розгалуженням (структура «ЯКЩО – ТО – ІНАКШЕ»).

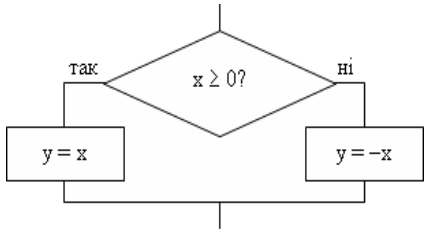
Умова – це твердження, яке може бути істинним чи хибним. Якщо твердження істинне, то вважається, що умова виконана. Якщо умова є істинною, то виконуються команди серії 1 (на блок-схемі − гілка «Так»), якщо ж умова не виконується (умова хибна), – команди серії 2 (гілка «Ні»). Після виконання однієї з серій команд виконавець переходить до наступної після розгалуження команди.

Приклад 1.61. Потрібно побудувати алгоритм обчислення значення функції $y = |x|$. Вона задається співвідношенням

$$y = \begin{cases} x, \text{ якщо } x \geq 0, \\ -x, \text{ якщо } x < 0. \end{cases}$$

При розв’язуванні цієї задачі потрібно виконати такі дії:

1) перевірити, умову чи є $x \geq 0$;
2) якщо $x \geq 0$, то присвоїти y значення x ($y = x$);
3) якщо $x < 0$, то присвоїти y значення, протилежне x ($y = -x$).



<u>Оператор</u>

<u>If...Then...Else...End If</u>

Для перевірки істинності умов і організації розгалуження у програмах мовою Visual Basic призначені умовні оператори *If...Then* та *If...Then...Else...End If.*

Оператор If використовує логічні змінні або вирази для запису умови й організації розгалуження в програмі.

Алгоритмічній конструкції «Повне розгалуження» відповідає умовний оператор *If...Then...Else...End If*

Синтаксис оператора:

If <умова> Then
<серія операторів 1>
Else
<серія операторів 2>
End If

або

If <умова> Then
< серія операторів 1>
Else : < серія операторів 2>
End If

Якщо результатом перевірки умови є значення *True*, то виконується блок дій *<оператор 1>*, який знаходиться після ключового слова *Then*. Якщо перевірка умови дала результат *False*, виконується блок дій *<оператор 2>*, вказаний після ключового слова *Else*. В другому з наведених варіантів може використовуватися як один оператор (його можна записати у тому ж рядку, що й *Else*, після знаку «:»), так і декілька (при цьому кожен оператор, починаючи з другого, записується в окремому рядку або теж через двокрапку).

Приклад 1.62. Змінну *A* треба збільшити на 1, якщо її значення ділиться на 3, і зменшити на 1 у протилежному випадку

```
If A Mod 3 = 0 Then
    A = A+1
Else : A = A–1
End If
```

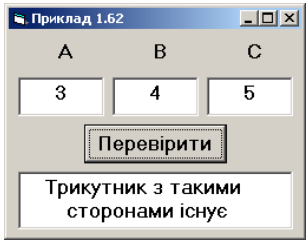
Приклад 1.63. Приклади умов, побудованих з використанням логічних операцій:

Not A <=3 – рівносильна виразу *A > 3*.

Age >= 10 And Age <= 18 – істинна тоді і тільки тоді, коли значення *Age* знаходиться в проміжку від 10 до 18 ($10 \le A \le 18$).

Age < 10 Or Age > 18 – істинна для всіх значень *Age*, які не належать проміжку від 10 до 18.

Приклад 1.64. Фрагмент програми для перевірки існування трикутника із сторонами *a, b, c*. З математики відома умова існування трикутника з певними довжинами сторін: сума двох будь-яких сторін повинна бути більшою за третю.

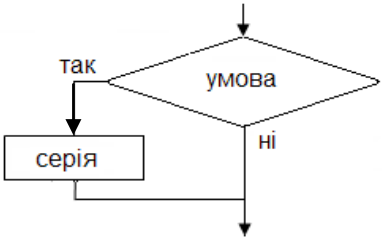



```
A = Val(Text1.Text)
B = Val(Text2.Text)
C = Val(Text3.Text)
If A < B + C And B < A + C And C < A + B Then
    Label4.Caption = "Трикутник з такими сторонами існує"
Else
    Label4.Caption = "Трикутника з такими сторонами не існує"
End If
```

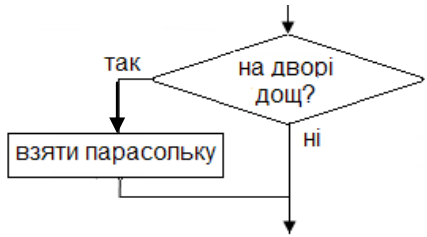
Конструкція If...Then

Досить часто при невиконанні умови не потрібно виконувати ніяких дій. Тоді використовується скорочена форма розгалуження − неповне розгалуження (структура «ЯКЩО-ТО»):

Блок-схема структури "ЯКЩО-ТО"



Приклад 1.65.



Оператор *If...Then* призначений для виконання деякої послідовності дій у тому випадку, якщо вказана в ньому умова є істинною.

Синтаксис оператора:

If <умова> Then <оператор>

Оператор *If* (Якщо) перевіряє істинність зазначеної умови. Якщо умова приймає значення *True* (Істина), програма виконає дію, зазначену в частині <оператор>. Якщо ж умова приймає значення *False* (Хибність), то команди будуть проігноровані і управління передається наступному після *If* оператору.

Значення змінних, що задіяні в логічному виразі, який використовується в якості умови, повинні бути визначені до початку оператора розгалуження.

Приклад 1.66. Нехай потрібно збільшити значення змінної *A* на одиницю, якщо її поточне значення менше 5. Відповідний оператор розгалуження має вигляд:

If A<5 Then A = A+1

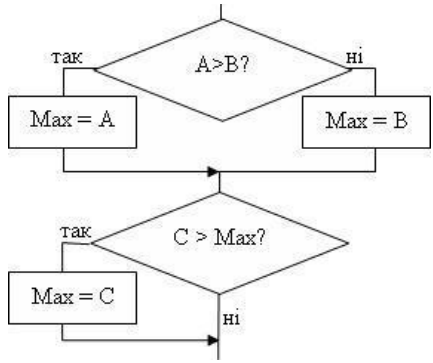
Оператор *A = A*+1 виконається тільки в тому випадку, коли істинна умова *A*<5:

Початкове значення A	Значення умови	Оператор A = A+1	Значення A після виконання оператора If...Then
1	True	виконується	2
5	False	пропускається	5
10	False	пропускається	10

Якщо ж у випадку істинності умови потрібно виконати послідовність дій, оператор *If...Then* записується за таким форматом:

Синтаксис	Приклад
If <умова> Then *<оператор 1>* *...* *<оператор N>* *End If*	*If A<5 Then* *A = A+1* *B = B*2* *End If*

Ключове слово *End If* вказує, де закінчується умовний оператор. Якщо умова хибна, управління передається оператору, записаному після *End If*.



Приклад 1.67. Дано три числа *A, B, C*. Визначити найбільше з цих чисел. Алгоритм пошуку найбільшого

з трьох чисел *A, B, C* реалізується за допомогою послідовного виконання повного та неповного розгалужень.

```
If A > B Then
    Max = A
Else: Max = B
End If
If C > Max Then Max = C
Text4.Text = Str(Max)
```

Елемент CheckBox (прапорець)

Елемент *CheckBox* застосовується, якщо користувачеві необхідно вибрати потрібні серед незалежних параметрів або характеристик. У панелі інструментів елемент керування *CheckBox* має вигляд: 

Під час роботи з елементом *CheckBox* основною властивістю, яка впливає на його відображення, є *Value* (Значення).

Таблиця 1.16

Значення властивості Value		
Значення	**Константа**	**Стан прапорця**
0	*vbUnchecked*	скинутий
1	*vbChecked*	встановлений
2	*vbGrayed*	недоступний

При кожній зміні користувачем стану прапорця, тобто при його встановленні або скиданні, для даного елемента відбувається подія *Click*. Вона виникає або після клацання на ньому лівою кнопкою миші, або після встановлення фокуса клавішею *Tab* з подальшим натисканням клавіші *Пропуск*.

Програмування прапорця. Після подвійного клацання елементу *CheckBox* на формі в режимі розробки, система **Visual Basic** створить процедуру обробки клацання *Check1_Click()*.

Подія *Click* (Клацання) відбувається як при встановленні, так і при скиданні прапорця. Щоб з'ясувати, встановлений прапорець чи скинутий, перевіряють значення властивості *Value* (табл. 2.5). Це роблять у процедурі *Check1_Click()* за такою схемою:

```
If Check1.Value = 1 Then
    <активізація параметрів, які пов'язані з прапорцем >
Else
    <відключення параметрів, які пов'язані з прапорцем >
End If
```

Приклад 1.68. Створення проекту, в якому користувач зможе вибирати різні значення властивості Font для елементу Label.

Властивості *Caption* об'єкта *Label1* надане значення *«Текст»*.

На формі розміщені три окремі елементи *CheckBox*, які мають імена *Check1, Check2, Check3*. Значення заголовків (*Caption*) змінені згідно з малюнком.

В залежності від стану прапорців застосовуються або відміняються параметри форматування шрифту для об'єкта *Label1*.

Властивості елемента *Label*, які відповідають за форматування тексту напису наведені в таблиці:

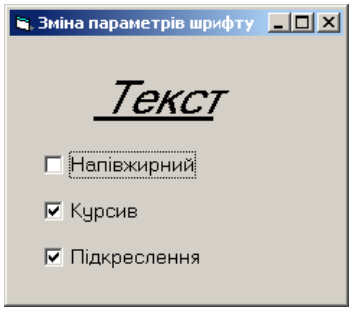
Властивість	Призначення	Можливі значення
FontBold	Напівжирний	True, False
FontItalic	Курсив	
FontUnderline	Підкреслення	

Перевірка стану прапорця *Check1* (Напівжирний) виглядає так:

```
If Check1.Value = 1 Then
    Label1.FontBold = True
Else: Label1.FontBold = False
End If
```

Перевірка стану прапорця *Check2* (курсив):

```
Private Sub Check2_Click()
    If Check1.Value = 1 Then
       Label1.FontItalic = True
    Else: Label1.FontItalic = False
    End If
End Sub
```



Питання для самоконтролю

1. *Дайте визначення алгоритмічної конструкції розгалуження.*
2. *Як записується і виконується умовний оператор у неповній формі? В якому випадку в неповному умовному операторі використовується ключове слово End If?*
3. *Як записується і виконується умовний оператор у повній формі?*
4. *Що слугує умовою в логічному операторі?*
5. *Запишіть у вигляді простої умови:*
   а) х – від'ємне число;
   б) число х не більше числа у;
   в) х – ділиться на 5.

6. *Запишіть у вигляді складеної умови:*
   а) 2 < x < 10;
   б) x не належить проміжку (2,10);
   в) x, y, z – додатні числа;
   г) хоча б одне з чисел x, y, z – додатне;
   д) жодне з чисел x, y, z – не додатне;
   є) x – парне число, більше від 10.
7. *Запишіть умовні оператори, які позначають такі дії:*
   а) перевірити, чи є число A парним;
   б) дано числа a і b. Від більшого числа відняти менше;
   в) перевірити, чи є серед чисел a,b,c рівні.
   г) упорядкувати числа a і b за неспаданням (якщо a>b, потрібно поміняти місцями значення a і b).
8. *Запишіть у вигляді умовного оператора обчислення функції Y = |X|.*
9. *Яких значень набудуть змінні A і B після виконання умовних операторів, наведених у таблиці, якщо початкові значення A=–3, B=5?*

A = –3, B = 5		A	B
1	*If A>B Then A=0 Else : B=0 End If*		
2	*If A<0 Then A=–A*		
3	*If A<>B Then A=B*		
4	*If A Mod 3 =0 Then A=A\3*		

A = –3, B = 5		A	B
5	*If A>B Then A=A-B Else : B=B-A End If*		
6	*If A<B Then A=2*A Else : B=B*A End If*		

10. *З якою метою в програмі може бути використаний елемент управління Прапорець? Яка подія є основною для нього?*
11. *Яка властивість дозволяє визначити, встановлений прапорець чи скинутий? Як організувати перевірку значення цієї властивості при зміні стану прапорця?*

1.10. Алгоритмічна структура Повторення. Цикл з параметром

Повторення (цикл)

Одним з найважливіших засобів програмування є можливість багаторазового повторення деякого набору команд. Команди, що повторюються, разом зі службовими словами, що забезпечують керування цим процесом, називаються *циклом*.

Цикл *– це алгоритмічна структура, за допомогою якої реалізується багаторазове повторення операторів.*

При цьому одна й та ж послідовність дій виконується доти, поки залишається істинною деяка умова. Серія команд, що

повторюється при кожному проході (ітерації) циклу, називається ***тілом циклу***.

Є два типи повторення: з передумовою та з післяумовою.

У випадку *повторення з передумовою* спочатку перевіряється умова, і якщо вона істинна, то тіло циклу виконується черговий раз, якщо ж ні – то виконання цих дій припиняється.

Якщо умова у вказівці повторення виявиться хибною при першій перевірці, то тіло циклу не виконається жодного разу.

Якщо ж при повторенні циклу умова незмінно залишається істинною, то виконання тіла циклу може повторюватися нескінченно (кажуть, що програма «зациклена).

Приклад 1.69. Алгоритм підрахунку суми N перших парних чисел (див. блок-схему). Позначимо через S шукану суму, через i – номер числа в послідовності парних чисел, a – поточне парне число. До початку циклу $S=0$ (ще нічого не підсумовували), $a=0$, $i=0$.

Щоб одержати наступне парне число, слід a збільшити на 2:

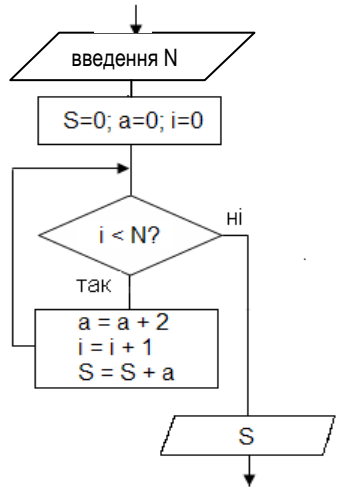
$$a = a + 2.$$

При переході до наступного доданка його номер збільшується на 1: $i=i+1$.

Наступний доданок додається до загальної суми: $S=S+a$.

Виконання циклу продовжується доти, поки $i < N$.

Зауважимо, що тіло циклу не виконається жодного разу, якщо на початку буде введене значення $N=0$. При першій перевірці умова $i<N$ виявиться хибною, тому відразу буде виведено нульове значення суми (змінної S).



Цикл із параметром

Цикл із параметром (або з лічильником) використовують тоді, коли треба виконати деякі дії певну кількість разів. Цикл із параметром реалізує оператор *For…Next*.

Синтаксис циклу *For…Next* такий:

For Змінна = ПочЗнач To КінЗнач [***Step Крок***]

<*оператори тіла циклу*>

Next **[*Змінна*]**, де:

- *For* – ключове слово, що позначає початок циклу;
- *Змінна* – ім'я змінної − параметра циклу;
- *ПочЗнач* – вираз − початкове значення параметра;
- *КінЗнач* – вираз − кінцеве значення параметра;
- *Step* – ключове слово, яке використовується, щоб вказати крок циклу (необов'язковий параметр);
- *Крок* – число, що задає крок циклу, тобто значення, на яке збільшується (чи зменшується) значення параметра після кожної ітерації;
- <*оператори тіла циклу*> – оператори, які повторюються в циклі. Як ви вже знаєте, кожне виконання тіла циклу називають ***ітерацією***;
- *Next [Змінна]* – ключове слово, що позначає кінець циклу *For...Next*. Вказувати після нього змінну-параметр не обов'язково, однак у деяких випадках це покращує читабельність коду, особливо за наявності вкладених циклів.

Виконується цикл за такою схемою:

1) параметр *Лічильник* одержує значення виразу *ПочЗнач*;
2) робиться перевірка, чи не перевищує значення змінної-параметра значення *КінЗнач*;
3) якщо *Лічильник* не перевищує *КінЗнач*, то виконуються оператори тіла циклу; якщо *Лічильник* перевищує *КінЗнач*, то цикл припиняється і керування переходить до рядка коду, який йде безпосередньо за ключовим словом *Next*;
4) якщо використовується ключове слово *Step*, параметр збільшується на величину *Крок*; якщо ж слово *Step* відсутнє, параметр *Лічильник* збільшується на одиницю і відбувається повернення до кроку 2.

Значення змінної-параметра змінюється на вказану величину або збільшується на одиницю щоразу після виконання тіла циклу.

Приклад 1.70. Програмний код, який реалізує алгоритм, представлений блок-схемою з прикладу 1.69:

```
N = InputBox("N?") ' Задання початкових значень змінних
S = 0 :  a = 0
For i = 1 To N    ' заголовок циклу: і приймає значення від 1 до N
      a = a + 2   ' оператори тіла циклу
      S = S + a
 Next i           ' кінець циклу
Print "S="; S     ' після завершення циклу виводиться значення S
```

Приклад 1.71. Параметр *A* збільшується на 2 на кожному кроці:

```
For A = 0 To 10 Step 2
      Print A
Next A
```

Приклад 1.72. Якщо необхідно змінювати значення параметра від більшого значення до меншого, то крок циклу роблять від'ємним:

```
For A = 10 To 0 Step –1
      S = S & Str(A)
Next A
```

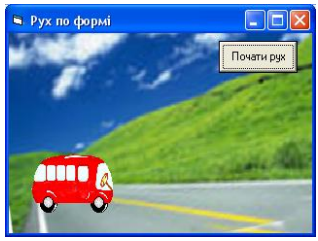
Приклад 1.73. Якщо кінцеве значення параметра циклу менше ніж початкове і крок циклу не зазначений, то тіло циклу взагалі не буде виконуватися:

```
For A = 10 To 0 ' Цей цикл не виконається жодного разу
      S = S & Str(A)
Next A
```

Приклад 1.74. Якщо значення змінної-параметра циклу змінюється в тілі циклу, є ризик ніколи не досягти кінцевого значення і програма «зациклиться».

```
For A = 1 To 10 ' Цей цикл буде працювати нескінченно
      A = A – 1
Next A
```

Приклад 1.75. В якості початкового та кінцевого значень параметра циклу можуть використовуватися вирази відповідного типу. Рух об'єкта по формі можна реалізувати за допомогою оператора циклу *For*:



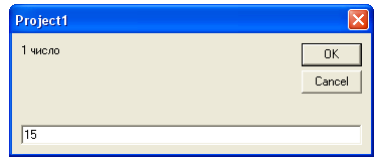
```
Dim i As Integer
Image1.Left = 0
For i = 1 To Form1.Width - 300
    Image1.Left = Image1.Left + 1
Next i
```

Приклад 1.76. Обчислення середнього арифметичного 5 дійсних чисел, що вводяться з клавіатури:

```
Dim i As Integer, A As Single, Sr As Single
Sr = 0
For i = 1 To 5
    A = InputBox (Str(i) & "число")
    Sr = Sr + A
Next i
Sr = Sr / 5
MsgBox "Середнє арифметичне " & Str(Sr)
```

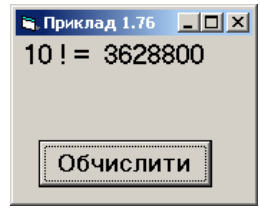
Приклад 1.77. З клавіатури вводяться 5 дійсних чисел, серед яких можуть бути і додатні, і від'ємні числа. Знайти середнє арифметичне додатних чисел.

```
Dim i As Integer, A As Single, Sr As Single, K As Integer
Sr = 0
K = 0 'лічильник додатних чисел
For i = 1 To 5
    A = InputBox (Str(i) & "число")
    If A >0 Then
        Sr = Sr + A
        K = K + 1
    End If
Next i
Sr = Sr / K
MsgBox "Середнє арифметичне додатних чисел " & Str(Sr)
```



Приклад 1.78. ***n!*** (читається «n-факторіал») обчислюється за формулою: n! = 1·2·3·…·n.

В програмному коді реалізоване обчислення ***F*** = 10! із застосуванням прийому накопичення добутку:

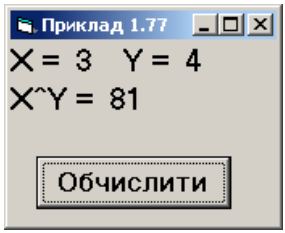



```
Dim i As Integer, N As Integer, F As Long
F = 1 : N = 10
```

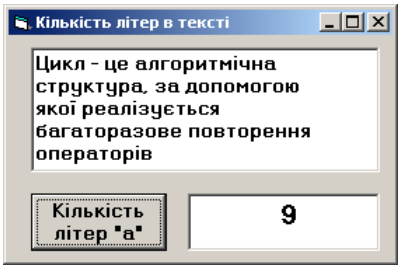
```
For i = 1 To N
    F = F * i
Next i
Print N; "! = "; F
```

Приклад 1.79. Не у всіх мовах програмування передбачено операцію піднесення до степеня $\mathbf{x}^{\mathbf{y}}$. Виконання цієї операції можна реалізувати за допомогою циклу *For*.

```
Dim X As Single, Y As Integer,
Dim P As Single, i As Integer
    P = 1
    X = InputBox ("X=?")
    Y = InputBox ("Y=?")
    For i = 1 To Y
      P = P * X
    Next i
    Print "X^Y = "; P
```



Приклад 1.80. Оператор циклу *For* зручно використовувати для аналізу значень рядкових величин (функції для роботи з даними типу *String* розглядалися в п.1.7). Підрахуємо кількість літер «а» в тексті *S*, який вводиться у текстове поле *Text1*. Для зручності, значення властивості *Multiline* об‘єкта *Text1* дорівнює *True*:




```
Dim S As String
Dim i As Integer
Dim K As Integer
Private Sub Command1_Click()
    S = Text1.Text
    For i = 1 To Len(S)
      If Mid(S, i, 1) = "a" Then K = K + 1
    Next i
    Text2.Text = Str(K)
End Sub
```

Приклад 1.81. Існує клас задач, розв’язування яких зводиться до перебору різних варіантів, в пошуках того, що задовольняє умові задачі. До цього класу належить задача пошуку дільників цілого числа.

Ціле число *K* є дільником числа *N*, якщо остача від ділення *N* на *K* дорівнює 0. Щоб знайти всі дільники, достатньо перебрати всі числа від 2 до *N* \ 2 і перевірити, чи є вони дільниками.

```
Dim N As Integer, K As Integer
N = InputBox ("N=?")
For K = 2 to N \ 2
    If N Mod K = 0 Then Print K;
Next K
```

Питання для самоконтролю

1. *Поясніть порядок виконання циклу з параметром. Скільки разів виконується тіло циклу?*
2. *З яким кроком змінюється значення параметра? Як змінюється значення параметра, якщо крок циклу не вказаний?*
3. *У якому випадку застосовують від'ємне значення кроку циклу?*
4. *Чому дорівнює S після виконання циклу:*

1	2	3	4
S = 0 *For A=5 To 7* *S = S + 1* *Next A*	*S = 0* *For A=5 To 7* *S = S + A* *Next A*	*S = 0* *For A=10 To 5 Step –1* *S = S + 1* *Next A*	*S = 0* *For A=10 To 5 Step–1* *S = S + A* *Next A*

5. *Скільки слів «Hello» виведеться на форму?*

```
For i = -2 To 7
    Print "Hello"
Next i
```

6. *Що надрукують на формі наведені програми?*

1	2	3	4
S = 1 *For i=1 To 5* *If i>2 Then S = S * i* *Next i* *Print S*	*S = 0* *A = 3* *For i =1 To 10* *S = A + i* *Next A* *Print S*	*S = 0* *A = 5* *For i = 1 To 3* *S = 2 * A* *Next A* *Print S*	*S = 0* *A = 5* *For i =1 To 3* *S = S + 2 * A* *Next A* *Print S*

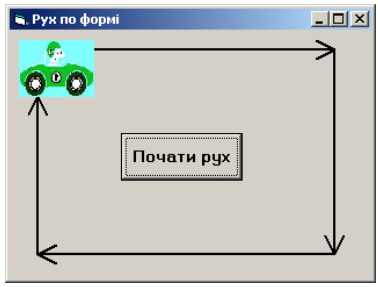
7. *Складіть таблиці зміни значень змінних для циклів:*

1	2	3
S=0 : P=1 *For I=1 To 5* *P=P*I* *S=S+P* *Next I*	*C =2* *For I=1 To 4* *C=1/(1-C)* *Next I*	*S=0 : P = 1* *For I=1 To 5* *P =–P*2* *S = S+P* *Next I*

8. *Використовуючи прийом накопичення суми, запишіть програмний код для знаходження:*
   а) суми квадратів чисел від 1 до N (число N вводиться з клавіатури);
   б) суми парних чисел в діапазоні від 100 до 200;
   в) суми парних чисел від 1 до N (число N вводиться з клавіатури);
   г) виразу R = 2N+4N+…+14N (число N вводиться з клавіатури);

д) виразу B=(A+3)(A+2.8)(A+2.6)…(A+1) (число A вводиться з клавіатури);

9. *Запишіть програмний код, який реалізує обчислення добутку цілих чисел N*M без використання операції множення.*
10. *Запишіть програмний код для підрахунку кількості літер B в тексті S, якщо значення змінної B типу String вводиться за допомогою функції InputBox.*
11. *Створіть проект, в якому реалізується рух об'єкта Image1 по формі за вказаною на малюнку траєкторією.*



12. *Розв'яжіть задачі за допомогою прийому перебору варіантів:*
    а) Перевірити, чи є задане ціле число A кубом іншого цілого числа.
    б) Задача Аль-Хорезмі (прибл. 780-850). Розкласти число 10 на 2 доданки, сума квадратів яких дорівнює 58.

1.11. Тематична робота «Базові поняття програмування»

Див. робочий зошит «Інформатика. Третій рік – єдиний курс. 11 клас.» /Бондаренко О.О., Ковшун М.І., Пилипчук О.П – Шепетівка: «Аспект», 2011.

2. Растрова комп'ютерна графіка та анімація

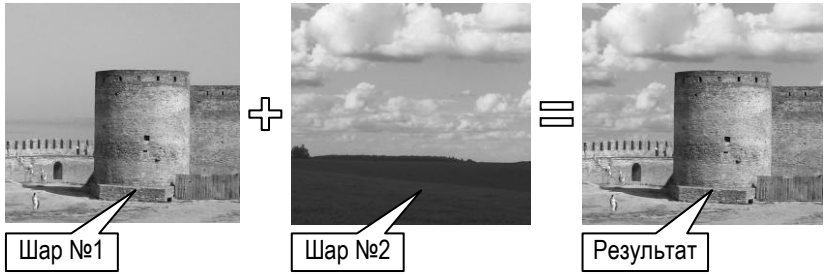
2.1. Середовище графічного редактора

Можливості сучасних графічних редакторів

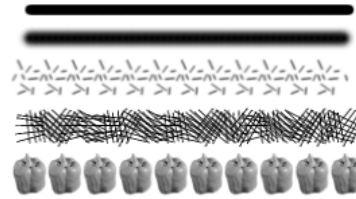
Для роботи з растровими зображеннями ви вже використовували графічний редактор ***Paint***. Незважаючи на те, що можливості ***Paint*** дуже скромні, його часто використовують для підготовки простих ілюстрацій. Адже він є на кожному комп'ютері, де встановлена операційна система ***Windows***.

Для виконання більш складних операцій з растровими зображеннями застосовують редактори професійного рівня ***GIMP***, ***Adobe Photoshop*** та ін. Серед основних можливостей цих програм:

- робота з шарами. Зображення будується з окремих шарів, які накладені один на інший так, ніби намальовані на прозорій



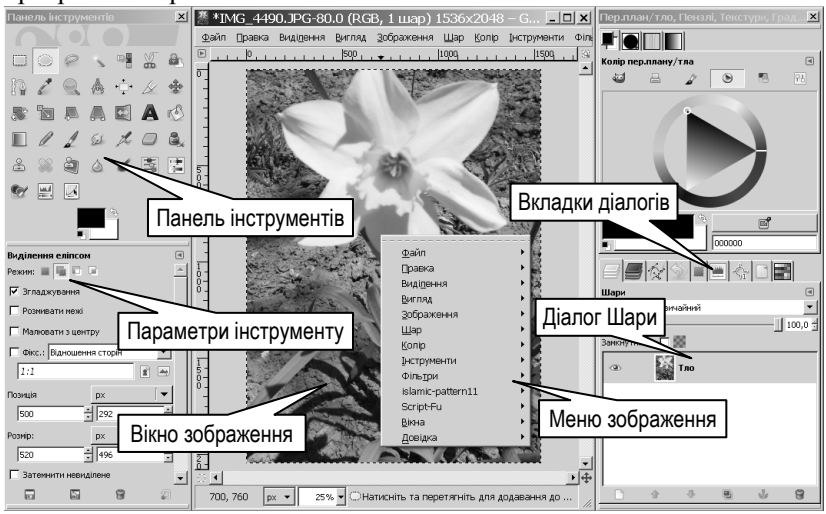
плівці. Користувач може в редагувати кожен шар незалежно від інших, спостерігаючи за зображенням в цілому;
- різноманітні інструменти для виділення фрагментів (правильної форми, довільної форми, подібних за кольором та інші);
- масштабовані пензлі різної форми. На малюнку показано лінії отримані різними пензлями;
- інструменти для перетворення зображення: обертання, зміна розмірів, викривлення тощо;
- потужні засоби для роботи з кольором: доступ до кожної з колірних складових зображення (каналів), регулювання яскравості, контрасту, прозорості, балансу кольорів та ін.;



- великий набір декоративних фільтрів;
- засоби для автоматизації роботи та інше.

Графічний редактор GIMP

Для запуску графічного редактора GIMP (англ. GNU Image Manipulation Program – програма для роботи з зображеннями з ліцензією GNU) користуються ярликом (див. мал.) або командою головного меню Пуск ⇨ Програми ⇨ GIMP ⇨ GIMP 2. Інший спосіб – відкрити файл з розширенням **xcf**, оскільки саме в таких файлах зберігаються проекти, що розроблюються в GIMP. Тут буде розглянута робота з програмою версії 2.6.11.



Середовище графічного редактора обов'язково містить 2 елементи: панель інструментів та вікно зображення. Рядок меню міститься у вікні зображення, завдяки чому, відкривши одночасно декілька зображень, легко зорієнтуватись, до якого з них буде

застосована вибрана команда. Доступ до меню зображення можна отримати й іншими способами: клацнувши правою кнопкою на зображенні (див. мал.) або клацнувши кнопку (*Меню*) у лівому верхньому куті вікна зображення. Подальша робота з меню відбувається так само, як і в вивчених раніше програмах.

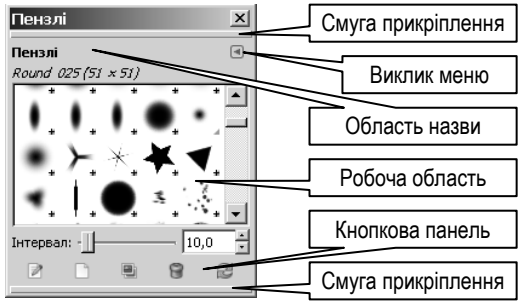
Поле зі списком *Масштаб* призначене для вибору або введення з клавіатури масштабу перегляду зображення. При цьому саме зображення не змінюється, а лише демонструється збільшеним або зменшеним. Якщо натиснути кнопку *Масштабування*, то масштаб зображення буде змінюватися одночасно зі зміною розмірів вікна. Вимикається цей режим повторним натисканням кнопки.

Якщо зображення видно не повністю, то швидко перейти до іншої його частини дозволяє кнопка *Навігація*. При її натисканні і утримуванні біля неї з'являється вікно зі зменшеним зображенням. На ньому слід вказати мишею потрібне місце і відпустити кнопку.

Діалоги

Інші засоби доступні користувачеві у вигляді *діалогових вікон* (далі − *діалогів*) і, за потреби, можуть бути увімкнені або вимкнені. Щоб увімкнути потрібний діалог, достатньо вибрати команду меню *Вікна* ⇨ *Діалоги з підтримкою прикріплення*, а потім клацнути у списку назву діалогу. Діалог з'явиться у вигляді окремого вікна (на малюнку – діалог *Пензлі*). Закривають діалог, як і інші вікна, кнопкою .

Порівняно з іншими вікнами діалоги мають деякі особливості, які дозволяють упорядковувати їх розміщення (див.мал.). Якщо навести вказівник на область назви (не плутайте з заголовком вікна!) і почати перетягувати, то біля вказівника з'явиться прямокутник з назвою діалогу. Перетягування завершують на іншому діалозі, при цьому результат залежить від положення вказівника. Якщо він вказує:



- на одну зі смуг прикріплення, то два вікна будуть об'єднані в одне так, що буде видно назви та робочі області обох діалогів;

- на рамку робочої області, то над назвою з'являться ще й вкладки обох діалогів, проте видно буде тільки один (див. мал. на попередній сторінці). Щоб перейти до іншого діалогу, слід клацнути його вкладку.

*Якщо при активному вікні зображення натиснути **Tab**, то панель інструментів та діалоги будуть приховані аж до наступного натискання **Tab**.*

Якщо прикріплений діалог перетягти за межі вікна, він набуде вигляду окремого вікна.

Робота з файлами

Крім відкриття графічних файлів командою *Файл* ⇨ *Відкрити…*, GIMP дозволяє створювати зображення й іншими способами:

- команда меню *Файл* ⇨ *Створити…* (клавіші **Ctrl+N**) відкриває діалогове вікно для створення нового порожнього документу. У вікні вказують ширину та висоту зображення, його роздільну здатність та деякі інші параметри. Також є список для вибору одного з готових шаблонів;
- команда меню *Файл* ⇨ *Створити* ⇨ *З буферу обміну* дозволяє відкрити для редагування зображення, скопійовані в буфер обміну в іншому вікні, або середовищі інших програм, або натисканням клавіші *PrintScreen*;
- щоб зробити знімок всього екрану або одного з вікон вибирають команду меню *Файл* ⇨ *Створити* ⇨ *Знімок екрану*;
- команда меню *Файл* ⇨ *Створити* ⇨ *Сканер/Камера…* дозволяє отримати зображення за допомогою відповідних пристроїв.

Зберігають файл, вибравши команду *Файл* ⇨ *Зберегти* та вказавши у діалоговому вікні ім'я файлу та папку. GIMP підтримує багато форматів файлів: **tiff**, **png**, **jpg**, **gif**, **bmp** тощо.

*Проте, якщо робота не завершена, то потрібно вказати розширення **xcf**.*

При цьому буде збережено найбільше даних про процес розробки: структуру шарів, їх прозорість, області виділень тощо.

Виділення фрагменту правильної форми

Для ефективної роботи з зображеннями потрібно вміти швидко виділяти той чи інший його фрагмент.

Якщо нічого не виділено, то виділеним вважається увесь малюнок.

Саме у межах виділеної частини зображення діють інструменти малювання, заповнення, регулювання тощо.

Для виділення фрагментів GIMP надає широке коло засобів. Найпростіші з них, це інструменти *Прямокутне виділення* і *Виділення еліпсу*. Їх використання нагадує подібні інструменти графічного редактора **Paint**: клацнувши кнопку на панелі інструментів, у вікні зображення перетягуванням виділяють фрагмент. Після відпускання кнопки миші на зображенні з'являється рухома штрихова *лінія виділення*. Проте є й відмінності. Якщо виділення було виконане приблизно, то відразу після відпускання кнопки миші є можливість його уточнити, перетягуючи кути і сторони прямокутника, що містить область виділення.

На малюнку показано діалог властивостей інструменту *Прямокутне виділення*. Кнопки з групи *Режим* дозволяють керувати взаємодією нової та попередньої областей виділення: – замінити поточну область виділення новою; – додати нову область виділення до наявної; – відняти нову область виділення від наявної; – створити область виділення з перетину з поточною. Ці кнопки є спільними для всіх інструментів виділення.

Щоб побудова прямокутника розпочиналась не з кута, а з його центру, слід встановити прапорець *Малювати з центру*.

Прапорець *Фікс.* вмикає засоби для кращого контролю за розмірами області виділення. Якщо він встановлений, то область виділення матиме задані ширину, висоту або відношення сторін. Потрібні значення вводять у розміщене нижче текстове поле. Розміщені поруч кнопки дозволяють швидко поміняти місцями введені значення, а отже й повернути на 90° майбутню область виділення.

Якщо вибрати параметр *Закруглені кути*, то з'явиться відповідний регулятор, яким задають радіус закруглення, виражений у крапках.

Часткове виділення крапок

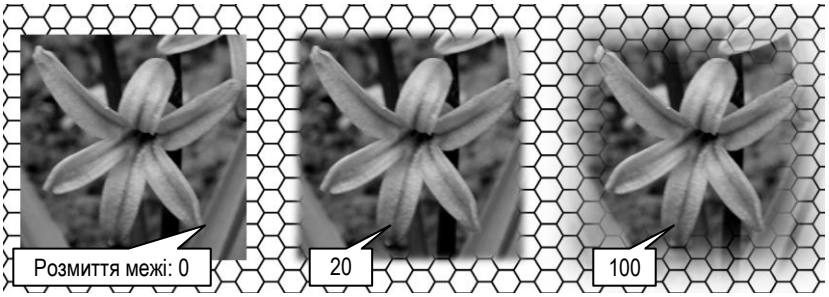
При виділенні фрагменту в простих графічних редакторах кожна з крапок зображення може бути або виділеною, або не виділеною. GIMP підтримує також часткове виділення крапок.

Ступінь «виділеності» крапки набуває значень від 0 (не виділено) до 255 (виділено повністю).

При вставлянні скопійованого фрагменту, що містить частково виділені ділянки, вони виглядають напівпрозорими.

При встановленні прапорця *Закруглені кути* стає доступним прапорець *Згладжування*. Якщо його встановити, то крапки поблизу лінії виділення стануть виділеними частково. В багатьох випадках це покращує сприйняття фрагменту, при доданні його до іншого зображення.

Щоб підсилити цей ефект, встановлюють прапорець *Розмивати межі* і вказують регулятором радіус розмиття. Після цього межі виділеного фрагменту будуть мати плавний перехід від повністю виділених до прозорих крапок. На малюнку показано результат вставляння на сітчасте тло квадратного фрагменту зі стороною 388 крапок, без заокруглення кутів але з різними значеннями розмиття межі: 0, 20 і 100 крапок. При розмитій межі



скопійований фрагмент має більші розміри, ніж прямокутник виділення, тому що

лінія виділення розмежовує точки виділені більше, ніж наполовину і менше ніж наполовину.

Параметр *Розмивання межі* є спільним для всіх інструментів виділення.

Напрямні

При побудові складних областей виділення та для точнішого розміщення фрагментів зображення стануть в нагоді *напрямні лінії* (далі − *напрямні*). Вони мають вигляд горизонтальних або вертикальних штрихових ліній, що нанесені поверх зображення.

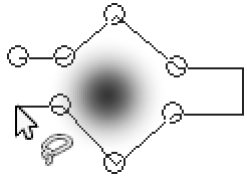
Для додання напрямної слід навести вказівник на одну з лінійок і виконати перетягування.

Завдяки напрямним зникає потреба занадто точно встановлювати вказівник перед натисканням або відпусканням кнопки миші. Достатньо наблизити його до напрямної або до перетину напрямних і натиснути (відпустити) кнопку. Спрацює «прилипання»: вказівник автоматично зміститься саме до найближчої напрямної або їх перетину.

Щоб вилучити зайву напрямну її просто перетягують за межі малюнка.

Виділення фрагменту довільної форми

Продовжимо розгляд інструментів виділення. Піктограмою з зображенням петлі (лассо) позначений інструмент *Вільне виділення*, призначений для виокремлення фрагментів довільної форми. Вибравши його на панелі інструментів слід уточнити параметри, після чого навести вказівник на початкову точку майбутньої лінії виділення і скористатись такими прийомами:



- клацати послідовно точки, щоб отримати прямолінійні сегменти;
- перетягуванням малювати криволінійні ділянки.

При цьому будується *контур виділення*, вузли якого при наближенні до них вказівника, позначаються невеликими колами. Їх можна перетягувати, щоб уточнити форму контуру.

Завершують побудову контуру виділення у початковій точці, або подвійним клацанням.

В другому випадку точка подвійного клацання та початкова точка будуть з'єднані прямолінійним сегментом.

Після цього контур зникає, і з'являється лінія виділення.

Виділення фрагменту з урахуванням кольору

Досить часто контур фрагменту, який потрібно виділити, утворений лініями, що розмежовують різні кольори. В такому випадку замість *Вільного виділення* зручніше скористатись інструментом *Розумні ножиці*:

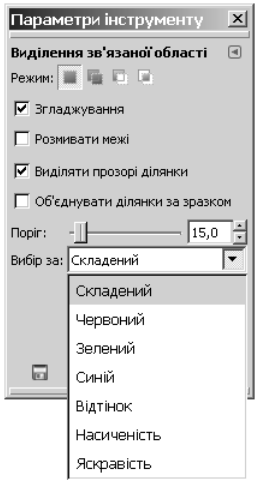
1) вибрати інструмент та уточнити параметри (режим, розмивання, згладжування);
2) клацнути початкову точку;
3) послідовно клацати точки межі на деякій відстані одна від одної.

При цьому GIMP намагатиметься автоматично відшуковувати

ділянку межі між точками. Якщо ділянка межі не дуже викривлена, відстань між точками можна робити більшою. Контур виділення буде зображуватись лінією з круглими вузлами (див. мал.);

4) обійшовши весь фрагмент, повторно клацнути початкову точку, щоб зробити контур виділення замкнутим;
5) натиснути **Enter**: побудований контур перетвориться на лінію виділення.

До натискання **Enter** користувач може уточнювати форму контуру, перетягуючи окремі вузли, а також додаючи нові вузли між наявними. Щоб додати вузол, достатньо клацнути лінію контуру між двома вузлами.



Інструмент *Виділення зв'язаної області* призначений для виділяння фрагменту, що містить крапки, подібні до початкової за кольором, насиченістю або яскравістю (див. мал.). Важливим параметром при цьому є *Поріг*, який встановлюють відповідним регулятором. Він визначає, на скільки виділені крапки можуть відрізнятися від початкової. При значенні 0 буде виділено одноколірний фрагмент, а зі збільшенням значення до нього будуть включатись все більше крапок, подібних до початкової.

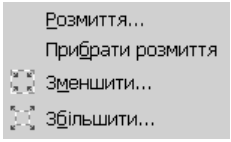
Так само використовують інструмент *Виділення за кольором*. Його дія відрізняється тим, що виділяються точки на всьому зображенні, а не тільки ті, що утворюють зв'язаний фрагмент.

Швидко виділити фрагмент, який відрізняється кольором від тла дозволяє інструмент *Виділення переднього плану*. Його використовують так:

1) виділяють фрагмент, подібно до того, як це роблять інструментом *Вільне виділення*. При цьому захоплюють не лише потрібну ділянку, але й частину тла;
2) наводять вказівник на ділянку, *що має бути виділена*, і, натиснувши ліву кнопку миші, «замальовують» якомога більше крапок цієї ділянки;

3) після відпускання кнопки «замальовані» та подібні до них крапки будуть видимі повністю, а інші – затемнені. Якщо затемнена частина потрібних крапок, то продовжують «малювати» по тих ділянках, намагаючись не зачепити тло;
4) натискають **Enter**, і незатемнені крапки стають виділеними.

Редагування області виділення



Після того, як фрагмент виділено, область виділення можна додатково відредагувати, скориставшись командами меню *Виділення* (див. мал.).

Незалежно від значень параметрів інструментів, якими виконувалось виділення, є можливість додати або прибрати розмиття межі, зменшити або збільшити розміри виділеної ділянки на певну кількість крапок.

Інколи буває легше виділити ту частину зображення, яку не потрібно редагувати. Наприклад, якщо треба редагувати прямокутну рамку зображення, то спочатку виділяють її внутрішню прямокутну частину, а потім виконують *інвертування* виділеної області (команда меню *Виділення* ⇨ *Інвертувати* або ***Ctrl+I***).

При інвертуванні виділені крапки стають не виділеними, і навпаки.

Вже побудовану область виділення можна перемістити в інше місце. Для цього вибирають інструмент *Переміщення*  , встановлюють параметр *Виділення* (див. мал.) і виконують перетягування.

Маска

Виділену ділянку зручно редагувати в режимі швидкої маски. Вмикають і вимикають цей режим кнопкою *Маска* у лівому нижньому куті вікна зображення або клавішами **Shift+Q**.

Після вмикання маски, поверх зображення з'являється напівпрозорий шар рожевого кольору, який наочно показує, до якої міри є виділеною кожна крапка.

Прозорі ділянки маски відповідають повністю виділеним крапкам, найщільніші – невиділеним. Проміжні відтінки рожевого позначають частково виділені крапки.

На масці можна малювати, використовуючи інструменти *Олівець*  , *Пензель*  , витирати ділянки інструментом

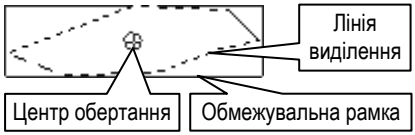
Гумка, виконувати заповнення різними способами, використовувати фільтри тощо (детально ці інструменти розглянемо далі). При виході з режиму маски на її основі формується нова область виділення.

Геометричні перетворення виділеного фрагменту

Геометричні перетворення фрагменту виконують, обираючи відповідні інструменти:

> *Перед застосуванням інструменту на панелі параметрів вказують, що буде перетворено: виділений фрагмент чи тільки область виділення .*

- *Віддзеркалення*. Слід уточнити параметр, відносно якої осі віддзеркалювати: горизонтальної чи вертикальної, після чого клацнути зображення;
- *Обертання*. Після клацання у вікні зображення відкриється діалогове вікно, навколо лінії виділення з'явиться обмежувальний прямокутник, а всередині – позначка центру обертання. Після цього перетягують, за потреби, центр і виконують обертання вручну (перетягуванням), або встановлюють у діалоговому вікні координати центра і кут повороту. В обох випадках дії підтверджують, натиснувши **Enter** або клацнувши кнопку *Повернути* у діалоговому вікні. Якщо утримувати клавішу **Ctrl**, то обертання здійснюватиметься кроками по 15°;



- *Масштабування*. Вибір інструменту теж супроводжується появою діалогового вікна і обмежувального прямокутника з позначкою центру масштабування. Для того, щоб довжина і ширина фрагменту змінювалися пропорційно, у діалоговому вікні натискають кнопку-перемикач з зображенням ланцюжка. На малюнку «ланцюжок» розірваний, отже розміри можна буде змінювати незалежно один від одного. Якщо кнопку клацнути ще раз, то «ланцюжок» з'єднається і довжина й ширина змінюватимуться одночасно.

Щоб вилучити виділений фрагмент, натискають ***Delete*** *або вибирають команду Правка ⇨ Очистити.*

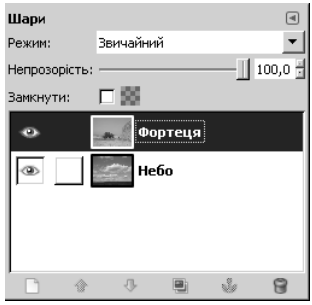
Питання для самоконтролю

1. *Назвіть растрові графічні редактори професійного рівня.*
2. *Опишіть можливості сучасних графічних редакторів.*
3. *Які елементи складають середовище GIMP?*
4. *Опишіть елементи вікна зображення.*
5. *Які особливості мають діалоги GIMP, порівняно з іншими вікнами?*
6. *Файл якого формату зберігає найбільше даних про зображення?*
7. *Опишіть дію кнопок Режим при виділянні фрагменту.*
8. *Як виділити прямокутний фрагмент з відношенням сторін 3:4?*
9. *Поясніть вислів «часткове виділення крапок».*
10. *Опишіть порядок роботи з напрямними.*
11. *Як виділити на малюнку фрагмент довільної форми?*
12. *Опишіть дію інструменту Розумні ножиці.*
13. *Чим схожі і чим відрізняються інструменти Виділення зв'язаної області та Виділення за кольором?*
14. *Опишіть порядок використання інструменті Переміщення.*
15. *Як редагують область виділення?*
16. *Як використовують маску?*
17. *Опишіть порядок використання інструментів для геометричних перетворень виділеного фрагменту.*

2.2. Шари. Інструменти малювання та заповнення

Багатошарова структура зображення

Як вже було сказано, GIMP дозволяє будувати зображення з елементів, які розміщені на окремих шарах. Для керування багатошаровою структурою зображення використовують меню *Шар* та діалог *Шари*.



При створенні нового документу з'являється один шар з назвою *Тло*. Графічні формати **jpg**, **bmp**, **png** тощо не передбачають збереження структури шарів зображення, тому при їх відкритті також з'являється один шар. Натомість, у файлах форматів **gif**, **tif**, **xcf** та деяких інших багатошарова структура зберігається і при їх відкритті вона буде відображена у діалозі *Шари* (див. мал.).

У робочому полі діалогу показаний стос шарів (у нашому випадку – шари *Фортеця* і *Небо*). При побудові зображення на екрані, шари перемальовуються, починаючи з нижнього. Таким

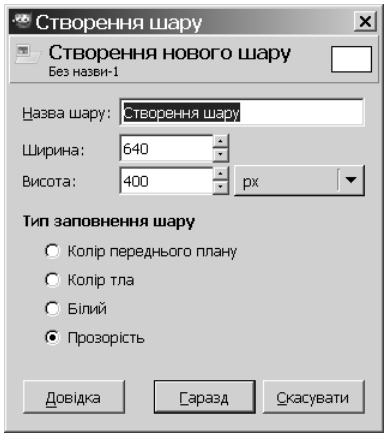
чином, шар *Фортеця* частково або повністю закриває шар *Небо*. Один з шарів є *поточним* або *активним*. Щоб зробити активним інший шар достатньо клацнути його назву. Саме над виділеною областю активного шару виконуються всі операції редагування.

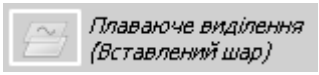
Якщо при редагуванні зображення на екрані не змінюється, можливо, активним є шар, якого не видно.

Керування шарами

Додати до зображення новий шар можна кількома способами:

- вибрати команду меню *Шар* ⇨ *Створити шар* і в діалоговому вікні (див. мал.) вказати параметри шару: назву, розміри та тип заповнення. Це саме діалогове вікно з'явиться, якщо клацнути кнопку  в діалозі *Шари*;
- відкрити папку з графічним файлом і перетягти його значок у вікно зображення. Вміст файлу буде доданий до структури зображення у вигляді одного або декількох шарів;
- вибрати команду меню *Шар* ⇨ *Дублювати шар* або клацнути кнопку  у діалозі *Шари*: над поточним шаром з'явиться його копія;
- виділити фрагмент, скопіювати його і вставити. При цьому з'являється особливий шар *Плаваюче виділення*, який командою *Шар* ⇨ *Створити шар* можна перетворити на звичайний шар.





Якщо замість цього клацнути за межами виділення, то новий шар не буде створено, а фрагмент буде додано до поточного шару.

Так само діє кнопка з зображенням якоря  в діалозі *Шари*.

За наявності плаваючого виділення більшість операцій з шарами стають недоступними.

Переміщують активний шар вгору або донизу відповідними кнопками діалогу *Шари* (,) або перетягуючи мишею.

Вилучають шар командою меню *Шар* ⇨ *Видалити шар* або клацнувши кнопку у діалозі *Шари*.

Діалог Шари

Розглянемо детальніше засоби діалогу *Шари*.

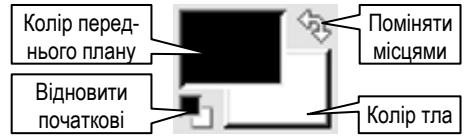
Якщо регулятором *Непрозорість* встановити значення 0%, то шар стане повністю прозорим. Крім керування загальною прозорістю шару, є можливість впливати на прозорість кожної його крапки. Для цього шар повинен містити так званий альфа-канал, який додають командою контекстного меню шару *Додати альфа-канал*. За наявності альфа-каналу, якщо вилучити фрагмент або скористатись інструментом *Гумка*, на шарі з'являться прозорі ділянки.

При встановленому прапорці *Замкнути* на прозорих ділянках не діють інструменти малювання і заповнення.

На початку рядка з назвою шару є дві кнопки-перемикачі.

Перша з них дозволяє повністю вимкнути шар: при цьому зображення будується так, ніби цього шару немає. Проте його можна в будь-який момент увімкнути, клацнувши цю ж кнопку (на ній при цьому з'явиться зображенням ока). Друга кнопка дозволяє тимчасово зв'язати декілька шарів для того, щоб над ними одночасно виконати певні перетворення (обертання, нахил, переміщення тощо). Зображення ланцюжка на цій кнопці свідчить, що даний шар зв'язаний з іншими.

Вибір кольорів



У нижній частині панелі інструментів показані зразки кольорів переднього плану та тла (див. мал.). Якщо їх немає, слід вибрати команду меню *Правка* ⇨ *Параметри*, у діалоговому вікні, що відкриється вибрати *Панель інструментів* і встановити прапорець *Колір переднього плану та тла*.

Щоб змінити один з кольорів, достатньо клацнути його зразок: з'явиться діалогове вікно з палітрою та іншими засобами вибору кольору. Підтверджують вибір натиснувши *Гаразд*.

Як видно з малюнка, за допомогою окремих кнопок можна швидко відновити початкові кольори переднього плану та тла, а також поміняти їх місцями.

Інструменти малювання. Пензлі

До інструментів малювання відносять *Олівець*  , *Пензель*  , *Аерограф*  , *Перо*  та *Гумку*  . Порядок їх використання однаковий:

- вибрати інструмент;
- вибрати форму пензля або пера. *Олівець*, *Пензель*, *Аерограф* та *Гумка* використовують спільну бібліотеку пензлів, один з яких вибирають після натискання кнопки у діалозі параметрів інструменту (див. мал.);

В якості першого пензля у першому ряду використовується вміст буферу обміну.

- вибрати колір переднього плану (для *Гумки* – колір тла);
- встановити рівень прозорості (регулятор та лічильник *Непрозорість*) та розмір пензля (регулятор та лічильник *Масштаб*);

- вибрати режим з відповідного списку. Режим визначає, як пензель впливає на наявне зображення (детальніше див. пункт «Режими шарів»);
- за потреби уточнити значення інших параметрів;
- малювати у вікні зображення, утримуючи натиснутою ліву кнопку миші.

Щоб отримати прямолінійний відрізок, клацають початкову точку, а потім, утримуючи Shift, кінцеву.

Проте кожен з інструментів має певні особливості. Лінія, отримана *Олівцем* не має напівпрозорих ділянок. *Аерограф* імітує роботу розпилювача фарби: при короткому натисканні лівої кнопки залишається нещільний слід; якщо ж кнопку утримувати довше, то щільність поступово зростає. На шарі, що не має альфа-каналу,

Гумка діє як *Пензель* з кольором тла. Якщо ж шар має альфа-канал, то гумка залишає прозорий або напівпрозорий слід.

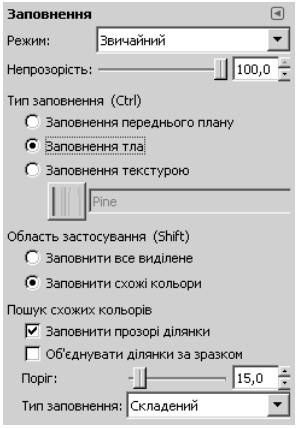
На доповнення до наявних пензлів користувач може створювати власні за допомогою вбудованого редактора пензлів. Викликають його кнопкою *Створити пензель* у діалозі *Пензлі*.

Інструмент *Перо* імітує малювання чорнилом за допомогою пера.

Інструменти заповнення

GIMP дозволяє виконувати заповнення (заливку) різних видів: однорідне, градієнтне, текстурою, а також створювати власні градієнти та текстури.

Градієнтом у графіці називають плавний перехід від одного кольору до іншого.
Текстура – це зображення, що використовується як візерунок при заливці.



Для заповнення кольором чи текстурою використовують інструмент *Заповнення*  . На малюнку наведено діалог його параметрів.

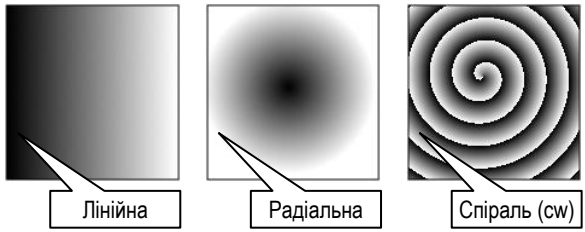
Залежно від стану перемикача *Тип заповнення*, заливка буде виконана кольором переднього плану, кольором тла або текстурою. Щоб вибрати текстуру слід натиснути кнопку, розміщену під перемикачем. У переліку, що з'явиться, на першому місці буде текстура, створена зі вмісту буфера обміну.

Якщо перемикачем *Область застосування* встановити значення *Заповнити схожі кольори*, то стануть доступні засоби, розміщені нижче. Зокрема встановлений прапорець *Об'єднувати ділянки за зразком*, означає, що при побудові області заповнення будуть використані дані не тільки з поточного шару, але й з інших видимих шарів.

При роботі з інструментом *Градієнт*  користувач має обрати один з готових варіантів градієнтного заповнення зі списку, що відкривається при натисканні кнопки (див. мал. на наступній сторінці). Поряд з кнопкою є прапорець, при встановленні якого

порядок кольорів градієнту змінюється на протилежний. Зауважимо, що деякі з пропонованих градієнтів мають фіксований набір кольорів, натомість в інших використовуються поточні кольори переднього плану та тла. На малюнку наведено зразки градієнтних заповнень, отриманих при різних значеннях параметра *Форма*.

Крім готового набору градієнтів GIMP має вбудований редактор градієнтів, що дозволяє користувачеві створювати власні набори кольорів та зберігати їх для подальшої роботи. Викликають його кнопкою *Створити градієнт* у діалозі *Градієнти*.



Режими шарів

Список *Режим* дозволяє вказати, як вміст поточного шару впливає при побудові зображення на розміщені нижче шари. У найпростішому випадку (режим *Звичайний*), крапки поточного шару просто малюються поверх відповідних крапок нижніх шарів. В інших режимах, для отримання коду кольору, над кодами кольорів нижніх і активного шарів виконуються певні математичні дії.

При зміні режиму шару дані про колір його крапок не змінюються.

Отже, увімкнувши для шару режим *Звичайний*, можна побачити власне сам шар.

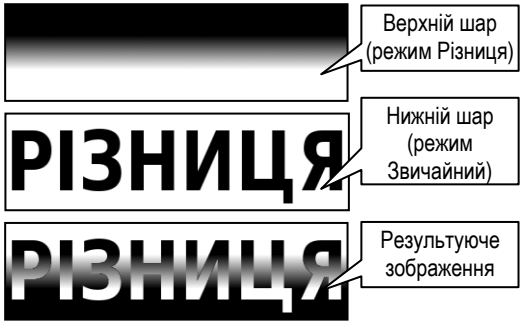
Приклад. Режим *Розсіяне світло* використовують тоді, коли окремі ділянки зображення (наприклад, обличчя), виглядають занадто темними:

- додайте прозорий шар над тим, який потрібно виправити;
- встановіть для нього режим *Розсіяне світло*;
- виберіть інструмент *Пензель*, встановіть круглу форму пензля достатнього діаметра з розмитими краями і білий колір переднього плану;

- малюйте поверх темних ділянок: ви побачите, що темні місця нижнього шару стають світлішими.

Змінюють вплив поточного шару на нижні шари регулятором Непрозорість.

Приклад. Якщо для шару увімкнути режим *Різниця*, то під його білими ділянками зображення ставатиме негативним, тобто всі кольори змінюватимуться на протилежні (див. мал.).

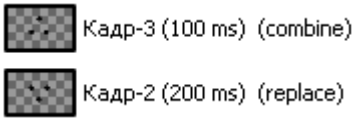


Створення GIF- анімації

Як вже було сказано, у файлі формату GIF можуть зберігатись декілька шарів. При перегляді такого файлу шари демонструються по черзі, утворюючи анімацію.

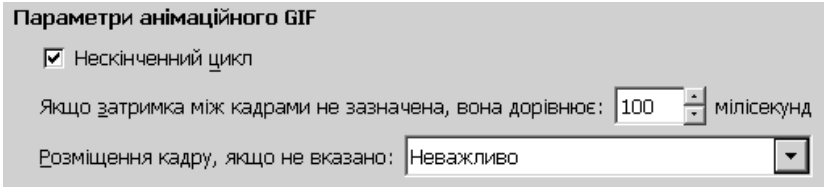
Послідовність створення анімаційного файлу така:

1) оскільки кількість кольорів у GIF-файлі обмежена числом 256, то слід увімкнути для зображення режим *Індексоване* (меню *Зображення* ⇨ *Режим* ⇨ *Індексоване*). Від обраної при цьому кількості кольорів залежить розмір майбутнього файлу;
2) підготувати багатошарове зображення, розмістивши кожен кадр на окремому шарі, враховуючи, що демонструватимуться вони, починаючи з нижнього шару. Назви шарів у цьому випадку використовують ще й для налаштування параметрів анімації, вказуючи в дужках, протягом якого часу демонструватиметься кадр та як він впливатиме на попередній кадр. Наприклад (див. мал.), кадр з назвою *Кадр-2* замінить (*replace*) попередній і демонструватиметься 200 мс, а *Кадр-3* домалюється (*combine*) до попереднього і до появи наступного кадру пройде 100 мс;



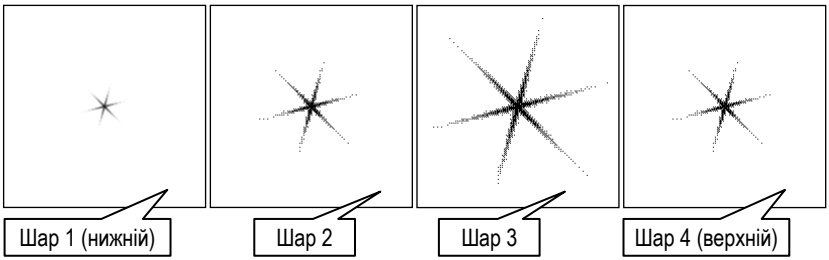
3) зберегти файл у форматі GIF. У процесі збереження з'явиться діалогове вікно *Експорт файлу*, в якому слід вибрати режим

Зберегти як анімацію і клацнути *Експорт*. У вікні *Збереження як GIF* (див. мал.) слід вказати параметри анімації:



- встановити прапорець *Нескінченний цикл*, якщо анімація має повторюватись безперервно;
- вказати час затримки між кадрами;
- вибрати зі списку один з варіантів розміщення кадрів: *Один кадр на шар*, якщо перед виведенням нового кадру попередній має бути стертий; *Накладання шар*ів − у протилежному випадку; *Неважливо* − якщо цей параметр несуттєвий для даного проекту.

Приклад. Пульсуюча зірка. Якщо створити 4 шари з



показаними на малюнку зображеннями зірки і зберегти як анімацію, то при перегляді побачимо пульсуючу зірку. При цьому, якщо шари мають прозоре тло, слід подбати, щоб кожен з них замінював попередній (*replace*), інакше при перегляді анімації на тлі шару 3 не буде видно шару 4.

Питання для самоконтролю

1. *Які можливості надає багатошарова структура зображення?*
2. *Файли яких форматів можуть містити декілька шарів?*
3. *Як додати до зображення ще один шар?*
4. *У стосі шарів з'явився шар Плаваюче виділення. Що робити?*
5. *Як змінити порядок розміщення шарів у стосі?*
6. *Що таке «альфа-канал»?*
7. *Що означає зображення*  *перед назвою шару?*
8. *Біля назв декількох шарів є знаки*  *. Що це означає?*
9. *Як змінити колір переднього плану? колір тла?*
10. *Що спільного у використанні інструментів малювання?*

11. *Чим відрізняються інструменти Олівець та Пензель?*
12. *Заливку яких видів дозволяє виконувати GIMP?*
13. *Опишіть особливості використання інструменту*  .
14. *Як виконати градієнтну заливку області?*
15. *Яку роль відіграє режим шару?*
16. *Опишіть послідовність підготовки GIF-анімації.*

2.3. Засоби корекції зображення

Типові недоліки зображень

Під час фотозйомки не завжди вдається створити умови для отримання знімка бажаної якості. Завадити можуть недостатня або надмірна освітленість об'єкту, який фотографують, зайві джерела світла тощо. Наприклад, при фотографуванні у сонячний день в кімнаті, кольорові штори на вікнах можуть створити загальний колірний фон, який спотворить інші кольори. Фахівець значну частину недоліків може усунути ще перед натисканням кнопки на фотоапараті, але якщо цього не зроблено, то покращити знімок допоможуть засоби графічного редактора.

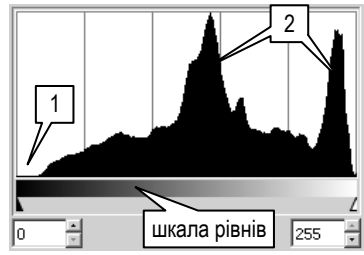
До типових недоліків фотозображень належать: мала або занадто велика яскравість, недостатня або надлишкова контрастність (див. мал.), недостатня різкість, спотворення кольорів, наявність зайвих деталей тощо.

Розглянуті далі інструменти та прийоми стануть в нагоді не лише при корекції фотографій але й при виконанні різноманітних творчих графічних робіт.

Корекція гістограми зображення

Перш ніж розпочати виправлення недоліків слід з'ясувати, у чому вони полягають. Це не завжди буває легко зробити.

Унаочнити колірний склад зображення дозволяє ***гістограма зображення*** − діаграма, що демонструє розподіл яскравості між крапками зображення.

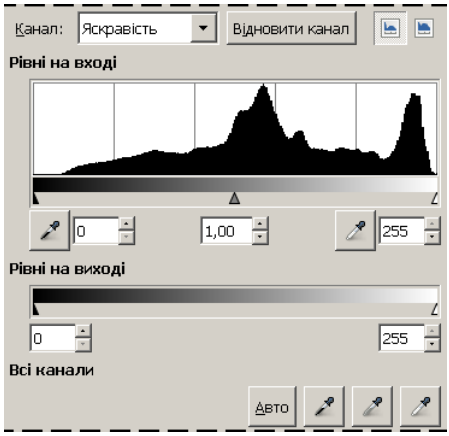


На малюнку показано фрагмент діалогу *Гістограма*. У нижній частині розміщена шкала рівнів яскравості, а над нею чорним кольором показано, скільки крапок зображення мають такий рівень. У нашому випадку чорні та близькі до них крапки на зображенні практично відсутні (область 1), а найбільше крапок мають середню та високу яскравість (області 2).

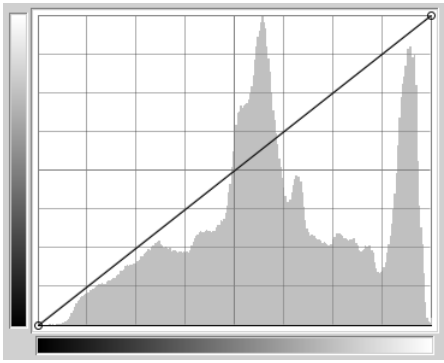
Гістограма є допоміжним елементом інструментів *Криві* та *Рівні*, розміщених в меню *Колір*.

*Дія інструментів **Рівні** та **Криві** не поширюється на шарі з індексованими кольорами.*
*Інструментом **Рівні** дозволяє: зробити зображення світлішим або темнішим, змінити його контраст, підсилити або ослабити окремі колірні складові.*

При виборі інструменту *Рівні* відкривається діалог *Коригування рівнів кольорів*, частина якого показана на малюнку. Під гістограмою є трикутні маркери для встановлення «чорної», «сірої» та «білої» точок. Зі списку *Канал* обирають, що потрібно змінити: зображення в цілому чи один колірних каналів. Ці точки можна також вказати безпосередньо на малюнку, скориставшись кнопками з зображеннями чорної, сірої та білої піпеток.



Якщо встановлено прапорець *Попередній перегляд*, то при пересуванні маркерів та клацанні «піпетками» на зображенні, воно змінює вигляд, що дозволяє оцінити кінцевий результат.

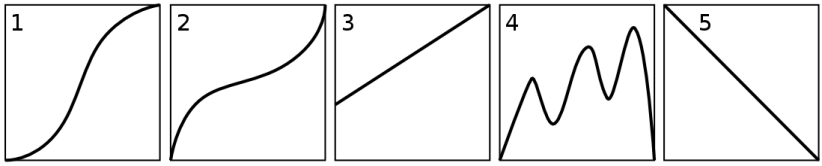


Дещо ширші можливості для корекції яскравості й кольору надає інструмент *Криві* (на малюнку – основна частина відповідного діалогу). Гістограма демонструє наявні у зображенні значення яскравості або рівня колірних складових, градієнтна смужка ліворуч демонструє рівні кінцевого зображення. Зв'язок між цими рівнями користувач встановлює, змінюючи форму кривої.

Спочатку крива має вигляд діагонального відрізка, як на малюнку. Це означає, що кінцеве зображення повністю відповідатиме початковому. Перетягуючи окремі точки кривої її форму ускладнюють, отримуючи, відповідно, складніші зміни в зображенні.

Розглянемо декілька прикладів (див. мал.):

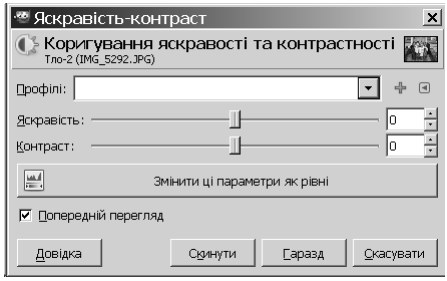
1) темні ділянки зроблено ще темнішими, а світлі – світлішими. Це відповідає збільшенню контрастності;
2) навпаки, темні ділянки освітлено, а світлі – затемнено: контрастність зменшилась;
3) найсвітліші ділянки майже не змінились, а темні стали світлішими. Причому, що темніша ділянка зображення, то вищим буде ступінь освітлення;
4) дуже складний зв'язок між вхідними й вихідними рівнями. Подібним способом можна отримати досить цікаві й несподівані колірні ефекти;
5) світлі ділянки зображення стали темними, а темні світлими, тобто отримано негативне зображення.



Регулювання контрастності, яскравості, балансу кольорів

Крім інструментів *Криві* та *Рівні* GIMP надає для коригування зображення ще й низку простіших засобів. До них зокрема належать інструменти *Яскравість-контраст* та *Баланс кольорів* з меню *Колір*.

Діалог, що відкривається при виборі команди *Колір* ⇨ *Яскравість-контраст* містить два регулятори для відповідних регулювань. Точні числові значення зміни параметрів можна ввести у поля лічильників праворуч від регуляторів (додатні − для збільшення, а від'ємні − для зменшення). Якщо встановлено прапорець *Попередній перегляд*, то у вікні зображення зразу ж будуть відображатись результати регулювань. При натисканні кнопки *Змінити ці параметри як рівні* буде відкрито діалог *Рівні*, засоби керування якого відображатимуть поточний стан регуляторів яскравості та контрастності.



Виправити недоліки кольору зображення або отримати цікаві ефекти можна, регулюючи баланс кольорів. На малюнку наведено фрагмент діалогу, що з'являється при виборі команди *Колір* ⇨ *Баланс кольорів*. Перемикачем *Вибір ділянки для зміни* вказують. на яку частину зображення впливатиме регулювання: найтемніші ділянки (*Тіні*), ділянки середньої яскравості (*Напівтони*) чи світлі частини. Трьома регуляторами здійснюють регулювання співвідношення між протилежними кольорами: бірюзовим та червоним, пурпуровим та зеленим, жовтим та синім. При натисканні кнопки *Відновити область* регулятори для обраної ділянки встановлюються у початкове положення.

Це не впливає на регулювання, зроблені для інших ділянок.

Щоб при регулюванні балансу кольорів яскравість зображення не змінилась, встановлюють відповідний прапорець (див. мал.).

Приклад. При фотографуванні людини з використанням фотоспалаху, розміщеного близько до об'єктиву фотоапарата, часто з'являється недолік зображення – червоні очі. Один зі способів виправлення включає такі кроки:

- виділити червоні очі;
- знебарвити виділену ділянку (наприклад, скориставшись командою меню *Колір* ⇨ *Знебарвити*);
- за потреби, знизити яскравість ділянки.

Інструмент Штамп

Інструмент *Штамп* призначений для копіювання ділянки зображення у формі обраного пензля. Завдяки різноманітності форм та розмірів пензлів *Штамп* дозволяє легко усувати зайві деталі зображення, «замальовуючи» їх фрагментами ділянки-зразка. Порядок використання *Штампа* такий:

- вибрати інструмент на панелі інструментів або командою *Інструменти* ⇨ *Малювання* ⇨ *Штамп*;
- на панелі властивостей вибрати форму пензля та уточнити інші параметри;
- навести вказівник на ділянку-зразок і, утримуючи клавішу ***Ctrl***, клацнути лівою кнопкою. На місті клацання з'явиться хрестоподібна позначка;
- навести вказівник на те місце, де має бути копія, і малювати клацанням або перетягуванням.

Інструмент Лікувальний пензель

Дрібні дефекти зображення та зайві деталі, особливо ті, що розташовані на однорідному тлі, зручно усувати за допомогою інструменту *Лікувальний пензель*. Його дія полягає в тому, що текстура з ділянки зображення, вказаної як зразок, переноситься на ділянку, що містить недолік. При цьому

застосовуються кольори, наявні поблизу дефекту. Наприклад, при редагуванні фотопортрету для усунення цятки на обличчі достатньо вказати як зразок ділянку чистої шкіри на обличчі: її текстура буде перенесена на ділянку з цяткою. Інструмент діє поступово, тобто при повторних клацаннях або переміщенні вказівника з утриманням лівої кнопки його дія підсилюється.

Порядок використання *Лікувального пензля* такий:

- вибрати інструмент на панелі інструментів або командою *Інструменти* ⇨ *Малювання* ⇨ *Лікувальний пензель*;
- на панелі властивостей вибрати форму пензля та уточнити інші параметри. Розмір пензля має бути значно більший за розмір дефекту, який треба усунути;
- навести вказівник на однорідну ділянку-зразок, яка не містить дефектів, і, утримуючи клавішу ***Ctrl***, клацнути лівою кнопкою. На місті клацання з'явиться хрестоподібна позначка;
- навести вказівник на недолік зображення і клацанням або перетягуванням усунути його.

На малюнку наведено приклад застосування *Лікувального пензля*. Для усунення недоліку було вибрано пензель круглої форми, діаметр якого видно на малюнку праворуч.

Питання для самоконтролю

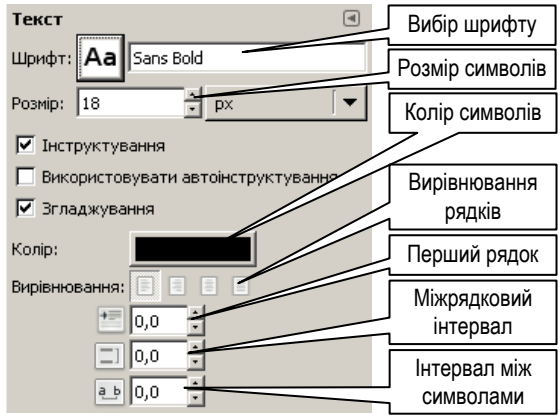
1. *Які бувають недоліки фотографій?*
2. *Що демонструє гістограма зображення?*
3. *Які можливості надає інструмент Рівні?*
4. *Яке призначення трьох кнопок з зображеннями піпеток?*
5. *Опишіть порядок використання інструменту Криві.*
6. *Як змінити контрастність зображення?*
7. *Як діє інструмент Баланс кольорів?*
8. *Коли на фотознімку виникає ефект «червоні очі»?*
9. *Як усунути ефект «червоні очі»?*
10. *Опишіть порядок використання інструменту Штамп.*
11. *Як діє інструмент Лікувальний пензель?*
12. *При фотографуванні у кімнаті з зеленими шторами на знімку вийшли спотворені кольори. Запропонуйте спосіб виправлення недоліку.*

2.4. Написи. Фільтри

Написи

Для додання до зображення написів використовують інструмент  *Текст*. Якщо вибрати його в панелі інструментів і клацнути на зображенні, з'явиться вікно простого текстового редактора. Текст, що набирають у ньому. зразу відображається й на

малюнку. При цьому створюється новий шар особливого типу – текстовий шар.



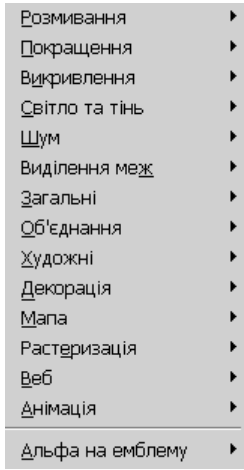
Властивості тексту залежать від значень, встановлених на панелі властивостей (див. мал.). Кнопка *Шрифт* розкриває список шрифтів, в якому наведено не тільки назви, але й зображення двох літер, за якими легко оцінити вигляд символів. Список поруч з лічильником *Розмір* призначений для вибору одиниць вимірювання висоти символів. Кнопка *Колір* відкриває стандартний діалог вибору кольору для символів.

Нижче розміщені засоби для встановлення формату абзацу, призначення яких зрозуміле з малюнка.

Щоб відредагувати текст, слід вибрати текстовий шар та інструмент Текст, після чого клацнути в межах тексту.

Що таке фільтр? Огляд фільтрів

Крім інструментів для ручного малювання та редагування сучасні графічні редактори включають засоби для швидкого виконання більш складних операцій. Їх називають *фільтрами*.



Дія деяких фільтрів полягає у послідовному автоматичному використанні низки інструментів редактора, подібно до того, як працюють макроси у текстовому процесорі. Інші ж фільтри є програмами, що обробляють зображення. При використанні більшості фільтрів відкриваються діалогові вікна, в яких необхідно вводити значення параметрів обробки.

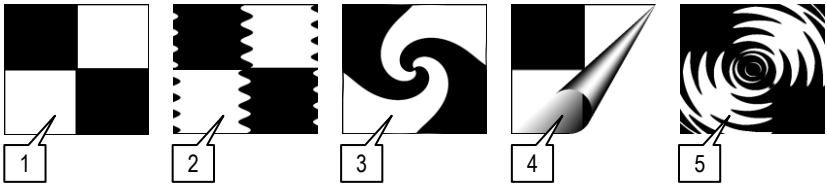
Для зручності фільтри поділяють на групи, список яких міститься в меню *Фільтри* (на малюнку – основна частина цього списку). Оскільки різноманітних фільтрів дуже багато,

то розглянемо лише деякі з них. З іншими варто ознайомитися самостійно.

Група *Розмивання*. Фільтр *Гаусове розмивання* дозволяє розмити, тобто зробити нерізким, все зображення або виділений фрагмент (наприклад, об'єкти на задньому плані портретної фотографії). Фільтр *Вибіркове Гаусове розмивання* діє тільки на однорідні ділянки і практично не впливає на межі між ділянками різного кольору або тону, завдяки чому зникають найдрібніші деталі, а зображення в цілому виглядає різким.

Група *Покращення* включає зокрема фільтр *Нечітка маска*, який вважають найкращим засобом для покращення різкості зображення.

Фільтри групи *Викривлення* імітують різноманітні деформації малюнка. Призначення багатьох з них зрозуміле з назви. На малюнку 1 – початкове зображення, а на наступних – результат дії фільтрів *Брижі* (2), *Вихор та щипок* (3), *Загнута сторінка* (4), *Хвилі* (5).



До групи *Світло та тінь* увійшли фільтри для створення різноманітних оптичних ефектів. Наприклад, результатом застосування фільтру *Наднова* буде додане до малюнка зображення яскравої зірки з напівпрозорими променями, що розходяться аж до країв малюнка.

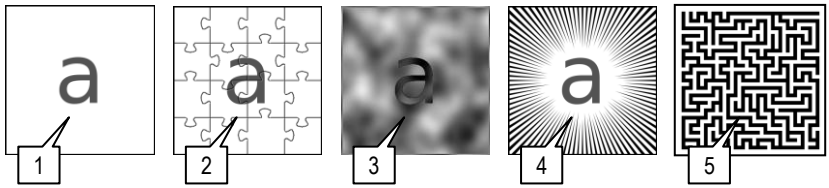
У групі об'єднання є фільтр *Фотоплівка*, що дозволяє об'єднати декілька фотографій в один малюнок, оформлений у вигляді фрагменту фотоплівки, де в якості «кадрів» можна використати будь-які з відкритих зображень. При цьому малюнок-результат буде відкритий в іншому вікні.

Фільтри групи *Художні* виконують перетворення, що імітують різноманітні техніки виконання малюнка. Крім класичних художніх прийомів (*Комікс*, *Кубізм*, *Олійна фарба*) тут є й цілком штучні (*Фотокопія*, *Хижак*).

Для додання різних елементів оформлення використовують фільтри з групи *Декорація*: *Додати рамку*, *Додати фаску*, *Кавові плями*, *Закруглені кути* тощо.

Група *Мапа* об'єднує фільтри, дія яких полягає в зміщенні частин зображення, багаторазовому використанні цілого зображення (можливо, деформованого) тощо. Фільтр *Без швів* створює зображення, яке не утворює швів при укладанні багатьох копій поряд. Це потрібно, наприклад, при підготовці текстур для інструменту *Заливка*.

Різноманітні ефекти, пов'язані з автоматичним створенням зображень, отримують за допомогою фільтрів з групи *Растеризація*. Деякі з них впливають безпосередньо на активний шар, інші – додають до зображення окремий шар з відповідним ефектом. На малюнку 1 наведено початкове зображення, а на



наступних – результати застосування різних фільтрів з цієї групи: *Головоломка* (2), *Різні плями* (3), *Лінійна наднова* (4), *Лабіринт* (5). В останньому випадку початкове зображення знищене, оскільки цей фільтр працює з активним шаром. Цікаві візуальні ефекти отримують, застосовуючи послідовно декілька різних фільтрів.

Підготувати малюнки для оформлення веб-сторінок допоможуть фільтри з групи *Веб*.

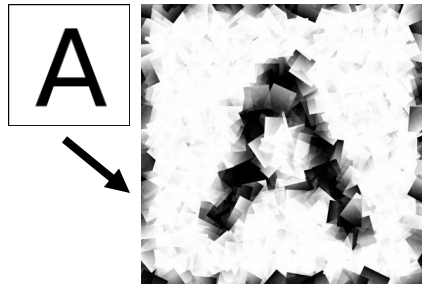
Прискорити розробку простих анімацій дозволяють фільтри, зібрані в групу *Анімація*. При використанні деяких з них результуюче зображення виводиться в окремому вікні. Для перегляду анімації безпосередньо в середовищі GIMP слід вибрати фільтр *Відтворення*.

Детально вивчати можливості різних фільтрів найкраще, експериментуючи з ними. Розглянемо декілька прикладів.

Фільтр Кубізм з групи Художні

Кубізм став революційною течією в образотворчому мистецтві XX століття (на малюнку – фрагмент картини Пабло Пікассо

«Портрет Амбруаза Воллара»). Фільтр з такою ж назвою (група *Художні*) допоможе перетворити будь-який малюнок на зображення в стилі кубізму.

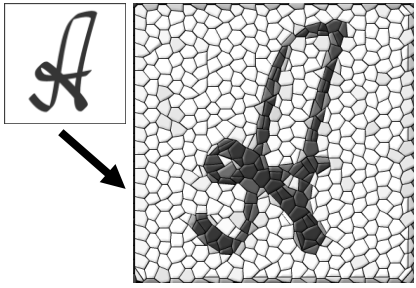


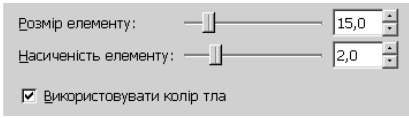
При його застосуванні «кубістичне» зображення будується з окремих елементів квадратної форми, отже й основним параметром у діалоговому вікні фільтру є розмір елементу, що встановлюється відповідним регулятором. Іншим регулятором змінюють насиченість (прозорість) елементів малюнка. Якщо встановити прапорець *Використовувати колір тла*, то при формуванні елементів малюнка крім наявних на ньому кольорів буде використано ще й колір тла.

Фільтр Мозаїка (група Викривлення)

Одним з найдавніших видів мистецтва є мозаїка – викладання візерунків та картин з окремих елементів невеликого розміру (камінців, шматочків скла тощо). На малюнку показано фрагмент мозаїки зі знаменитого Софіївського собору, що в Києві.

Подібний декоративний ефект дозволяє отримати фільтр *Мозаїка* (група *Викривлення*). При його застосуванні на початкове зображення накладається сітка, кожна комірка якої перетворюється на елемент мозаїки. Основні параметри, які вказують у діалоговому вікні:





- форму елементів обирають зі списку *Елементи мозаїки*. Можливі значення: *квадрати*, *шестикутники*, *восьмикутники й квадрати*, *трикутники*;

- розмір елементу та інтервал між елементами;
- охайність елементів: 1 – елементи матимуть правильну форму, при зменшенні до 0 зростає спотворення форми елементів.

Підвищення різкості зображення

Якщо різкість зображення недостатня, наприклад, внаслідок неточного фокусування при фотозйомці, то виправити цей недолік дозволяють фільтри з групи *Покращення*: *Підвищення різкості* та *Нечітка маска*.

Перший з них дуже простий у керуванні. Його діалогове вікно містить лише один регулятор *Різкість*. Якщо встановлено прапорець *Перегляд*, то у діалоговому вікні можна переглянути результат застосування фільтру. Завершують регулювання натисканням кнопки *Гаразд*.

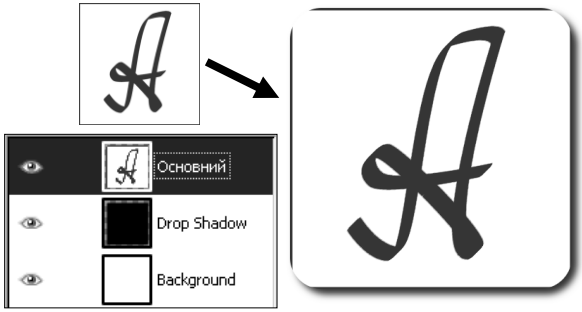
Фільтр *Нечітка маска* більш досконалий. Його параметри: *Радіус, Величина* та *Поріг* по різному впливають на результуюче зображення, тому, змінюючи їх за допомогою регуляторів, також потрібно орієнтуватися на вікно попереднього перегляду.

Фільтр Закруглені кути (група Декорація)

Цей фільтр дозволяє додати до зображення оформлення у вигляді заокруглених кутів та тіні (див. мал.). У діалоговому вікні фільтру слід вказати значення таких параметрів:

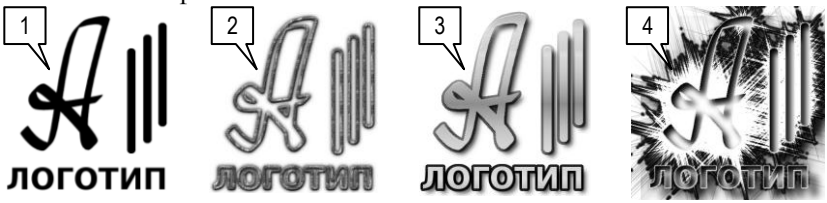
- зсув тіні вздовж осей X та Y вказують у пікселях. Додатні значення означають зсув відповідно праворуч і донизу, а від'ємні – ліворуч і вгору;
- радіус розмивання тіні у пікселях;

Тінь малюється на новому шарі *Drop Shadow*, який додається під поточним, і лише при встановленому прапорці *Додати падаючу тінь*. При встановленому прапорці *Додати тло*, під шаром з тінню додається шар Background, залитий кольором тла.



<u>Група фільтрів Альфа на емблему</u>

Фільтри цієї групи допоможуть швидко розробити ефектний логотип (емблему). Назва групи пояснюється тим, що для побудови емблеми не використовуються дані про колір крапок малюнка, а лише його альфа-канал.



Отже, спочатку потрібно підготувати прозорий шар, на якому є непрозора заготовка майбутньої емблеми (мал.1). Це може бути напис або будь-який малюнок. Зробивши цей шар активним, слід вибрати один з фільтрів з групи *Альфа на емблему*. Засоби керування, зібрані у діалоговому вікні фільтру, у різних фільтрів дуже відрізняються. На малюнку наведено результати застосування фільтрів *Об'ємний контур* (2) , *Глянцевий* (3), *Крига* (4).

Питання для самоконтролю

1. *Як додати напис до зображення?*
2. *Які особливості має текстовий шар?*
3. *Як відредагувати текст на текстовому шарі?*
4. *Опишіть засоби панелі властивостей інструменту Текст.*
5. *Поясніть дію кнопки Текст за контуром.*
6. *Що називають фільтром?*
7. *Як працює фільтр Вибіркове Гаусове розмивання?*
8. *До якої групи належать фільтри Загорнута сторінка та Хвилі?*
9. *Яка група об'єднує фільтри, що імітують оптичні ефекти?*
10. *Як перетворити фотографію на зображення в стилі кубізму?*
11. *Яку особливість має зображення. оброблене фільтром Без швів?*
12. *Назвіть основні параметри фільтру Мозаїка.*
13. *Як виправити зображення з недостатньою різкістю?*
14. *При застосуванні фільтру Закруглені кути з'явилось зображення тіні. Як її прибрати?*
15. *Поясніть назву та призначення групи фільтрів Альфа на емблему.*

2.5. Тематична робота «Растрова комп'ютерна графіка та анімація»

Див. робочий зошит «Інформатика. Третій рік – єдиний курс. 11 клас.» /Бондаренко О.О., Ковшун М.І., Пилипчук О.П – Шепетівка: «Аспект», 2011.

3. Бази даних. СУБД

3.1. Загальні відомості

База даних являє собою файл або сукупність файлів спеціального формату, які містять належним чином структуровані дані, призначені для зберігання, накопичення, обробки та використання за допомогою комп'ютера.

Це можуть бути архіви, описи майна і матеріалів, бухгалтерські документи, особові справи працівників у відділі кадрів, інформаційне наповнення веб-сайту тощо. І всюди для користування інформацією необхідні засоби для її редагування, систематизації і швидкого пошуку.

Більшість сучасних баз даних (БД) є *реляційними* (*див. далі*). Реляційною називається база даних, у якій всі дані, що доступні користувачеві, організовані у вигляді *таблиць*, що зв'язані між собою, а всі операції, що виконуються з даними, зводяться до дій із цими таблицями.

Таблиця складається з рядків і стовпців. Кожний стовпець містить дані одного *типу*. У базах даних рядки таблиці називають *записами*, а стовпці – *полями* (*див. мал.*).

Поля та зв'язки між таблицями утворюють *структуру бази даних*, а записи складають інформацію, що у ній міститься.

Для того щоб зрозуміти, що таке структура бази даних, уявіть порожню базу, у якій поки що немає ніяких даних. Незважаючи на відсутність даних у базі, інформація в ній все-таки є. Це опис структури бази, тобто типів даних та зв'язків між ними. Структура визначає, що і в якому вигляді може бути записане в базу даних.

Системи управління базами даних (СУБД)

СУБД *– це програми для введення, зберігання, пошуку та обробки даних в базі даних.*

Основні функції, що реалізуються СУБД:

- забезпечення введення даних у комп'ютер з одночасною перевіркою їх правильності;
- видача даних користувачу на принтер або на екран у відповідності до його запиту;
- забезпечення одночасного доступу декількох користувачів;
- виконання найнеобхідніших видів обробки даних: сортування даних, пошук потрібного запису тощо;
- забезпечення неушкодженості бази даних при припиненні електропостачання та в інших аварійних ситуаціях;
- коректне внесення змін у базу даних при одночасній роботі з нею багатьох користувачів.

Основні об'єкти бази даних

База даних містить об'єкти чотирьох основних типів:

- ***Таблиці*** (*Таблицы*) – основні об'єкти бази даних, де зберігаються дані;
- ***Запити*** (*Запросы*) – для вибору потрібних даних із двох або кількох зв'язаних таблиць;
- ***Форми*** (*Формы*) – для зручного введення, перегляду та корегування взаємозв'язаних даних;
- ***Звіти*** (*Отчёты*) – для підготовки даних до друкування.

Людей, що працюють з базою даних, умовно поділяють на розробників та користувачів. Розробник вдосконалює структуру бази даних, створює всі об'єкти. Від нього залежить, чи буде база даних надійною і зручною в користуванні. Користувач, здебільшого, не є спеціалістом з теорії розробки баз даних, тому працює лише з формами та звітами. Він може нічого не знати про таблиці, зв'язки тощо, і при цьому успішно вносити дані до бази або отримувати їх у відповідь на свої дії з формою.

Ієрархічна, мережна, реляційна моделі баз даних

Ієрархічна база даних розглядає дані як сукупність різних об'єктів, де об'єкти нижнього рівня підпорядковані об'єктам

вищого рівня. ***Мережна база даних*** описує сукупність об'єктів, кожен з яких може бути зв'язаний з іншим.

Реляційна база даних складається із взаємозв'язаних таблиць. Кожна таблиця описує певну сутність з предметної області бази даних. Така модель найбільш поширена серед сучасних СУБД.

Приклад реляційної моделі баз даних

Як приклад використання реляційної бази даних, розглянемо купівлю квитка в залізничній касі. Спочатку ми звертаємось до розкладу руху поїздів у вигляді таблиці, де записані номери поїздів, кінцеві станції, час прибуття і відправлення. В цій таблиці вибираємо № поїзда, який нас влаштовує. Крім цієї таблиці десь повинні бути інші *взаємозв'язані* таблиці:

- дані про вагонний склад вибраного поїзда;
- наявність вільних місць у кожному вагоні за датами;
- дані про відстань до станцій за маршрутом руху поїзда;
- дані про вартість проїзду у вагонах кожного типу.

Ви кажете касиру: «Дайте, будь ласка, квиток на поїзд № 87 в плацкартному вагоні до Запоріжжя на 20 березня».

Кожне робоче місце касирів обладнане комп'ютером з потрібними програмами, з'єднаним через мережу з комп'ютерами в інших містах та центральним комп'ютером, на якому зберігається база даних про всі поїзди, що курсують залізницями України.

Касир вводить запит і комп'ютер звертається до центральної бази даних, де знаходить:

- за номером поїзда – плацкартні вагони від 9 до 16, серед яких вибирає, наприклад, № 9;
- у вагоні № 9 на 20 березня вільні місця від 26 до 54, серед яких вибирає 27;
- за номером поїзда – відстань до станції Запоріжжя 568 км;
- за відстанню 568 км і плацкартним типом вагона – вартість проїзду, наприклад, 58 грн.

Після виконання такої процедури на принтері друкується квиток, до відповідних таблиць центральної бази даних вносяться зміни, що 20.03.2012 р. у поїзді № 87 у вагоні № 9 місце 27 зайняте до станції Запоріжжя.

Особливості реляційної бази даних

Концепцію реляційних баз даних, в основі яких покладено математичне поняття відношення (від англ. relation), запропонував

Е.Ф.Кодд у 1970 р. Реляційна база даних складається з взаємопов'язаних двовимірних таблиць (*відношень*).

Двовимірна таблиця складається із ***записів*** *(рядків) і* ***полів*** *(стовпчиків).*

Заголовок стовпчика – це назва певного атрибуту сутності, описаної в таблиці, а вміст стовпчика – значення цього атрибуту для різних об'єктів. Приклади сутностей, їхніх атрибутів та можливих значень наведено в таблиці:

Сутність	Атрибут	Значення
Автомобіль	Марка	ЗАЗ Форца
	Рік випуску	2011
Змагання	Вид спорту	Волейбол
	Клас	11
	Рівень	Районний

Зрозуміло, що *поле* може містити значення лише одного типу – текст, число, дату тощо, а *запис* – дані про один об'єкт. Наприклад, у таблиці з відомостями про учнів у кожному з рядків-записів розміщено дані про одного учня, а атрибутами учня є: прізвище, ім'я, адреса, дата народження тощо. Для одного з учнів ці атрибути мають такі значення: Бондарчук, Іван, вул. Драйзера 36а, кв. 1, 05.11.1987.

Запис є ***інформаційною моделлю*** *певного об'єкта, а поля запису описують* ***властивості*** *цього об'єкта.*

База даних складається, як правило, з декількох таблиць. Кожна таблиця повинна мати *ключове поле* (або *ключ*). Ключове поле містить дані, які є унікальними для кожного запису, тобто жодні два записи не можуть мати однакових даних у ключовому полі. Ключем може бути, наприклад, порядковий номер учня в класному журналі, табельний номер працівника на заводі або число, яке генерується автоматично при додаванні даних у таблицю. В деяких випадках ключ складається не з одного, а з декількох полів.

За допомогою ключових полів створюються зв'язки між таблицями бази даних, що дозволяє змоделювати досить складні відношення між різними сутностями.

У реляційних базах даних виділяють три типи зв'язків:

- ***«один-до-одного»*** – коли одному запису таблиці А відповідає не більше ніж один запис у таблиці В, і навпаки – одному запису таблиці В відповідає не більше ніж один запис у таблиці А.

Такий зв'язок використовують рідко, бо відповідні дані можна помістити в одну таблицю;

- ***«один-до-багатьох»*** – коли одному запису таблиці А може відповідати багато записів у таблиці В, але кожному запису таблиці В відповідає не більше ніж один запис у таблиці А. Сторона «один» у зв'язку «один-до-багатьох» називається головною таблицею, а сторона «багато» – зв'язаною таблицею;
- ***«багато-до-багатьох»*** – коли одному запису таблиці А може відповідати багато записів у таблиці В, і навпаки – одному запису таблиці В може відповідати багато записів у таблиці А.

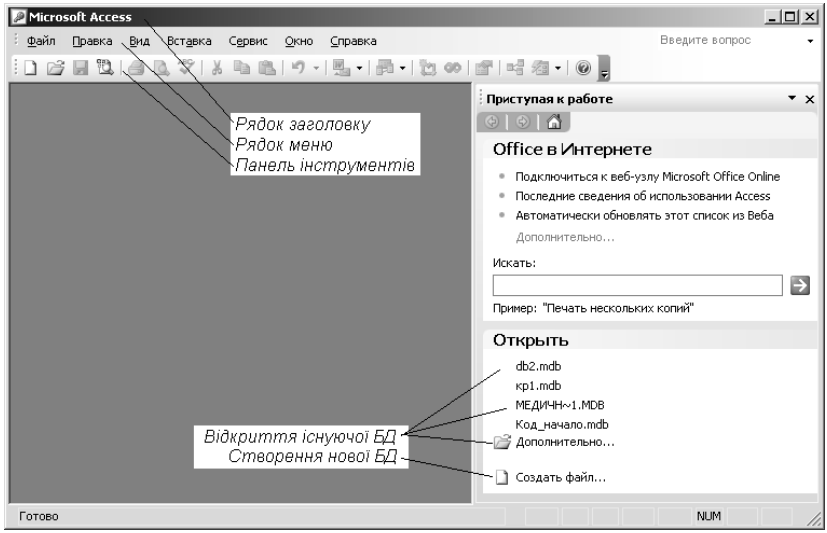
Наприклад, база даних для зберігання інформації про учнів може включати таблиці та зв'язки, схематично показані на малюнку. Як бачимо, база даних містить інформацію про такі сутності: учнів, навчальні предмети та оцінки. В таблиці *Учні* з ключовим полем *КодУчня* «Іванов» має унікальний код 1, і цей код визначає його в таблиці *Успішність*.

Кожен учень може отримати *багато* оцінок, але кожна оцінка належить лише *одному* учневі, тому між таблицями *Учні* та *Успішність* встановлено зв'язок «один-до-багатьох». Зв'язок такого ж типу є між таблицями *Предмети* та *Успішність*, адже з кожного предмета може бути *багато* оцінок, але кожна оцінка ставиться лише з якогось *одного* предмета. Через таблицю *Успішність* між таблицями *Учні* та *Предмети* встановлено зв'язок «багато-до-багатьох»: кожен учень може отримати оцінки з *багатьох* предметів, а з кожного предмета можуть отримати оцінки *багато* учнів.

СУБД Microsoft Access

Для запуску СУБД *Access* достатньо на робочому столі двічі клацнути на ярлику програми, після чого з'являється вікно СУБД *Microsoft Access* (*див. мал.*).

У вікні *Access*, крім рядків заголовка і меню, панелі інструментів, з'являється область завдань ***Приступаючи до роботи*** (*Приступая к работе*), засобами якої можна створити нову базу



даних або відкрити наявну.

Для виходу з програми користуються одним з таких способів:

- вибрати команду меню ***Файл*** ⇨ ***Вихід*** (*Выход*);
- клацнути на кнопці ✕ закриття вікна програми *Access*;
- натиснути комбінацію клавіш ***Alt + F4***.

Питання для самоконтролю (Тест ТЕМА-2-1):

1. *Яке призначення баз даних?*
2. *Що таке «структура бази даних»?*
3. *Що таке СУБД?*
4. *Які основні функції реалізує СУБД?*
5. *Які основні типи об'єктів містить база даних?*
6. *Для чого служать таблиці бази даних?*
7. *Для чого служать запити бази даних?*
8. *Для чого служать форми бази даних?*
9. *Для чого служать звіти бази даних?*
10. *Наведіть приклади сутностей, атрибутів та значень атрибутів.*

11. Поясніть слова: «реляційна база даних є відображенням моделі «сутність-зв'язок»».
12. Що містить поле двовимірної таблиці бази даних?
13. Що містить запис двовимірної таблиці бази даних?
14. Для чого служить ключове поле таблиці бази даних?
15. Які типи зв'язків можуть бути у реляційних базах даних?

3.2. Навчальна база даних «Борей»

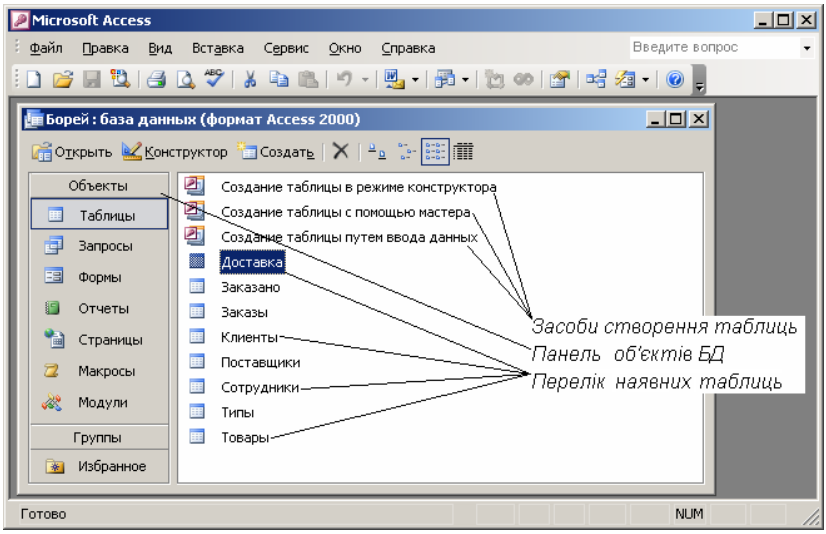
База даних *Борей* містить дані, які дозволяють тренуватися в роботі з її об'єктами: таблицями, запитами, формами, звітами тощо, що полегшує розуміння основних властивостей *Microsoft Access*.

Вона допомагає зрозуміти структуру реляційних баз даних і принципи взаємодії їх об'єктів, ілюструє процеси введення, зберігання, обробки і друкування даних.

Щоб відкрити базу даних *Борей,* потрібно скористатись командою меню ***Довідка*** ⇨ ***Приклади баз даних*** ⇨ ***Навчальна база даних «Борей»*** (*Справка Примеры баз данных...* ⇨ *Учебная база данных «Борей»*). Заставку слід закрити, клацнувши ***ОК***; головну кнопкову форму, що з'явиться, також закрити. На екрані залишиться вікно бази даних, в якому можна досліджувати її об'єкти.

Таблиці

На панелі об'єктів (*див. мал.*) вибрано режим роботи з таблицями. Як бачимо, інформація бази даних *Борей* зберігається у



8 таблицях. Праворуч знаходяться засоби для створення нових таблиць і піктограми існуючих таблиць (*Доставка*, *Заказано*, *Заказы*, *Клиенты*, *Поставщики* тощо).

Щоб відкрити таблицю, достатньо двічі клацнути на її піктограмі або виділити її піктограму і вибрати на панелі команду ***Відкрити*** *(Открыть).*

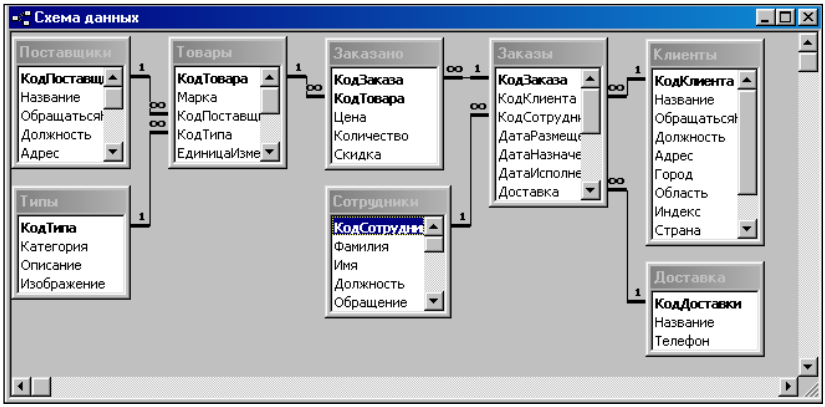
Сотрудники : таблица

	Код сотрудника	Фамилия	Имя	Должность	Дата рождения	Дата найма	Адрес	Город
+	1	Белова	Мария	Представитель	08-Грг-1968	01-Тра-1992	ул. Нефтяников, 14-4	Москва
+	2	Новиков	Павел	Вице-президент	19-Лют-1952	14-Сер-1992	Судостроительная ул., 12-245	Москва
+	3	Бабкина	Ольга	Представитель	30-Сер-1963	01-Кві-1992	Крещатик, 34-55	Киев
+	4	Воронова	Дарья	Представитель	19-Вер-1958	03-Тра-1993	ул. Пехотинцев, 1-34	Киев
+	5	Кротов	Андрей	Менеджер по продажам	04-Бер-1955	17-Жов-1993	Зеленый просп. 24-78	Москва
+	6	Акбаев	Иван	Представитель	02-Лип-1963	17-Жов-1993	Студенческая ул., 22-15	Москва
+	7	Кралев	Петр	Представитель	29-Тра-1960	02-Січ-1994	Сиреневый бульв. 11-11	Москва
+	8	Крылова	Анна	Внутренний координатор	09-Січ-1958	05-Бер-1994	Лесная ул. 12-456	Москва
+	9	Ясенева	Инна	Представитель	02-Лип-1969	15-Лис-1994	Родниковый пер. 1	Киев

Про вміст таблиці свідчить її назва. Наприклад, якщо відкрити таблицю *Сотрудники*, можна побачити дані про працівників.

Схема даних

Графічне подання зв'язків між таблицями (*див. мал.*) переглядають у вікні ***Схема даних*** (*Схема данных*), яке відображається після натискання відповідної кнопки панелі інструментів або за командою ***Сервіс*** ⇨ ***Схема даних*** (*Сервис* ⇨ *Схема данных*).



В даному випадку видно, що декілька пар таблиць зв'язані між собою зв'язком «один-до-багатьох» (*Поставщики-Товары*, *Сотрудники-Заказы*, *Поставщики-Заказано* (через таблицю *Товары*) тощо). Між деякими таблицями виникли зв'язки «багато-до-

багатьох»: *Сотрудники-Доставка* (через таблицю *Заказы*), *Товары-Клиенты* (через таблицю *Заказано*).

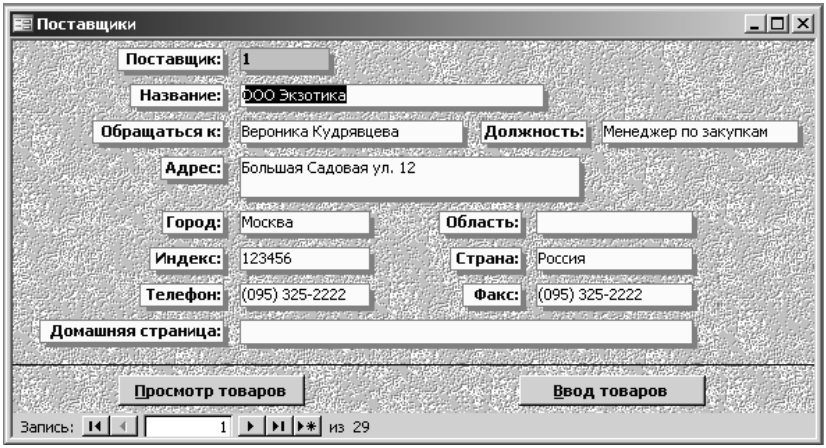
Запити

Запити в базі даних використовуються *для відбору даних із однієї або кількох зв'язаних таблиць*, пошуку даних за певними умовами, а також для обчислення підсумкових значень.

Деякі запити використовуються як джерело даних для форм і звітів. Інші слугують для проведення різноманітних операцій над даними. Результати виконання цих операцій можна переглянути у режимі таблиці запиту.

Якщо клацнути на кнопці ***Запити*** (*Запросы*), праворуч з'являться піктограми наявних запитів.

Для відкриття запиту достатньо двічі клацнути на відповідній піктограмі. Наприклад, при виконанні запиту *Товары по типам* одержимо таблицю, подібну до наведеної на малюнку.



Форми

Форми бази даних *Борей* демонструють зручні способи подання на екрані вмісту таблиць і результатів запитів.

Наприклад, форма *Поставщики* (*див. мал.* на попередній сторінці) дозволяє працювати із записами таблиці *Поставщики*. Тут можна переглянути чи доповнити записи про постачальників та перейти на іншу форму для перегляду списку товарів кожного з постачальників (*Просмотр товаров*) чи його доповнення (*Ввод товаров*).

Звіти

Для друкування кінцевих результатів у базі даних *Борей* є декілька звітів різної складності (*див. мал. вище*).

Хоч звіти призначені для виведення даних на друк, проте їх можна переглядати на екрані, зокрема, для того, щоб перед друком оцінити, як виглядатиме документ і, за потреби, внести зміни. Щоб переглянути звіт на екрані, слід на панелі об'єктів клацнути кнопку *Отчёты* і вибрати один зі звітів (наприклад, *Товары по типам)*.

Товары по типам

10-Кві-2004

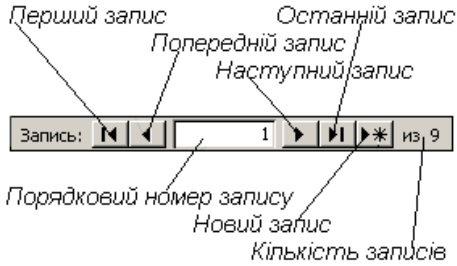
Категория: Кондитерские изделия	
Марка:	**На складе:**
Chocolade	15
Gumbar Gummibarchen	15
Maxilaku	10
NuNuCa Nuss-Nougat-Creme	76
Teatime Chocolate Biscuits	25
Valkoinen suklaa	65
Zaanse koeken	36
Число товаров: 13	

Категория: Молочные продукты	
Марка:	**На складе:**
Camembert Pierrot	22
Flotemysost	26
Geitost	112
Gorgonzola Telino	0
Gudbrandsdalsost	26
Queso Manchego La Pastora	86
Raclette Courdavault	79
Число товаров: 10	

Пошук інформації в базі даних

Пошук потрібної інформації в базі даних виконується кількома способами. Найпростіший з них – переглядаючи таблицю, шукати запис за його вмістом. Пересування по таблиці виконується, як і в електронних таблицях *Excel,* за допомогою клавіш ↓, ↑, ***Home*** і

End, комбінації клавіш ***Ctrl + Home*** і ***Ctrl + End***, а також за допомогою спеціальної панелі, що внизу таблиці (*див. мал.*). Перейти до запису з певним номером можна, якщо ввести цей номер у поле на панелі і натиснути ***Enter***.



Потрібні дані можна шукати так само, як і в *Word* чи *Excel*: за допомогою засобів ***Знайти*** (*Найти*) і ***Замінити*** (*Заменить*). Для зручності пошуку дані попередньо впорядковують.

Найбільш ефективним і гнучким є пошук за допомогою фільтрів та запитів. За допомогою фільтру *із наявної таблиці* вибираються дані, які задовольняють заданим умовам. При виконанні запиту *із однієї або кількох таблиць формується нова таблиця* із заздалегідь заданими полями та умовами відбору.

Питання для самоконтролю (Тест ТЕМА-2-1):

1. *Яке призначення баз даних?*
2. *Що таке СУБД?*
3. *Яке призначення і функції має СУБД?*
4. *Як поділяються БД за структурою?*
5. *В чому суть ієрархічної і мережної моделей БД?*
6. *У чому особливості реляційної моделі БД?*
7. *Які функціональні можливості має MS Access?*
8. *З яких об'єктів може складатися база даних Access?*
9. *Яким чином можна пересуватися по таблиці?*
10. *Як переглянути схему даних? Що на ній зображується?*
11. *Для чого використовується запит?*
12. *Для чого використовується форма?*
13. *Для чого використовується звіт?*
14. *За допомогою яких засобів проводиться пошук інформації у базі даних?*
15. *Чим відрізняється фільтр від запиту?*

3.3. Проектування бази даних

Проектування бази даних починається зі створення на папері її структури, при цьому виконуються такі роботи:

- проаналізувати предметну область і створити модель «сутність-зв'язок»;
- визначити перелік даних, які будуть зберігатися;
- визначити кількість і вміст таблиць для зберігання даних;
- визначити для таблиць назви полів, їх тип та ключові поля.

Розміщення даних у зв'язаних таблицях дозволяє позбавитись від даних, що повторюються.

Прикладом нераціональної «бази даних» може бути класний журнал, у якому на кожній сторінці повторюються однакові дані про кожного учня.

Повторення даних в різних таблицях знижує надійність бази даних в цілому: помилившись при введенні, наприклад, прізвища в одну з таблиць, надалі, під час пошуку, ми ризикуємо отримати неповну інформацію про людину. Краще застосувати декілька зв'язаних таблиць замість однієї великої: в одній розмістити прізвище, ім'я, по батькові кожного учня; в другій – перелік предметів, що вивчаються; в третій – поточні оцінки, дати опитування і коди учня та предмета з перших двох таблиць. В таких таблицях не буде даних, що повторюються.

Створення «порожньої» бази даних

Проектування БД можна виконати у режимах ***Майстри*** (*Мастера*), ***Сторінки й проекти бази даних*** (*Страницы и проекты базы данных*) і ***Нова база даних*** (*Новая база данных*).

Перший спосіб не потребує особливих умінь та знань: користувачу пропонуються на вибір зразки (шаблони) баз даних з порожніми таблицями, формами та звітами, які потрібно заповнити за допомогою спеціальних майстрів. Такий спосіб використовується рідко через низьку ефективність і громіздку структуру «спроектованої» бази даних.

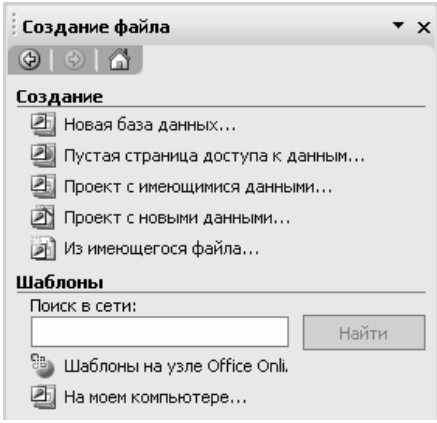
Другий спосіб вимагає розуміння основ проектування баз даних і дозволяє користувачу створити ефективну та оптимальну базу даних. Саме цей спосіб буде розглянуто далі.

Створення файлу порожньої (без таблиць) бази даних та поступове створення її об'єктів дозволяє користувачу самому визначити структуру бази даних і створити її елементи.

Для цього виконуються такі кроки:

- завантажити програму *Access*;
- в області завдань вибрати команду ***Створити файл*** (*Создать файл*), а потім – ***Нова база даних*** (*Новая база данных*);
- відкриється діалогове вікно ***Файл нової бази даних*** (*Файл новой базы данных*);
- у списку ***Папка*** вибрати папку для зберігання файлу БД;

- у полі ***Ім'я файлу*** (*Имя файла*) задати назву файлу (якщо цього не зробити, то новий файл отримає назву: db1, db2 тощо.)
- натиснути кнопку ***Створити*** (*Создать*).



Після виконання цих дій нова порожня база даних буде створена і відкриється її вікно, в якому можна розпочинати створювати її об'єкти: таблиці, запити, форми, звіти. Про це мова піде далі.

Приклад створення структури таблиць бази даних

Першим кроком при створенні бази даних є проектування таблиць, які міститимуть усі необхідні дані.

При створенні структури таблиці бази даних:

- описують поля (ім'я поля, тип даних, опис і властивості);
- зберігають структуру таблиці.

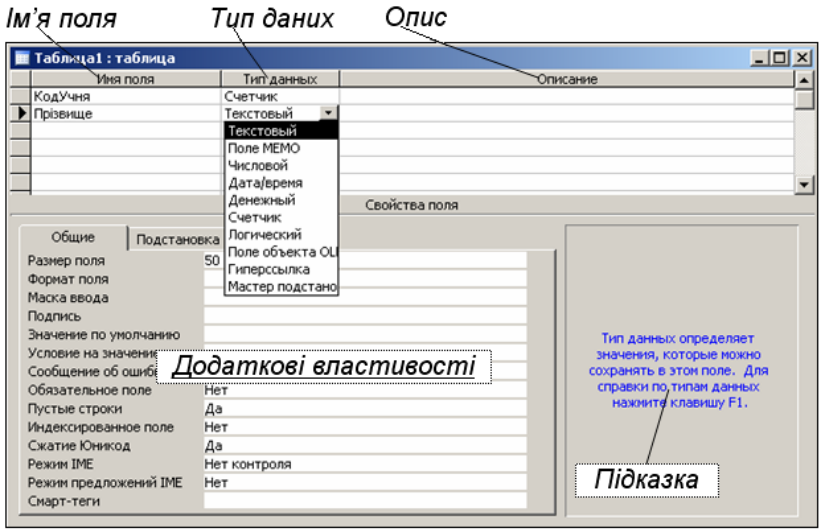
Щоб створити таблицю у режимі *Конструктора*, необхідно вибрати команду ***Створення таблиці в режимі конструктора*** (*Создание таблицы в режиме конструктора*) або натиснути кнопку ***Конструктор*** на панелі інструментів. Відкриється вікно (*див. мал.*), у якому будуть виконуватися всі дії зі створення структури таблиці.

Поле, в якому знаходиться курсор, (*див. Прізвище*) називається *поточним* і має зліва позначку ▸.

У кожному рядку Конструктора описують одне поле таблиці, що створюється.

Описуючи поле, слід вказати його ім'я та тип даних.

В іменах полів слід уникати пропусків, апострофа та інших спеціальних символів, оскільки це може створити проблеми при подальшому розвитку проекту.

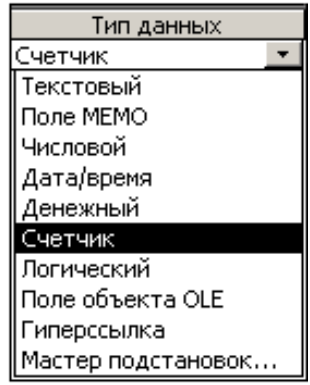


Можна додати опис поля (СУБД його ніяк не використовує). В нижній частині вікна перелічені інші властивості поля, які, при потребі, можна змінити. Створимо таблицю *Учні*, в якій будуть поля: *КодУчня*, *Прізвище*, *Імя*, *Адреса*, *РікНар* з такими типами і властивостями:

Ім'я поля	Тип даних	Додаткові властивості
КодУчня	Счётчик	за мовчазною згодою
Прізвище	Текстовый	розмір – 20; обов'язкове поле;
Імя	Текстовый	розмір – 10; обов'язкове поле;
Адреса	Текстовый	розмір – 45; необов'язкове поле;
РікНар	Дата/время	необов'язкове поле;

Поле створюють таким чином:

- у стовпчику ***Ім'я поля*** (*Имя поля*) ввести ім'я, наприклад, *КодУчня*;
- курсор перемістити на стовпчик ***Тип даних*** (*Тип данных*). У ньому з'явиться значення ***Текстовий*** (*Текстовий*) і кнопка, що відкриває список ▼;
- зі списку вибрати потрібний тип, наприклад, ***Лічильник*** (*Счётчик*);
- за потребою курсор перемістити на стовпчик ***Опис*** (*Описание*) і ввести опис;



- у нижній частині вікна задати потрібні властивості, якщо встановлені програмою не задовольняють вимогам.

Основні типи даних, що використовуються: ***Лічильник*** (*Счётчик*), ***Текстовий*** (*Текстовый*), ***Числовий*** (*Числовой*), ***Дата/час*** (*Дата/время*).

Для поля *КодУчня* пропонується використовувати тип ***Лічильник*** (*Счётчик*). Завдяки цьому, при введенні у таблицю даних про нових учнів, їхні номери будуть створюватися автоматично, причому кожний новий номер буде відрізнятися від наявних у таблиці.

Для поля *Прізвище*, вказавши тип ***Текстовий***, потрібно змінити властивості: ***Розмір поля*** (*Размер поля*) – з 50 на 20 (навряд чи будуть прізвища довші, ніж 20 символів, а пам'ять комп'ютера обмежена і її слід економити); у полі ***Обов'язкове поле*** (*Обязательное поле*) відкрити список і вибрати ***Так*** (*Да*).

При встановленні типу даних ***Дата/час*** (*Дата/время*) бажано в нижній частині вікна відкрити вкладку ***Загальні*** (*Общие*), встановити вказівник на поле ***Формат поля***, відкрити список і вибрати потрібний формат (*див. мал.*).

Полный формат даты	19.06.1994 17:34:
Длинный формат даты	19 червня 1994 р.
Средний формат даты	19-Чер-94
Краткий формат даты	19.06.1994
Длинный формат времени	17:34:23
Средний формат времени	5:34
Краткий формат времени	17:34

Наприклад, ***Короткий формат дати***: 12.04.2008.

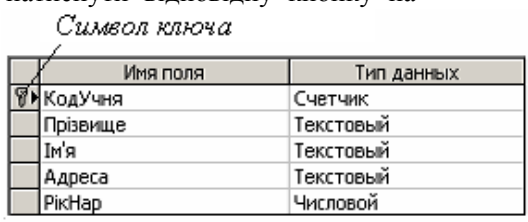
Первинний ключ

Ключів у таблиці може бути декілька. З них вибирається один (як правило, найкоротший), який надалі буде представляти кожний запис таблиці. Такий ключ називається первинним.

Наприклад, якщо в таблиці *Учні* за первинний ключ взяти *Прізвище*, то це означає, що не повинно бути учнів з однаковими прізвищами. Але ж прізвища можуть повторюватися, тому доведеться доповнювати ключ ще й іменем тощо. Первинний ключ стає довгим і незручним для використання. Оскільки він служить не лише для того, щоб відрізняти один запис від іншого, але й для організації зв'язків між таблицями, доцільно включити в таблицю додаткове поле *КодУчня*, і саме його використати як *первинний ключ*.

Для створення первинного ключа слід:

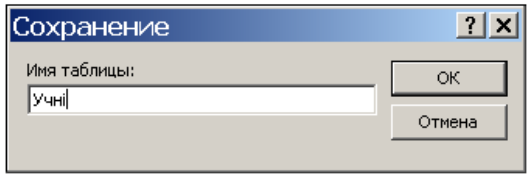
- виділити поле, яке буде ключовим (у нас *КодУчня*);
- вибрати команду меню ***Правка*** ⇨ ***Ключове поле*** (*Ключевое поле*) або натиснути відповідну кнопку на панелі інструментів – зліва від імені виділеного поля з'явиться символ ключа – ознака того, що дане поле є ключовим.



Після заповнення всіх полів вікно конструктора набуде наведеного на малюнку вигляду (поля *Опис* та *Властивості поля* не показані). Послідовно виділяючи поля ***Тип даних*** (*Тип данных*), слід перевірити, чи властивості відповідають завданню (*див. вище*).

Зберігають структуру таблиці так:

- вибрати команду меню ***Файл*** ⇨ ***Зберегти*** (*Сохранить*) або натиснути відповідну кнопку на панелі інструментів;



- у діалоговому вікні, що відкриється, ввести ім'я таблиці (до 64 символів; автоматично пропонується ім'я *Таблица1*);
- натиснути кнопку ***ОК***.

За такою ж методикою створюються та зберігаються інші таблиці бази даних, а саме:

- таблиця *Предмети*:

КодПред	*Счётчик*	за мовчазною згодою;
НазваПред	*Текстовый*	розмір – 15; обов'язкове поле;
Учитель	*Текстовый*	розмір – 20; обов'язкове поле;

- таблиця *Успішність*:

КодОцінки	*Счётчик*	за мовчазною згодою;
КодУчня	*Числовой*	обов'язкове поле;
КодПред	*Числовой*	обов'язкове поле;
Оцінка	*Числовой*	обов'язкове поле;

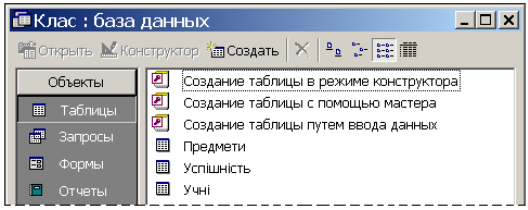
Оцінка не повинна перевищувати 12, тому, для захисту від помилки при введенні, потрібно ввести обмеження:

- у режимі *Конструктора* у таблиці *Успішність* у стовпчику ***Тип даних*** (*Тип данных*) виділити клітинку в рядку *Оцінка*;

- у нижній частині вікна (***Властивості поля*** (*Свойства поля*)) вибрати вкладку ***Загальні*** (*Общие*);
- у рядку ***Умова*** (*Условие на значение*) ввести <13.

Тепер при спробі ввести недопустиме значення оцінки на екрані з'явиться повідомлення про порушення умов на значення поля і пропозиція виправити помилку.

Після створення структури таблиць вікно бази даних *Клас* набуде наведеного на малюнку вигляду: у вікні з'являться значки поки що порожніх таблиць *Предмети*, *Успішність*, *Учні*.



Потрібно ввести по декілька записів у кожну таблицю. На прикладі цих записів перевіряється, наскільки правильно створена база даних і чи відповідає вона поставленим вимогам.

Введення даних у режимі таблиці

Використання режиму таблиці є найпростішим способом введення даних і виконується так:

- подвійним клацанням на значку відкривається таблиця;
- послідовно вводяться дані в поля запису, з натисканням щоразу клавіші ***Tab*** або ***Enter***;
- у поле, яке має тип ***Лічильник*** (*Счётчик*), числа вводяться автоматично, кожного разу збільшуючись на 1;
- поля, які визначені як необов'язкові, можна залишати порожніми;
- після натискання клавіші ***Tab*** або ***Enter*** в останньому полі запису курсор переходить на початок наступного запису, поля якого заповнюються в такій самій послідовності.

Перехід до наступного запису заблоковано, якщо не заповнено хоча б одне обов'язкове поле.

Після введення перших записів таблиця *Учні* матиме вигляд:

Учні : таблица

КодУчня	Прізвище	Ім'я	Адреса	РікНар
1	Іванов	Віталій	вул. Грушова, буд. 32	12.09.1987
2	Токаренко	Леонід	вул. Шкільна, буд. 8	08.08.1986

Збереження кожного запису відбувається автоматично після переходу до наступного.

Зліва від першого поля таблиці знаходиться область вибору запису, в якій з'являються піктограми, що вказують його стан: ▶ – даний запис поточний (вибраний); ✎ – у запис вводяться дані.

При введені даних у таблицю *Успішність* виникають певні труднощі: в ній замість прізвищ учнів і назв предметів треба вказувати їхні числові коди згідно з таблицями *Учні* і *Предмети*. Виглядатиме це так, як показано на малюнку. Звичайно, було б краще, якби замість кодів відображались відповідні їм текстові дані. Це стає можливим після зв'язування таблиць і виконання підстановок (*див. далі*).

Успішність : таблиця

КодОцінки	КодУчня	КодПред	Оцінка
1	1	1	7
2	1	2	9
3	2	1	10
4	2	2	8

Копіювання і переміщення даних

Копіювання даних із одного поля в інше виконується так:

- виділити потрібне поле;
- вибрати команду ***Копіювати*** (*Копировать*) або ***Вирізати*** (*Вырезать*) з панелі інструментів або з контекстного меню;
- клацнути в полі, куди копіюються або переміщаються дані;
- вибрати команду ***Вставити*** (*Вставить*).

Інформація з'явиться у новому полі і залишиться (при копіюванні) або зникне (при вирізанні) з попереднього поля.

Скасування виконаних дій

Залежно від ситуації, скасовують дії такими способами:

- щойно введені дані (до переходу в наступне поле) – клавіша ***Esc***; повторне натискання ***Esc*** скасовує всі щойно зроблені зміни у записі (до переходу до іншого запису);
- виконана дія – кнопка ***Скасувати*** (*Отменить*) на панелі інструментів;
- всі дані в щойно введеному записі – команда ***Правка*** ⇨ ***Скасувати запис*** (*Отменить запись*) або комбінація клавіш ***Ctrl + Z***.

Питання для самоконтролю (Тест ТЕМА-2-2):

1. *Як створити «порожню» базу даних?*
2. *Яким чином зберегти у власній папці нову базу даних?*

3. *Які елементи має таблиця бази даних?*
4. *Що містить запис таблиці? Що таке поле таблиці БД?*
5. *Що таке «первинний ключ» таблиці БД? Для чого він призначений?*
6. *Які типи даних використовуються в базі даних?*
7. *Коли і для чого використовується тип даних Лічильник?*
8. *Як створити структуру бази даних у режимі Конструктора?*
9. *Як встановити тип даних поля таблиці БД?*
10. *Як вибрати потрібний формат виведення дати або часу?*
11. *Як створити у певному полі первинний ключ?*
12. *Які дії потрібно виконати для збереження структури БД?*
13. *За допомогою яких клавіш можна переміщатися таблицею БД?*
14. *Які дії і яким чином можна відмінити при заповненні таблиці БД?*
15. *Як скасувати щойно виконане введення запису БД?*

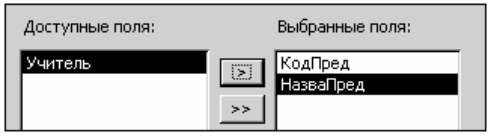
3.4. Зв'язування таблиць

Виконання підстановок

Виставляти оцінки в таблиці *Успішність* незручно: замість прізвищ учнів і назв предметів вказуються їх коди. *Access* дозволяє, не змінюючи структури бази даних, виводити замість кодів інформацію у зв'язаній таблиці в текстовому вигляді.

Щоб у таблиці *Успішність* замість кодів предметів автоматично з'являлися відповідні назви, потрібно налаштувати їх підстановку з таблиці *Предмети*:

- відкрити таблицю *Успішність* в режимі *Конструктора*;
- вибрати поле *КодПред*;
- перейти у стовпець *Тип данных* і зі списку вибрати команду ***Майстер підстановок…*** (*Мастер подстановок...*);
- у вікні ***Створення підстановки*** (*Создание подстановки*) клацнути на кнопці ***Далі*** (*Далее*);
- вибрати таблицю *Предмети* і клацнути на кнопці ***Далі*** (*Далее*);
- у наступному вікні зі списку полів таблиці *Предмети* вибрати назву поля *КодПред* і клацнути на кнопці >, те ж саме виконати для поля *НазваПред*. Ці назви повинні з'явитися в полі ***Обрані поля*** (*Выбранные поля*). Клацнути ***Далі*** (*Далее*);



- у наступному вікні двічі клацнути в полі назви предмета або перетягуванням підібрати ширину підставленого стовпця і клацнути на кнопці ***Далі*** (*Далее*);
- у вікні з фінішним прапорцем клацнути кнопку ***Готово***.

З’явиться вікно ***Створення підстановки*** (*Создание подстановки*), де натиснути ***Так*** (*Да*);

- закрити вікно *Конструктора*.

Після виконання вказаних дій таблиця *Успішність* набуде наведеного на малюнку вигляду: замість числових кодів виводяться назви відповідних предметів з таблиці *Предмети*.

Успішність : таблиця

КодОцінки	КодУчня	КодПред	Дата	Оцінка
7	7	іноземна мова	15.01.2008	7
8	1	інформатика	22.01.2008	12
9	2	фізика	24.01.2008	2
10	5	фізкультура	28.01.2008	5
(Счетчик)	0			0

Створення зв’язків між таблицями

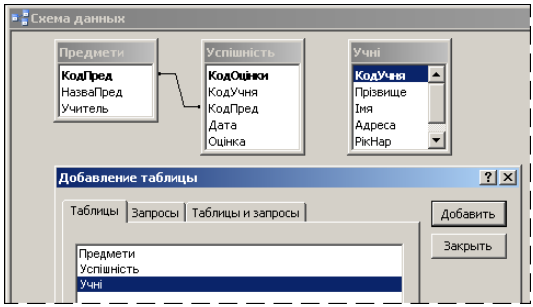
Перед створенням зв’язків слід налаштувати бажані підстановки між відповідними полями.

Зв’язок «один-до-багатьох» встановлюють так:

- відкрити базу даних;
- вибрати команду меню ***Сервіс*** (*Сервис*) ⇨ ***Схема даних*** (*Схема данных*) або натиснути відповідну кнопку на панелі інструментів;

- у вікні ***Додавання таблиці*** (*Добавление таблицы*), яке з’явиться, послідовно виділяти назву таблиці і натискати кнопку ***Додати*** (*Добавить*) – відповідна таблиця буде з’являтися у вікні ***Схема даних*** (якщо вікно ***Додавання таблиці*** не з’являється, натиснути кнопку ***Відобразити таблицю*** (*Отобразить таблицу*) на панелі інструментів); інший спосіб – перетягти потрібну таблицю із вікна БД у вікно схеми даних;

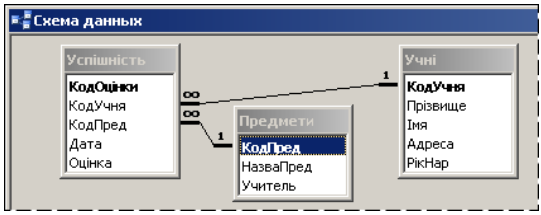
- натиснути кнопку ***Закрити*** (*Закрыть*) – на екрані залишиться вікно ***Схема даних***. Зверніть увагу, що поля, для яких виконана підстановка, сполучені лінією;



- для встановлення зв’язку потрібно виділити ключове поле *КодУчня* головної

таблиці *Учні*, перетягти його на таблицю *Успішність*, встановити на поле з такою ж назвою і відпустити – з'явиться діалогове вікно ***Зміна зв'язків*** (*Изменение связей*);

- у цьому вікні встановити прапорець ***Забезпечення цілісності даних*** (*Обеспечение целостности данных*), а також прапорці ***Каскадне оновлення зв'язаних полів*** (*Каскадное обновление связанных полей*), щоб при зміні значення поля зв'язку в головній таблиці автоматично змінювалося значення у відповідному полі в підлеглій, і ***Каскадне видалення зв'язаних полів*** (*Каскадное удаление связанных записей*) (при видаленні запису із головної таблиці будуть видалені зв'язані записи в підлеглих таблицях);



- натиснути кнопку ***Створити*** (*Создать*) – в схемі даних з'являться лінії, які показують тип зв'язку (тут «*один-до-багатьох*»).

Зв'язок між таблицями *Предмети* і *Успішність*, що виник при виконанні підстановки, також слід налаштувати для забезпечення цілісності даних. Для цього лінію зв'язку клацають правою кнопкою і вибирають команду ***Змінити зв'язок*** (*Изменить связь*) – з'явиться описане вище вікно ***Зміна зв'язків***.

Для збереження схеми даних потрібно вибрати команду ***Зберегти*** (*Сохранить*) і відповісти ***Так*** (*Да*) при появі відповідного вікна.

Зовнішній ключ

У наведеному прикладі таблиці *Учні* і *Предмети* є головними стосовно таблиці *Успішність*, ключове поле головної таблиці зв'язується із відповідним полем зв'язаної таблиці.

У цьому випадку поле зв'язаної таблиці називають її ***зовнішнім ключем***.

Таблиця *Успішність* має два зовнішні ключі: поле *КодУчня* і поле *КодПред*. Очевидно, що до зовнішнього ключа не ставиться вимога унікальності. Адже кожному запису головної таблиці (наприклад, *Предмети*) у зв'язаній таблиці може відповідати більше ніж один запис, і всі вони матимуть однакове значення зовнішнього

ключа (тобто однаковий код предмету). Саме зовнішні ключі забезпечують каскадне оновлення та видалення зв'язаних полів.

Редагування таблиці

Треба розрізняти два види редагування: вмісту таблиці і структури таблиці.

При редагуванні *вмісту таблиці* користуються прийомами, засвоєними під час роботи в текстовому процесорі *Word* (вставка і вилучення символів, використання буфера обміну тощо). При переміщенні по клітинках за допомогою клавіш керування курсором, щоб почати редагування вмісту клітинки, треба натиснути ***F2*** або клацнути в клітинці.

Для видалення запису (рядка) потрібно його виділити (клацнути область вибору, коли вказівник набуде вигляду ➡) і вибрати команду *Правка* ⇨ ***Видалити запис*** (*Удалить запись*) або натиснути клавішу ***Del***.

Змінюють структуру таблиці (назви і властивості полів) в режимі Конструктора.

Для редагування *структури таблиці* виконують такі дії:

- відкрити потрібну таблицю, двічі клацнувши на її значку;
- перейти в режим *Конструктора*, клацнувши на значку, що на панелі інструментів.

У режимі *Конструктора* вносяться потрібні виправлення шляхом зміни:

- ***назви поля*** – виділити і редагувати як звичайний текст;
- ***типу даних*** – відкрити список ***Тип даних*** (*Тип данных*) праворуч від назви поля і вибрати в ньому потрібне;
- ***властивостей*** – внести зміни в нижній частині вікна.

Питання для самоконтролю (Тест ТЕМА-2-4):

1. *Яким чином виконується зв'язування таблиць бази даних?*
2. *Як виконується додавання таблиці БД у вікно Схема даних?*
3. *Яку кнопку слід натиснути, щоб з'явилось вікно Додавання таблиці?*
4. *Після яких дій з'являється вікно Зміна зв'язків?*
5. *Які дії потрібно виконати для створення зв'язків між таблицями БД?*
6. *Опишіть призначення прапорців у вікні Зміна зв'язків?*
7. *Що дає режим Каскадне оновлення зв'язаних полів?*
8. *Як будується зв'язок «один-до-багатьох» між таблицями БД?*
9. *Коли і для чого в БД виконуються підстановки?*
10. *Які дії потрібно виконати для створення підстановок?*
11. *Якого вигляду набудуть таблиці після виконання підстановок в БД?*
12. *Які два види редагування проводяться над таблицями БД?*

13. Як відредагувати вміст таблиці БД?
14. Як змінити структуру таблиці БД?
15. Які виправлення в таблиці виконують за допомогою Конструктора?

3.5. Впорядкування, пошук та фільтрація даних

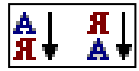
Впорядкування (сортування) даних

При введенні даних у таблицю записи розташовуються у такому порядку, в якому вони вводилися. Це не завжди зручно при перегляді вмісту таблиці. Бажано дані згрупувати і впорядкувати за певними ознаками, щоб у них було легко орієнтуватися. Наприклад, розташувати прізвища за алфавітом або упорядкувати записи за зменшенням років народження.

При впорядкуванні записи розташовуються в новому порядку відповідно до даних вибраного поля.

Впорядкування даних проводиться так:

- відкрити таблицю для проведення впорядкування даних;
- встановити курсор на поле (стовпчик), за даними якого треба провести впорядкування;
- на панелі інструментів натиснути одну із кнопок ***Сортувати за зростанням*** (*Сортировка по возрастанию*) або ***Сортувати за спаданням*** (*Сортировка по убыванию*). Такі ж команди можна вибрати в меню *Записи* ⇨ ***Сортування*** (*Сортировка*) або за допомогою контекстного меню;



- для сортування записів за даними декількох *суміжних* полів необхідно виділити відповідні стовпчики і провести впорядкування.

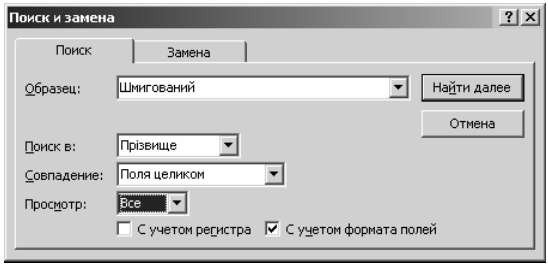
Пошук даних за зразком

Пошук даних за зразком проводиться у такій послідовності:

- відкрити таблицю;
- встановити курсор на довільну клітинку стовпчика, де буде проводитися пошук;
- вибрати команду *Правка* ⇨ ***Знайти*** (*Найти*) або натиснути кнопку ***Знайти*** (*Найти*) на панелі інструментів;

- у вікні, що з'явиться, вибрати вкладку ***Пошук*** (*Поиск*);
- у поле ***Зразок:*** (*Образец:*) ввести зразок розшукуваних даних;

- поле ***Шукати в:*** (*Поиск в:*) містить назву стовпчика, де буде проводитися пошук. Для проведення пошуку в усій таблиці слід відкрити відповідний список і вибрати назву таблиці;



- зі списку ***Збіг:*** (*Совпадение:*) вибрати, яка частина поля має збігатися зі зразком: все поле (*Поля целиком*), довільна частина поля (*С любой частью поля*) чи початок поля (*С начала поля*);
- встановивши прапорець ***Враховувати регістр*** (*С учётом регистра*), обмежують пошук лише тими полями, що збігаються з текстом зразка не лише за змістом, але й за регістром (наприклад, за зразком «Клас» НЕ БУДЕ знайдене поле зі словом «клас»);
- встановивши прапорець ***Враховувати формат*** (*С учётом формата полей*), обмежимо пошук тими полями, що відповідають зразку на вигляд (наприклад, за зразком 15/11/09 НЕ БУДЕ знайдена дата, яка була введена саме так, але на екрані має вигляд 15.11.2009);
- натиснути кнопку ***Знайти далі*** (*Найти далее*) – в таблиці буде виділений текст, який збігається із заданим зразком;
- для продовження пошуку натиснути кнопку ***Знайти далі***.

Якщо за заданим зразком не буде знайдено жодного запису, з'явиться повідомлення «*Поиск ... завершён. Образец не найден*».

За допомогою кнопки *Больше>>* можна уточнити напрям пошуку у вікні, що відкриється.

Для забезпечення більшої гнучкості пошуку, записуючи зразок, використовують маски:

* – відповідає будь-якій кількості символів, використовується на початку або в кінці зразка;

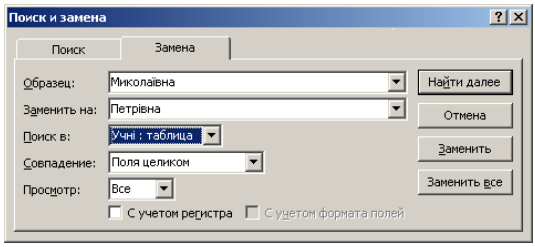
? – будь-який ОДИН символ;

– будь-яка ОДНА цифра.

Наприклад, для пошуку прізвищ, що закінчуються на «енко», використовується маска «*енко».

Пошук і заміна даних

Пошук і заміна даних проводиться у такій послідовності:



- встановити курсор на довільну клітинку стовпчика, де буде виконуватися пошук і заміна;
- вибрати команду ***Правка*** ⇨ ***Замінити*** (*Заменить*) або у вікні для пошуку вибрати вкладку ***Заміна*** (*Замена*);
- у поле ***Зразок:*** (*Образец:*) ввести зразок для пошуку і заміни;
- у поле ***Замінтити на:*** (*Заменить на:*) ввести текст, що замінить знайдений текст зразка;
- призначення інших полів та прапорців таке ж, як при пошуку у таблиці;
- у полі ***Перегляд:*** (*Просмотр*) уточнюють область пошуку: всі записи (*Всі*), вище (*Вгору*) чи нижче (*Вниз*) від активного запису;
- натиснути кнопку *Найти далее* – в таблиці виділиться текст, який збігається зі зразком;
- натиснути кнопку (*Заменить*) для виконання заміни або ***Знайти далі*** (*Найти далее*), щоб її не виконувати і шукати далі;
- для виконання відразу всіх замін слід клацнути кнопку ***Замінити всі*** (*Заменить все*).

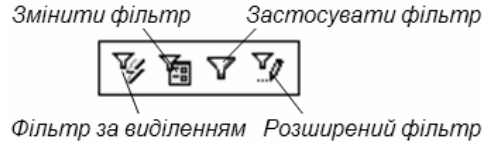
Фільтри

При упорядкуванні даних змінюється розташування записів, проте не зменшується кількості тих, які доводиться переглядати. Більш зручним засобом для перегляду потрібних записів є фільтри. Використання фільтрів дозволяє у великих таблицях знаходити і виводити тільки ті дані, які потрібні для перегляду і аналізу.

Є три види фільтрів:

- ***фільтр за виділеним зразком*** – відбір даних, які містять у своєму складі виділений фрагмент;
- ***простий фільтр*** – відбір даних відповідно до заданого вмісту поля;
- ***розширений фільтр*** – відбір даних за спеціально оформленими складними умовами пошуку.

Фільтр за виділеним зразком та простий фільтр створюють безпосередньо у вікні таблиці, а для створення розширеного фільтра відкривається спеціальне вікно *Конструктора фільтра* (*див. далі*).



Для роботи з фільтрами на панелі інструментів таблиці є командні кнопки, дію яких розглянемо далі.

Знищення фільтра

Перед створенням будь-якого фільтра потрібно впевнитися у тому, що для даної таблиці не встановлено умови фільтрації. Якщо на таблицю не накладено жодних умов відбору, то кнопка ***Застосувати фільтр*** (*Применить фильтр*) буде неактивна.

Для знищення фільтра необхідно на панелі інструментів вибрати команду ***Змінити фільтр*** (*Изменить фильтр*), натиснути кнопку ✕ (*Очистить бланк*) і потім натиснути кнопку ***Застосувати фільтр***. Після цього потрібно зберегти таблицю, натиснувши кнопку ***Зберегти*** (*Сохранить*).

Фільтр, збережений з цією таблицею раніше, буде знищено, а кнопка ***Застосувати фільтр*** стане неактивною.

Використання фільтра за виділеним зразком

Для створення найпростішого і швидкого засобу відбору даних необхідно виконати такі дії:

- відкрити таблицю;
- виділити елемент, за яким буде проводитись фільтрація;
- вибрати команду меню ***Записи*** ⇨ ***Фільтр*** (*Фильтр*) ⇨ ***Фільтр за виділеним*** (*Фильтр по выделению*) або натиснути відповідну кнопку на панелі інструментів.

Як наслідок, відбудеться фільтрація і в таблиці будуть показані тільки ті записи, які містять дані, що збігаються з виділеними.

Можна здійснити фільтрацію за умовою «не містить виділеного значення», для чого необхідно вибрати команду меню ***Записи*** ⇨ ***Фільтр*** (*Фильтр*) ⇨ ***Виключити виділене*** (*Исключить выделенное*).

Фільтр не змінює даних у таблиці, а лише приховує ті, які не цікавлять користувача.

Скасовують дію фільтра (не знищують фільтр!), щоб побачити всі записи таблиці, за допомогою команди ***Записи*** ⇨ ***Видалити фільтр*** (*Удалить фильтр*) або натиснувши відповідну кнопку на панелі інструментів.

Фільтр можна в будь-який момент застосувати повторно командою *Записи* ⇨ ***Застосувати фільтр*** (*Применить фильтр*) або повторно натиснувши ту ж кнопку на панелі інструментів. Ця кнопка фіксується в одному з положень: ***Видалити фільтр*** або ***Застосувати фільтр***.

Використання простих фільтрів

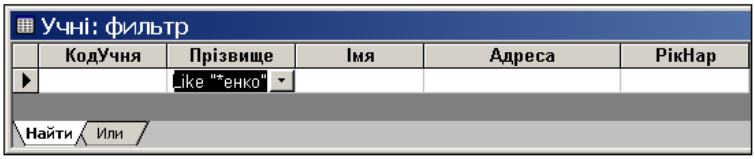
Для створення звичайного фільтра необхідно виконати дії:

- відкрити потрібну таблицю;
- вибрати команду *Записи* ⇨ ***Фільтр*** (*Фильтр*) ⇨ ***Змінити фільтр*** (*Изменить фильтр*) або натиснути відповідну кнопку на панелі інструментів;

- у вікні, що відкриється, будуть перелічені умови, задані для поточного фільтра.

Перед створенням нового фільтра потрібно знищити попередній, інакше результатом фільтрації буде виконання умов двох (і більше) фільтрів.

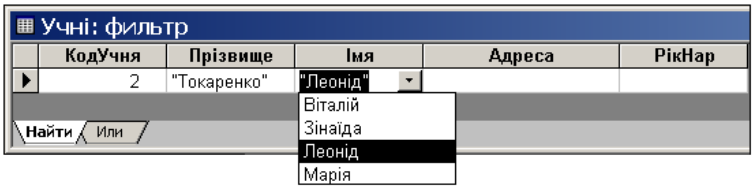
Якщо, наприклад, в таблиці *Учні* раніше використали фільтр за виділеним зразком для пошуку учнів, у яких прізвища закінчуються на «енко», то конструктор простого фільтра матиме вигляд.



Оператор *Like* (англ. *такий, як…*; *подібний до…*) з'являється у конструкторі автоматично після виконання будь-якого фільтрування. Щоб знищити цей фільтр, потрібно виконати вищеописані дії.

У верхній частині вікна розташовано рядок з назвами всіх полів вибраної таблиці. Другий рядок містить умови фільтрації. У цьому рядку можна змінити наявні умови або ввести нові. Умови відбору можуть вводитися з клавіатури або зі списку вибору, який відкривається при клацанні кнопки ▼, що справа у відповідному полі.

Наприклад, щоб знайти запис з прізвищем «Токаренко», потрібно відкрити список у полі *Прізвище* і вибрати зі списку потрібне. Після натискання кнопки ***Застосувати фільтр*** (*Применить фильтр*) у таблиці будуть показані тільки записи з прізвищем «Токаренко».



Можна проводити фільтрацію за кількома ознаками. Нехай потрібно знайти запис зі значеннями «Токаренко Леонід». Необхідно в полі *Прізвище* вибрати зі списку «Токаренко», перейти в поле *Ім'я* і вибрати «Леонід».

При встановленні умов фільтрації можна користуватися масками з використанням символів *, ? і #, як було описано вище.

Умови, введені в одному рядку для кількох полів, розглядаються як об'єднані операцією ***І*** (AND).

Наприклад, для відбору всіх учнів з прізвищами на букву «В» і 1988 року народження потрібно ввести для поля *Прізвище* умову **В*** і для поля *РікНар* – ***1988**.

У нижній частині вікна розташовані вкладки ***Знайти*** (*Найти*) та ***Або*** (*Или*). Вкладка ***Або*** (*Или*) призначена для введення іншого набору умов фільтрації. У рядку на цій вкладці вводять умови, які будуть зв'язані з попередніми умовами операцією ***Або*** (OR).

Наприклад, щоб відібрати крім учнів, вказаних у попередніх умовах, ще й таких самих, але 1989 року народження, потрібно на вкладці *Или* ввести для поля *Прізвище* умову **В*** і для поля *РікНар* – **1989**. При цьому з'явиться ще одна вкладка *Или* і т.д., що дозволяє розширювати умови пошуку.

Умови фільтрації можуть містити вирази з використанням операцій < (менше), > (більше), <> (не дорівнює), AND (і), OR (або), NOT (не; відмінний від вказаного).

Наприклад, для одержання попереднього результату можна було на вкладці ***Знайти*** для поля *РікНар* ввести вираз: *1988 OR *1989.

Для застосування фільтра за новими або відредагованими умовами потрібно вибрати команду *Записи* ⇨ ***Застосувати фільтр*** (*Применить фильтр*) або натиснути відповідну кнопку на панелі інструментів.

Щоб скасувати дію фільтра і побачити всі записи таблиці, вибирають команду *Записи* ⇨ ***Видалити фільтр*** (*Удалить фильтр*) або натискають ту ж саму кнопку на панелі інструментів.

Використання розширеного фільтра

У розширеному фільтрі зберігаються встановлені до його використання умови фільтрації і сортування.

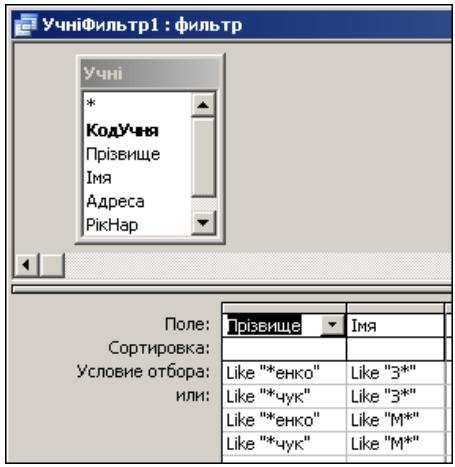
Ці умови можна доповнити або вилучити і створити новий фільтр. Для знищення фільтра потрібно виконати вищеописані дії.

Для створення розширеного фільтра виконують такі кроки:

- відкрити потрібну таблицю;
- вибрати команду ***Записи*** ⇨ ***Фільтр*** (*Фильтр*) ⇨ ***Розширений фільтр*** (*Расширенный фильтр*) – відкриється вікно наведеного вигляду.

У верхній частині вікна відображається список полів поточної таблиці (і, можливо, головних таблиць для полів з підстановками). В нижній частині вікна відображається *бланк запиту*, в якому задаються умови відбору записів. В рядку *Поле* вводяться назви полів, для яких будуть встановлюватися умови фільтрації. Вводити назви полів можна шляхом вибору зі списку, який відкривається після натискання на кнопку ▼ (*див. мал.*).

У рядку ***Сортування*** (*Сортировка*) можна вибрати тип сортування для одного або кількох полів. У рядку ***Умова відбору*** (*Условие отбора:*) вводяться умови фільтрації, в рядку ***Або*** (*Или*) – додаткові умови.



Наприклад, щоб відібрати учнів з прізвищами, що закінчуються на «енко» або «чук», з іменами, що починаються на літери «М» або «З», потрібно виконати такі дії:

- у першій клітинці рядка ***Поле*** (*Поле*) вибрати зі списку *Прізвище*;
- у відповідній клітинці рядка ***Умова відбору*** записати «*енко»;

- в рядку ***Поле*** в наступній клітинці вибрати із списку *Ім'я*;
- у відповідній клітинці рядка ***Умова відбору*** записати «З*»;
- нижче у клітинках рядка *или* та наступних записати ще три варіанти умов пошуку (*див. мал.*).

*Умови, записані в одному рядку, об'єднуються операцією **І** (тобто діють одночасно). Після цього умови з різних рядків об'єднуються операцією **АБО**.*

Для застосування фільтра досить натиснути кнопку ***Застосувати фільтр*** (*Применить фильтр*) на панелі задач. Можливий результат показано на малюнку.

Учні : таблиця

КодУчня	Прізвище	Ім'я	Адреса	РікНар
3	Бондаренко	Марія	пр-кт Миру, буд. 10, кв. 58	06.07.1984
5	Бондарчук	Зінаїда	вул. Тургенєва, буд. 31	21.02.1987

Збереження фільтра

Створені фільтри завжди зберігаються автоматично при збереженні таблиці. При повторному відкритті таблиці збережений фільтр є поточним і може бути викликаний командою *Записи* ⇨ ***Застосувати фільтр*** (*Применить фильтр*).

Якщо створюється новий фільтр, він заміняє фільтр, що був збережений з таблицею.

Питання для самоконтролю (Тест ТЕМА-2-5):

1. *У чому полягає операція впорядкування даних в таблиці БД?*
2. *Яким чином проводиться впорядкування даних в таблиці БД?*
3. *У чому полягає операція пошуку даних за зразком в таблиці БД?*
4. *Яке застосування в масці фільтра мають символи «*», «?», «#»?*
5. *У якій послідовності проводиться пошук і заміна даних?*
6. *Як провести пошук і заміну даних за вибором?*
7. *Для чого використовується фільтрація даних?*
8. *Які види фільтрів застосовуються в БД?*
9. *Які кнопки має панель інструментів для роботи з фільтрами?*
10. *Як знищити фільтр, що зберігається з таблицею?*
11. *Як виконується відбір даних за виділеним зразком?*
12. *Як скасувати дію фільтра? Як поновити його дію?*
13. *Які дії потрібно виконати для створення простого фільтра?*
14. *Яке призначення має вкладка Або при використанні фільтра?*
15. *Як зберігають створений фільтр?*

3.6. Створення запитів

Більш гнучкий доступ до інформації у базі даних забезпечується за допомогою запитів. Запити дозволяють відібрати дані, які зберігаються в різних таблицях, а також здійснювати відбір згідно із заданими умовами.

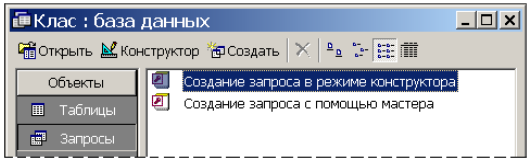
При відкриванні запиту на екрані відображається таблиця. Проте слід мати на увазі, що запит, як об'єкт бази даних, на відміну від таблиці, не містить в собі даних. Запит зберігає лише опис правила, за яким з бази даних можна отримати певну інформацію. Коли користувач відкриває запит, відбувається пошук даних у таблицях бази у відповідності з цим правилом. Потім знайдена інформація виводиться у вигляді таблиці.

За своїми можливостями запити потужніші за фільтри, оскільки фільтр діє в межах таблиці, для якого створений, а запит може обробляти декілька зв'язаних об'єктів одночасно.

Запити за призначенням та результатами поділяються на:

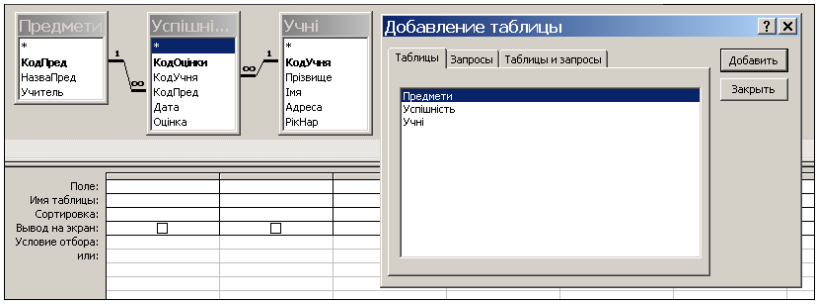
- прості запити;
- запити з параметрами;
- перехресні запити;
- запити на зміну даних.

Створення простого запиту



Щоб створити у базі даних простий запит за допомогою конструктора, потрібно виконати такі дії:

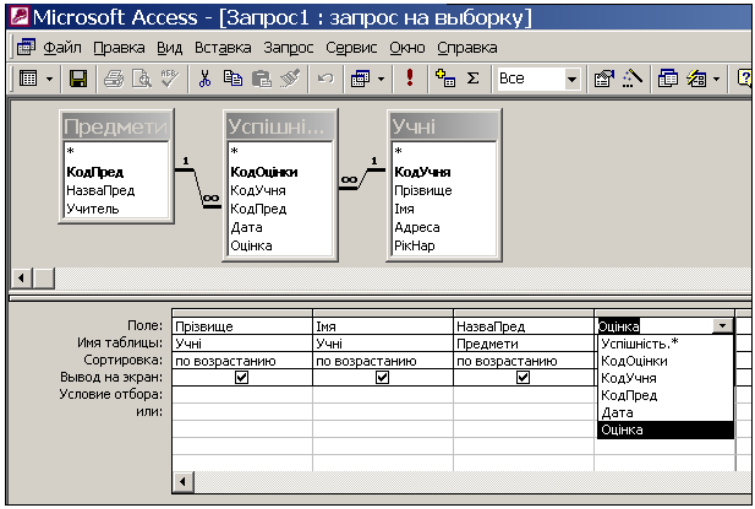
- відкрити вкладку ***Запити*** (*Запросы*);
- вибрати команду ***Створення запиту в режимі конструктора*** (*Создание запроса в режиме конструктора*) – відкриється вікно ***Запит 1: запит на вибірку*** (*Запрос 1: запрос на выборку*) і зразу ж на його фоні діалогове вікно ***Додавання таблиці*** (*Добавление таблицы*) для вибору таблиць, на основі яких буде створюватися запит;
- вибрати послідовно таблиці (*Предмети*, *Успішність*, *Учні*) і ввести їх у запит натисканням кнопки ***Додати*** (*Добавить*);
- закрити вікно. У верхній частині вікна *Конструктора запитів*, яка називається *Схема даних запиту*, з'являться списки полів доданих таблиць.



Нижня частина вікна *Конструктора* називається *Бланк запиту*. В рядку ***Поле:*** потрібно ввести імена полів, які повинні відображатися в підсумковій таблиці.

Це можна зробити так:

- клацнути в рядку ***Поле:*** – з'явиться кнопка ▼;
- клацнути на кнопці ▼ – відкриється список полів вибраних для запиту таблиць у форматі *Назва таблиці. Назва поля*;
- клацнути на потрібному полі – назва з'явиться в клітинці, нижче з'явиться назва таблиці, з якої буде вибиратися це поле;
- при потребі, у наступному рядку вибрати вид впорядкування (***Сортування***);
- у рядку ***Виведення на екран:*** (*Вывод на экран:*) автоматично встановиться прапорець. Якщо його зняти, то вміст цього поля не буде виводитись у підсумковій таблиці;



- перейти на наступний стовпчик, де повторити такі ж дії для наступного поля (на малюнку послідовно введені поля *Прізвище*, *Ім'я*, *НазваПред*, *Оцінка*).

Щоб у вікні запиту було видно панель інструментів (*див. на малюнку третій рядок*), потрібно вибрати команду ***Вигляд*** (*Вид*) ⇨ ***Панелі інструментів*** (*Панели инструментов*) ⇨ ***Конструктор запитів*** (*Конструктор запросов*).

Для видалення стовпчика достатньо його виділити (клацнути над ним при появі ⬇) і вибрати команду меню *Правка* ⇨ ***Видалити стовпці*** (*Удалить столбцы*).

Запуск і збереження запиту

Для запуску запиту потрібно на панелі інструментів натиснути кнопку ***Запуск*** або перевести запит у ***Режим таблиці*** натисканням кнопки ***Вигляд*** (*Вид*). При натисканні на ▼ відкривається три режими: ***Конструктор***, ***Режим таблиці*** (*Режим таблицы*), ***Режим SQL.***

Після виконання запиту кнопка ***Вигляд*** (*Вид*) перетворюється на кнопку, натискання якої призводить до повернення в режим *Конструктора запитів*. Послідовне натискання цих кнопок дає зручну можливість оперативно переглядати результати виконання запиту. Після виконання запиту одержимо таблицю, наведену на малюнку (див. наступну сторінку).

Для збереження запиту необхідно виконати команду меню *Файл* ⇨ ***Зберегти*** (*Сохранить*) або натиснути відповідну кнопку на панелі інструментів. Відкриється діалогове вікно ***Збереження*** (*Сохранение*), у якому слід ввести ім'я запиту і натиснути кнопку ***ОК***.

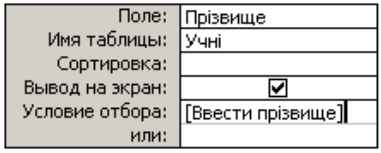
Прізвище	Ім'я	НазваПред	Оцінка
Бондаренко	Марія	Алгебра	11
Бондаренко	Марія	Інформатика	11
Бондаренко	Марія	Хімія	10
Бондарчук	Зінаїда	Алгебра	5
Бондарчук	Зінаїда	Інформатика	4
Бондарчук	Зінаїда	Фізика	11
Бондарчук	Зінаїда	Хімія	4
Іванов	Віталій	Алгебра	8
Іванов	Віталій	Алгебра	11

Запит з параметрами

Параметричний запит надає користувачу додаткові можливості. Наприклад, перед виконанням запиту можна ввести прізвище,

за яким буде здійснено подальший пошук. Для створення запиту з параметрами виконують такі дії:

- відкрити існуючий запит у режимі *Конструктора*;
- у «Бланк запиту» в потрібному стовпчику в рядку ***Умова відбору:*** (*Условие отбора*) ввести текст у квадратних дужках (наприклад, в стовпчику *Прізвище* записати *[Ввести прізвище]*); умови відбору можна встановлювати за маскою, за кількома полями (наприклад, прізвище та ім'я) або за логічною операцією ***АБО***;



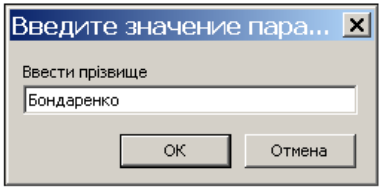
- відкрити запит, натиснувши кнопку ***Запуск***;
- у діалоговому вікні, що з'явиться, ввести значення параметра (наприклад, Бондаренко);
- натиснути кнопку ***ОК*** – підсумкова таблиця міститиме записи, які відповідають заданим параметрам;
- для збереження запиту з новим ім'ям потрібно вибрати команду ***Зберегти як…*** (*Сохранить как...*), ввести нове ім'я і натиснути ***ОК***.

	*Щоб скасувати введення параметрів, потрібно в режимі Конструктора очистити рядок* ***Умова відбору****.*

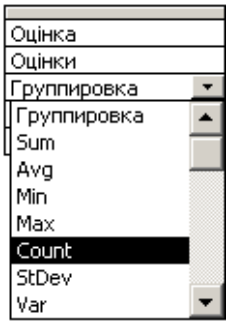
Створення запиту про кількість оцінок

Для створення запиту про кількість оцінок необхідно:

- відкрити вкладку ***Запити*** (*Запросы*);
- вибрати ***Створення запиту в режимі Конструктора*** (*Создание запроса в режиме Конструктора*);



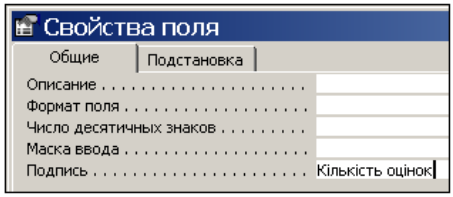
- додати потрібні таблиці (в нашому випадку – *Предмети*, *Успішність*, *Учні*), закрити вікно ***Додавання таблиці*** (*Добавление таблицы*);
- у бланк запиту послідовно ввести поля (тут *Учні.Прізвище*; *Учні.Ім'я*, *Предмети.НазваПред*; *Успішність.Оцінка*);
- на панелі інструментів натиснути кнопку ***Групові операції*** (*Групповые операции*) або вибрати команду



Вигляд (*Вид*) ⇨ ***Групові операції*** – в бланку запиту з'явиться новий рядок ***Групові операції*** зі значенням ***Групування*** (*Группировка*) для всіх полів;

- у стовпчику *Оцінка* відкрити список ***Групування*** (*Группировка*) і вибрати в ньому *Count* – це функція для підрахунку кількості записів;
- відкрити запит і впевнитися, що в підсумковій таблиці для кожного учня визначається кількість всіх одержаних ним оцінок. Цьому полю буде автоматично надана назва *Count_Оцінка***.**

- щоб змінити запропоновану назву, потрібно перейти в режим *Конструктора*, виділити в бланку запиту стовпчик *Оцінка* і натиснути кнопку ***Властивості*** (*Свойства*) на панелі інструментів;



- в діалоговому вікні ***Властивості поля*** (*Свойства поля*), що відкриється, в рядку ***Підпис*** (*Подпись*) ввести потрібний текст (тут *Кількість оцінок*);
- закрити вікно ***Властивості поля*** (*Свойства поля*) і знову відкрити запит, щоб пересвідчитися, що стовпчик має щойно встановлену назву;
- зберегти запит з потрібним іменем (наприклад, *Кількість*).

Запрос1 : запрос на выборку

Прізвище	Ім'я	НазваПред	Кількість оцінок
Бондарчук	Зінаїда	Хімія	1
Іванов	Віталій	Алгебра	2
Іванов	Віталій	Інформатика	2
Іванов	Віталій	Фізика	4
Іванов	Віталій	Хімія	3

У нашому випадку після виконання описаних дій можемо одержати наведену на малюнку таблицю.

Обчислення середнього бала

Для обчислення середнього бала необхідно виконати такі дії:

- відкрити потрібний запит (наприклад, *Кількість*), перейти в режим *Конструктора*;
- додати в кінці бланку запиту ще одне поле *Оцінка*;

- якщо бланку запиту не має рядка ***Групові операції*** (*Групповые операции*), потрібно на панелі інструментів натиснути відповідну кнопку або вибрати команду ***Вигляд*** (*Вид*) ⇨ ***Групові операції*** (*Групповые операции*);
- в *останньому* стовпчику *Оцінка* відкрити список ***Групування*** (*Группировка*) і вибрати в ньому *Avg* – це функція для підрахунку середнього значення;
- натиснути кнопку ***Властивості*** (*Свойства*) на панелі інструментів;

- у вікні ***Властивості поля*** (*Свойства поля*) ввести ***Формат поля*** – ***Фіксований*** (*Фиксированный*), ***Кількість десяткових знаків*** (*Число десятичных знаков*) – 1, ***Підпис*** (*Подпись*) – *Середній бал*;
- відкрити запит і пересвідчитися, що стовпчик має встановлені назву і вміст;
- за допомогою команди ***Зберегти як…*** (*Сохранить как...*) зберегти створений запит з НОВИМ іменем (наприклад, *Середнє*).

У нашому випадку після виконання описаних дій для створення запиту одержимо таблицю, подібну до наведеної на малюнку.

Середнє : запрос на выборку

Прізвище	Ім'я	НазваПред	Кількість оцінок	Середній бал
Іванов	Віталій	Алгебра	2	9,5
Іванов	Віталій	Інформатика	2	9,0
Іванов	Віталій	Фізика	4	7,5
Іванов	Віталій	Хімія	3	9,3

Створення запиту для відбору оцінок високого рівня

Щоб створити запит для відбору оцінок високого рівня, скористаємось раніше створеним запитом, у якому треба зняти встановлені умови відбору і встановити нову умову *>9* у полі *Оцінка*:

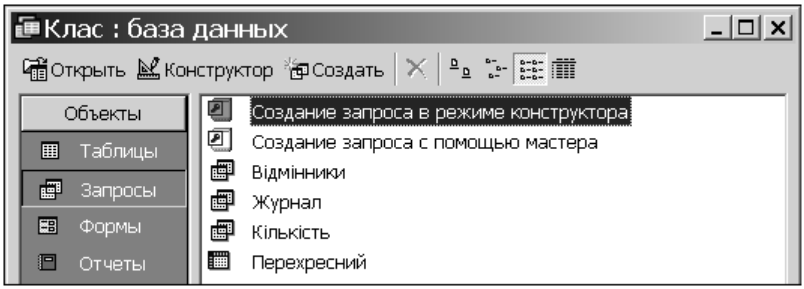
- відкрити потрібний запит (наприклад, *Журнал*) у режимі *Конструктора*; зняти всі умови відбору, якщо вони є;
- у стовпчику *Оцінка* в поле ***Умова відбору*** (*Условие отбора*) ввести умову *>9*;
- відкрити запит, впевнитися в правильності його роботи (результат може бути таким, як на малюнку на наступній сторінці);

Журнал : запрос на выборку

	Прізвище	Ім'я	НазваПред	Оцінка
▶	Іванов	Віталій	Фізика	10
	Іванов	Віталій	Хімія	12
	Бондаренко	Марія	Хімія	10

- за допомогою команди ***Зберегти як…*** (*Сохранить как...*) зберегти запит *з новим іменем* (наприклад, *Відмінники*).

Після створення запитів вікно бази даних *Клас* на вкладці ***Запити*** (*Запросы*) може мати наведений на малюнку вигляд.



При розробці баз даних, зорієнтованих на тривале використання і часту зміну наповнення, створюють *запити на додавання*, *оновлення* та *видалення* даних. Вони дозволяють швидко виконувати відповідні операції одночасно для багатьох записів, не порушуючи при цьому умов цілісності даних.

Питання для самоконтролю (Тест ТЕМА-2-7):

1. *Для чого використовуються запити?*
2. *Чим запити відрізняються від фільтрів?*
3. *На які види поділяються запити за призначенням?*
4. *Як створити простий запит?*
5. *Які команди служать для відкриття створеного запиту?*
6. *Як зберегти запит з потрібним іменем?*
7. *Як створюють запит з параметрами?*
8. *Як працювати із запитом з параметрами?*
9. *Як можна створити запит про успішність учнів?*
10. *Яке призначення має функція Count?*
11. *Як для поля ввести більш придатне ім'я?*
12. *Як провести обчислення середнього бала?*
13. *Яке призначення має функція Avg?*
14. *Як створити запит для відбору оцінок високого рівня?*
15. *Які запити спрощують внесення у базу великих масивів нових даних?*

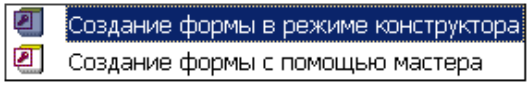
3.7. Форми. Звіти

Створення форми

Для побудови інтерфейсу користувача бази баних у СУБД Access створюють ***форми***.

Форма являє собою зручний бланк для перегляду на екрані, заповнення і редагування вмісту таблиць та перегляду результатів запитів. Продуманий інтерфейс користувача дає змогу доручити роботу з базою даних персоналу невисокої кваліфікації.

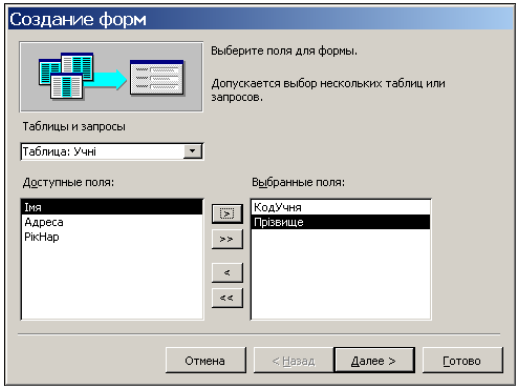
Після завантаження бази даних для створення форми обирають один із засобів, доступних на вкладці ***Форми*** (*Формы*) (*див. мал.*).



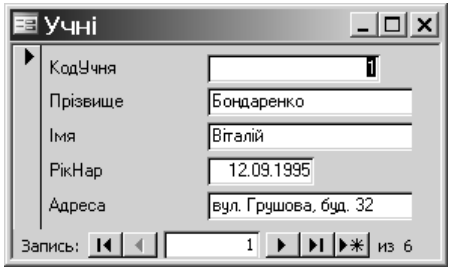
Створення форми в режимі *Майстра* дає хороший результат, і саме цей режим буде далі розглянуто. *Майстер* задає детальні питання про записи, поля, макет, формат і створює форму на основі відповідей. Доопрацьовують і редагують форму у режимі *Конструктора*.

Для створення форми в режимі *Майстра* виконують дії:

- відкрити базу даних, перейти на вкладку ***Форми*** (*Формы*);
- вибрати команду ***Створення форми з допомогою майстра*** (*Создание формы с помощью Мастера*) – відкриється вікно ***Створення форми*** (*Создание форм*);



- у полі ***Таблиці і запити*** (*Таблицы и запросы*) вибрати потрібну таблицю або запит (у нас вибрано *Таблица: Учні*). У полі ***Доступні поля*** (*Доступные поля*) будуть перелічені поля вибраної таблиці;
- за допомогою кнопки «>» вибрати поля, які міститиме форма (вони при натисканні з'являтимуться у списку ***Вибрані поля*** (*Выбранные поля*), або скористатися кнопкою «>>», щоб вибрати всі поля разом. Скасовують вибір кнопками «<» та «<<»;

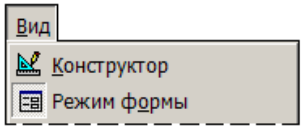


- натиснути кнопку ***Далі*** (*Далее*) – з'явиться наступне вікно, у якому вибрати варіант розміщення ***в один стовпець*** (*в один столбец*) (пропонуються ще ***стрічковий*** (*ленточный*), ***табличний*** (*табличный*), ***вирівняний*** (*выровненный*) тощо);
- натиснути кнопку ***Далі*** (*Далее*) – з'явиться вікно, в якому вибрати стиль оформлення, наприклад, ***Стандартний*** (*Стандартный*);
- натиснути кнопку ***Далі*** (*Далее*) – з'явиться наступне вікно з фінішним прапорцем, у якому можна змінити ім'я форми, вибрати подальші дії (відкрити форму для роботи чи продовжити розробку в режимі *Конструктора*) і натиснути кнопку ***Готово***.

У результаті виконаних дій буде збережена і відкриється форма для заповнення таблиці *Учні* (див. мал.) Внизу розміщені кнопки переходу. Розміри вікна форми змінюють, перетягуючи межі та кнопками керування вікном.

Перехід до режиму Конструктора

Перехід до режиму *Конструктора* здійснюється за допомогою меню ***Вигляд*** (*Вид*) або відповідної кнопки панелі інструментів. Повернутися до режиму форми можна за допомогою команди ***Вигляд*** (*Вид*) ⇨ ***Режим форми*** (*Режим формы*) або відповідної кнопки панелі інструментів.



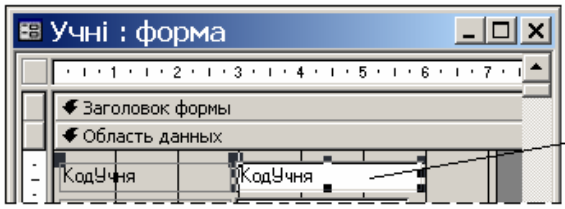
Редагування форми

Форма *Учні* містить елементи двох видів: написи та поля. При редагуванні форми слід мати на увазі, що написи не виводять даних з бази, а лише певний текст (підписи полів, пояснення), який відображається і в режимі *Конструктора*. У поля, що мають вигляд білих прямокутників, при роботі з формою виводяться дані, а в режимі *Конструктора* – назви полів таблиці чи запиту, звідки ці дані отримані. Видалення напису ніяк не впливає на працездатність форми, а після видалення поля буде втрачений доступ до зв'язаної з

ним інформації. Здебільшого поля мають підписи, розміщені у *зв'язаних* з ними написах.

Для редагування форми необхідно виконати такі дії:

- відкрити форму і перейти в режим *Конструктора*;



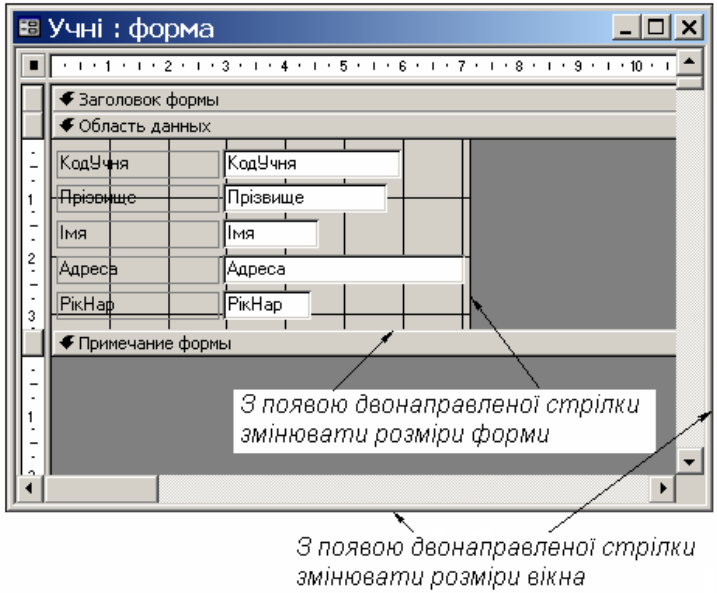
- для ***видалення поля*** (наприклад, *КодУчня*), потрібно клацнути лівою кнопкою миші в полі і після появи чорних квадратів натиснути клавішу ***Delete***. Поле буде видалене разом з пояснювальним написом;
- для ***переміщення поля*** необхідно просто перетягти його на нове місце (вказівник миші буде таким –). Маркер у вигляді великого чорного квадратика дозволяє переміщувати окремо кожен зі зв'язаних елементів (вказівник –);
- для ***переміщення всіх об'єктів*** форми разом – вибрати команду ***Правка*** ⇨ ***Виділити все*** (*Выделить всё*) (або натиснути комбінацію клавіш ***Ctrl+A*** англ.) і виконати перетягування;

Щоб зняти виділення, достатньо клацнути на вільному місці форми.

- для ***зміни розмірів*** поля потрібно його виділити, встановити вказівник миші на один із маленьких чорних квадратів і перемістити його до одержання потрібного розміру;
- для ***зміни шрифту*** слід виділити потрібне поле і за допомогою кнопок панелі інструментів встановити розмір, написання, колір і вид шрифту (після збільшення розмірів шрифту одного поля, можливо, доведеться перемістити інші поля).
- зберегти відредаговану форму, натиснувши кнопку ***Зберегти*** (*Сохранить*) на панелі інструментів.

Оформлення форми

Крім області даних форма може мати заголовок та примітку, на яких розміщують як корисну інформацію, так і елементи дизайну. Ці частини форми залишаються видимими, навіть якщо для перегляду області даних необхідна смуга прокрутки. Наприк-



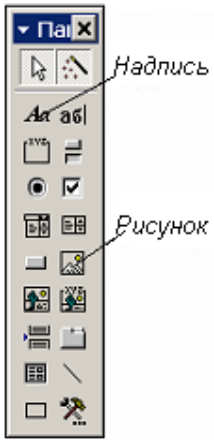
лад, для стрічкової (рос. – *ленточной*) форми в області заголовка зручно розмістити назви стовпців.

Щоб змінити зовнішній вигляд форми, виконують такі дії:

- відкрити форму, перейти в режим *Конструктора*;
- змінити розміри вікна і форми, для чого встановити вказівник миші на межу і при появі двонаправленої стрілки перетягти її до потрібного розміру вікна;
- розширити область заголовка (примітки), перетягнувши її нижню межу.

Щоб додати напис, користуються панеллю елементів (при її відсутності натиснути на панелі інструментів кнопку ***Панель елементів*** (*Панель элементов*)):

- на панелі елементів клацнути кнопку ***Aa*** (***Напис***), потім клацнути на вільному місці форми і ввести потрібний текст (наприклад, «Учні 9-В класу»);
- натиснути клавішу ***Enter*** – стане доступною панель інструментів;
- для напису встановити потрібний шрифт (наприклад, Arial, 18 пт, червоний, напівжирний);
- щоб змінити колір фону напису, натиснути на панелі інструментів кнопку ▾ і вибрати потрібний колір.



На форму можна вставити малюнок:

- на панелі елементів натиснути кнопку ***Малюнок*** (*Рисунок*) і *перетягуванням* виділити область, наприклад, у лівій частині заголовка форми – з'явиться вікно ***Вибір малюнка*** (*Выбор рисунка*), у якому знайти потрібний малюнок і натиснути ***ОК***;
- вставивши малюнок, за допомогою кутових маркерів зменшити його до необхідних розмірів, при цьому частина малюнка може стати невидимою;
- відкрити вікно властивостей малюнка за допомогою відповідної кнопки ***Властивості*** (*Свойства*) на панелі інструментів, у якому відкрити вкладку *Макет*;
- у полі ***Встановлення розмірів*** (*Установка размеров*) відкрити список і встановити значення ***За розміром рамки*** (*По размеру рамки*);
- закрити вікно властивостей малюнка.

Після виходу з режиму *Конструктора* (меню ***Вигляд*** (*Вид*) ⇨ ***Режим форми*** (*Режим формы*)), форма набуде вигляду, показаного на малюнку.

Використовуючи створену форму, можна вводити в таблицю дані про нових учнів.

Створення звіту

Для підготовки даних до друку створюють звіти. В ***Access*** звіт створюють за допомогою *Майстра звітів* або *Конструктора звітів*. При використанні *Майстра звітів* виконуються такі дії:

- у відкритій базі даних перейти на вкладку *Отчёты*;
- вибрати команду ***Створення звіту з допомогою Майстра*** (*Создание отчёта с помощью Мастера*) – відкриється вікно ***Створення звітів*** (*Создание отчётов*);
- у полі ***Таблиці і запити*** (*Таблицы и запросы*) вибрати таблицю або запит – джерело даних для звіту (у нас вибрано *Таблица: Учні*). У полі ***Доступні поля*** (*Доступные поля*) будуть показані поля вибраної таблиці;

- за допомогою кнопки «>» вибрати потрібні поля або скористатися кнопкою «>>» для вибору всіх полів разом;
- натиснути кнопку ***Далі*** (*Далее*) – з'явиться вікно, призначене для задання рівнів групування. Для прийняття рівня, заданого за мовчазною згодою, натиснути кнопку ***Далі***;
- вибрати тип впорядкування, натиснути кнопку ***Далі***;
- вибрати вид макета (***у стовпчик*** (*в столбец*), ***табличний*** (*табличный*), ***вирівняний*** (*выровненный*)) та орієнтацію аркуша паперу (***книжкова*** (*книжная*), ***альбомна*** (*альбомная*)), натиснути кнопку ***Далі***;
- вибрати стиль оформлення звіту (***Діловий*** (*Деловой*), ***Звичайний*** (*Обычный*), ***Напівжирний*** (*Полужирный*), ***Стислий*** (*Сжатый*), ***Спокійний*** (*Спокойный*), *Строгий*), натиснути кнопку ***Далі***;
- у вікні з фінішним прапорцем ввести назву звіту *Учні*, натиснути кнопку *Готово*.

У результаті виконаних дій (макет – *табличний*, орієнтація – *книжкова*, стиль – *строгий*) з'явиться звіт *Учні* (*див. мал*).

Учні

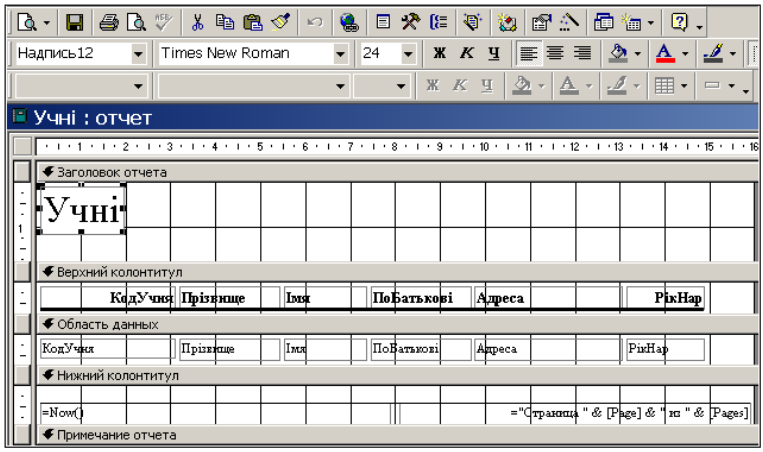
КодУчня	Прізвище	Ім'я	ПоБатькові	Адреса	РікНар
1	Іванов	Віталій	Іванович	вул. Грушова, буд. 32	12.09.1987
2	Токаренко	Леонід	Петрович	вул. Шкільна, буд. 8	08.08.1986
3	Бондаренко	Марія	Сидорівна		06.07.1984
4	Токаренко	Наталія	Олексіївна		19.08.1986
5	Сидоренко	Олег	Карлович	вул. Червона, буд. 20	22.09.1984

Редагування і оформлення звіту

Редагування і оформлення звіту виконується в режимі *Конструктора* подібно до того, як це робилось для форми (*див. вище*). Наприклад, для редагування заголовка *Учні* необхідно виділити текст (*див. мал.*) і засобами панелі інструментів виконати потрібні дії. Крім відповідно оформлених даних, звіт може містити такі частини:

- заголовок – друкується на початку першої сторінки;
- верхній та нижній колонтитули – друкуються вгорі та внизу кожної сторінки;
- примітку – друкується в кінці останньої сторінки.

Для збереження звіту слід натиснути кнопку ***Зберегти*** (*Сохранить*) на панелі інструментів; закривають звіт кнопкою ✕.



Друкування звіту

Перед друком, щоб не зіпсувати папір, рекомендується використати режим попереднього перегляду, для чого, *не відкриваючи звіт*, натиснути відповідну кнопку на панелі інструментів.

Для друкування звіту *потрібно його відкрити* і натиснути кнопку ***Друк*** (*Печать*) на панелі інструментів або вибрати команду ***Файл*** ⇨ ***Друк*** (*Печать*).

Питання для самоконтролю (Тест ТЕМА-2-8):

1. *Для чого використовуються форми?*
2. *Якими засобами можна створити форму?*
3. *За допомогою якого засобу можна редагувати форму?*
4. *Як перемістити поле в інше місце форми?*
5. *Яким чином перемістити всі поля форми разом?*
6. *Як змінити параметри шрифту окремого поля?*
7. *Які дії слід виконати для створення заголовка форми?*
8. *Яким чином форму оздобити малюнком?*
9. *Для чого створюють звіти?*
10. *Якими засобами можна створити звіт?*
11. *Як створити звіт за допомогою Майстра?*
12. *Як редагувати звіт у режимі Конструктора?*
13. *Яким чином змінити параметри шрифту у звіті?*
14. *Що дає попередній перегляд звіту?*
15. *Як надрукувати звіт?*

3.8. Тематична робота «Бази даних. СУБД»

Див. робочий зошит «Інформатика. Базовий курс. 9 клас» / Пилипчук О.П., Шестопалов Є.А. – Шепетівка: «Аспект», 2009.

4. Автоматизоване створення та публікація веб-ресурсів

4.1. Сайти. Мова HTML. Хостинг.

Інформаційна модель сайту

Веб-сайт *(англ.* website*, місце, майданчик в Інтернеті), або* ***сайт*** *(англ.* site*, місце, майданчик) – сукупність* ***веб-сторінок****, доступних у Інтернеті, які об'єднані як за змістом, так і навігаційно.*

Створення веб-сайту починається зі створення інформаційної моделі сайту, тому спочатку необхідно сформулювати вимоги до інформаційного наповнення та завдання, що мають бути вирішені при створенні веб-сайту. Наприклад, при плануванні розробки сайту шкільного гуртка «Бісероплетіння» може з'явитися така інформаційна модель:

- Головна сторінка (історія розвитку бісерної творчості);
- Матеріали та інструменти для роботи з бісером;
- Види низання:
  - Низання голчате (технологія виготовлення виробу);
  - Низання дугами (технологія виготовлення виробу);
  - Паралельне низання (технологія виготовлення виробу);
- Майстер-клас з виготовлення квітів;
- В гостях у майстрині (інформація про керівника гуртка);
- Наші роботи;
- Гостьова книга;
- Блог.

Проектуючи сайт, слід мати на увазі, що ***будь-яку веб-сторінку оцінюють за двома параметрами***: змістом та зовнішнім виглядом.

Зовнішній вигляд кожного веб-сайту є унікальним, однак в усіх сайтах можна знайти спільні за функціональністю частини. На будь-якому сайті першою відкривається *головна (домашня) сторінка*. Її розробці приділяють особливу увагу, бо, як показують дослідження, люди не здатні читати інформацію, що відображається на екрані монітора, так уважно, як книжки або журнали, вони зазвичай лише поверхово її переглядають, ніби рекламу. Якщо головна сторінка містить те, що шукає відвідувач

сайту, він працюватиме з сайтом далі, в іншому разі − перейде до інших сайтів, яких в мережі Інтернет дуже багато.

У верхній частині головної сторінки зазвичай розташовують *назву сайту*, яку дублюють на інших сторінках сайту.

Щоб забезпечити швидкий перехід до основних тематичних *розділів сайту*, створюють *меню сайту*.

Меню сайту *– сукупність гіперпосилань на його розділи.*

Горизонтальне меню найчастіше розміщують під назвою сайту, іноді дублюючи його в нижній частині сторінки, а вертикальне – переважно в лівій частині сторінки, у місці, звідки відвідувач починає її переглядати. Меню є одним із найважливіших

компонентів сайту, користувач постійно звертає на нього увагу, і тому вимоги до нього високі. Меню повинне бути зручним, помітним і зрозумілим, інакше користувач не знатиме, як перейти до потрібного розділу, і залишить сайт. Пункти меню мають бути чітко відділені один від одного.

Гіперпосилання, розміщені в тексті або у вигляді графічних об'єктів, дозволяють переходити на різні сторінки сайту або на інші сайти.

Як організувати веб-сайт

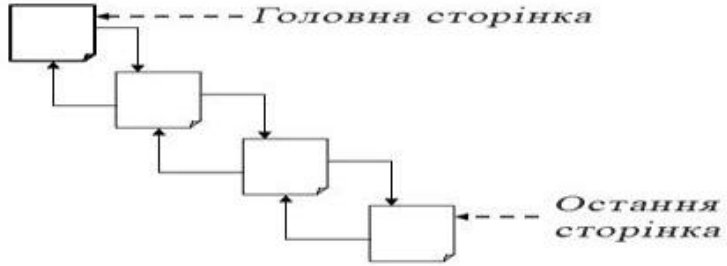
При проектуванні сайту потрібно добре обміркувати його загальну структуру, зміст та зв'язки між сторінками (посилання).

Нижче наведено три варіанти структури сайту. Кожен з них має свої недоліки та переваги.

Стандартна. Головна веб-сторінка містить посилання на інші документи веб-сайту, а документи містять посилання, відповідно, на головну веб-сторінку.

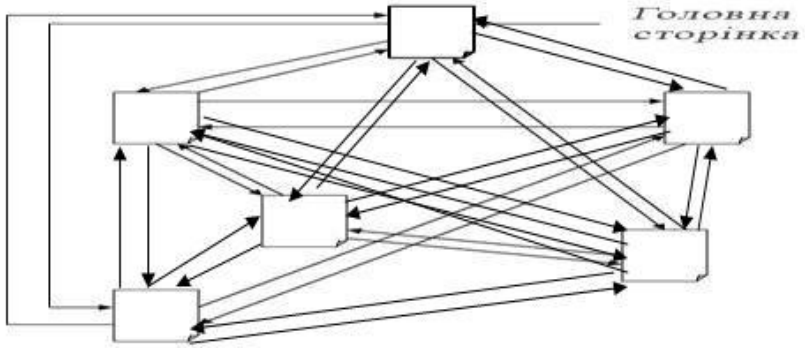
Це найпростіший спосіб організації веб-сайту і зустрічається він досить часто.

Каскад. У цьому випадку посилання у документах задані таким чином, що існує тільки один шлях обходу сторінок веб-сайту. Схему каскадного способу організації веб-сайту наведено на малюнку.



При каскадному способі відвідувачі сайту можуть з кожної сторінки переміщуватись тільки до наступної або до попередньої.

Павутина. У цьому випадку усі сторінки веб-сайту містять посилання на інші сторінки, і користувач може легко перейти з будь-якої сторінки практично на будь-яку іншу. Ця схема може перетворитися у лабіринт, якщо вийде з-під контролю, але вона популярна у тих випадках, коли посиланнями на документи користуються не надто часто, або коли кількість сторінок сайту не дуже велика. На схемі показано простий приклад такої організації веб-сайту.



При побудові веб-сайту здебільшого поєднують стандартний метод і метод павутини: відвідувач можете перейти до будь-якого документу безпосередньо з головної сторінки, при цьому і самі документи також посилаються один на інший.

Інформаційні розділи. Проектуючи сайт, необхідно обґрунтувати назви його розділів, їх кількість та чітку структурованість схеми сайту, виходячи з завдань які він вирішуватиме. Наприклад, сайт школи може містити такі розділи:

1. Головна сторінка (шкільні новини);
2. Історія школи (коротка історія школи);
3. Наші вчителі (фото вчителів школи);
4. Кращі учні школи (фото кращих учнів школи);
5. Фотогалерея (розміщення фото за певними темами);
6. Каталог статей (статті для батьків, учнів);
7. Каталог файлів (розробки виховних заходів);
8. Гостьова книга;
9. Форум;
10. Блог.

Класифікація веб-сайтів

За рівнем доступу

Відкриті – всі сервіси сайту доступні кожному з відвідувачів.

Напіввідкриті – для доступу необхідно зареєструватися (зазвичай безкоштовно).

Закриті – службові сайти організацій, особисті сайти приватних осіб тощо. Такі сайти доступні для вузького кола людей. Доступ новим людям зазвичай дається через т. зв. інвайт (запрошення).

За природою вмісту

Статичні – весь вміст готується заздалегідь. Файли користувачу передаються в тому вигляді, в якому вони зберігаються на сервері.

Динамічні – вміст генерується спеціальними програмами (скриптами) на основі інформації з бази даних. Наприклад, розробник сайту для інтернет-магазину, в якому продають десятки тисяч різних товарів, не складає для кожного товару окрему сторінку. Натомість, щоб показати сторінку товару, запускається скрипт, з бази даних вибираються дані про назву, ціну, опис, зображення товару, потім формується веб-сторінка і передається браузеру користувача. На самому сервері ця сторінка не зберігається.

За фізичним розташуванням

Зовнішні сайти мережі Інтернет. Наприклад, сайт Українського центру оцінювання якості освіти *http://www.testportal.gov.ua/*.

Локальні сайти, доступні тільки в межах локальної мережі. Це можуть бути як сайти організацій, так і сайти приватних осіб в локальній мережі провайдера.

За категорією вирішуваних завдань

Сайт-візитка – містить загальні дані про власника сайту (організацію, підприємця тощо): вид діяльності, історія, прайс-лист, контактні дані, реквізити, схема проїзду і т.п. Фахівці розміщують своє резюме.

Представницький сайт – так іноді називають сайт-візитку з розширеною функціональністю, що включає докладний опис послуг, портфоліо, відгуки, форму зворотного зв’язку і т.д.

Корпоративний сайт – містить повну інформацію про компанію-

власника, послуги, продукцію, події в житті компанії. Відрізняється від сайту-візитки і представницького сайту повнотою наданої інформації, часто надає різні інструменти для роботи з контентом (пошук і фільтри, календарі подій, фотогалереї, корпоративні блоги, форуми). Може містити закриті розділи для тих чи інших груп користувачів – співробітників, дилерів, контрагентів і т.ін.

Каталог продукції. Містить докладний опис товарів та послуг, сертифікати, технічні та споживчі дані, відгуки експертів і т. д.

Інтернет-магазин – веб-сайт з каталогом продукції, за допомогою якого клієнт може замовити потрібні товари. Використовуються різні системи розрахунків: від пересилання товарів післяплатою до розрахунків за допомогою банківських карт та електронних розрахункових систем.

Промо-сайт – сайт про конкретну торгову марку або продукт. На таких сайтах розміщується вичерпна інформація про бренд, проводяться рекламні акції (конкурси, вікторини, ігри і т. п.).

Сайт-квест – Інтернет-ресурс, на якому організовано змагання з розгадування послідовності взаємопов'язаних логічних задач.

Особисті сторінки користувачів (англ. *home pages*), які лише за традицією називаються сторінками, хоча багато з них являють собою повноцінні сайти (*див. мал.*). Оскільки створюють такі сторінки користувачі різної кваліфікації, зміст і дизайн таких сторінок не завжди досконалі.

За інформаційним наповненням

Тематичний сайт – надає вичерпну інформацію з певної теми.

Тематичний портал – це дуже великий веб-ресурс, який надає вичерпну інформацію з певної галузі знань. На відміну від тематичного сайту, портал містить засоби взаємодії з користувачами і дозволяє користувачам спілкуватися в рамках порталу (*форуми, чати*). Це – середовище «існування» користувача.

Веб-сервіс – зазвичай вирішує конкретне завдання, безпосередньо пов'язане з використанням мережі Інтернет:

- **Пошукові сервіси** – наприклад, *http://yandex.ru* (Яндекс), *http://www.google.com.ua/* (Google).
- **Поштовий сервіс** – наприклад www.i.ua.
- **Форум –** наприклад, форум вчителів інформатики *http://informatic.org.ua/forum/*.
- **Блоговий сервіс** – наприклад, *http://www.blogger.com/*
- **Фотохостинг** – наприклад, Flickr, ImageShack, Panoramio, Photobucket.
- **Зберігання відео** – наприклад, YouTube, RuTube.

Дошка оголошень

Каталог сайтів – наприклад, *http://zakladka.org.ua/* (Закладка), *http://www.dmoz.org/* (Open Directory Project).

Поняття про мову HTML

Основою однієї з найпопулярніших служб Інтернету – всесвітньої «павутини» (World Wide Web або WWW) – є мова гіпертекстової розмітки HTML.

***HTML** – набір домовленостей для розмітки документів, які визначають вигляд документів на екрані комп'ютера при доступі до них з використанням програми браузера.*

Документи, підготовлені з використанням HTML, називають HTML-документами. Одержати уявлення про те, як виглядає код HTML, ви зможете, якщо завантажите HTML-документ у браузер і виберете команду *Вигляд ⇨ У вигляді HTML*.

***HTML-документ** (або веб-сторінка) – це текстовий файл з розширенням .htm або .html, що розмічений засобами HTML і призначений, переважно, для публікації в Інтернеті.*

Код HTML є дуже компактним, і HTML-документи мають розміри значно менші, ніж подібні документи, підготовлені сучасними текстовими процесорами (наприклад, Microsoft Word). Це одна з причин широкого застосування мови HTML для розмітки текстів, які розміщуються в Інтернеті.

HTML-документи зазвичай розміщуються в мережі не поодинці, а у вигляді сайтів.

Розмітка документа за допомогою HTML виконується у два етапи:

1) документ розбивається на елементи: заголовки, абзаци, малюнки, таблиці тощо;
2) для кожного елемента задається команда мови *HTML*, що називається тегом (або дескриптором).

***Команда** містить інформацію про те, як повинен виглядати даний елемент на Web-сторінці, які зв'язки він має з іншими елементами або документами.*

У мові HTML є досить багато тегів, серед яких: теги створення заголовку документа, опису параметрів шрифту, креслення ліній, вставляння гіперпосилань, вставляння графічних елементів і т.д. Тому веб-сторінка, крім тексту і посилань, може містити зображення, звуки, відео.

***Тег** (англ. tag – вказівник, мітка) – це фрагмент коду, який описує певний елемент HTML-документа і міститься в кутових дужках < >.*

Деякі теги: <p> − початок абзацу; … − курсивне написання символів; … − напівжирне написання;
 − розрив рядка. У таблиці наведено приклад HTML-коду та результат його відображення браузером:

HTML-код	***Результат***
<p>Сонце заходить, гори чорніють, пташечка тихне, поле німіє…	**Сонце** заходить, *гори* чорніють, ***пташечка** тихне*, поле німіє…

Теги бувають парні (контейнери) та не парні.

Наприклад, теги та є парними (тобто, утворюють контейнер). При цьому тег − називають відкриваючим, бо він вмикає відповідний режим, а тег − називають закриваючим, оскільки він припиняє дію цього режиму.

Тег
 не є парним.

Хостинг. Розміщення веб-сайту на хостинг-сервері

Хостинг *(від англійського слова hosting – сумісне розміщення; to host – приймання гостей) – це розміщення веб-сайту на обладнанні компанії. Ресурси сервера і ліній зв'язку використовуються при цьому сумісно багатьма клієнтами.*

Зараз є багато веб-серверів, які надають безплатний хостинг (наприклад, українські сервери *www.ukrbiz.net, www.freehost.kiev.ua*, російські сервери *www.chat.ru, www.narod.ru, www.boot.ru*). Такі сервери не вимагають оплати за підтримку веб-сайтів клієнтів і працюють за рахунок коштів рекламодавців. Зареєстрованому клієнту надається певний простір на диску сервера (типово – декілька сотень мегабайт), якого вистачає для розміщення досить солідного сайту. Єдине зобов'язання, яке бере на себе клієнт безкоштовного веб-сервера, − це розміщення на своєму сайті банерів – маленьких помітних зображень (часто анімованих), призначених для реклами. Згоди на це у клієнтів, як правило, не питають – банери встановлюються і оновлюються без участі господаря сайту.

Безкоштовні веб-сервери мають також обмеження щодо сервісу, який вони надають: вони, як правило, не підтримують сучасні Інтернет-технології PHP, JSP, ASP та інші. Але свій сайт, який складається зі звичайних HTML-файлів, ви зможете розмістити на безкоштовному хостингу без особливих проблем. На деяких хостингах користувач отримує готову систему керування вмістом сайту (детальніше − далі).

Питання для самоконтролю:

1. *Що називається веб-сайтом?*
2. *Як називається сторінка, яка відкривається у вікні броузера першою на будь-якому веб-сайті?*
3. *Що розташовується у верхній частині головної сторінки веб-сайту?*
4. *Що таке меню веб-сайту? Який вигляд має меню веб-сайту?*
5. *З чого починається створення веб-сайту?*
6. *За якими параметрами оцінювати веб-сторінку?*
7. *Перелічи варіанти структури веб-сайту.*
8. *Охарактеризуй стандартний варіант структури веб-сайту.*
9. *Охарактеризуй каскадний варіант веб-сайту.*
10. *Охарактеризуй модель веб-сайту «Павутина».*
11. *Яким методом найзручніше користуватись при створенні веб-сайту? В чому його суть?*
12. *За якими критеріями класифікують веб-сайти?*
13. *Які інформаційні ресурси ти знаєш?*
14. *Наведи приклади веб-сервісів.*
15. *Що називається HTML-документом?*

4.2. Засоби автоматизованої розробки веб-сайтів.

Засоби автоматизованої розробки веб-сайтів

Спеціальні програми, які дозволяють створювати веб-сторінки, називаються ***HTML-редакторами.***

Для створення файлів HTML придатний практично будь-який текстовий редактор, наприклад, ***Блокнот*** (*Notepad*), або будь-який спеціалізований, наприклад, ***KompoZer***. Деякі засоби дозволяють створювати код HTML без його фактичного написання.

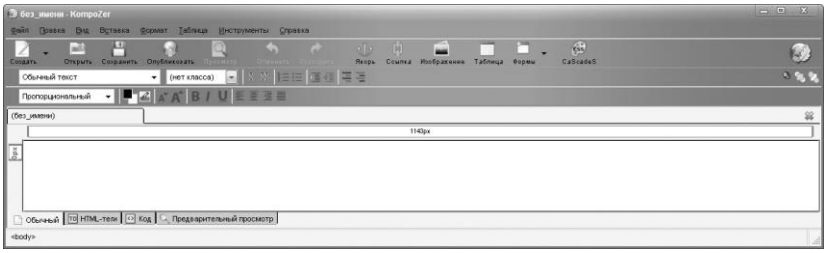
HTML-редактори поділяють на дві великі категорії: графічні і програмні редактори. Редактори обох типів мають спільні риси: і перші і другі нагадують сучасні текстові процесори. Відрізняються ж вони методом візуального відображення елементів, які входять до складу веб-сторінки.

Графічні HTML-редактори відображають і дозволяють редагувати сторінку такою, як вона буде виглядати у вікні броузера. Саме тому їх іноді називають редакторами категорії WYSIWYG (англ. What You See Is What You Get – «що бачиш – те отримуєш»).

Програмні HTML-редактори виводять на екран вихідний текст мовою HTML, надаючи при цьому в розпорядження розробника потужні засоби генерації коду, які позбавляють від необхідності писати його вручну.

Веб-сторінки можуть містити не тільки текстову, але й графічну інформацію. На основі графічних файлів можна за допомогою спеціального програмного забезпечення створювати анімаційні ролики та відеоролики. Також при перегляді веб-сторінок можливе використання звукового супроводу. Користуватися редакторами категорії WYSIWYG легше, оскільки вони ізолюють автора від важкого для сприйняття коду HTML. Майже всі популярні редактори цього типу, крім ***Microsoft FrontPage і Netscape*** Navigator Gold, дозволяють також виконувати безпосереднє редагування тексту HTML.

KompoZer − один із кращих візуальних веб-редакторів. Він має привабливий інтерфейс, велику кількість функцій, легкий у користуванні, швидкість роботи робить його одним із кращих програм свого класу. Програма не вимагає інсталяції. Завантажити її можна із сайту *http://kompozer.sourceforge.net*, або з сайту *http://ut0aza.at.ua* молодіжної колективної радіостанції UT0AZA, центру дитячої та юнацької творчості м. Глухів (на сторінці



http://ut0aza.at.ua/load/kompozer/8-1-0-15). Це потужний редактор веб-сторінок, який використовує візуальну технологію WYSIWYG.

KompoZer є повною системою веб-розробки, що поєднує в собі простоту використання і великі можливості які є у професійних програмах на зразок ***Microsoft FrontPage*** або ***Adobe DreamWeaver***.

KompoZer дуже простий у використанні, що робить його ідеальним для початківців, які хочуть швидко створити привабливий веб-сайт. ***KompoZer*** дозволяє зробити це без знання HTML або веб-програмування.

Основні **особливості** включають у себе:

1) Комплексне управліннння файлами за FTP-протоколом. Дозволяє зайти на свій сайт і здійснювати навігацію по файлах, редагуючи веб-сторінки «на льоту». Створений HTML код буде працювати у всіх популярних браузерах.
2) Перехід меж режимом WYSIWYG та редагуванням HTML за допомогою вкладок.
3) Потужна підтримка для форм, таблиць, шаблонів.

Система керування вмістом веб-сайту (СКВ)

Система керування вмістом (контентом) *(англ.Content management system, CMS) − інформаційна система або комп'ютерна програма, призначена для створення, редагування і керування вмістом сайту.*

Система керування вмістом дозволяє оперувати різноманітними даними: документами, фільмами, фотографіями, номерами телефонів і т.ін. Користувач СКВ отримує зручні засоби для опублікування цих даних на сайті, розмежуванням доступу до них користувачів різних категорій, зміни дизайну сторінок сайту тощо.

Ще кілька років тому для створення і подальшої підтримки власного веб-сайту треба було мати спеціальні знання. Але зараз в

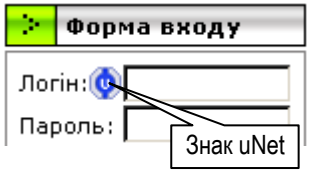
Інтернеті дуже швидко розвиваються так звані соціальні сервіси. Одним з них є можливість отримати безкоштовний хостинг для власного веб-сайту, на якому користувачеві надають не лише дисковий простір на сервері для власних файлів, але й можливість вибрати один з багатьох шаблонів оформлення свого майбутнього сайту та підтримувати його за допомогою СКВ.

Хостинг *http://www.ucoz.ua* – один із сайтів де до послуг користувача пропонується потужна система керування вмістом сайту. Залежно від потреб можна підключати різноманітні шаблони: каталоги статей, щоденник (блог), фотоальбом, форум, гостьову книгу і т.ін.

Зовнішній вигляд майбутнього веб-сайту можна вибрати з великої кількості готових зразків дизайну. Взявши за основу готовий шаблон оформлення можна внести до нього зміни на власний смак. Врешті-решт, можна розробити повністю власний шаблон і встановити його.

Для розміщення сайту надається дисковий простір 400 Мбайт. Проте, коли користувач працює над своїм сайтом як адміністратор, цей простір поступово збільшується (на екрані при цьому відображається лічильник).

Як створити сайт на ucoz.ua



Щоб отримувати можливість створювати скільки завгодно сайтів, слід зареєструватися. Для цього треба зайти на сайт http://www.ucoz.ua/ і пройти процедуру реєстрації у системі uNet. Це дозволить заходити в якості користувача на сайти системи, вказавши замість імені користувача електронну адресу. Сайти, які дозволяють такий спосіб входу, мають відповідну позначку біля поля введення імені користувача (див. мал.).

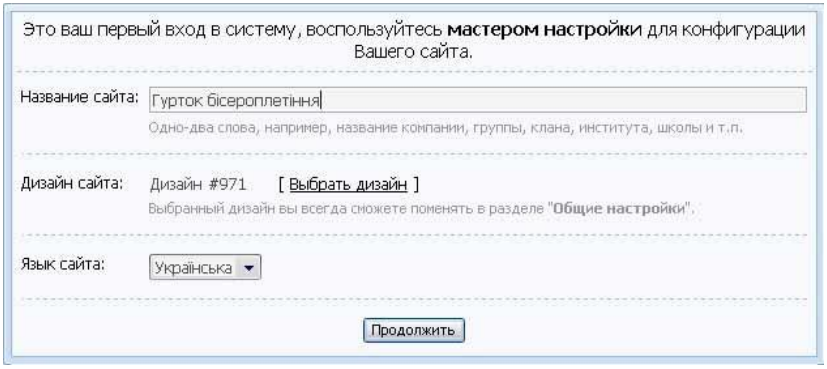
Для продовження створення сайту треба перейти за посиланням, вказаним у листі-підтвердженні. Ви будете перенаправлені до робочого столу користувача системи – так званого вебтопу. Тут слід вибрати новий пароль, вже для доступу до вебтопу, вибрати секретне питання та відповідь, ввести пароль користувача, який вказували при реєстрації, і вибрати команду **Зберегти (Сохранить).**

Вебтоп *– це панель, в якій користувач системи uCoz може обрати доменне ім'я для свого майбутнього сайту. Саме через вебтоп створюються сайти на uCoz. Пароль, вибраний для вебтопу надалі використовується для входу до* ***Панелі керування сайтом (Панель управления сайтом)****.*

З'являється вікно *Управление сайтами*. На вкладці *Создание сайтов* вписуємо бажану адресу майбутнього сайту, ставимо позначку поряд з *С правилами согласен* і вибираємо команду *Продолжить*. Якщо сайт з такою адресою вже зареєстровано, то з'явиться відповідне повідомлення і треба буде обрати іншу адресу. Нарешті з'явиться напис «Сайт успешно создан».

Після цього у вікні вводимо назву сайту і вибираємо команду *Выбрать дизайн*.

З запропонованих шаблонів обираємо той, який найбільше сподобався і вибираємо команду *Продолжить***.**

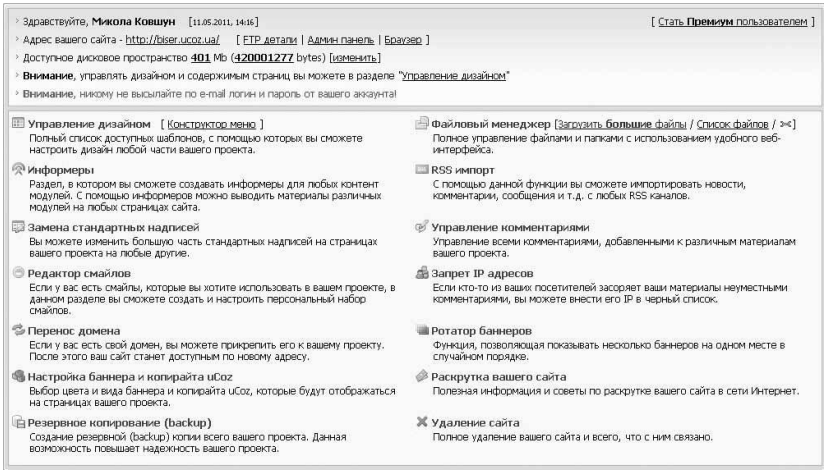


Далі uCoz запропонує обрати модулі для сайту. Оберемо потрібні, виберемо команду *Продолжить* і потрапимо у панель керування сайтом uCoz (на малюнку нижче). Надалі модулі можна буде додавати, перейшовши на вкладку *Неактивные*.

Редагування сайту uCoz

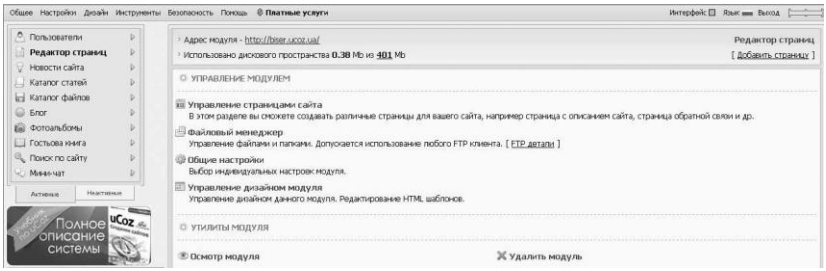
Отже, сайт створено. Тепер треба його відредагувати і наповнити контентом.

Робити виправлення і вносити вміст сайту краще робити через **панель адміністратора uCoz**. Для цього слід перейти на сайт, у формі входу написати e-mail і пароль (той, що вказувався при



реєстрації користувача) і зайти на сайт. Оскільки ви є адміністратором сайту, вгорі з'явиться *панель адміністратора*, а внизу сторінки − *кнопки редагування*: *у візуальному режимі* (кнопка з оком) і в *режимі HTML* (кнопка поряд).

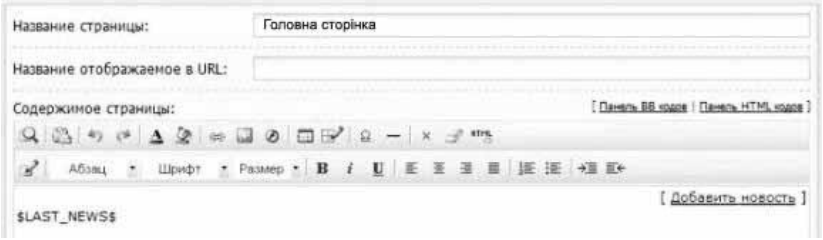
Спочатку ваш сайт має назву «Мій сайт». Її слід змінити відповідно до тематики вашого сайту. Для цього на панелі адміністратора вибираємо команду *Конструктор* ⇨ *Увімкнути конструктор* (*Включить конструктор*). Сайт трохи змінює свій вигляд, ставимо курсор на стару назву і пишемо нову. Потім на панелі адміністратора знову вибираємо команду *Конструктор* ⇨ *Зберегти зміни* (*Сохранить изменения*). Візуально назва сайту змінена, але, якщо вона не була вказана при створенні сайту, то у заголовку вікна браузера відображатиметься назва сайту, запропонована системою, а саме «Персональный сайт». Для того, щоб це змінити, слід зайти у панель управління, на панелі адміністратора вибравши команду



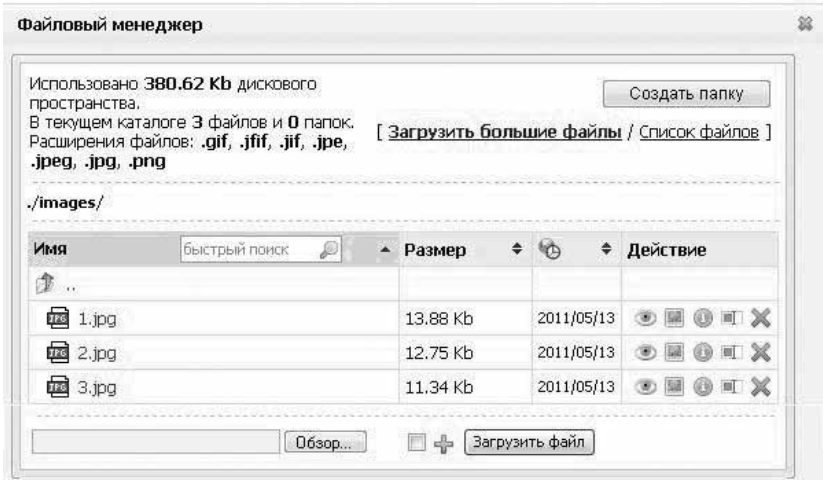
Загальне (*Общее*) ⇨ *Вхід у панель управління* (*Вход в панель управления*). Далі – *Редактор страниц* ⇨ *Общие настройки* і ввести бажану назву у відповідне поле (див. мал.).

Наповнення сайту контентом

Тепер треба наповнити вмістом головну сторінку сайту. Для цього клацнемо кнопку редагування у візуальному редакторі (із зображенням ока).



У полі введення тексту сторінки вже є напис **$LAST NEWS$**, який позначає місце, де будуть виводитись останні новини сайту. Поряд з цим написом (або замість нього) вставляємо текст, що буде знаходитись на даній сторінці, і приводимо до відповідного вигляду за допомогою панелі редагування – вирівнюємо абзаци, вибираємо



гарнітуру, колір і т.ін. Панель форматувння схожа на відповідну панель у Microsoft Word. По закінченні редагування сторінки слід вибрати команду *Сохранить*.

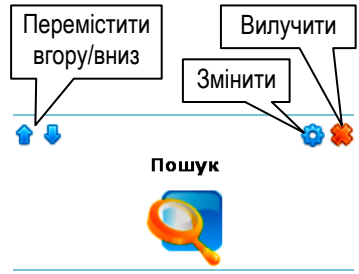
Розглянемо, як додати до тексту ілюстрації (малюнки, фотографії). Щоб вставити зображення на сторінку треба натиснути на кнопку у вигляді картинки, з'явиться вікно *Изображение*. Клацнувши кнопку з зображенням папки поряд з полем *Путь*, активуємо *файловий менеджер*. Далі, клацнувши кнопку **Обзор** обираємо файл із зображенням на своєму комп'ютері, і вибираємо команду *Загрузить файл* (див. мал. на попередній сторінці).

Ім'я файлу з зображенням з'явиться у списку файлового менеджера. Клацаємо по ньому і у вікні *Image* уточнюємо налаштування: відступи, тип рамки, розмір картинки. Якщо хочемо щоб текст обтікав зображення, ставимо вирівнювання ***«ліворуч»*** і вибираємо команду *Вставить* (*Insert*).

Редагування бічних блоків і меню сайту

Для того, щоб відредагувати, додати або вилучити бічні блоки, на панелі адміністратора слід вибрати команду *Конструктор* ⇨ *Включити конструктор* (*Включить конструктор*). Бічні блоки змінять свій вигляд: біля них з'являться кнопки керування, призначення яких зрозуміле з малюнка.

Блок *Меню* має додаткову кнопку редагування у вигляді ключа. Якщо на неї натиснути, то з'явиться вікно *Управление меню*. Тут можна видалити пункт меню, додати або змінити порядок простим перетягуванням.



Також у режимі конструктора можна додати новий блок. Для цього у режимі конструктора слід вибрати команду *Конструктор* ⇨ *Додати блок* (*Добавить блок*), дати йому назву і перетягти в потрібне місце. Далі натиснувши на значок редагування і заповнити блок вмістом. І після цього вибрати команду *Конструктор* ⇨ *Зберегти зміни* (*Сохранить изменения*).

Додавання нової сторінки на сайт

Щоб додати на сайт нову сторінку, на панелі адміністратора вибираємо команду *Додавання* (*Добавление*) ⇨ *Редактор сторінок* (*Редактор страниц*). У вікні редактора вписуємо назву сторінки, вставляємо вміст і, подібно до головної сторінки, вибираємо команду *Зберегти* (*Сохранить*).

Після додавання кожної сторінки у меню автоматично додається відповідний пункт.

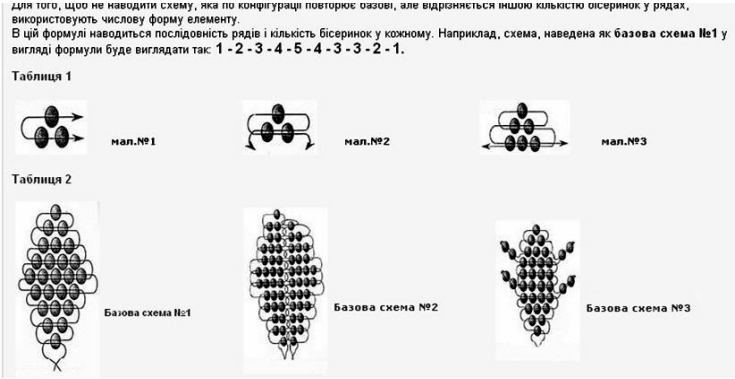
При виникненні запитань щодо розробки сайту, відповідь можна знайти на офіційному *форумі uCoz* за адресою *http://forum.ucoz.ru*.

Структурування інформації за допомогою таблиць

У звичайних текстових редакторах таблиці використовують для наочного подання числової чи текстової інформації.

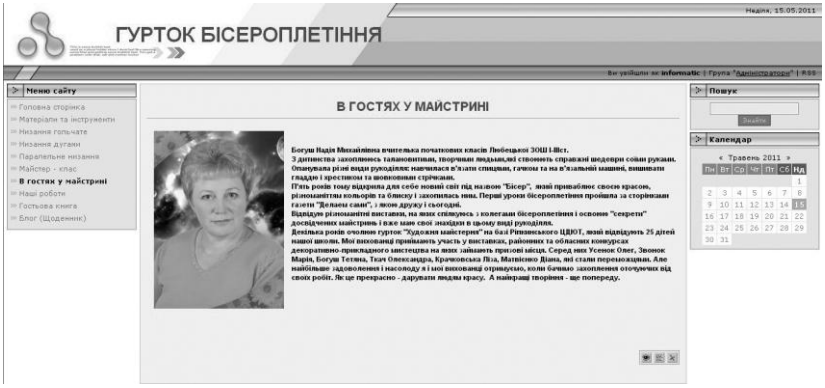
У веб-дизайні роль таблиць дещо ширша. Часто їх використовують для позиціювання графічних чи інших об'єктів на веб-сторінці. При цьому межі таблиці роблять невидимими, а в клітинках розташовують малюнки, текст, інші таблиці тощо.

Наприклад, відкривши сайт http://biser.ucoz.ua/ та зайшовши на сторінку «Паралельне низання», відкриваємо візуальний редактор, під назвою сторінки вставляємо завчасно підготовлений текст. Після тексту потрібно вставити декілька малюнків.



Вирівняємо їх за допомогою таблиці розміром 1х3. У кожну комірку вставимо малюнок, який раніше завантажили у папку **images** файлового менеджеру. Вибравши команду **Зберегти**, отримаємо сторінку, фрагмент якої зображено на малюнку.

Далі, відкривши сторінку «В гостях у майстрині», переходимо



у візуальний редактор. У візуальному редакторі вставляємо таблицю 1х2. У ліву комірку вставляємо фото, а в праву текст. Вибравши команду **«Зберегти»** та перейшовши на цю сторінку сайту, побачимо результат (див. мал.).

Питання для самоконтролю:

1. *Що називають HTML-редактором?*
2. *На які категорії поділяють HTML-редактори?*
3. *Які можливості має графічний HTML-редактор?*
4. *Назвати основні особливості візуального редактора KompoZer.*
5. *Що називають системою керування вмістом (контентом) веб-сайту?*
6. *Опишіть порядок реєстрації на сайті www.ucoz.ua ?*
7. *Як створити сайт на uCoz?*
8. *Що таке вебтоп?*
9. *Як виконати редагування сайту uCoz?*
10. *Як змінити назву сайту uCoz?*
11. *Як завантажити малюнок до файлового менеджеру сайту uCoz?*
12. *Як наповнити сторінку сайту текстовою та графічною інформацією?*
13. *Як відредагувати бокові блоки сайту uCoz?*
14. *Як відредагувати меню сайту uCoz?*
15. *Як структурувати інформацію на сайті uCoz за допомогою таблиць?*

4.3. ***Принципи дизайну веб-сторінок***

Як оформляти текст веб-сторінки

При наповненні веб-сторінок інформацією, треба враховувати, що відвідувачі, зазвичай, приходять на сайт, щоб отримати інформацію. Оскільки основним засобом подання інформації є текст, то від того, як він буде написаний і оформлений, залежить загальне враження від сайту і його корисність для відвідувачів.

При розміщенні тексту на веб-сторінці слід дотримуватись таких правил:

- Текст повинен бути лаконічним, зрозумілим при читанні з першого разу. З тексту бажано вилучити всі зайві звороти і слова. Заголовки повинні бути зрозумілими і інформативними. При перерахуванні різних об’єктів, понять або подій користуйтесь маркованими списками. Намагайтесь економити мережний час відвідувачів сайту і пам’ятайте, що більшість користувачів спочатку проглядають сторінку «по діагоналі», звертаючи увагу лише на заголовки і виділені слова.
- Тексти значного об’єму краще розбивати на кілька сторінок, які будуть пов’язані гіперпосиланнями. Якщо текст має заголовки різного ієрархічного рівня, то цю ієрархію варто відобразити в структурі зв’язаних між собою сторінок. Це допоможе

користувачу краще орієнтуватися у великому за обсягом текстовому матеріалі.

- Тексти веб-сторінок повинні бути написані грамотно. Неграмотно написаний текст ставить під сумнів якість наведених у ньому даних.

Основні правила веб-дизайну

Відомо, що будь-яка технологія, будь-який творчий процес підкоряється певним правилам і законам, невиконання яких призводить до негативних наслідків. Веб-сайт як комплекс художніх рішень також підлягає цілому ряду правил «хорошого тону», яких слід дотримуватись, щоб сайт виглядав гарно.

При плануванні і створенні веб-ресурсу важливо пам'ятати, що головним критерієм, на який треба орієнтуватись при розробці сторінок, є зручність для кінцевого користувача. Звідси витікає необхідність певної стандартизації підходів до веб-дизайну, виробленню правил, які можуть задовольнити всю вашу потенційну аудиторію.

Фахівці вважають, що якісні сайти повинні коректно відображатись при роздільній здатності екрану 640x480 крапок з колірною палітрою на 256 кольорів.

Зауваження. При відображенні HTML-документа, розрахованого на перегляд з роздільною здатністю екрану 800x600 точок, на комп'ютері, налаштованому на режим 640x480 точок, у нижній частині головного вікна браузера з'являється горизонтальна смуга прокрутки, що значно утруднює читання документу і викликає нарікання користувачів.

Для того щоб веб-сторінка коректно відображалась при використанні екранної палітри на 256 кольорів, включаючи до складу документу графічні елементи, треба подавати якомога більше графіки у форматі GIF і лише там, де це справді необхідно, – у форматі JPG.

Веб-сторінка повинна однаково відображатись у популярних браузерах: IE, Opera та Mozilla FireFox тощо.

Багато користувачів використовують для виходу в Інтернет dial-up та інші повільні види з'єднання. Тому для них швидкість передачі даних грає вирішальну роль. Виходячи з цього, всі сторінки веб-сайту, а також всі інтегровані в них графічні та інтерактивні елементи мають бути мінімальними за обсягом.

Це досягається шляхом використання при розробці сайту спеціальних графічних компресорів, а також ряду прийомів, які дозволяють видалити з HTML-документа зайвий код і цим самим зменшити розмір HTML-файлу.

Створена веб-сторінка повинна обов'язково включати навігаційні елементи, які охоплюють всі розділи сайту, причому ці елементи повинні бути завжди на виду. Якщо вони розміщені у верхній частині сторінки і випадають з поля зору після прокрутки вниз (скролінгу), треба продублювати їх у нижній частині сторінки. Графічні посилання і активні елементи слід повторити у текстовій формі, розраховуючи на користувачів, у браузерах яких відключено відображення графіки або відсутня підтримка Java.

Треба намагатись витримати весь проект в одному дизайнерському стилі, оформляти різні його розділи таким чином, щоб загальне художнє вирішення було однаковим для всього сайту.

Не треба використовувати на одній веб-сторінці більше ніж три різних шрифти, включаючи шрифти, які застосовуються у графічних елементах.

Слід використовувати лише коректні колірні схеми і не застосовувати при оформленні документів більш ніж три різних кольори. Виняток тут можна допустити лише для напівтонів одного й того ж кольору, що застосовуються, наприклад, при контекстному виділенні рядків у таблицях.

З точки зору людської психології поєднання кольорів може значною мірою впливати на сприйняття наведеної інформації. Тому при підборі кольорів, наприклад тексту з фоном, рекомендується виходити принаймні з точки зору здорового глузду: текст повинен легко читатись.

Підібрати швидко колірну схему для сайту дозволяють спеціальні інструменти. Наприклад, за адресою http://kuler.adobe.com/ розміщено онлайновий програмний засіб для роботи з колірними схемами від компанії Adobe. А користувачі браузера Mozilla Firefox можуть встановити доповнення Colorzilla для подальшого використання без підключення до Інтернету.

Проблема оцінки сайту

Об'єктом мови розмітки HTML є структура документа, а не його зовнішній вигляд. Мова HTML призначена саме для розмітки логічної, а не візуальної структури, а зовнішній вигляд документа у вікні браузера є лише побічним ефектом логічної розмітки.

Проте цього принципу дотримувались лише на початку існування мови. Надалі до HTML було уведено велику кількість візуально орієнтованих засобів, які не мають ніякого структурного значення і застосовуються виключно для управління зовнішнім виглядом документа у графічному середовищі.

Обмежуючись лише набором тегів і незважаючи на оформлення, можна одержати для свого документа цілком логічний, послідовний і строгий дизайн. Саме такий академічний стиль характерний для освітніх та наукових сайтів і є найкращим рішенням для тих, кому дизайн не надто важливий.

Стрімке зростання кількості ресурсів мережі Інтернет і відсутність універсальних стандартів призвели до усвідомлення значення практично апробованих принципів проектування. Важливим завданням є визначення специфічних елементів або параметрів мережі, які впливають на комунікацію. Виділяють такі параметри, які найбільш часто використовуються для опису і оцінки мережевих сайтів:

- оформлення документа (використання шрифтів, формату сторінок і кольору для передавання змісту тексту);
- структура сайту;
- використані засоби розробки;
- зміст повідомлень;
- привабливість;
- доступність.

Ці параметри вказують напрямки, на які треба звертати особливу увагу під час розробки сайтів.

Розробляючи сайт, слід також звернути увагу на результати дослідження впливу традиційних друкарських знаків (шрифту) на швидкість і зручність роботи з документом:

- читачі однаково читають шрифти як із засічками, так і без них;
- довжина рядка впливає на швидкість роботи;
- текст, набраний курсивом або лише великими літерами, більш складний для читання, ніж звичайний текст;
- контрастність друкарських елементів (середній розмір, чіткість, яскравість) більш важливі, ніж малюнок шрифту;
- використання білих літер на кольоровому фоні є ефективним засобом, що привертає увагу до тексту.

Структура сайту є основою, на якій розробники створюють навігаційні засоби. Дослідження проблем, які виникали у

користувачів внаслідок неефективності структури сайту, дозволили виділити такі загальні проблеми:

- дезорієнтація користувача (незнання того, де користувач знаходиться відносно структури сайту);
- відволікання від головної мети;
- ненадійність навігації (функції або ознаки навігації – колір, форма кнопок і гіперпосилань – змінюються від сторінки до сторінки).

Дослідження свідчать, що користувачі чекають близько 8 секунд, поки що-небудь з'явиться на відкритій ними сторінці, потім готові чекати до 20 секунд, поки сторінка повністю завантажиться, і ще 6 секунд їм треба на те, щоб вирішити, продовжувати перегляд сайту чи ні.

Стосовно розміщення інформації на сторінках є такі рекомендації: на головній сторінці має бути мало тексту і посилання на інші сторінки сайту; основний обсяг інформації вміщують на інших сторінках. Існує тест на змістовність сторінки: якщо її роздрукувати, то не менше ¾ повинно бути заповнено інформативним текстом (це не стосується головної сторінки).

Параметри оцінки якості сайту

Зовнішній вигляд

Шрифти: придатність шрифту для електронного подання; розбірливість шрифту; виправданість виділення (грубий шрифт або підкреслення), достатність контрасту між шрифтом і фоновим кольором.

Формат: наявність відповідного простору, довжина рядка (занадто довгий або занадто короткий); наявність достатніх пробілів, які оточують текст і графіку.

Колір: відповідність кольорів, застосовуваних для конкретної аудиторії і з певною метою.

Структура сайту

Підтримка навігації: тип навігаційної підтримки, забезпечуваний сайтом, її достатність; корисність додаткових видів навігаційної підтримки для споживача.

Інформаційна архітектура: структура сайту; можливість «інтуїтивної» роботи із сайтом; взаємозв'язки та ієрархія.

Застосовувані засоби

Відповідність застосування засобів: інтерактивність гіпертексту, фрейми, мультимедіа, звук, відеозапис.

Обмеження цілісності: урахування обмеження цілісності для користувача, зокрема, часу завантаження графіки, часу завантаження вмісту сторінки, версії програм перегляду, апаратних платформ, швидкості з'єднання.

Повідомлення

Зміст: забезпечення повного і достатнього змісту; точність, повнота і актуальність змісту; стиль викладу змісту.

Зрозумілість: відповідність повідомлення цільовій аудиторії; наявність у споживачів знань, необхідних для розуміння тексту.

Привабливість

Рівень складності для аудиторії; підтримка інтересу до сайту; релевантність інформації, пов'язана з інтересами і мотивацією користувачів; досягнення користувачами намічених цілей; повернення користувачів на сайт.

Культурні відмінності: відображення сайтом культурних відмінностей, включаючи мову, жести, піктограми, соціальні звичаї.

Потреба у спеціальному доступі: повідомлення про потребу у спеціальному доступі для людей з фізичними вадами, включаючи осіб з поганим зором, слухом, рухливістю та ін.

Питання для самоконтролю:

1. *З якою метою відвідувачі заходять на веб-сайт?*
2. *Які є правила оформлення тексту на веб-сторінці?*
3. *Поясніть основні правила веб-дизайну.*
4. *Якими параметрами характеризується зовнішній вигляд сайту?*
5. *Якими параметрами можна охарактеризувати структуру сайту?*
6. *Які особливості впливають на привабливість веб-сайту?*
7. *Якими параметрами можна охарактеризувати повідомлення розміщене на сайті?*
8. *В чому полягають проблеми неефективної структури сайту?*
9. *Назвати параметри, які часто використовуються для опису і оцінки сайтів.*
10. *Назвати браузери в яких коректно повинен відображатись веб-сайт.*
11. *Який формат графічних файлів доцільніше використовувати при створенні сайту?*
12. *Які вимоги існують до навігаційних елементів веб-сайту?*

4.4. Технології Веб 2.0. Веб-спільноти

Веб 2.0 – друге покоління мережевих сервісів Інтернету. На відміну від першого покоління сервісів, орієнтованих, переважно, на читання, Веб 2.0 дозволяє користувачам діяти спільно – обмінюватися інформацією, зберігати посилання та мультимедійні документи, створювати та редагувати публікації, тобто відбувається

соціальна взаємодія. Тому технології Веб 2.0. ще називають соціальними сервісами Інтернету.

Принципи функціонування Веб 2.0

Важливим аспектом Веб 2.0 є зміна пріоритетів та акцентів у використанні технологій та задоволенні потреб користувачів. Так, раніше Інтернет орієнтувався на розвиток технологій комп'ютерної взаємодії, а Веб 2.0 розвиває технології орієнтовані на користувачів. Нові інформаційні технології суттєво впливають на колективні способи спілкування, мислення та дій.

Головним принципом Веб 2.0 є невід'ємне право користувачів самостійно створювати контент, маніпулювати ним та керувати зв'язками між своїми та чужими матеріалами, отже мова йде про скоординовану активність окремих користувачів щодо формування та наповнення мережі контентом.

Активність характеризується підвищеним рівнем комунікації, координації та включення користувачів у процес використання та створення ресурсів, поповнення сервісів, визначення стратегії розвитку ресурсу в цілому.

Основні принципи функціонування технологій Веб 2.0:

Простота та доступність. Сюди відноситься максимальна легкість опанування технологій користувачами-початківцями, функціональність інтерфейсу, простота навігації, наочність та не переобтяженість дизайну.

Контент, який створюють користувачі. Інформацію (текстову, мультимедійну тощо) розміщує на сайті будь-хто з користувачів, а інші відвідувачі можуть використовувати, поліпшувати, оцінювати, коментувати, тобто контролювати процес формування інформаційних ресурсів.

Архітектура співучасті і, як наслідок, мережеві ефекти, а саме ефект «колективного інтелекту», принцип «координація замість контролю», принцип «деякі права застережені» («some rights reserved») – мінімальний захист інтелектуальної власності.

Фолксономія – народна класифікація, практика колективної (спільної) категоризації інформації (текстів, фото, посилань, відео тощо). Для класифікації використовуються ключові слова, які називаються мітками або тегами. Ключові слова задають самі користувачі. Поширена форма візуалізації сукупності всіх тегів ресурсу – «хмаринка тегів». «Хмаринка тегів» відображає область інтересів авторів, які розміщують контент на даному ресурсі.

Вибравши тег, можна отримати список категоризованих матеріалів з даної теми.

Веб, як платформа. Надання можливості користувачам застосовувати прикладну програму виключно за допомогою веб-оглядача.

Сервісність. Проекти Веб 2.0 пропонують користувачам вирішення конкретної задачі: організацію власної фото- або відео-галереї, пошук інформації, створення набору закладок на улюблені сайти та інше.

Постійна бета. Сервіси покращуються постійно, в режимі реального часу, більше не існує релізів програмного забезпечення.

Класифікація соціальних сервісів Інтернету

Класифікацію соціальних сервісів здійснено за видами діяльності членів мережевого співтовариства. До основних типів відносять:

- соціальні пошукові системи;
- засоби для збереження закладок;
- соціальні сервіси збереження мультимедійних ресурсів;
- мережеві щоденники (блоги);
- вікі (wiki);
- карти знань;
- соціальні геосервіси.

Соціальні пошукові системи – системи, які дозволяють користувачам самим визначати в якому напрямку вести пошук, які сайти переглядати насамперед, на які слова звертати першочергову увагу і яким чином подавати знайдені результати. Пошук можна адаптувати до певної тематики та співтовариства.

Засоби для зберігання закладок – призначені для зберігання посилань на веб-сторінки, які Ви регулярно відвідуєте. На відміну від традиційних методів зберігання закладок, нові соціальні сервіси дозволяють:

- додавати посилання з будь-якого комп'ютера, підключеного до мережі Інтернет;
- посилання будуть доступні з будь-якого комп'ютера, підключеного до мережі Інтернет;
- позначати кожну закладку одним або кількома тегами (мітками-категоріями).

Засоби зберігання закладок відносяться до так званих народних класифікаторів.

Соціальні сервіси збереження мультимедійних ресурсів – сервіси мережі Інтернет, які дозволяють безкоштовно зберігати, класифікувати, обмінюватися цифровими фотографіями, аудіо- і відеозаписами, текстовими файлами, презентаціями, а також організовувати обговорення контенту.

Мережеві щоденники (блоги) – сервіс Інтернету, що дозволяє будь-якому користувачеві вести записи з довільної тематики. За аналогією з особистими щоденниками блоги називають мережними щоденниками.

Вікі (Wiki) – соціальний сервіс, що дозволяє будь-якому користувачеві редагувати текст сайту (писати, вносити зміни, видаляти, створювати посилання на нові статті).

Різні варіанти програмного забезпечення вікі-систем дозволяють завантажувати на сайти зображення, файли, що містять текстову інформацію, відеофрагменти, звукові файли і т.д.

Карти знань (англ. Mind map) – спосіб зображення процесу загального мислення за допомогою схем. В українських перекладах термін може звучати по-різному – карта розуму, карта пам'яті, інтелект-карта, майнд-меп тощо.

Соціальні геосервіси – сервіси мережі Інтернет, які дозволяють з досить високою точністю знаходити, позначати, коментувати, доповнювати фотографіями різні об'єкти на карті

Землі. Використовуються реальні дані, отримані за допомогою штучних супутників Землі.

Віртуальні співтовариства (спільноти)

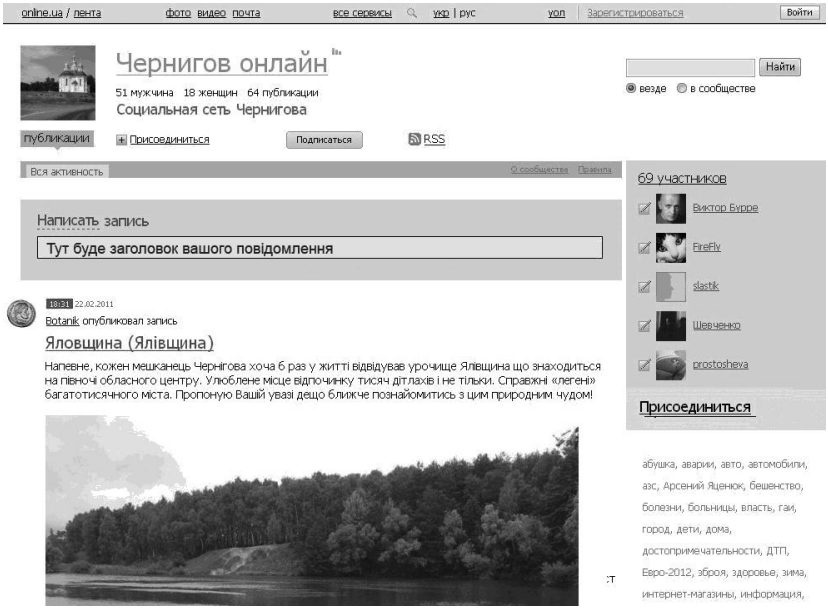
Віртуальні співтовариства *(англ. virtual communities, e-communities) — новий тип співтовариств, які виникають і функціонують в електронному просторі (перш за все за допомогою мережі Інтернет) з метою сприяння вирішенню професійних, політичних завдань, задоволення інтересів у мистецтві, дозвіллі тощо.*

Віртуальні співтовариства (спільноти) — це цілком реальні групи людей, які для обміну інформацією використовують електронні засоби та мережі. У свою чергу такі співтовариства теж можуть об'єднуватись у мережеві структури на підставі спільних інтересів.

В мережі Інтернет є багато сервісів, які надають можливість створювати віртуальні співтовариства (спільноти).

Наприклад: mail.ru, meta.ua, online.ua, Я.ru, Facebook, тощо.

Щоб стати учасником співтовариства (спільноти) треба обрати сервіс де вже створене співтовариство, яке вас цікавить та зареєструватись на цьому сервісі і приєднатись до співтовариства.



На малюнку зображено сторінку співтовариства «Чернігів онлайн».

Для створення співтовариства (спільноти) на сервісі online.ua, треба зайти на цей сервіс та зареєструватись на ньому. Далі, перейти за посиланням «Створити спільноту» і пройти процедуру створення спільноти, наприклад: «Наша школа», «Мій населений пункт» тощо.

Кожна спільнота створена в мережі Інтернет керується певними правилами.

Орієнтовні правила спільноти

1. Всі учасники спільноти, включно з лідером, діють на основі «Угоди користувача».

2. Створення спільноти:

2.1. Тематика спільноти встановлюється при її створенні, і не може бути змінена в подальшому.

2.2. Лідером спільноти стає користувач, який її створив.

2.3. Спільнота не є власністю якогось користувача, але керує нею лідер спільноти.

2.4. Лідер спільноти несе відповідальність за її створення та всю активність учасників згідно законодавства України.

3. Обмеження:

3.1. Не допускається створення спільнот, які порушують законодавство України та загальноприйняті етичні правила.

3.3. Не допускається створення спільнот такої тематики:

- політика;
- релігія, культи.

4. Діяльність спільноти:

4.1. Активність спільноти повинна проводитись у рамках вказаної тематики.

5. Управління спільнотою:

5.1. Лідер спільноти володіє виключним правом на управління спільнотою. Він встановлює основні правила спільноти, які не можуть суперечити загальним правилам спільнот. Лідер спільноти має доступ до налаштувань спільноти; призначає редакторів спільноти; має доступ до управління публікаціями та коментарями всіх учасників спільноти. До повідомлень спільноти лідер може залишати коментарі довжиною до 500 символів.

5.2.Редактор спільноти призначається лідером спільноти. Редактор має доступ до управління публікаціями і коментарями учасників спільноти. Записи редактора не підлягають промодерації

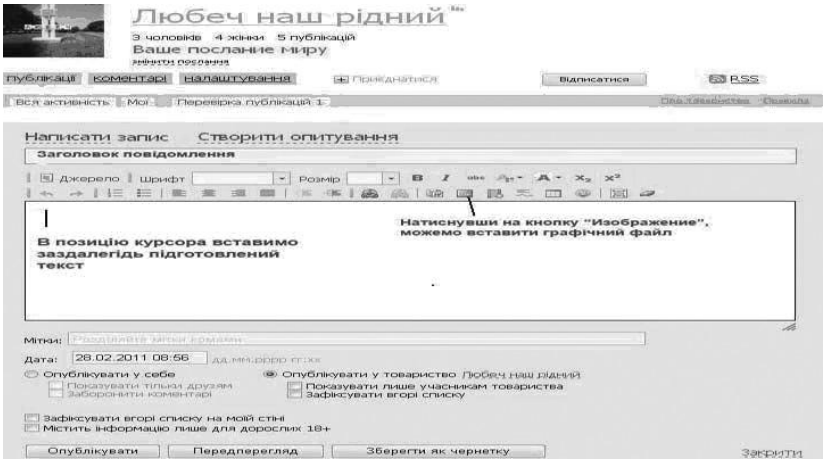
лідером спільноти, але лідер спільноти має можливість вилучати записи редакторів. До записів спільноти редактор спільноти може залишати коментарі довжиною до 500 символів.

Публікація на сайті спільноти повідомлень і файлів

Для опублікування на сайті спільноти повідомлень та файлів, треба:

- зайти на сторінку спільноти, де ви зареєстровані;
- пройти процедуру входу (у форму ввести логін і пароль).

У вікні візуального редактора, який видно на малюнку, можна вставляти заздалегідь підготовлений текст, графічні файли, відеофайли, структурувати повідомлення за допомогою таблиць.



Після завершення вставки тексту, графічних файлів, слід натиснути на кнопку **Опублікувати** і повідомлення буде опубліковане у спільноті.

Питання для самоконтролю:

1. *Яка відмінність між 1-м та 2-м поколіннями веб-сервісів?*
2. *В чому полягає суть ключових факторів успіху Веб 2.0?*
3. *Назвати головний принцип функціонування Веб 2.0*
4. *Назвати принципи функціонування технологій Веб 2.0*
5. *Назвіть типи соціальних сервісів Інтернету.*
6. *Що таке соціальні пошукові системи?*
7. *Які соціальні сервіси Інтернету тобі відомі?*
8. *Що таке віртуальне співтовариство (спільнота)?*
9. *Які ти знаєш сервіси, що підтримують технологію Веб 2.0?*
10. *Опиши загальні правила спільноти.*
11. *Хто такий лідер співтовариства (спільноти)?*
12. *Хто такий редактор співтовариства (спільноти)?*

13. *Як створити спільноту (співтовариство) на online.ua/?*
14. *Створення яких тематик уникають у спільноті?*
15. *Опишіть порядок опублікування повідомлення у спільноті.*

4.5. Вікі-технології. Спільна робота з документами

Вікі-сервіс *– це сайт, що дозволяє відвідувачам редагувати та обговорювати розміщені на ньому матеріали, створювати нові сторінки.*

В основі ***вікі-технології*** лежить принципово нова ідея – використання «колективного розуму».

Сторінки на ***вікі-сервісі*** називаються статтями.

Перша ***вікі-система*** називалася WikiWikiWeb. Її назвали на честь автобусного маршруту, що діє в аеропорті Гонолулу (Гаваї) – «Wiki Wiki», що означає «швидко-швидко». Порівняльний аналіз деяких можливостей Вікі та звичайних веб-сайтів наведено в таблиці 1.

Веб-сайт	***Вікі-сайт***
Наповненням займається одна людина	Наповненням займається спільнота
Дизайн має значення	Дизайн не має значення
Необхідні знання html–тегів	Необхідні знання простих Wiki-тегів
Оновлення через FTP-протокол	Оновлення через веб-протокол
Створення нових сторінок передує створенню посилань	Посилання на нові сторінки передують створенню нових сторінок
При оновленні сайту попередня інформація знищується	Всі сторінки сайту залишаються в базі даних
Для кожної сторінки всередині сайту можна отримати перелік сторінок, на які вона посилається	Для кожної сторінки можна отримати список сторінок, на які вона посилається, та список сторінок, які містять посилання на сторінку
Карта сайту створюється централізовано	Карта сайту створюється автоматично та відображає інтереси учасників спільноти
Індивідуальна редакційна політика	Колективна редакційна політика

Вікі характеризується такими ознаками:

- можливість багаторазового виправлення засобами самого середовища, без використання зовнішніх редакторів;
- особлива мова розмітки – так звана вікі-розмітка, яка дозволяє легко та швидко створювати в тексті структурні елементи та оформлювати окремі елементи;
- відображення змін відразу після їх внесення;

- розподіл вмісту на іменовані сторінки;
- колективна робота;
- облік змін (облік версій) тексту – можливість порівняння редакцій та відновлення попередніх версій.

Вікі-технологія – потужний інструмент для швидкого створення та редагування колективного гіпертексту. При цьому дописувач або група учасників проекту не відволікаються на HTML-кодування та встановлення зв'язків між різними частинами тексту, адже цю роботу виконує спеціальний програмний агент. Це дозволяє користувачам, які не мають спеціальних знань, створювати та редагувати статті.

Іншою важливою рисою вікі є контроль за версіями статей. Всі виправлення фіксуються та супроводжуються інформацією щодо часу, дати та автора правки. У систему вбудований модуль контролю версій, що дозволяє порівнювати початковий та відредагований текст статті. Будь-яка версія статті може бути відновлена. Оскільки такі зміни супроводжуються «підписами» авторів, користувач може оперативно зв'язатися з учасниками, що редагували статтю, та обговорити подальшу спільну роботу над нею.

Крім функцій створення та публікації матеріалів підтримуються засоби колективної та індивідуальної комунікації. Інформаційні матеріали та середовище обміну повідомленнями знаходяться в єдиному просторі. Кожна стаття має свою сторінку обговорення. Окрім того, такі функції, як облік змін, порівняння версій та журнал правок, створюють віртуальний простір, в якому члени мережевого співтовариства можуть спостерігати за спільною діяльністю.

Порівняно зі звичайними сайтами, у вікі діє інша ідеологія створення нових сторінок. За правилами побудови веб-сайтів, спочатку створюється сторінка, а вже потім робиться посилання на неї. У вікі ж *посилання на ще не створені сторінки* – не тільки норма, але й єдиний можливий спосіб створення сторінок: для створення нової сторінки спочатку необхідно вказати в тексті посилання на неї.

Взаємозв'язок сторінок та колективні зусилля – саме ці риси виділяють вікі-технологію серед інших соціальних сервісів. Учасники з різних географічних областей, фахівці з різних галузей знань можуть незалежно один від одного працювати над

створенням статей. Взаємодія між учасниками встановлюється через взаємодію між статтями. Взаємодія між статтями встановлюється автоматично у відповідності до правила – назва статті є потенційним посиланням на цю статтю в тексті інших статей.

Переваги вікі-сервісу:

- для редагування тексту на вікі-сайті не потрібно знання HTML;
- наявність власної мови розмітки, яка, на відміну від мови HTML, більш проста і зручна у використанні;
- для введення і редагування матеріалу використовується простий on-line редактор;
- внесені виправлення миттєво відображаються на сайті, не потрапляючи на попередню перевірку в руки редактора або адміністратора сайту;
- в середовищі on-line редактора є панель інструментів, яка робить написання й форматування тексту справою не більш складною, ніж у Word'і.
- вікі-системи дозволяють стежити за всіма змінами на сайті;
- надається місце для дискусії з приводу будь-якого опублікованого матеріалу (зверху кожної статті є вкладка Редагувати);
- можливість присвоїти статті певну категорію дозволяє миттєво знаходити матеріали, що належать до цієї категорії;
- використання механізму шаблонів дозволяє швидко створювати статті або фрагменти статей однієї структури, передавати новини на сторінки користувачів в межах певного проекту, змінюючи дані лише в одній статті-шаблоні.

Недоліком вікі-сервісу є неможливість одночасного редагувати статті кількома користувачами.

Найвідомішим прикладом застосування вікі-технології є ***Вікіпедія*** − багатомовна онлайнова енциклопедія. Розглянемо на її прикладі деякі обмеження вікі.

Обмеження статті з погляду читацького сприйняття

Кожна стаття Вікіпедії перебуває в процесі «еволюції» і, імовірно, буде рости й далі. Після того, як одні автори статті закінчують свою роботу, інформацію продовжують додавати інші. Це не є проблемою, тому що, із практичної точки зору, Вікіпедія має у своєму розпорядженні необмежений обсяг. Разом з тим,

занадто довгі статті можуть бути незручними для читання, навігації й редагування.

Середній читач зазвичай утомлюється при прочитанні вже 6-10 тис. слів, що приблизно відповідає 40-50 тис. знаків видимого (чистого) тексту. Якщо ж стаття суттєво довша, то для полегшення сприйняття може знадобитися винести частину інформації в окремі підстатті, залишивши в основній статті лише коротке резюме. Це слід робити у всіх випадках, коли обсяг видимого (чистого) тексту починає перевищувати зазначені розміри.

Рекомендується створювати статті 20000–30000 знаків, що відповідає орієнтовно 4-6 сторінкам А4 і відповідно 8-12 екранним сторінкам на комп'ютері.

Технічні обмеження

Повідомлення про повний обсяг вікі-тексту з'являється при кожному відкритті вікна редагування, починаючи з того моменту, коли він перевищив 32 КБ.

Статті, що перевищують 400 КБ, можуть відображатися некоректно або навіть не відображатися взагалі при використанні специфічних видів доступу й деяких версій браузерів. Такі довгі статті настійно рекомендується розбивати на дві або більше статей.

Обмеження за змістом

У Вікіпедії діє принцип: «Одна стаття – одна ідея!». Наприклад, вікі-стаття про таблицю Мендєлєєва не може висвітлювати біографічні відомості визначного хіміка. Це – не недолік, а скоріше корисна особливість, адже при цьому зміст статей стає цілком прогнозованим, що спрощує роботу з енциклопедією в цілому.

Середовище GoogleDocs

Кожен з зареєстрованих користувачів пошукового сервісу Google має можливість використовувати засоби середовища GoogleDocs для оприлюднення документів різних типів та спільної роботи над ними.

Програми, що відкриваються безпосередньо у вікні браузера дають можливість користувачам працювати з текстовими, табличними документами та презентаціями і здатні у більшості випадків замінити Microsoft Word, Excel та PowerPoint. Крім цього вони мають щільну інтеграцію з онлайновими засобами Google Gmail (Пошта) і Groups (Пошта).

З **документами** можна виконувати такі операції:

- Завантажувати документи Word, OpenOffice, RTF, HTML чи текстові файли (або сотворювати нові документи).
- Використовувати простий редактор WYSIWYG для форматування документів, перевірки правопису тощо.
- Запропонувати іншим користувачам (наприклад, електронною поштою) редагування або перегляд документів та таблиць.
- Редагувати документи разом з іншими людьми.
- Переглядати історію версій документів і таблиць і переходити до будь-якої версії.
- Публікувати документи в Інтернеті для всіх користувачів у вигляді веб-сторінок або поміщати документи у свій блог.
- Відсилати документи електронною поштою як вкладення.

Що можна робити з **таблицями:**

- Імпортувати та експортувати файли формату .xls, .csv, .txt та .ods і вивести як .pdf і html.
- Форматувати і редагувати формули в таблицях.
- Чат в режимі реального часу з користувачами, яким ви надали доступ до таблиці.
- Ввести таблицю або фрагмент таблиці, у ваш блог чи на сайт.

Ось що ви можете зробити з **презентаціями:**

- Редагувати презентації разом з друзями і співробітниками.
- Імпортувати наявні презентації з файлів типів .ppt та .pps.
- Експортувати презентацію за допомогою команд меню **Файл** ⇨ **Зберегти як PDF** і **Зберегти як PPT.**
- Змінити презентацію за допомогою простого WYSIWYG-редактора.
- Включити зображення і відео.
- Перглядати презентацію в режимі реального часу.
- Публікувати і вставляти презентації у веб-сайт.

Переваги сервісу GoogleDocs

- Використовуючи цей сервіс, користувач не лише економить на офісному пакеті, а повністю перебудовує свою роботу з документами.
- Всі важливі документи, які бажано мати «під рукою», можна тепер зберігати в мережі, редагувати з будь-якого комп'ютера і легко та швидко надсилати електронною поштою.

Але головне – це можливість спільної роботи з документами. Ці переваги дозволяють по-новому організувати роботу у мережевому проекті.

Недоліки сервісу та обмеження його застосування

Обмеження за обсягами:

Документи. Найбільший розмір документа – 500 КБ, а розмір вставленого зображення – 2 МБ.

Таблиці. В кожній таблиці може бути не більш, ніж 10 000 рядків, 256 стовпців, 100 000 комірок і 40 аркушів.

Презентації. Найбільший розмір файлів у форматі .ppt і .pps не повинен перевищувати 10 МБ; розміри файлів, завантажених з Інтернету, не повинні перевищувати 2 МБ; розміри файлів, що надсилаються електронною поштою, не повинні перевищувати 500 КБ.

Можливості керувати доступом користувачів

Власники. Можуть редагувати документи, таблиці і презентації, і запрошувати співавторів і спостерігачів. Можуть знищити документи, таблиці і презентації і закрити таким чином доступ співавторам і спостерігачам. Слід звернути увагу: щоб повністю знищити документ, таблицю чи презентацію та закрити доступ до них, їх треба знищити, а потім очистити кошик (Empty Trash).

Співавтори. Можуть редагувати документи, таблиці і презентації. Можуть запросити або вилучити інших співаторів і спостерігачів (якщо власник дав таке право). Можуть експортувати копію документу, таблиці і презентації на власні носії.

Спостерігачі. Можуть бачити більшість останніх версій документу, таблиці і презентації, але не можуть зробити жодних змін. Можуть експортувати копію документу, таблиці і презентації на власні носії.

Обмін з користувачами Google. Запрошуючи людей редагувати або переглядати документ, таблицю чи презентацію, слід використовувати список Gmail-контактів, щоб забезпечити простий клавіатурний доступ до їхніх електронних адрес та імен. Завдяки цьому, завжди, коли ви друкуєте ім’я того, хто вже записаний до вашого списку контактів *Gmail*, відповідна електронна адреса з’явиться у запрошенні, як тільки ви надрукуєте декілька літер.

Питання для самоконтролю:

1. *Що таке вікі-сервіс?*
2. *В чому полягає відмінність між веб-сайтом і вікі-сайтом?*
3. *За якими ознаками можна охарактеризувати вікі?*
4. *В чому суть вікі-технологій?*
5. *Назви переваги вікі-сервісу.*
6. *Назви недоліки і обмеження вікі-сервісу.*
7. *Для чого потрібні вікі?*
8. *Як відбувається спільне редагування документів у сервісі GoogleDocs?*
9. *Назви переваги і недоліки сервісу GoogleDocs.*
10. *Як відбувається обмін з користувачами Google?*
11. *Які існують технічні обмеження до статей у вікі?*
12. *Які існують обмеження за змістом до статей у вікі?*
13. *Назвати обмеження до статті у вікі, з точки зору читацького сприйняття.*
14. *Які існують можливості обмежувати доступ користувачів?*

4.6. Тематична робота «Робота з веб-ресурсами»

Див. робочий зошит «Інформатика. Третій рік – єдиний курс. 11 клас» / Пилипчук О.П., Ковшун М.І., Сальнікова І.І. – Шепетівка: «Аспект», 2011.

4.7. Тематична робота «Робота з веб-ресурсами» (продовження)

Див. робочий зошит «Інформатика. Третій рік – єдиний курс. 11 клас» / Пилипчук О.П., Ковшун М.І., Сальнікова І.І. – Шепетівка: «Аспект», 2011.