

**ДРОГОБИЦЬКИЙ ДЕРЖАВНИЙ ПЕДАГОГІЧНИЙ УНІВЕРСИТЕТ  
ІМЕНІ ІВАНА ФРАНКА**

**Ірина Шаклеїна, Надія Ших**

# **ОСНОВИ ІНТЕРНЕТ-ТЕХНОЛОГІЙ**

**Методичні рекомендації  
до виконання лабораторних робіт  
(для студентів напряму підготовки «Інформатика»)**

**Дрогобич  
2012**

**УДК.004(07)**  
**ББК 32.988.р.**  
**Ш17**

**Рекомендовано до друку вченою радою Дрогобицького державного педагогічного університету імені Івана Франка як методичні вказівки до лабораторних робіт (протокол № 3 від 22.03.2012 р.)**

**Рецензенти:**

**Берко Андрій Юліанович**, кандидат технічних наук, професор кафедри «Інформаційні системи та мережі» національного університету «Львівська політехніка»;

**Сікора Оксана Володимирівна**, кандидат технічних наук, доцент кафедри інформатики та обчислювальної математики Дрогобицького державного педагогічного університету імені Івана Франка.

Шаклеїна І., Ших Н.

**Ш17 Основи Інтернет-технологій : методичні рекомендації до виконання лабораторних робіт.** – Дрогобич: Редакційно-видавничий відділ Дрогобицького державного педагогічного університету імені Івана Франка, 2012. – 152 с.

Посібник укладено відповідно до програми навчальної дисципліни «Основи Інтернет-технологій» для підготовки фахівців ОКР «Бакалавр» галузі знань: «0403 Системні науки та кібернетика» напряму підготовки «6.040302 Інформатика», затвердженої вченою радою Дрогобицького державного педагогічного університету імені Івана Франка (протокол № 2 від 17.02.2011).

**УДК.004(07)**  
**ББК 32.988.р.**

## Зміст

Передмова .....	4
Лабораторна робота 1. Система адресації та ідентифікація комп'ютерів. Робота з утилітами .....	5
Лабораторна робота 2. Веб-браузери .....	12
Лабораторна робота 3. Пошуковий сервіс мережі Інтернет .....	20
Лабораторна робота 4. Електронна пошта .....	29
Лабораторна робота 5. Створення власного веб-сайту за допомогою MS Publisher .....	37
Лабораторна робота 6. Формування веб-сторінок засобами візуального редактора MS Frontpage. Робота з текстом та малюнками. Створення гіперпосилань .....	45
Лабораторна робота 7. Редактор MS Frontpage. Робота з фреймами та формами. Ефекти динамічного HTML .....	54
Лабораторна робота 8. Мова HTML. Основні елементи створення html-сторінок. Робота з тегами .....	62
Лабораторна робота 9. Елементи створення html-сторінок. Створення посилань. Посилання на малюнок та фон сторінки. Карта. Створення таблиць .....	71
Лабораторна робота 10. Елементи створення html-сторінок. Створення списків та фреймів в html-документах .....	80
Лабораторна робота 11. Елементи створення html-сторінок. Використання форм при створенні html-документів .....	91
Лабораторна робота 12. Каскадні таблиці стилів (CSS). Формат .....	100
Лабораторна робота 13. Таблиці стилів CSS. Блочні об'єкти. Обрамлення .....	110
Лабораторна робота 14. Основи JavaScript. Змінні, операції, стрічки. Масиви .....	121
Лабораторна робота 15. Основи JavaScript. Розгалуження, цикли, форми .....	131
Додатки .....	142
Зразок оформлення звіту .....	148
Література .....	149

## Передмова

Зараз Інтернет інтенсивно завойовує інформаційний ринок, будучи не лише засобом спілкування, передачі інформації, але і повноцінним економічним і політичним середовищем. Інтернет-технології – це технології створення та підтримки інформаційних ресурсів у комп'ютерній мережі Інтернет, оволодіння якими допоможе на більш високому рівні використовувати інформаційні ресурси, надасть можливість розміщення власної інформації в Інтернет, більш осмислено використовувати навігацію як в межах сайту, так і між окремими веб-сайтами.

Методичні рекомендації до виконання лабораторних робіт з курсу «Основи Інтернет-технологій» укладені на базі семестрового курсу, який читається для студентів спеціальності «Інформатика». До посібника увійшли лабораторні роботи щодо адресації та ідентифікації комп'ютерів в мережі, роботі з веб-браузерами, основними службами Інтернет, пошуковими системами, а також основам створення веб-сторінок засобами MS Office та за допомогою мови розмітки веб-сторінок HTML. Передбачено ознайомлення з каскадними таблицями стилів CSS та основами Java Script. Всі лабораторні роботи мають однакову структуру: тема, мета, хід роботи, теоретичні відомості, завдання для самостійного виконання та питання для самоконтролю за темою роботи. Теоретичні відомості охоплюють основний матеріал, який потрібно знати студенту для виконання завдань лабораторної роботи. У процесі роботи, крім завдань, подано рекомендації до їх виконання. Для перевірки рівня знань у кожній лабораторній роботі передбачено контрольні запитання.

Під час підготовки до заняття студент повинен прочитати теоретичні відомості, спробувати самостійно виконати завдання, передбачені в лабораторній роботі та вміти дати відповіді на контрольні запитання. У деяких випадках при підготовці до роботи потрібно скористатися додатковою літературою, поданою наприкінці посібника.

Заняття з курсу проходять у три етапи: допуск до роботи, виконання роботи та захист роботи. Під час допуску студенти повинні показати мінімальний рівень підготовки, потрібний для виконання роботи. Під час виконання роботи студент оформляє звіт згідно зі зразком, запропонованим у посібнику. Після виконання роботи її потрібно захистити.

## ТЕМА. Система адресації та ідентифікація комп'ютерів. Робота з утилітами

**МЕТА:** ознайомлення з характеристиками та використанням різних класів IP-адрес, набуття навичок визначення класу IP-адреси; набуття навичок роботи з мережевими утилітами.

### ХІД РОБОТИ

1. Ознайомитися з характеристиками та використанням різних класів IP-адрес.
2. Визначити IP-адресу робочого ПК, встановити її клас та діапазон.
3. Використовуючи відповідні програми, встановити IP-адреси вказаних вузлів та переглянути таблицю маршрутизації.
4. Прослідкувати роботу маршрутизатора за допомогою утиліти `tracert`.

### ТЕОРЕТИЧНІ ВІДОМОСТІ

#### Імена і адресація

Кожний вузол TCP/IP визначається своєю логічною **IP-адресою**, яка ідентифікує положення комп'ютера в мережі. IP-адреса є 32-х розрядним бінарним числом, яке містить у собі адресу мережі і вузла.

IP-адреса складається з двох частин: номера мережі і номера вузла. Номер мережі може бути обраний адміністратором довільно, або призначений за рекомендацією спеціального підрозділу Internet, якщо мережа повинна працювати як складова частина Internet. Номер вузла в протоколі IP призначається незалежно від локальної адреси вузла.

IP-адреса має довжину 4 байти і зазвичай записується у вигляді чотирьох чисел, що представляють значення кожного байта в десятковій формі і розділених крапками (див. додаток 1). Наприклад:

- 128.10.2.30 – традиційна десяткова форма представлення адреси;
- 10000000.00001010. 00000010. 00011110 – двійкова форма представлення цієї ж адреси.

Яка частина адреси є номером мережі, а яка – номером вузла, визначається значеннями перших біт адреси. Якщо адреса починається з 0, то мережа належить до класу A і номер мережі займає один байт, останні 3 байти інтерпретуються як номер вузла в мережі. Мережі класу A мають

номери в діапазоні від 1 до 126. (Номер 0 не використовується, а номер 127 зарезервований для спеціальних цілей). Кількість вузлів в мережах класу А може досягати  $2^{24}$ , тобто 16 777 216 вузлів. Мережі класу А належать найбільшим світовим постачальникам послуг Internet.

Якщо перші два біта адреси рівні 10, то мережа належить до класу В. У мережах класу В під номер мережі і під номер вузла відводиться по 16 біт. Отже, мережа класу В є мережею середніх розмірів з максимальним числом вузлів  $2^{16}$ , що складає 65 536 вузлів. Такі мережі мають найбільші університети та великі організації.

Якщо адреса починається з послідовності 110, то це мережа класу С. У цьому випадку для номера мережі відводиться 24 біти, а для номера вузла – 8 біт. Мережі цього класу найбільш поширені, число вузлів у них обмежене  $2^8$ , тобто 256 вузлами. Саме до цього класу належать мережі переважної більшості провайдерів Internet.

Якщо адреса починається з послідовності 1110, то вона є адресою класу D і позначає особливу, групову адресу – multicast. Якщо у пакеті як адреса призначення вказана адреса класу D, то такий пакет повинні отримати всі вузли, яким присвоєна ця адреса.

Якщо адреса починається з послідовності 11110, то це означає, що ця адреса належить до класу Е. Адреси цього класу зарезервовані для майбутніх застосувань.

Структуру IP-адреси для основних класів можна подати у вигляді таблиці:

Клас мережі	Діапазон значень першого октету	Загальна кількість мереж	Максимальна кількість вузлів в мережі
A	1 - 126	126	16 777 214
B	128 - 191	16 382	65 534
C	192 - 223	2 097 150	254

### **Використання масок в IP-адресації**

Важливим елементом розбиття адресного простору Internet є підмережі. **Підмережа** – це підмножина мережі, що не перетинається з іншими підмережами. Підмережі використовуються для того, щоб обійти обмеження фізичних мереж на число вузлів у них і максимальну довжину кабелю в сегменті мережі. Найменша мережа класу С може складатися із 254 вузлів. Для того, щоб досягти цієї цифри треба об'єднати декілька

фізичних сегментів мережі. Зробити це можна або за допомогою фізичних пристроїв (наприклад, репітерів), або за допомогою машин-шлюзів. У першому випадку розбиття на підмережі непотрібне, оскільки логічно мережа виглядає як одне ціле. При використанні шлюзу мережа розбивається на підмережі.

Розбиття мережі на підмережі використовує ту частину IP-адреси, яка закріплена за номерами хостів. Адміністратор мережі може замаскувати частину IP-адреси і використовувати її для призначення номерів підмереж. Фактично, спосіб розбиття адреси на дві частини тепер застосовуватиметься до адреси хоста з IP-адресою мережі, в якій організовується розбиття на підмережі.

**Маска підмережі** – це число, яке складається із чотирьох октетів, що визначають, яка частина IP-адреси є мережевою адресою, а яка – адресою хоста. Наприклад, IP-адреса 207.29.170.193, а маска – 255.255.255.0. Тоді IP-адреса і маска в двійковому вигляді будуть, відповідно

11001111.11011011.10101010.11000001 і

111111.11111111.11111111.00000000.

Всі числа, «накриті» маскою, є номерами підмереж, а останнє десяткове число (8 біт) залишене для адрес хостів підмережі. При організації зв'язків між комп'ютерами в мережі маски використовуються для визначення того, чи є цільовий хост у тій же підмережі, що і початковий.

Наприклад, маска 255.255.255.0 дає змогу розбити мережу класу В на 254 підмережі по 254 вузли в кожній. Для стандартних класів мереж маски мають такі значення:

- клас А – 11111111. 00000000. 00000000. 00000000 (255.0.0.0);
- клас В – 11111111. 11111111. 00000000. 00000000 (255.255.0.0);
- клас С – 11111111. 11111111. 11111111. 00000000 (255.255.255.0).

Забезпечуючи кожну IP-адресу маскою, можна відмовитися від понять класів адрес і зробити гнучкішою систему адресації. Наприклад, адреса 185.23.44.206 потрапляє в діапазон 128-191, тобто адреса належить до класу В. Тому номер цієї мережі 185.23.0.0, а номер вузла – 0.0.44.206. Якщо цю адресу асоціювати з маскою 255.255.255.0, то номером підмережі буде 185.23.44.0, а не 185.23.0.0, як це визначено системою класів.

Для визначення місцеположення цільового хоста комп'ютер застосовує операцію ANDing для IP-адрес і масок обох хостів. Операція ANDing працює так: узявши в двійковому вигляді IP-адресу і маску, сервер порівнює

їх, і якщо у відповідному розряді адреси і маски стоїть 1, то результат буде 1. Інакше результат – 0. Якщо результати її виконання рівні – обидва комп'ютери знаходяться в одній підмережі.

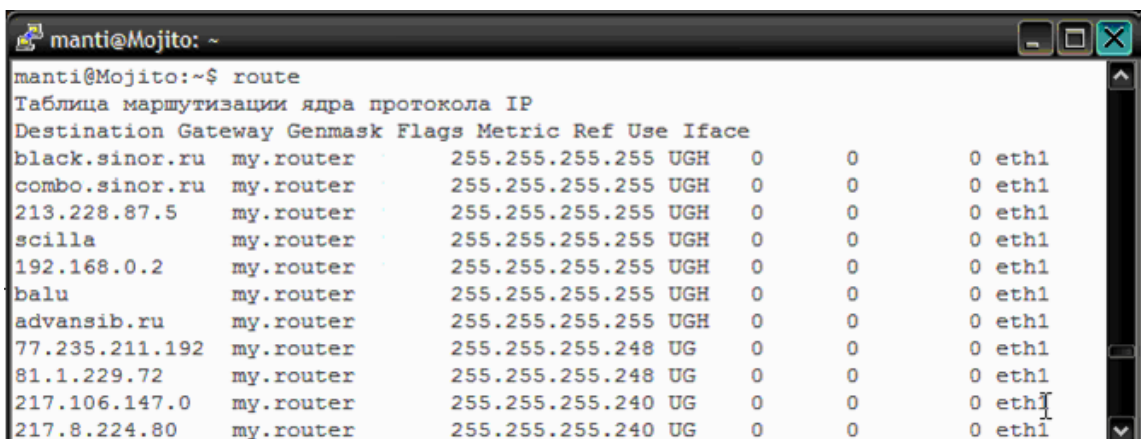
### Мережеві утиліти

У мережевих операційних системах існує велике число спеціальних програм, призначених для управління і аналізу мережевих з'єднань – утиліт. Розглянемо деякі з них.

**Утиліта ipconfig** – дає змогу проглянути поточну конфігурацію адрес TCP/IP для всіх встановлених на цьому комп'ютері мережевих адаптерів і комутованих з'єднань. З її допомогою можна визначити IP-адресу цього комп'ютера. Запущена без параметрів команда **ipconfig** видає як результат поточну конфігурацію адреси TCP/IP для всіх встановлених на цьому комп'ютері мережевих адаптерів і комутованих з'єднань (рис. 1). Команду **ifconfig** слід першою використовувати для діагностування можливих проблем із з'єднанням TCP/IP. З її допомогою можна визначити чи була призначена IP-адреса мережевому адаптеру та дізнатися адресу шлюзу.

Запустити утиліту **ipconfig** можна з командного рядка DOS: Пуск/Програми/стандартні/командний рядок, і потім ввести «**ipconfig/all**» у командному рядку – буде одержано докладну інформацію про мережеве налаштування комп'ютера.

**Утиліта route** дає змогу отримати докладну інформацію про поточну таблицю маршрутизації (рис. 1) і змінювати цю таблицю.



```
manti@Mojito:~$ route
Таблица маршрутизации ядра протокола IP
Destination Gateway Genmask Flags Metric Ref Use Iface
black.sinor.ru my.router 255.255.255.255 UGH 0 0 0 eth1
combo.sinor.ru my.router 255.255.255.255 UGH 0 0 0 eth1
213.228.87.5 my.router 255.255.255.255 UGH 0 0 0 eth1
scilla my.router 255.255.255.255 UGH 0 0 0 eth1
192.168.0.2 my.router 255.255.255.255 UGH 0 0 0 eth1
balu my.router 255.255.255.255 UGH 0 0 0 eth1
advansib.ru my.router 255.255.255.255 UGH 0 0 0 eth1
77.235.211.192 my.router 255.255.255.248 UG 0 0 0 eth1
81.1.229.72 my.router 255.255.255.248 UG 0 0 0 eth1
217.106.147.0 my.router 255.255.255.240 UG 0 0 0 eth1
217.8.224.80 my.router 255.255.255.240 UG 0 0 0 eth1
```

Рис. 1. Виведення таблиці маршрутизації за допомогою команди **route**.

**Утиліта netstat** дає змогу отримати детальну інформацію про активні на цей час з'єднання (рис. 2). Додаткові ключі уможливають отримання інформації про мережеві порти, про IP-адреси комп'ютерів, що беруть участь у підключенні, а також про інші мережеві параметри.



```

manti@Mojito: ~
Активные соединения с интернетом (w/o servers)
Proto Recv-Q Send-Q Local Address Foreign Address State
tcp      0      0 Mojito.local:57716  advansib.ru:65012  ESTABLISHED
tcp      0      0 Mojito.local:57725  advansib.ru:65012  ESTABLISHED
tcp      0      0 Mojito.local:55821  advansib.ru:65012  ESTABLISHED
tcp      0      0 Mojito.local:37316  advansib.ru:65012  ESTABLISHED
tcp      0      0 Mojito.local:55832  advansib.ru:65012  ESTABLISHED
tcp      0      0 Mojito.local:56022  balu:https          ESTABLISHED
tcp      0      0 Mojito.local:55829  advansib.ru:65012  ESTABLISHED
tcp      52      0 Mojito.local:ssh    scilla:3712         ESTABLISHED
tcp      0      1 Mojito.local:59906  balu:3128           SYN_SENT
tcp      0      0 Mojito.local:55830  advansib.ru:65012  ESTABLISHED
tcp      0      0 Mojito.local:37317  advansib.ru:65012  ESTABLISHED
tcp      0      0 Mojito.local:37314  advansib.ru:65012  ESTABLISHED
tcp      0      0 Mojito.local:55823  advansib.ru:65012  ESTABLISHED
tcp      0      0 Mojito.local:55828  advansib.ru:65012  ESTABLISHED
tcp      0      0 Mojito.local:55831  advansib.ru:65012  ESTABLISHED
tcp      0      0 Mojito.local:55820  advansib.ru:65012  ESTABLISHED
tcp      0      0 Mojito.local:37315  advansib.ru:65012  ESTABLISHED
Активные сокеты домена UNIX (w/o servers)
Proto RefCnt Flags      Type      State      I-Node  Путь
unix  2      [ ]      DGRAM    State      3134    @/com/ubuntu/upstart
unix  2      [ ]      DGRAM    State      3339    @/org/kernel/udev/udev
unix  2      [ ]      DGRAM    State      7814    @/org/freedesktop/hal/udev_event
unix  21     [ ]      DGRAM    State      6338    /dev/log
:

```

Рис. 2. Виведення активних підключень за допомогою команди netstat.

```

manti@Mojito: ~
manti@Mojito:~$ netstat -r
Таблица маршрутизации ядра протокола IP
Destination Gateway Genmask Flags MSS Window irtt Iface
217.70.106.29 my.router 255.255.255.255 UGH 0 0 0 eth1
combo.sinor.ru my.router 255.255.255.255 UGH 0 0 0 eth1
213.228.87.5 my.router 255.255.255.255 UGH 0 0 0 eth1
scilla my.router 255.255.255.255 UGH 0 0 0 eth1
192.168.0.2 my.router 255.255.255.255 UGH 0 0 0 eth1
balu my.router 255.255.255.255 UGH 0 0 0 eth1
advansib.ru my.router 255.255.255.255 UGH 0 0 0 eth1
77.235.211.192 my.router 255.255.255.248 UG 0 0 0 eth1
81.1.229.72 my.router 255.255.255.248 UG 0 0 0 eth1
217.106.147.0 my.router 255.255.255.240 UG 0 0 0 eth1
217.8.224.80 my.router 255.255.255.240 UG 0 0 0 eth1
192.168.240.0 * 255.255.255.224 U 0 0 0 tap0
80.89.133.32 my.router 255.255.255.224 UG 0 0 0 eth1
82.200.114.0 my.router 255.255.255.224 UG 0 0 0 eth1
81.1.229.96 my.router 255.255.255.224 UG 0 0 0 eth1
81.1.229.32 my.router 255.255.255.224 UG 0 0 0 eth1
81.1.229.128 my.router 255.255.255.128 UG 0 0 0 eth1
79.175.39.0 my.router 255.255.255.128 UG 0 0 0 eth1
81.1.232.0 my.router 255.255.255.0 UG 0 0 0 eth1
80.89.143.0 my.router 255.255.255.0 UG 0 0 0 eth1
192.168.240.0 192.168.240.1 255.255.255.0 UG 0 0 0 tap0
212.192.163.0 my.router 255.255.255.0 UG 0 0 0 eth1
82.117.68.0 my.router 255.255.255.0 UG 0 0 0 eth1

```

Рис. 3. Виведення таблиці маршрутизації за допомогою команди netstat.

**Утиліта ARP** служить для виводу і зміни записів кеша протоколу ARP, який містить одну або декілька таблиць, що використовуються для зберігання IP-адрес і відповідних їм фізичних адрес Ethernet або Token Ring. Для кожного мережевого адаптера Ethernet або Token Ring, встановленого в комп'ютері, використовується окрема таблиця.

```
manti@Mojito: ~
manti@Mojito:~$ arp
Адрес HW-тип HW-адрес  флаги Маска  Интерфейс
my.router             ether  00:24:8c:69:ba:10  C           eth1
manti@Mojito:~$
```

Рис. 4. Виведення таблиці поточного протоколу ARP для всіх інтерфейсів.

**Утиліта ping** перевіряє зв'язок з віддаленим комп'ютером. Формат команди:

ping [-<Sw>] [<ім'я\_кінцевого\_комп'ютера>]

де <Sw> – комбінація додаткових параметрів; <ім'я\_кінцевого\_ комп'ютера> – IP-адреса або доменне ім'я віддаленого хосту.

Ця програма дає змогу перевірити правильність функціонування DNS-серверів: якщо деякий вузол "відгукується" на IP-адресу, але "не відгукується" на доменне ім'я, то або DSN-сервер непрацездатний, або він неправильно вказаний у конфігурації.

```
C:\Documents and Settings\Користувач>ping 192.168.0.1
Обмен пакетами с 192.168.0.1 по 32 байт:
Ответ от 192.168.0.1: число байт=32 время<1мс TTL=64
Ответ от 192.168.0.1: число байт=32 время<1мс TTL=64
Ответ от 192.168.0.1: число байт=32 время<1мс TTL=64
Ответ от 192.168.0.1: число байт=32 время<1мс TTL=64
Статистика Ping для 192.168.0.1:
  Пакетов: отправлено = 4, получено = 4, потеряно = 0 (0% потерь),
Приблизительное время приема-передачи в мс:
  Минимальное = 0мсек, Максимальное = 0 мсек, Среднее = 0 мсек
```

Рис. 5. Приклад відповіді від діючого хосту.

**Утиліта tracert** дає змогу визначити маршрут проходження тестового пакету з даними та встановити, на яких ділянках маршруту затримка пакетів є максимальною.

Формат команди: Tracert [-<Sw>] [<ім'я\_кінцевого\_комп'ютера>], де <Sw> – комбінація додаткових параметрів; <ім'я\_кінцевого\_комп'ютера> – IP-адреса або доменне ім'я віддаленого вузла.

## ЗАВДАННЯ

1. Для заданих IP-адрес класів А, В і С і запропонованих масок (див. варіанти завдань) визначити:
  - а) клас адреси;
  - б) максимально можливу кількість підмереж;

в) діапазон зміни адрес підмереж;

г) максимальне число вузлів в підмережах.

1	адреса	109.18.107.14
	маска	11111111.10000000.00000000.00000000
2	адреса	200.131.197.27
	маска	11111111.11111111.11111111.11000000
3	адреса	135.209.23.246
	маска	11111111.11111111.11111111.11111000
4	адреса	211.184.171.100
	маска	11111111.11111111.11111111.00000000
5	адреса	11.231.241.248
	маска	11111111.11111000.00000000.00000000
6	адреса	156.131.138.69
	маска	11111111.11111111.11111100.00000000

2. За допомогою утиліт отримати необхідну інформацію і заповнити таблицю

Символьне ім'я комп'ютера	Адреса локальної мережі	IP-адреса комп'ютера, його номер в мережі	Маска підмережі	Адреси шлюзу	IP-адреса сервера DNS

3. Використовуючи програму ping, звернутись за IP-адресою: ping 127.0.0.1. ("замикання на собі") та перевірити відгук власного комп'ютера: ping IP\_adress\_of\_Local\_host.

4. За допомогою утиліти ping визначити IP-адреси вузлів з доменними іменами: www.mail.ru, www.yandex.ru, www.jahoo.com.

5. Використовуючи програму route, переглянути таблицю маршрутизації на вашому комп'ютері: Пуск/ Програми/стандартні/командний рядок route print

6. Перевірити роботу маршрутизатора за допомогою утиліти tracert, відправивши пакети на вузол www.opennet.ru. (ввести: tracert www.opennet.ru)

7. Додати в таблицю маршрутизації комп'ютера рядок для пересилки пакетів у мережу 172.21.0.0 (маска 255.255.0.0) через мережевий інтерфейс комп'ютера: route add 172.21.0.0 mask 255.255.0.0 192.168.1.4 METRIC 3.

Перевірити роботу внесених змін за допомогою утиліти tracert.

8. Оформити звіт з виконання лабораторної роботи.

### **КОНТРОЛЬНІ ЗАПИТАННЯ**

1. Яке основне призначення та структура IP-адреси?
2. Яку інформацію можна отримати, проаналізувавши IP-адресу?
3. Мережі якого класу мають максимальну кількість вузлів? Мінімальну кількість вузлів?
4. Що таке підмережа?
5. Яку структуру має маска підмережі?
6. Що таке утиліти?
7. Наведіть приклади утиліт.
8. Для чого призначена утиліта ipconfig? Як її викликати?
9. Які дані виводяться у разі виклику утиліти ipconfig без додаткових ключів?
10. Як протестувати з'єднання з віддаленим хостом?
11. Як визначити доменне ім'я хосту з відомою IP-адресою?
12. Як визначити IP-адресу хосту із зазначеним доменним ім'ям?
13. Яке призначення DNS-серверів?
14. Які операції можна виконати з використанням команди netstat?
15. Чи можна за допомогою команди ifconfig визначити IP-адресу комп'ютеру?

### **Лабораторна робота №2**

#### **ТЕМА. Веб-браузери**

**МЕТА:** набуття навичок налаштування властивостей браузерів, перегляду, збереження і друку веб-сторінок, створення закладок для швидкого завантаження відвідуваних веб-сайтів.

#### **ХІД РОБОТИ**

1. Ознайомитися з інтерфейсом та налаштуваннями браузера Internet Explorer.
2. Ознайомитися з інтерфейсом та налаштуваннями браузера Opera.

3. Навчитися завантажувати та зберігати веб-сторінки за допомогою веб-браузерів Internet Explorer і Opera.
4. Порівняти можливості цих веб-браузерів.

## ТЕОРЕТИЧНІ ВІДОМОСТІ

**Веб-браузер** (веб-оглядач) – це програма перегляду інформаційних ресурсів, розміщених у всесвітній мережі Інтернет. Вона дає змогу отримати доступ до всіх веб-документів, картинок, звукових і відеофайлів в мережі. Оглядач відображає веб-документ на екрані, користуючись командами (тегами). Правила запису тегів визначаються спеціальною мовою розмітки гіпертексту – HTML (HyperTextMarkupLanguage), подібною до мови програмування. Найважливішою рисою веб-сторінки є наявність гіперпосилань. Гіперпосилання – це текст чи малюнок, розміщений на веб-сторінці, після клацання на якому виконується дія завантаження іншого веб-ресурсу.

Усі веб-ресурси зібрані в тематичні групи, які називаються веб-сайтами і зберігаються на спеціально відведених для цього комп'ютерах – серверах. Для швидкого доступу, файли з потрібною інформацією мають персональну адресу. Адреса будь-якого файлу у всесвітньому масштабі визначається уніфікованим покажчиком ресурсу – URL.

URL-адреса складається з трьох частин:

- 1) назви служби, яка здійснює доступ до цього ресурсу (ім'я прикладного протоколу), наприклад: для служби WWW служить протокол `http://`, для FTP-архівів – `ftp://`, для перегляду новин UseNet – `news://`, для доступу до файлів на локальному комп'ютері – `file://` ;
- 2) назви доменного імені комп'ютера (сервера), на якому зберігається ресурс, наприклад, `http://www.abcde.com`;
- 3) назви повного шляху доступу до файлу, наприклад, `http://www.abcde.com/Files/abcd.zip`.

Великі і малі літери в назвах каталогів вважаються різними.

У повному доменному імені вузла присутні частини, які називаються доменами. Домени найчастіше вказують на регіональні ознаки або на характер діяльності установи, яка володіє. Домени можуть мати такі значення:

<b>.ua</b> – Україна;	<b>.com</b> – комерційні організації;
<b>.ru</b> – Росія;	<b>.edu</b> – освітні організації;
<b>.uk</b> – Великобританія;	<b>.gov</b> – урядові установи;
<b>.de</b> – Німеччина;	<b>.net</b> – організації, що працюють з мережею;
<b>.it</b> – Італія;	<b>.org</b> – державні і суспільні установи.

Найпоширенішими на сьогоднішній день веб-оглядачами є такі: Internet Explorer, Opera, Netscape Navigator, Google Chrome, Mozilla FireFox. Далі розглянемо роботу з Internet Explorer та Opera.

Браузер можна запустити різними способами. При роботі в Windows найпростіші з них – клікнути на значку браузера на робочому столі або вибрати його з головного меню.

### **Браузер Internet Explorer**

Internet Explorer розробляється корпорацією Microsoft починаючи з 1995 року і міститься у складі операційних систем. Останньою версією браузера є Internet Explorer 9.0.

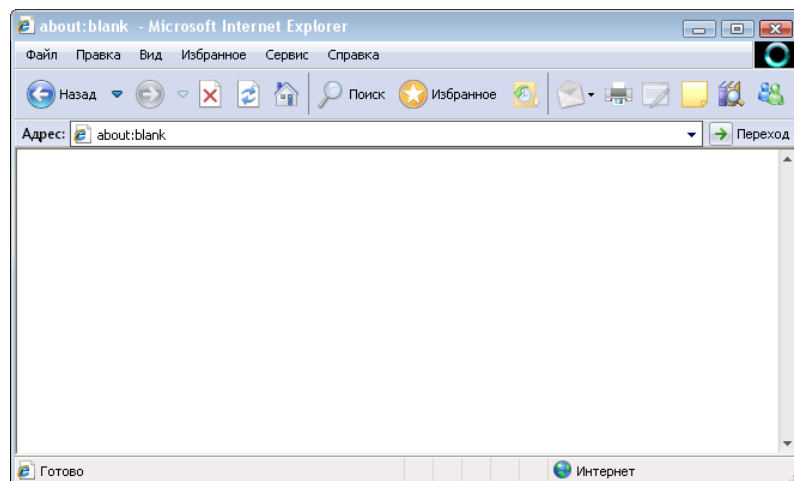


Рис. 1. Вікно браузера Internet Explorer.

Браузер Internet Explorer 9 забезпечує легкий вибір і навігацію за вкладками завдяки відображенню ескізів всіх відкритих вкладок в одному вікні. Вікно програми має стандартний вигляд (рис. 1).

### **Призначення елементів вікна**

Допускається використання трьох панелей інструментів:

- *Звичайні кнопки* – панель з кнопками для швидкого виклику команд, що часто використовуються. Далі вона називається просто панеллю інструментів;
- *Адресний рядок* – панель для введення URL-адреси документа, що відкривається. Тут містяться останні 25 адрес.

- *Посилання* – панель посилань на веб-вузли, на які радить звернутися Microsoft.
- *Рядок стану* – тут відображається поточна інформація, наприклад, час завантаження веб-ресурсу.

Щоб встановити або зняти панелі у вікні використовується меню **Вигляд**⇒ **Панель інструментів**⇒...

Вікно браузера можна налаштовувати відповідно до потреб користувача за допомогою меню **Сервіс**⇒ **Властивості оглядача**⇒ **Параметри**⇒...

- встановити обрану веб-сторінку як початкову сторінку при запуску браузера за допомогою підменю **...Загальні**⇒ **Домашня сторінка**;

- для зміни розміру папки **Тимчасові файли Інтернет** використовують підменю **...Загальні**⇒ **Тимчасові файли**⇒ **Налаштування**. Зберігання браузером деяких із переглянутих сторінок називається кешуванням.

Зміна розміру шрифту для біжучого режиму **Вигляд**⇒ **Розмір шрифту**. Змінити розмір шрифту можна через меню **Сервіс**⇒ **Властивості оглядача**⇒ **Загальні**⇒ **Шрифти**.

Зміна кольорів для відображення тексту, фону, посилань виконується за допомогою меню **Сервіс**⇒ **Властивості оглядача**⇒ **Загальні**⇒ **Кольори**.

Браузер автоматично веде записи про кожен сеанс користувача в Інтернеті. Такі записи зберігаються у папці **Журнал**. Для редагування властивостей журналу використовується **Сервіс**⇒ **Властивості оглядача**⇒ **Журнал**.

Для забезпечення простоти і швидкості доступу до веб-вузлів призначене меню **Вибране**⇒ **Додати у вибране**.

Існує декілька способів відкриття та завантаження веб-сторінок:

- 1) за допомогою меню **Файл**⇒ **Відкрити**⇒...;
- 2) введенням url-адреси в полі **Адреса** браузера;
- 3) клацанням по гіперпосиланнях на відкритій веб-сторінці;
- 4) вибором потрібної сторінки з папки **Вибране**;
- 5) вибором посилання на раніше відкриті документи збережені у папці **Журнал**.

## Відкриття вікон

Для збереження даних веб-сторінки потрібно виконати наступну послідовність дій **Файл**⇒ **Зберегти як...** У діалоговому вікні вказати папку, ім'я файлу, тип файлу: *веб-сторінка повністю* – збереже все у вигляді двох файлів; *веб-сторінка, тільки HTML* – збереже інформацію на веб-сторінці без рисунків у вигляді HTML-коду; *текстовий файл* – збереже у вигляді тексту у файлі з розширенням .txt.

Для збереження без відкриття слід використати команду контекстного меню **Зберегти об'єкт як...** на посиланні.

Браузер Internet Explorer дає змогу надрукувати як усю веб-сторінку, так і будь-який її фрагмент. Крім того, можна надрукувати документи, на які на цій веб-сторінці є посилання та таблицю посилань. Для цього у меню **Файл** потрібно вибрати команду **Друк** та у вікні, що з'явиться, натиснути кнопку **Друк**.

Якщо веб-сторінка має складну структуру, тобто має фрейми, то для виводу інформації на друк використовують відповідні параметри:

- *Виведені на екран* – весь документ;
- *Лише вибрані кадри* – активний фрейм;
- *Всі кадри окремо* – кожен фрейм на окремому аркуші.

Заради безпеки кожен вузол слід віднести до зони безпеки за замовчуванням (Інтернет). **Сервіс**⇒ **Властивості оглядача**⇒ **Безпека**:

- **Інтернет** – середній рівень безпеки;
- **Надійні вузли** – низький рівень безпеки;
- **Обмежений** – містить вузли, яким Ви взагалі не довіряєте.

Можна організувати роботу так, що веб-сторінки будуть завантажуватися для автономної роботи без виходу в Інтернет. Щоб зберегти **Вибране**⇒ **Додати у вибран**⇒ **Режим завантаження**⇒ **Глибина посилань**⇒ **Тип синхронізації**.

## Браузер Opera

Браузер Opera є достатньо популярним, оскільки розробники поставили за мету високу швидкість, коректне відображення сторінок та доступність для людей з особливими потребами (впроваджено голосовий інтерфейс). Вікно програми має стандартний вигляд (рис. 2).

До **переваг** браузера Opera належать такі:

- Висока швидкість виведення веб-сторінок та реакції на дії користувача;



- Багатовіконний інтерфейс;
- Зміна масштабу документа від 20% і до 1000%;
- Перемикання відображення картинок у процесі роботи і для кожного вікна окремо.
- Перемикання відображення документа між налаштуваннями документа і налаштуваннями користувача (колір фону, колір і шрифт тексту і посилань).

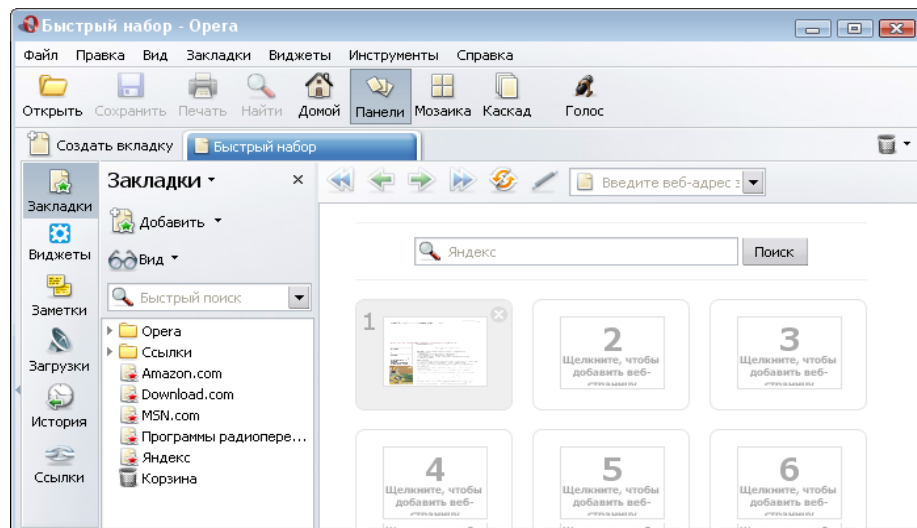


Рис. 2. Вікно браузера Опера

- Наявність інтегрованого поштового клієнта, засобів для переглядання RSS, IRC та інших інструментів (також підтримується використання зовнішніх програм-аналогів);
- Зручне управління за допомогою миші (жестів), клавіатури (гарячі клавіші) та голосу;
- Вбудовано засоби блокування спливаючих вікон;
- Можливість переглядання WAP сайтів та XML файлів;
- Можливість вбудовування голосового модуля, що зачитуватиме текст сторінки. Допомагає людям з ослабленим зором або тим, хто одночасно займається декількома справами;
- Можливість зберегти набір відкритих вкладок, щоб згодом відновити їх або почати роботу із сторінок, які були відкриті під час останнього закриття браузера Опера. Збережена сесія містить історії і налаштування для кожної вкладки і вікна, може бути скопійована і використана на іншому комп'ютері.

- Speed Dial (Швидкий набір). Він уможлиблює легкий доступ до найулюбленіших сайтів. Досить відкрити нову вкладку, щоб побачити Speed Dial.


### **Недоліки браузера Opera**

- Вбудований поштовий клієнт не підтримує створення повідомлень у вигляді HTML.
- Вбудований FTP-клієнт дає змогу завантажувати лише окремі файли, а не папку повністю, а також унеможлиблює редагування та зміну файлів на віддаленому сервері.

## **ЗАВДАННЯ**


1. У браузері Opera додати в меню Закладки адресу сторінки сайту Міністерства освіти та науки, молоді та спорту України [www.mon.gov.ua](http://www.mon.gov.ua).

Алгоритм виконання:

- Запустіть браузер Opera клацнувши на піктограмі  на робочому столі або виконавши послідовність Пуск⇒ Всі програми⇒ Opera.
- У рядку Адреса введіть [www.mon.gov.ua](http://www.mon.gov.ua) та натисніть кнопку Перейти або просто клацніть клавішу <Enter>.
- Після завантаження веб-сторінки в меню Закладки виберіть пункт Створити закладку. Відредагуйте назву створеної закладки в полі Ім'я та натисніть кнопку ОК.

2. Створити в браузері Opera закладку Speed Dial (Швидкий набір) завантаження прогнозу погоди із сайту [www.meteorprog.com.ua](http://www.meteorprog.com.ua) для міста Дрогобич.

Алгоритм виконання:

- Запустіть браузер Opera клацнувши на піктограмі  на робочому столі або виконавши послідовність Пуск⇒ Всі програми⇒ Opera.
- У рядку Адреса введіть [www.meteorprog.com.ua](http://www.meteorprog.com.ua) та натисніть кнопку Перейти або просто клацніть клавішу <Enter>.
- Після завантаження веб-сторінки знайдіть посилання на стан погоди у місті Дрогобич та відкрийте його.
- Створіть нову закладку, клацнувши на ярлику Створити закладку або скориставшись комбінацією клавіш Ctrl+T.

- На вкладці Швидкий набір клацніть по одному вільному з дев'яти фреймів лівою клавішею миші. У вікні, що з'явиться, в переліку Відкриті сторінки виберіть ту, котра вказує на веб-сторінку прогнозу погоди [www.meteorprog.com.ua](http://www.meteorprog.com.ua) та клацніть ОК.
3. Зберегти фотографію одного із замків, розмішених на сайті Замки та храми України ([www.castles.com.ua](http://www.castles.com.ua)) та зберегти всю веб-сторінку з інформацією про замки Львівщини.
  4. У браузері Internet Explorer завантажити сторінку [www.bigmir.net](http://www.bigmir.net). Перегляньте програму телепередач на сьогодні. Зробити головну сторінку [www.bigmir.net](http://www.bigmir.net) домашньою сторінкою браузера.
  5. За допомогою браузера Internet Explorer зберегти веб-сторінку [www.kmu.gov.ua](http://www.kmu.gov.ua) у вигляді html-файлу, текстового файлу та веб-архіву. З'ясувати, чим відрізняються результати застосування цих методів збереження.
  6. У Internet Explorer переглянути журнал відвідування веб-сторінок за останні два тижні, завантажити одну із веб-сторінок з переліку.
  7. Створити у папці Вибране браузера Internet Explorer папку Прогноз погоди і додати до неї посилання на веб-сторінки [www.meteorprog.com.ua](http://www.meteorprog.com.ua) і [www.weather.bigmir.net](http://www.weather.bigmir.net).
  8. У браузері Internet Explorer завантажити сторінку [www.drohobych.net.ua](http://www.drohobych.net.ua), попередньо задавши завантаження лише тексту без елементів мультимедія та графіки. Змінити розмір шрифту та тип кодування тексту. Оновити завантажену сторінку, задавши режим відображення графіки й елементів мультимедія. Порівняти швидкість завантаження і вигляд сторінки при різних параметрах завантаження.
  9. Оформити звіт з виконання лабораторної роботи

## **КОНТРОЛЬНІ ЗАПИТАННЯ**

1. Для чого призначені браузери?
2. Як користуватися браузером?
3. Що таке домашня сторінка браузера?
4. Які прийоми навігації ви знаєте?
5. У який спосіб можна зберігати веб-сторінки для подальшого використання?
6. Як створити власний список корисних сайтів?

7. Назвіть основні елементи вікна програми Internet Explorer.
8. Що таке URL-адреса, які вимоги до її запису?
9. Що відображається в стрічці стану?
10. Як відобразити (приховати) панелі у вікні програми Internet Explorer?
11. Як змінити початкову сторінку при завантаженні браузера?
12. Вкажіть способи відкриття веб-сторінок.
13. Які основні кодування використовуються в українських (російських) сайтах? Як змінити вид кодування?
14. Як відкрити нову сторінку, не закриваючи попередню?
15. Як задати ваші кольори і шрифти веб-сторінки, щоб переважали вони, а не параметри задані розробником?
16. Як додати веб-сторінку у нову папку в пункті “Вибране”?
17. Як впорядкувати вибрані посилання (перейменувати, перемістити, видалити)?
18. Як зберегти веб-сторінку (зберегти всю сторінку, у вигляді тексту, без рисунків)?
19. Як вивести веб-сторінку на друк?
20. Як видрукувати окремих фрейм сторінки?

### Лабораторна робота №3

#### ТЕМА. Пошуковий сервіс мережі Інтернет

**МЕТА:** набуття практичних навичок пошуку інформації в Інтернет з використанням різних пошукових систем

#### ХІД РОБОТИ

1. Відкрити сайти пошукових серверів і ознайомитися з їхньою структурою, налаштуваннями та довідковою системою.
2. Провести пошук за ключовими словами та фразами.
3. Застосувати для пошуку оператори мови пошуку.
4. Ознайомитися із результатами пошуку.
5. Порівняти можливості пошукових систем.

## ТЕОРЕТИЧНІ ВІДОМОСТІ

### Пошукові системи мережі Інтернет

Найбільш популярною службою мережі Інтернет на сьогодні є World Wide Web (або просто Web). Пошук потрібної інформації є одним з основних завдань користувача у WWW. Для його ефективного виконання у Web-просторі існують спеціальні засоби – так звані пошукові системи. Практика доводить, що ефективно і правильно користуватися пошуковими системами вміють не більше 3% користувачів.

За принципом дії розрізняють кілька видів пошукових систем. Найбільш поширеними є два основні типи: *пошукові каталоги* та *індексні пошукові системи*.

Пошукові тематичні каталоги (Subject Directory) організовані за тим же принципом, що й тематичні каталоги бібліотек. На основній сторінці пошукового каталогу розташовано скорочений список великих тематичних категорій, наприклад таких, як **Освіта (Education)**, **Наука (Science)**, **Бізнес (Business)**, **Мистецтво (Art)** тощо. Кожний запис у списку категорій – це гіперпосилання. “Натискання” на нього відкриває наступну сторінку пошукового каталогу, на якому ця тема представлена детальніше. Елементами найнижчого рівня є посилання на окремі Web-сторінки і сервери разом із стислим описом їхнього змісту.

Пошукові каталоги створюються, як правило, вручну. Висококваліфіковані редактори особисто переглядають інформаційний простір WWW, відбирають те, що, на їхню думку, становить загальний інтерес і заносять адреси до каталогу. Оскільки каталоги складаються на основі експертних оцінок, то у них відображені кращі Web-ресурси. Недоліком тематичних каталогів є порівняно невелике охоплення існуючих ресурсів мережі.

В Інтернет функціонує декілька сотень індексних пошукових систем. Принцип роботи з індексними пошуковими системами ґрунтується на використанні *ключових слів*. Шукаючи інформацію з деякої теми, користувач повинен дібрати ключові слова, які описують цю тему, і задати їх індексній пошуковій системі як запит. Пошукова система знаходить у своїх базах даних (вони називаються *індексами* або *показчиками*) адреси веб-ресурсів, які містять ключові слова, і видає клієнту сторінку з посиланнями на ці ресурси.

Пошукові каталоги доцільно використовувати для пошуку інформації з

нової, раніше не вивченої теми. Коли можливості пошукового каталогу вичерпані і виникає необхідність провести поглиблений пошук, доцільно перейти до використання індексної пошукової системи.

Сьогодні найбільш розвинені пошукові системи Інтернет поєднують у собі обидва методи пошуку (за темами і за ключовими словами) і дають змогу використовувати найбільш придатний (рис.1).

Відомими пошуковими серверами є: **Google, AltaVista, Yahoo!, Rambler, Yandex, <Мета> Україна.**

Google (<http://www.google.com>) – найбільш потужна, надійна та високошвидкісна пошукова система. Забезпечує гарні результати пошуку як англійською, так і українською/російською мовами. Використовує базу даних і алгоритми пошуку Yahoo!. Має свій каталог web. Підтримує пошук у межах вказаного веб-сайту. Дає хороші результати при пошуку ресурсів, пов'язаних з інформаційними технологіями.

AltaVista (<http://www.altavista.com>) – це пошукова система, що охоплює весь Інтернет. Відрізняється великою кількістю проіндексованих документів, високою швидкістю, комплексним описом ресурсів, потужними і зручними функціями пошуку, особливо графічних зображень. Підтримує можливість складного пошуку. Має свій каталог.



Рис.1. Стартова сторінка пошукового серверу mail.ru.

Yahoo! (<http://www.yahoo.com>) – друга за величиною пошукова система у світі. Має велику кількість різноманітних сервісів і якісний web-каталог

відсортованих за великою кількістю розділів ресурсів. Не надає можливості пошуку українськомовного і російськомовного тексту.

Yandex (<http://www.yandex.ru>) – здійснює пошук у російській та українській частині Інтернету. Маючи потужний механізм підбору сайтів під запити, ця пошукова машина допомагає знайти найбільш відповідні web-сторінки, щодня проглядаючи сотні тисяч web-сторінок у пошуках змін або нових посилань.

<Мета> Україна (<http://www.meta.ua>) – українська пошукова система, що надає користувачеві зручні інструменти пошуку і додаткові інформаційні ресурси (наприклад, курси валют НБУ і прогноз погоди на тиждень у містах України). Тут можна здійснювати пошук як за тематичним каталогом, так і за ключовими словами. Також містить посилання на каталоги пошукових систем Європи і країн ближнього зарубіжжя.

### **Організація роботи пошукової системи**

Для завантаження обраної пошукової системи записують адресу веб-сторінки пошукової служби в рядку адрес браузера. На стартовій сторінці будь-якого пошукового сервера є форми для введення запиту на пошук (рис.1).

На сьогоднішній день всі популярні інформаційно-пошукові системи мають як мінімум два пошукові інтерфейси: *простий пошук* (simple search) і *розширений пошук* (advanced search). Основна відмінність між ними полягає у можливості складання запитів і пошукових вказівок різного ступеня складності. Обидва інтерфейси належать до графічного типу і реалізовані у вигляді web-сторінок. Як параметри пошуку використовуються ключові слова, за допомогою яких створюються пошукові запити (пошукові вказівки).

Поле для введення ключових слів і кнопка відправки запиту є обов'язковим для будь-якого типу інтерфейсу. Для пошуку інформації за одним ключовим словом необхідно набрати це слово в полі введення запитів і натиснути кнопку **Знайти (Найти, Search)**. Пошук за одним словом доцільно проводити в тому випадку, якщо це слово є рідкісним, маловживаним або власним іменем.

Набагато ефективнішим є пошук за кількома словами, причому важливу роль відіграє правило, яке вказує пошуковій системі як опрацьовувати групу слів. Наприклад, користувача можуть цікавити:

- документи, що містять / перше слово, / друге одночасно;

- документи, в яких ці слова зустрічаються *поруч* або *недалеко* одне від одного;
- документи, в яких зустрічається *АБО* перше слово, *АБО* друге, *АБО* обидва разом.

Отже, для ефективного пошуку за кількома ключовими словами потрібні спеціальні команди, які дають змогу пов'язати окремі слова між собою. Ці команди в пошукових системах утворюють спеціальну *мову запитів*.

Кожна індексна пошукова система використовує свою власну мову запитів. Ретельний перелік правил написання запитів для конкретної пошукової служби можна знайти на її сервері за посиланнями *Допомога*, *Підказка*, *Як скласти запит*, *Поради з пошуку*, *Правила складання запитів* тощо. Але є загальний принцип, згідно з яким усі команди можна поділити на три групи: *команди простого пошуку*, *команди мови запитів* і *команди розширеного пошуку*.

### **Команди простого пошуку**

**1. Пошук словоформ.** У зв'язку з тим, що в українській та російській мовах слова змінюються за відмінками, важливою властивістю пошукової системи є пошук словоформ. У більшості випадків пошукові системи дають змогу знаходити різні словоформи, наприклад, попередній запит на пошук *видатні фізики* рівносильний запиту *видатний фізик*.

**2. Значення великих літер.** Загальне правило для більшості пошукових систем полягає в тому, що великі літери на початку слова сприймаються як додаткова умова, що обмежує область пошуку. Наприклад, за запитом *Ліга Чемпіонів* будуть знайдені лише ті документи, які містять слова *Ліга Чемпіонів*. Проте пошук за запитом *ліга чемпіонів* відобразить список сторінок, в яких є слова *Ліга чемпіонів*, *ліга Чемпіонів*, *Ліга Чемпіонів*, *ліга чемпіонів*.

**3. Пошук однокореневих слів.** Більшість пошукових систем знаходить документи, які містять слова однокореневі з ключовими. Наприклад, пошук за запитом *модел* поверне документи, в яких є слова *модель*, *моделей*, *модельний*, *моделізм*, *моделює*, *моделювання*.

### **Команди мови запитів**

У розширеному пошуку, крім ключових слів, можна використовувати прості логічні оператори та логічні дужки.

*Таблиця 1.* Список операторів для складного пошуку інформації



Оператор	Опис
I	За допомогою цього оператора об'єднують два чи більше слів таким чином, щоб вони всі були в документі, наприклад <i>Ейнштейн I теорія I відносності</i> . В україно- та російськомовних пошукових системах списки слів і без такого оператора сприймаються так, ніби між ними стоїть оператор I. Але для більшості англomовних пошукових систем оператор I відіграє важливу роль, йому відповідають символи "&" та "+".
АБО	Оператор забезпечує пошук за будь-яким словом з групи, наприклад, <i>університет АБО академія</i> . У більшості пошукових систем оператор АБО записується у запиті як OR чи позначається символом "   ". За таким запитом будуть знайдені документи, що містять будь-яке з вказаних слів чи обидва слова одночасно.
НЕ	Оператор використовується, коли з результатів пошуку необхідно вилучити деяке ключове слово. Наприклад, <i>модем НЕ внутрішній</i> . За запитом будуть знайдені документи, що містять слово "модем", але не містять слово "внутрішній". У більшості пошукових систем оператор НЕ записується у запиті як NOT або позначається символом " – ".
()	Круглі дужки застосовуються, коли необхідно керувати порядком дій логічних операторів. Наприклад, пошук за запитом <i>(технічне АБО програмне) I забезпечення</i> поверне документи, в яких є слова <i>технічне забезпечення</i> або <i>програмне забезпечення</i> .
" "	Подвійні лапки дають змогу знаходити вказаний у них пошуковий вираз буквально. При цьому фіксується граматична форма слів, тобто за запитом <i>"погода в Дрогобичі"</i> будуть знайдені документи, в яких міститься таке саме словосполучення, – <i>погода в Дрогобичі</i> .
Пошук із зазначенням відстані	Дає можливість вказати, на який відстані одне від одного повинні розташовуватися слова в документі. В англomовних пошукових системах використовується оператор NEAR. Наприклад, <i>[5, інформаційні ресурси]</i> – для системи <META> (обидва ключові слова повинні належати одній групі довжиною не більше п'яти слів), <i>інформаційні / 2 ресурси</i> – для системи Яндекс (відстань між ключовими словами не повинна перевищувати 2 слова).
Title	Оператор Title дає змогу обмежити пошук назвами документів. Наприклад, за запитом <i>title("дистанційна освіта")</i> будуть знайдені документи, назви яких містять фразу <i>дистанційна освіта</i> .

### Команди розширеного пошуку

Засоби розширеного пошуку дають можливість знаходити документи за датами створення, наприклад, документи, опубліковані в певний день або після певної дати. Таким пошуком користуються клієнти, які регулярно контролюють публікації з деякої теми і слідкують за новинами, що з'явилися

з моменту останнього перегляду Web-ресурсів.

Рис.2. Форма розширеного пошуку на сервері Google.

Сучасні індексні пошукові системи за посиланням **Розширений пошук** пропонують користувачам спеціальні зручні форми, призначені для здійснення команд розширеного пошуку (рис. 2).

## ЗАВДАННЯ

1. Ознайомитися з правилами формування запитів на декількох пошукових серверах (н-д: [www.meta.com](http://www.meta.com), [www.rambler.ru](http://www.rambler.ru), [www.altavista.com](http://www.altavista.com), [www.google.com.ua](http://www.google.com.ua), [www.yandex.ru](http://www.yandex.ru)). За допомогою обраних пошукових систем відшукати документи, котрі містять слова *освіта* та *Інтернет* (у першому випадку документи, в яких зустрічається хоча б одне із заданих слів, у другому – лише ті документи, в яких зустрічаються усі задані слова одночасно).

Для виконання завдання проробіть такі дії:

- завантажте браузер Internet Explorer;
- у вікні браузера до поля **Адреса** введіть назву пошукового серверу [www.meta.com](http://www.meta.com) і натисніть **<Enter>**. Браузер у своєму вікні відобразить інтерфейс пошукового механізму в мережі Інтернет;
- оглядово ознайомтеся із цією сторінкою. З'ясуйте, як і де задавати ключові слова для пошуку. З'ясуйте, як задавати в пошуковій системі META декілька ключових слів для пошуку;
- у вікні початкової сторінки вузла введіть до поля **Запит** запис *освіта АБО Інтернет*. Натисніть у вікні кнопку **Знайти** або з клавіатури клавішу **<Enter>**;

- запишіть результат пошуку (кількість знайдених документів, час пошуку);
- у вікні початкової сторінки вузла введіть до поля **Запит** запис *освіта І Інтернет*. Натисніть у вікні кнопку **Знайти** або з клавіатури клавішу **<Enter>**. Запишіть результат пошуку (кількість знайдених документів, час пошуку);
- за запропонованою схемою (п. 2–7) розгляньте ще декілька пошукових систем (н-д: <http://www.rambler.ru>, <http://www.google.com.ua>, <http://www.altavista.com>, <http://www.yandex.ru>). Порівняйте роботу обраних вами пошукових серверів.

2. У будь-якій пошуковій системі задати для пошуку прізвище одного з викладачів Інституту фізики, математики та інформатики ДДПУ і відшукати список документів, опублікованих в Інтернет, у яких зустрічається вказане прізвище.

Для виконання завдання проробіть такі дії:

- завантажте браузер Internet Explorer;
  - у вікні програми до поля **Адреса** введіть назву пошукового серверу.
  - у вікні початкової сторінки вузла введіть до поля **Запит** прізвище викладача. Натисніть у вікні кнопку **Знайти** або з клавіатури клавішу **<Enter>**;
  - для уточнення даних про викладача вкажіть його ім'я за допомогою опції **Шукати у знайденому** або **Розширеного пошуку**. Натисніть кнопку **Знайти** або з клавіатури клавішу **<Enter>**;
3. За допомогою будь-якої пошукової системи визначити точну адресу офіційного сайту ДДПУ імені Івана Франка та рік його заснування.
4. Знайдіть в Інтернет інформацію про історію створення вашого міста. Якщо ви отримали дуже багато інформації, збільшіть кількість ключових слів. Пошукайте ту ж інформацію у декількох інших пошукових системах (н-д: <http://www.rambler.ru>, <http://www.google.com.ua>, <http://www.aport.ru>, <http://www.altavista.com>, <http://www.yandex.ru>). Порівняйте результати пошуку. Результати занесіть у таблицю:

Пошукова система	Ключові слова	К-сть знайдених документів	Тривалість пошуку

5. Знайдіть інформацію про обов'язки педагогів, скориставшись розширеним пошуком.

Задайте ключові слова, які мають входити до сторінок з пошуком та виберіть один з можливих методів пошуку. Під час пошуку використовуйте оператори для складного пошуку інформації (and, or, Title та інші).

6. За допомогою будь-якої пошукової системи знайдіть репродукцію картини Клода Моне «Маки». Збережіть знайдену інформацію у власній папці. Знайдіть ще декілька картин художника, використовуючи інші пошукові системи.

7. Використовуючи команди розширеного пошуку знайдіть в мережі Інтернет реферат на запропоновану викладачем тему. Зберегти знайдений документ у власній папці у форматі MS Word, зазначивши на початку документа адресу ресурса, з якого було скачано реферат.

8. Використовуючи розширений пошук, знайдіть декілька інформаційних ресурсів, що містять теоретичний матеріал з теми «Пошукова система WAIS».

9. За допомогою будь-якої пошукової системи визначте точну адресу астрономічного сайту інституту фізики, математики та інформатики ДДПУ імені Івана Франка та інформацію про його розробників.

10. Оформити звіт з виконання лабораторної роботи

## **КОНТРОЛЬНІ ЗАПИТАННЯ**

1. Що таке пошукові web-портали?
2. Що таке профіль користувача?
3. Як можна здійснити пошук інформації в Інтернет без використання пошукових систем?
4. Які ви знаєте пошукові системи в Інтернет?
5. Що таке пошукові каталоги? Яким чином вони створюються?
6. Яка відмінність між пошуковими каталогами та покажчиками?
7. Наведіть приклади пошукових систем у Інтернет.
8. Наведіть приклади відомих вам пошукових каталогів мережі Інтернет.
9. Що таке індекси (покажчики) пошукової системи?
10. Як правильно здійснити пошук у пошуковій системі?
11. Що таке розширений пошук?

12. Наведіть приклади утворення ключових слів для розширеного пошуку.
13. Що таке мова запитів і з якою метою вона використовується?
14. Які команди мови запитів ви знаєте?
15. Які символи та логічні операції можна застосовувати у процесі розширеного пошуку?
16. З якою метою при пошуку інформації використовується оператор Title?
17. Для чого використовується оператор Heading?
18. Наведіть приклади програм, які здійснюють пошук в Інтернет відразу на багатьох пошукових системах.
19. Яке призначення служби FTP?
20. Як можна зайти на сервер FTP?

#### Лабораторна робота № 4

### ТЕМА. Електронна пошта

**МЕТА:** набуття практичних навичок електронного листування за допомогою web-інтерфейсу та поштового клієнта.

#### ХІД РОБОТИ

1. Зареєструватися на поштовому сервері та одержати власну електронну адресу.
2. Створити і відправити звичайний лист та листи з вкладенням.
3. Перевірити пошту та відповісти на одержані повідомлення.
4. Надіслати святкові листівки одногрупникам.
5. Оцінити практичність сервісу поштового серверу.

#### ТЕОРЕТИЧНІ ВІДОМОСТІ

Електронна пошта (e-mail) – одна з найпопулярніших служб Інтернету. Кожного дня мільйони користувачів з усього світу надсилають один одному повідомлення.

**Електронна пошта** – це сукупність засобів, призначених для обміну інформаційними даними між користувачами комп'ютерної мережі. Для того, щоб надсилати й отримувати електронну пошту, потрібна спеціальна програма. Така програма може бути розташована у Вебі – тоді ви

працюватимете з нею за допомогою браузера. Досить часто програму для роботи з електронною поштою (цю програму ще називають *поштовим клієнтом*) розміщують безпосередньо на комп'ютері.

Існує багато різноманітних клієнтських програм. Це, наприклад, програма **Microsoft Outlook Express**, що належить до операційної системи Windows як стандартна. Потужніша програма, що інтегрує в собі, окрім підтримки електронної пошти, інші засоби діловодства, **Microsoft Outlook** (яка входить до складу відомого пакета **Microsoft Office**). Зі спеціалізованих поштових добре зарекомендували себе програми **The Bat!** та **Eudora Pro**.

Перевагою користування веб-поштою є доступ до поштової скриньки з будь-якого комп'ютера, підключеного до Інтернет, проте швидкість роботи у цьому випадку нижча, ніж під час використання поштового клієнта.

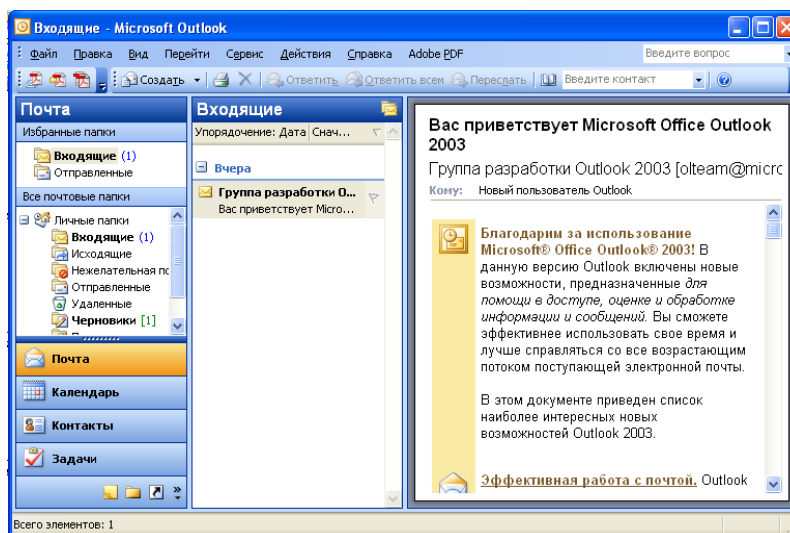


Рис. 1. Вікно програми Microsoft Outlook Express.

Поштова служба базується на двох прикладних протоколах: **SMTP** та **POP3**. За першим відбувається відправлення кореспонденції з комп'ютера на сервер, а за другим – прийом повідомлень, що надійшли.

### Регістрація на поштовому сервері

Надсилати та отримувати електронну пошту користувач може лише за наявності облікового запису електронної пошти. Його, зазвичай, створюють на сайтах, які надають поштові послуги всім бажаючим. Таким сайтом, наприклад, є hotmail.com чи ukr.net.

**Обліковий запис електронної пошти** – це набір параметрів, необхідних для роботи з поштою. Серед них є адреса вашої поштової скриньки. Адреса електронної пошти користувача записується з

використанням латинських літер, цифр і деяких спеціальних символів. Електронна адреса складається з двох частин, розділених символом @, наприклад student\_ddpu@ukr.net, student@rambler.ru. Перша частина адреси – це унікальне ім'я користувача (login), а друга – ім'я сервера, на якому розташована скринька цього користувача

Для отримання електронної скриньки потрібно зайти на сайт будь-якого поштового сервера, наприклад сайту Ukr.net (www.ukr.net). На стартовій сторінці поштового серверу у верхній лівій частині вікна знаходиться ділянка для роботи з електронною поштою. Після ознайомлення з правилами користування поштовим сервером можна приступати до реєстрації, клацнувши на відповідному посиланні: **Реєстрація** для www.ukr.net, **Завести пошту** для www.rambler.ru, **Завести поштову скриньку** для www.yandex.ru тощо.

Після згоди на реєстрацію з'явиться веб-сторінка з анкетною формою, поля якої потрібно заповнювати відповідно до правил (рис. 2). Реєстраційні листи на різних серверах дещо відрізняються, однак незмінними у них є поля: **логін, пароль, ім'я/прізвище, стать, дата народження**.

The image shows a registration form on the Ukr.net website. At the top left is the logo 'ukr.net це – мій Інтернет!'. To the right of the logo is the text 'Створити акаунт УКРНЕТ'. There are language selection buttons for 'Ukr', 'Рус', and 'Eng'. Below the header, a blue box contains the text: 'Після завершення реєстрації Ви отримаєте поштову скриньку FREEMAIL і віртуальну флешку e-Disk розміром 1Гб.' The form itself consists of several input fields, each with a green checkmark to its right, indicating they are filled or valid. The fields are: 'Логін' (username) with 'student\_ddpu@ukr.net', 'Пароль' (password) with masked characters, 'Повторіть пароль' (repeat password) with masked characters, 'Ваше ім'я' (name) with 'Адміністратор', 'Ваша стать' (gender) with radio buttons for 'чоловік' and 'жінка', 'Дата народження' (date of birth) with a dropdown for '28', a dropdown for 'жовтня', and a dropdown for '1995', 'Країна' (country) with a dropdown for 'Україна', 'Область' (region) with a dropdown for 'Львівська обл.', and 'Місто' (city) with a dropdown for 'Дрогобич'. Below these are 'Альтернативний e-mail' (alternative email) with 'pila-ifmi@rambler.ru' and 'Мобільний телефон' (mobile phone) with an empty field. To the right of these fields is a note: 'На e-mail або мобільний буде надіслано код активації акаунта. Обов'язково заповніть одне з цих полів.' At the bottom of the form is a CAPTCHA field with the text 'Введіть символи, які Ви бачите на картинці:' and a grid of characters 'ZXAZLA'. To the right of the CAPTCHA is a 'Оновити картинку' button. At the very bottom is a large green button labeled 'Реєстрація' and a small red icon with the text 'Реєстрація означає згоду з умовами користування скринькою'.

Рис.2. Форма реєстрації на сайті Ukr.net.

Особливим полем в анкеті є поле введення контрольних символів (цифр чи літер), які зображені на картинці. Ці символи є кодом безпеки, що підтверджує заповнення реєстраційного листа людиною. Бо, як відомо, електронні роботи не розрізняють зображень, а тому не зможуть створювати свої електронні скриньки. Після введення всієї потрібної

інформації потрібно клацнути на кнопці **Реєстрація (Зареєструватися)**. Якщо вибраний вами логін на сервері вже зареєстрований, вам буде запропоновано зробити це ще раз з іншим логіном та підтвердити пароль.

### **Інтерфейс поштової скриньки та правила листування.**

Щоб отримати можливість перевірити свою пошту та відправити повідомлення, необхідно спочатку увійти у свою поштову скриньку, вказати ім'я користувача та пароль, задані у вашому обліковому записі та натиснути кнопку **Увійти**.

За замовчуванням будь-яка поштова скринька містить стандартні папки: **Вхідні** (до неї надходять нові повідомлення електронної пошти), **Вихідні** (місце тимчасового зберігання невідправлених повідомлень), **Надісланні/Відправлені** (тут зберігаються копії надісланих повідомлень), **Чернетки** (призначена для зберігання незавершених повідомлень), **Видалені/Кошик** (місце тимчасового зберігання видалених повідомлень, які будуть знищені остаточно після виходу з поштової скриньки). При потребі можна створювати власні папки (наприклад, для повідомлень від друзів, одногрупників, колег).

Більшість поштових програм мають так звані адресні книги, в яких зберігаються не тільки e-mail адресатів, а і їхні реальні прізвища/імена та, при потребі, додаткова інформація. Дані в адресну книгу вносяться з клавіатури та записуються автоматично при листуванні.

Усі електронні повідомлення мають однакову структуру. На рис. 3. продемонстровано основні компоненти листа.

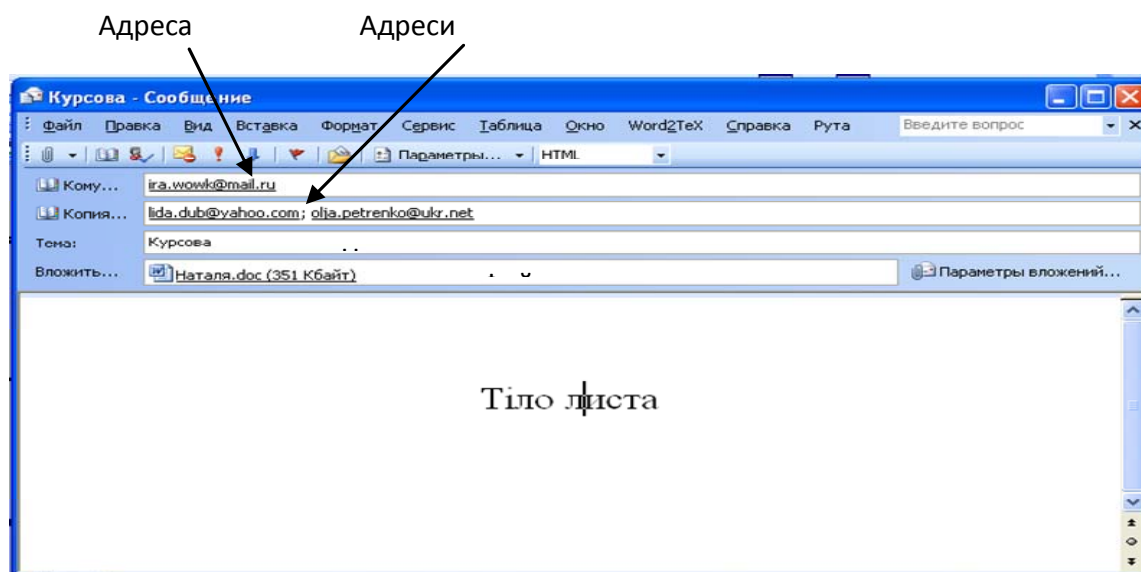


Рис.3. Приклад електронного повідомлення.



Отже, повідомлення має такі компоненти.

- **Адреса одержувача.** Так само, як і на звичайному конверті, ви маєте вказати адресу, за якою надсилаєте лист.

- **Ім'я та адреса відправника.** Це ваші власні ім'я та адреса, що автоматично додаються до повідомлення поштовою програмою. Звичайно у програмі передбачена можливість визначити, який вигляд вони будуть мати у повідомленні,

- **Тема повідомлення.** Кількома словами потрібно зазначити, про що йдеться в повідомленні. У поштовій програмі одержувача ці слова разом з адресою виводяться у списку вхідних повідомлень.

- **Час і дата.** Автоматично вставляються у повідомлення поштовою програмою, коли вона його відправляє.

- **Тіло повідомлення.** Це власне текст повідомлення. У полі **Текст листа** із звичайної чи віртуальної клавіатури вводиться текст звичайного повідомлення

- **Вкладення.** Ви можете додати до повідомлення будь-які файли, зокрема зображення, звукозаписи, програми та текстові документи. Якщо потрібно надіслати відео-, аудіо-файл, фотографію чи будь-який файл, створений в іншому програмному середовищі, слід вибрати посилання **Прикріпити файли**, вказати місцезнаходження файлу та натиснути кнопку **Відкрити**.

- **Копія.** За бажанням у цьому полі можна вказати електронну адресу іншої людини, щоб вона отримала копію повідомлення. Першому адресату буде відомо про копію повідомлення

- **Прихована копія.** Вказується адреса, куди має бути надіслана копія. Проте перший (основний) одержувач повідомлення про це не знатиме.

Після натискання кнопки **Відіслати листа** на екрані з'явиться повідомлення про успішне відправлення або про помилку відправлення. Складене повідомлення (з усіма зазначеними компонентами) ваш поштовий сервер надішле поштовому серверу одержувача, звідки той відкриє й прочитає повідомлення, скориставшись поштовим клієнтом або браузером (якщо це сервер у Вебі).

Щоб переглянути одержані повідомлення, потрібно натиснути посилання **Вхідні** або **Перевірити пошту**. У вікні перевірки пошти відображається список вхідних повідомлень з вказуванням їхньої назви (теми), адреси відправника, дати отримання, розміру та статусу (рис. 4).

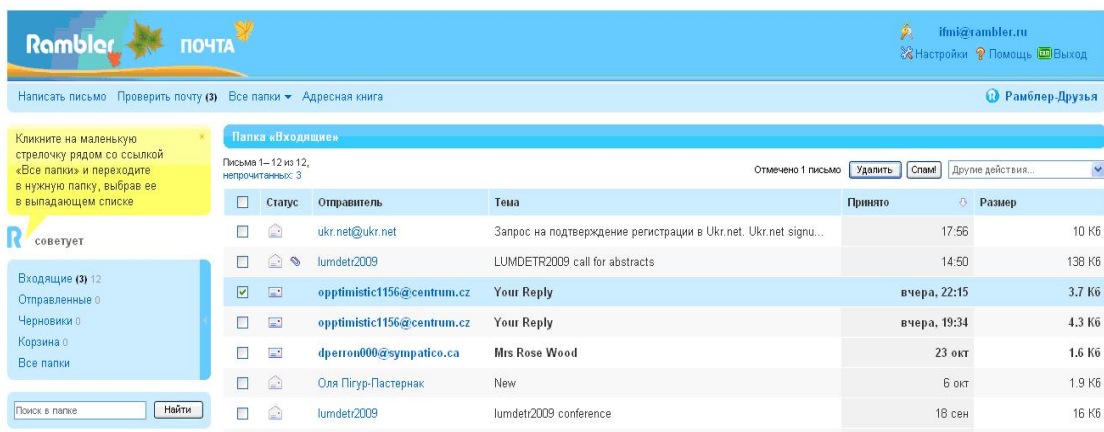


Рис. 4. Структура папки **Вхідні** на сервері Rambler.ru.

Статус повідомлення відображений відповідним значком: закритий конверт – непрочитане повідомлення, відкритий конверт – прочитане повідомлення, конверт зі скріпкою – повідомлення з вкладенням. Крім вмісту папки **Вхідні** та стандартних посилань на сторінці веб-пошти розміщені кнопки для роботи з кореспонденцією, зокрема кнопки **Видалити**, **Видалити як спам**, **Перемістити до іншої папки**, **Переслати**, **Відповісти**.

Завершити роботу у поштової скриньці можна лише натиснувши посилання **Вихід (Exit)** чи відповідну кнопку, піктограму. В жодному разі не можна виходити з поштової скриньки за допомогою кнопки закриття вікна. Після виходу з поштової скриньки всі повідомлення збережені у папці **Корзина** будуть знищені.

## ЗАВДАННЯ

1. Створити на поштовому сервері Ukr.net свою електронну поштову скриньку.

Для виконання завдання проробіть такі дії:

- завантажте браузер Internet Explorer;
- у вікні програми до поля Адреса введіть електронну адресу поштового серверу www.ukr.net. Дочекайтеся завантаження початкової сторінки цього вузла;
- у вікні *Реєстрація нового користувача* заповніть поля початкової анкети:
- після заповнення представлених полів натисніть кнопку **Продовжити**;

- заповніть поля анкети, що залишилися;
  - обов'язково введіть у поле цифри зображені на малюнку. Перевірте правильність вводу та клацніть на кнопці *Зареєструватися*. Поштова скринька буде створена;
  - запишіть електронну адресу своєї поштової скриньки (типу: <ваш логін>@ukr.net) та роздайте її одногрупникам. Запам'ятайте пароль.
2. Створити електронний лист у своїй поштовій скриньці і відіслати його на тестові адреси student\_ddpu@rambler.ru, student\_ddpu@ukr.net, а також комусь із одногрупників. Зберегти тестові електронні адреси та e-mail своїх одногрупників в Адресній книзі.

Для виконання завдання проробіть такі дії:

- завантажте браузер Internet Explorer;
  - у вікні програми до поля Адреса введіть електронну адресу www.ukr.net. Дочекайтеся завантаження початкової сторінки цього вузла;
  - введіть у відповідні поля свій логін і пароль та натисніть кнопку *Ввійти*;
  - на веб-сторінці своєї поштової скриньки виберіть посилання *Адресна книга*;
  - на сторінці адресної книги натисніть кнопку *Додати адресу*, в поле E-mail запишіть електронну адресу student\_ddpu@rambler.ru, в поле Ім'я запишіть "Тестова адреса";
  - збережіть в адресній книзі тестову адресу student\_ddpu@ukr.net та електронні адреси кількох одногрупників;
  - на веб-сторінці своєї поштової скриньки виберіть посилання *Написати листа*;
  - на веб-сторінці створення листа у полі *Кому*: вкажіть тестову електронну адресу, вибравши її з адресної книги. Для цього натисніть на посилання *Адресна книга* та клацніть на записі тестової адреси; аналогічно у полі *Копія*: вкажіть тестову електронну адресу student\_ddpu@ukr.net, у полі *Прихована копія* вкажіть електронні адреси одногрупників, а у полі *Тема*: вкажіть назву листа Тест. У полі *Текст листа* напишіть повідомлення: "Привіт. Це моя електронна адреса. Пишіть мені.";
  - натисніть кнопку *Відправити*.
3. Надіслати лист викладачу на повідомлену ним адресу. Лист обов'язково має містити інформацію про тему лабораторної роботи і студента, що її

виконує, та файл вкладення у будь-якому форматі.

4. Перевірити свою поштову скриньку. Відповісти на отримані повідомлення, відіславши електронні листи з файлами-вкладенням у різних форматах (текстовий, графічний, електронні таблиці тощо). Зберегти електронні адреси однокласників у папці Контакти.
5. Переглянути вміст папки Вхідні, створити у ній папку з назвою Важливі листи і перемістити туди кілька листів. Видалити непотрібні листи.
6. Відіслати на тестову адресу `student_ddpu@ukr.net` та однокласникам листівку з будь-якого поштового сервера (наприклад, `www.mail.ru`, `www.yandex.ru`).
7. Оформити звіт з виконання лабораторної роботи

### **КОНТРОЛЬНІ ЗАПИТАННЯ**

1. Що таке електронна пошта?
2. Яку структуру має адреса електронної пошти?
3. Що потрібно для того, щоб одержати власну поштову скриньку та електронну адресу?
4. Які поля у вікні електронного повідомлення обов'язкові для заповнення?
5. Чи можна відправити одне повідомлення одночасно різним адресатам?
6. Що може містити повідомлення?
7. Що таке повідомлення з вкладенням?
8. За якими ознаками можна визначити, що файл містить вкладення?
9. Які характеристики (атрибути) має поштове повідомлення?
10. На яких основних протоколах базується пошта?
11. Що таке адресна книга і як нею користуватися?
12. Що таке спам? Як обмежити його надходження?
13. Як відправити за допомогою електронної пошти листівку? Які поля є обов'язковими для заповнення?
14. Як правильно завершити роботу із поштовою скринькою?
15. Назвіть основні поштові програми та їхні переваги.

## ТЕМА. Створення власного веб-сайту за допомогою MS Publisher

**МЕТА:** ознайомитися з основами створення веб-документів за допомогою програми MS Publisher; створити веб-сайт на певну тематику, що зберігатиметься у власній папці та по можливості розмістити створений сайт в мережі Інтернет.

### ХІД РОБОТИ

1. Спланувати дизайн макету та інформаційне наповнення веб-сайту.
2. Розробити веб-сайт за допомогою програми MS Publisher.
3. Розширити вміст сайту шляхом використання звуку, зображень, таблиць.
4. Доповнити документ гіперпосиланнями на інші інформаційні ресурси.
5. Відредагувати сайт, розглянути правила розміщення створеного веб-ресурсу в мережі.

### ТЕОРЕТИЧНІ ВІДОМОСТІ

MS Publisher призначена для створення професійних публікацій. Зокрема, за допомогою цієї програми можна швидко і якісно створювати багато різноманітних типів публікацій для друку, а саме: різного типу оголошення, інформаційні бюлетені (стіннівки, міні-газети), листівки, запрошення на певні свята, власні візитні картки, брошури або плакати певного змісту тощо.

Крім того, MS Publisher дає можливість створювати прості веб-сайти (групу веб-сторінок, присвячену певній темі та розміщену в певному каталозі веб-сервера) і, тим самим, розташовувати інформацію в мережі Інтернет, що робить її доступною широкому колу користувачів та дає змогу суттєво зменшити витрати на друк.

Особисті веб-сайти можна з успіхом *використовувати* як:

- інформаційні ресурси для інших користувачів;
- для встановлення зв'язку з іншими студентами у світі;
- як засіб пошуку партнерів для здійснення власних проектів;
- для демонстрації процесу навчання, який охоплює вивчення матеріалів з мультимедійної енциклопедії, довідкової бібліотеки, Інтернету та інших

джерел;

- для опублікування результатів оглядів та анкетних опитувань;
- для представлення творів, наприклад, віршів, оповідань і наукових робіт;
- для відображення різноманітних подій, що відбуваються на факультеті чи в групі (наприклад, для розміщення матеріалів краєзнавчих експедицій, фотографій з екскурсій);
- для створення інтерактивної газети чи журналу, що відображає події з життя факультету, що дасть можливість заощадити кошти на друк.

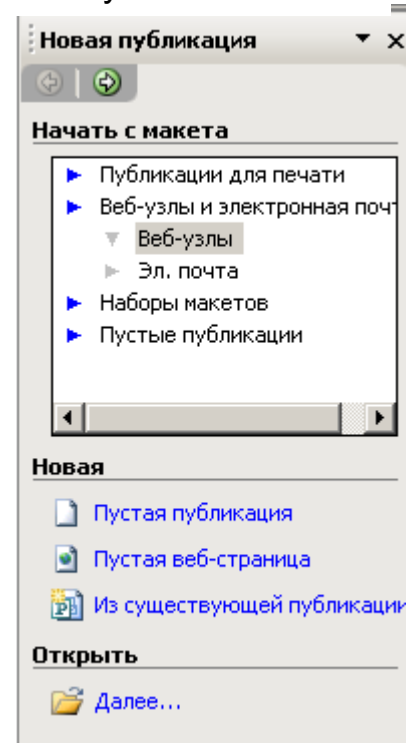
Редактор **MS Publisher**, який дає змогу створювати веб-сайти, що складаються з декількох сторінок, найкраще використовувати для створення простого веб-сайту, який буде швидко завантажуватися. Проте, слід зауважити, що цю програму недоцільно використовувати для розробки складного сайту або такого, що потребує постійного оновлення.

При розробці та створенні власного тематичного веб-сайту потрібно чітко дотримуватися поставленого завдання. При плануванні змісту веб-сайту *необхідно зауважувати чимало чинників*:

- Яке з'єднання з Інтернетом буде мати більша частина відвідувачів майбутнього веб-сайту: швидке чи повільне. Від цього суттєво залежать кількість і розміри зображень на веб-сайті,
- Які версії браузерів будуть використовувати відвідувачі веб-сайту: сучасні або старі (що не дають можливості розглядати сторінки з таблицями, фреймами та анімацією. Не слід забувати, що створений вами веб-сайт можуть відвідувати люди, які працюють з різними браузерами та різним за потужністю комунікаційним обладнанням.

Також потрібно звернути увагу на узгодженість елементів дизайну веб-сайту. На кожній сторінці веб-сайту *доцільно розмістити*:

- ◆ кнопки або гіперпосилання для переходу на головну сторінку;
- ◆ адресу електронної пошти;
- ◆ дату останньої модифікації веб-сайту;
- ◆ URL-адресу сайту (на даному етапі ви не зможете розмістити URL-адресу на веб-сторінці).



Доки веб-сайт не завершено, його потрібно зберігати як файл MS Publisher. У вигляді веб-сторінок зберігають лише готовий веб-сайт. Крім того, після збереження сайту у вигляді веб-сторінки не потрібно видаляти файли веб-сайту у форматі MS Publisher, щоб мати змогу з часом вносити в нього зміни.

Для того, щоб розробити за допомогою цього редактора власний тематичний сайт, потрібно одразу після запуску програми в діалоговому вікні **Нова публікація** у віконці **Почати з макету** обрати тип публікації **Веб-вузли**. Після цього можна буде переглянути шаблони веб-сайтів, зразки яких відображаються у правій частині вікна та вибрати макет, що максимально підходить сайту (рис. 1).

Після вибору макету відкриється вікно майстра Зручний конструктор веб-вузлів, в якому слід обрати потрібну кількість сторінок, з яких буде складатися створюваний сайт.

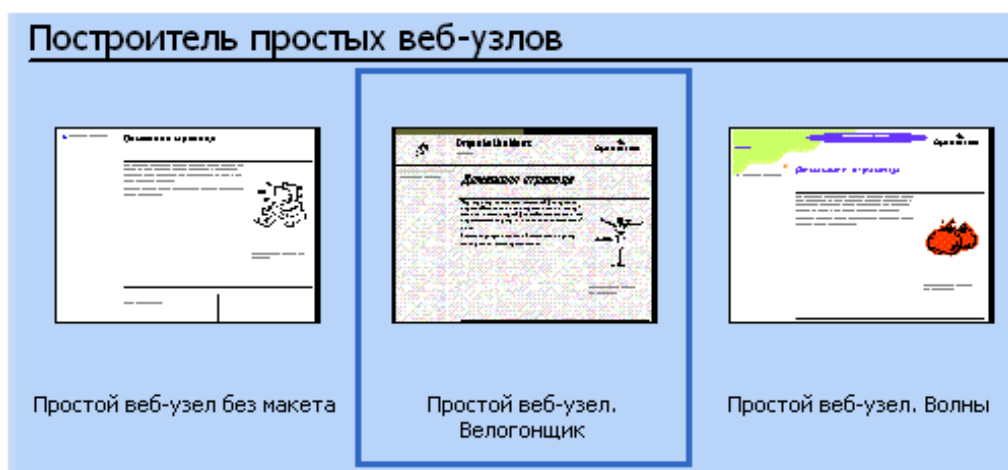


Рис. 1. Зразки макетів сайтів, що пропонуються в MS Publisher.

**Заголовки** веб-сторінок і **пункти навігаційного меню** у програмі MS Publisher пов'язані між собою: при редагуванні одного з цих елементів текст автоматично оновлюється і в іншому. Дуже важливо, щоб жодні два елементи на сторінці не накладалися, і щоб між текстом та графікою була відстань. Необхідно пам'ятати, що коли рамка текстового поля торкається іншого об'єкта або накладається на нього, напис зберігається як зображення. Таке зображення не може читатися як текстовий документ, а отже, вміст напису не буде доступним для редагування. Окрім того, це збільшує час завантаження веб-сторінок через Інтернет, а також впливає на відображення ефектів анімації.

Під час інформаційного наповнення сайту варто пам'ятати, що з метою прискорення завантаження сторінок деякі користувачі відключають відображення графічного змісту. Тому бажано передбачити для всіх малюнків, що містяться в публікації, альтернативний текст, який буде в зазначеному випадку виводитися замість малюнка. Щоб задати альтернативний текст для малюнка, треба викликати його контекстне меню і на закладці **Веб** команди **Формат рисунка...** ввести потрібний текст у відповідному вікні.

За допомогою програми MS Publisher можна перевірити, як виконано дизайн веб-сайта. Якщо при цьому буде виявлена проблема, на екрані відобразиться відповідне діалогове вікно. Цю можливість забезпечує інструмент **Перевірка макета** (рис. 2), який шукає порожні поля, текст в області переповнення, непропорційні зображення, об'єкти, що частково виходять за межі сторінки, текст, що перетворюється на графіку, порожній простір у верхній частині веб-сторінки, сторінки, з якими не можуть зв'язатися гіперпосилання, та об'єкти, що довго завантажуються.

За допомогою **гіперпосилань** можна зв'язати сторінки створюваного веб-сайту між собою або з будь-яким інформаційним об'єктом, що перебуває в мережі Інтернет. Для створення гіперпосилань між елементами створюваного веб-документу потрібно обрати об'єкт або текстовий блок, який потрібно зв'язати з іншою сторінкою веб-сайту ⇒ **Вставка** ⇒ **Гіперпосилання** ⇒ у діалоговому вікні **Додавання гіперпосилання** клікнути на панелі **Зв'язати з місцем у документі**. Після чого слід обрати сторінку, з якою потрібно зв'язати це гіперпосилання ⇒ **ОК**.

Якщо потрібно створити посилання на певний інформаційний ресурс, розміщений в мережі Інтернет, то у діалоговому вікні **Додавання гіперпосилання** потрібно обрати команду **Зв'язати з файлом, веб-сторінкою** і ввести у відповідному полі URL-адресу веб-сайту, на який створюється гіперпосилання (рис. 3).

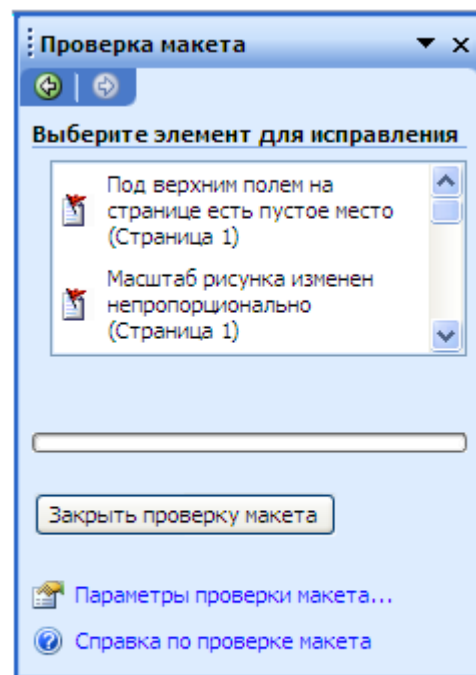


Рис. 2. Діалогове вікно для перевірки макета публікації.



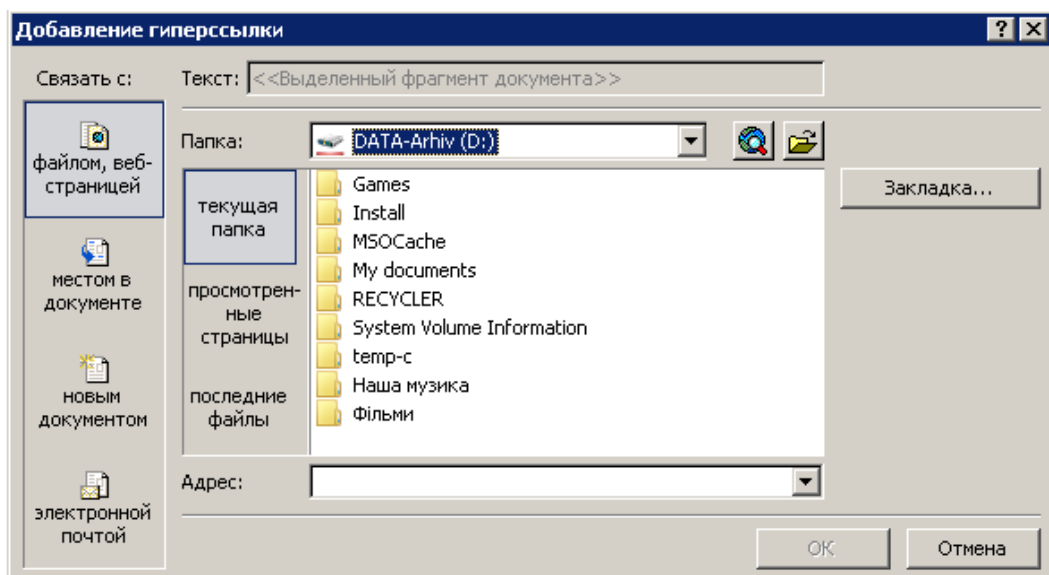


Рис. 3. Діалогове вікно Додавання гіперпосилання.

Щоб побачити, як створюваний веб-документ відобразиться у браузері, можна скористатися командою **Файл** ⇒ **Попередній перегляд веб-сторінки**, після чого автоматично завантажується веб-оглядач, що використовується в системі за замовчуванням, в якому буде відображено розроблюваний сайт.

Створений веб-сайт можна розмістити в мережі Інтернет на вузлі, призначеному для безкоштовного розміщення сайтів, або на вузлі вашого провайдера послуг Інтернету. На багатьох вузлах безкоштовного розміщення сайтів застосовують рекламу, яка може виявитися небажаною, або накладають певні обмеження для користувачів. Тому перед розміщенням веб-сайта в Інтернеті необхідно переконатися, що умови угоди про розміщення сайтів вас влаштовують.

Для передачі файлів на Web-сервер поширені дві можливості:

- з використанням протоколу HTTP;
- з використанням протоколу FTP.

В обох випадках необхідно зареєструватися на одному з безкоштовних Web-серверів, наприклад: <http://www.hostinger.ru>, <http://ayola.net>, <http://www.hut.ru>, <http://www.freehost.kiev.ua>, <http://www.topua.net>, <http://www.ukrbiz.net>.

Під час реєстрації необхідно заповнити відповідні реєстраційні поля, одне з яких передбачає введення електронної пошти. Після реєстрації автоматично буде відправлено на вказану пошту логін, пароль, адресу для

входу в панель керування сайтом, адресу новоствореного вашого сайту (поки що порожнього), FTP сервер, FTP логін, FTP пароль. Ці дані потрібні для керування сайтом.

Для передачі файлів за допомогою протоколу HTTP потрібно увійти у панель управління сайтом, ввівши одержані на пошту логін та пароль. У панелі керування треба знайти вкладку (посилання чи менеджер завантаження файлів). Після цього необхідно там створити папку з назвою index.files (можна іншу назву, але для певності краще цю). У створену папку на Web-сервері завантажити з власної папки свого комп'ютера всі файли, а в кореневий каталог на Web-сервері завантажити файл index.html. Після завершення завантаження можна зайти через будь-який Web-браузер на свій сайт за адресою, що вислана вам на пошту.

Можливий ще один варіант пересилання – це протокол FTP, що призначений для обміну файлами між комп'ютерами, пов'язаними між собою локальною чи глобальною мережею. У такому випадку комп'ютери взаємодіють один з одним за технологією «клієнт-сервер». Файли зберігаються в центральному комп'ютері (FTP-сервері), до якого підключені комп'ютери розподіленої мережі (FTP-клієнти). Клієнт посилає на сервер запит і одержує у відповідь необхідні йому файли. FTP допускає двобічний обмін файлами між сервером і клієнтом.

На комп'ютері-сервері повинна бути встановлена програма FTP-сервер, а на клієнтських комп'ютерах – програма-клієнт. Остання може бути як окремою програмою (наприклад, відома програма CuteFTP), так і модулем, вбудованим в іншу програму (наприклад, FAR, Total Commander).

### **Алгоритм розміщення сайту за допомогою програми Total Commander.**

1. Запустити Total Commander.
2. Вибрати **FTP** ⇒ **З'єднатися з FTP-сервером...**, в результаті чого з'явиться вікно З'єднання з FTP-сервером. У найбільшій ділянці вікна є перелік створених FTP-з'єднань.
3. Натиснути кнопку **Додати** ⇒ у діалоговому вікні **Налаштування FTP-з'єднання** вказати ім'я з'єднання та ввести адресу FTP-сервера, яку було вислано на пошту ⇒ **Ок**.
4. Вибрати створене FTP-з'єднання і натиснути кнопку **З'єднатися**.
5. Ввести послідовно ім'я користувача FTP (FTP-логін), FTP пароль і дочекатися з'єднання.

6. Після з'єднання на одній з панелей Total Commander буде відображатися вміст папок на Web-сервері.
7. На іншій панелі Total Commander слід перейти у папку, де міститься ваш сайт (на ПК, за яким ви працюєте).
8. Скопіювати потрібні папки та файли та від'єднатися.

Після завершення завантаження можна зайти через будь-який Web-браузер на свій сайт за адресою, що була вислана вам на пошту.

## **ЗАВДАННЯ**

1. Запустіть програму MS Publisher на виконання. Створіть нову публікацію і виберіть для неї макет веб-сайту.

Для виконання завдання проробіть такі дії:

- Пуск ⇒ Програми ⇒ MS Office ⇒ MS Publisher.
- В ділянці завдань Нова публікація у віконці Почати з макету виберіть тип публікації Веб-вузли та електронна пошта ⇒ виберіть посилання Веб-вузли.
- Перегляньте шаблони веб-сайтів, зразки яких відображаються у правій частині вікна та виберіть макет, що максимально підходить вашому сайту.
- У вікні майстра Зручний конструктор веб-вузлів встановіть прапорець на закладці Розміщення відомостей про профіль організації.

2. На екрані відобразиться діалогове вікно Особиста інформація. Заповніть у ньому відповідні поля. Якщо це вікно не відобразилося, виконайте наступні дії: Правка ⇒ Особисті дані ⇒ заповніть відповідні поля власною інформацією.

3. Задайте розмітку та оберіть кольорове оформлення для Вашого сайту.

Для виконання завдання проробіть такі дії:

- Формат ⇒ Параметри: веб-вузол ⇒ Панель навігації ⇒ виберіть варіант Вертикальна та горизонтальна.
- У ділянці завдань Параметри: веб-вузол оберіть Колірні схеми ⇒ Застосувати колірну схему ⇒ виберіть один із варіантів кольорового оформлення.
- Збережіть створений документ у власній папці. Програма автоматично додасть до імені файлу розширення .pub.

4. Заповніть веб-сторінку необхідною текстовою інформацією, картинками

- та фотографіями; оберіть відповідний логотип.
5. Створіть гіперпосилання на інші сторінки веб-сайту та на об'єкти, що розміщені в мережі Інтернет.
  6. Збережіть веб-сайт у форматі публікації MS Publisher. Перевірте макет сайту за допомогою засобів перевірки макета веб-сайту
    - Сервіс ⇒ Перевірка макета ⇒ в області завдань Перевірка макета у списку Виберіть елемент для виправлення відобразиться перелік виявлених помилок.
    - Виберіть команду Параметри перевірки макета ⇒ на вкладці Загальні визначте, в якому порядку буде відображатися інформація про знайдені помилки ⇒ перейдіть на вкладку Перевірки ⇒ конкретизуйте, які саме помилки слід шукати. Виправте знайдені помилки.
    - Перегляньте створений веб-сайт Файл ⇒ Попередній перегляд веб-сторінки.
  7. Збережіть веб-сайт у форматі веб-сторінки за допомогою команди Файл ⇒ Помістити на веб-вузол ⇒ відкрийте власну папку ⇒ в полі Ім'я файлу перевірте написання імені файлу ⇒ Зберегти. Оскільки це ім'я стане частиною URL-адреси вашого веб-сайту, необхідно переконатися, що це саме те ім'я, яким би ви хотіли користуватися, і що його легко запам'ятають відвідувачі.
  8. Перевірте роботу створеного веб-сайту. Для цього відкрийте Мій комп'ютер або Провідник і знайдіть папку, в якій зберігається веб-сайт ⇒ Двічі клацніть на файлі HTML з потрібним іменем.
  9. Знайдіть адреси декількох сайтів, що пропонують безкоштовне розміщення сайтів в мережі Інтернет. За потреби розмістіть створений веб-вузол в мережі Інтернет, використовуючи алгоритм, наведений в теоретичних відомостях.
  10. Оформіть звіт з виконання лабораторної роботи.

### **КОНТРОЛЬНІ ЗАПИТАННЯ**

1. Яке призначення програми MS Publisher?
2. Для чого створюють власні веб-сайти?
3. На що варто звернути увагу, плануючи зміст веб-сайту?
4. Як змінити розмітку сайту?
5. Як змінити заголовок веб-сторінки?

6. Як додати нову сторінку до веб-сайту?
7. Які об'єкти може містити веб-сторінка?
8. Як додати до веб-сайту зображення?
9. Як вставити таблицю у веб-сторінку?
10. Як створити гіперпосилання на певний файл?
11. Як створити гіперпосилання на Інтернет-сторінку?
12. Як створити гіперпосилання на іншу сторінку сайту?
13. Як можна змінити кольорове оформлення сайту?
14. Як зберегти створений сайт у форматі MS Publisher?
15. Як переглянути створений веб-сайт?
16. Як зберегти готовий сайт у форматі веб-сторінки?
17. Що необхідно здійснення FTP-обміну?
18. Які програми FTP-клієнти вам відомі?
19. Які є способи передачі файлів на Web-сервер?
20. Наведіть приклади серверів, що безкоштовно розміщують сайти.

#### Лабораторна робота № 6

### **ТЕМА. Формування веб-сторінок засобами візуального редактора MS Frontpage. Робота з текстом та малюнками. Створення гіперпосилань**

**МЕТА:** ознайомитися з основами створення веб-документів за допомогою програми MS Frontpage; створити веб-сайт на певну тематику, доповнити його малюнками та гіперпосиланнями.

#### **ХІД РОБОТИ**

1. Спланувати дизайн макету та інформаційне наповнення веб-сайту.
2. Розробити веб-сайт за допомогою програми MS Frontpage.
3. Розширити вміст сайту шляхом використання зображень.
4. Доповнити документ гіперпосиланнями на інші інформаційні ресурси.
5. Відредагувати створений документ.

#### **ТЕОРЕТИЧНІ ВІДОМОСТІ**

Найбільш поширеними візуальними редакторами для створення сайтів

є Frontpage і Dreamweaver. **Microsoft Frontpage XP** – сучасна інтегрована оболонка для побудови окремих веб-сторінок і цілих веб-вузлів. За допомогою програми FrontPage можна створити свій веб-вузол і розмістити його в мережі Інтернет. При цьому не обов'язково знати мови програмування в HTML-кодах, оскільки ця програма передбачена як для програмістів, так і для користувачів, не знайомих з програмуванням. Крім того, редактор Frontpage містить великий набір шаблонів і майстрів для створення сайтів з різної тематики.

Недоліком редактора Frontpage є його орієнтація на браузер Internet Explorer (слід перевіряти роботу створеного сайту в інших браузерах) та деяка надмірність готового коду HTML (редактор відстежує зміни в кодї сторінок і відновлює теги, видалені розробником).

### **Створення web-вузла за допомогою FrontPage**

**Веб-вузол** – це набір файлів у форматі HTML, розташованих у певній теці і зв'язаних один з одним гіперпосиланнями. Один з файлів веб-вузла є головним, відкривається в браузері користувача при підключенні до веб-вузла. Решта сторінок відображаються у вікні браузера в міру переходу до них за гіперпосиланнями. Окрім файлів HTML до складу вузла належить набір графічних об'єктів формату GIF або JPG, призначених для оформлення сторінок.

Щоб створити закінчений веб-вузол, недостатньо просто розмістити в одній теці декілька HTML-файлів. Грамотно побудований вузол має добре продуману структуру, що полегшує користувачеві пошук необхідної інформації.

На початку роботи в редакторі краще компонувати вузол за допомогою майстра веб-вузла, для чого потрібно виконати наступні кроки. Запустити Frontpage ⇒ **Файл** ⇒ **Створити** ⇒ **Сторінка або веб-вузол** ⇒ в ділянці завдань додатку відкриється вікно **Створення веб-сторінок** із списком шаблонів і майстрів, якими можна скористатися для побудови веб-вузла (рис.1).

Для того, щоб створити веб-вузол, слід обрати потрібний шаблон. Після чого заповнити відповідні поля діалогового вікна майстра створення веб-вузла (рис. 2).

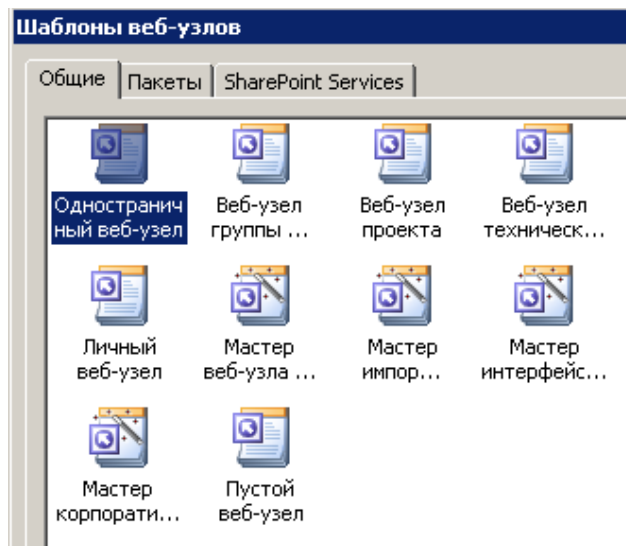


Рис 1. Майстри і шаблони веб-вузлів.

Майстер згенерує новий веб-вузол і відкриє його в режимі перегляду завдань із списком дій, які необхідно виконати для отримання закінченого вузла. У процесі розробки вузла можна додавати нові завдання. Список завдань зберігається разом з файлами веб-вузла і не дасть забути про незавершені операції.

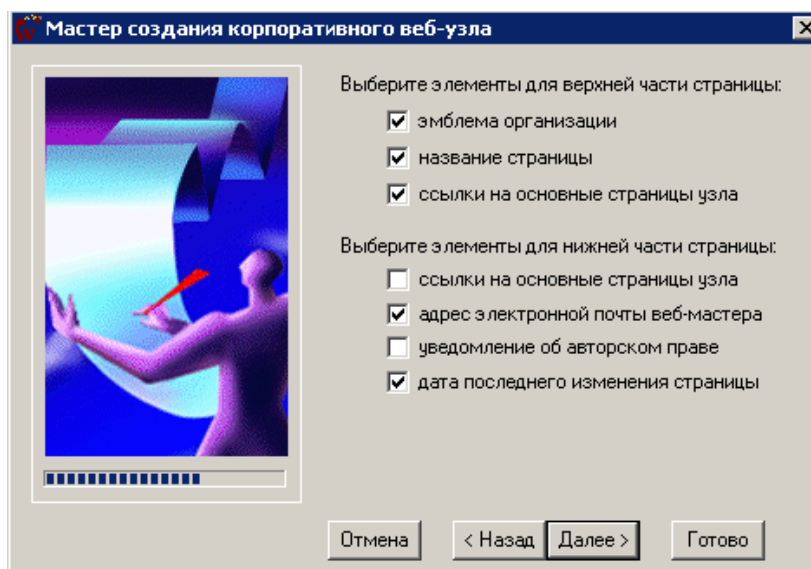


Рис. 2. Налаштування оформлення сторінок за допомогою майстра корпоративного веб-вузла.

### Режими перегляду

Frontpage пропонує шість режимів перегляду веб-вузла, перемикаючи які можна за допомогою команд меню Вигляд:

- **Сторінка** – режим перегляду і редагування окремої веб-сторінки. У цьому режимі веб-сторінці можна задати тему її оформлення, розмістити

на ній текст з використанням елементів форматування, графічні і відеозображення, гіперпосилання, таблиці, фрейми, проглядати HTML-код, на основі якого генерується сторінка.

- **Теки** – список тек і файлів веб-вузла з їхніми характеристиками, подібний до списку вікна Провідника Windows, відображає не всі файли веб-вузла, а тільки ті, які несуть певне змістове навантаження. Допоміжні теки, що містять графічні файли стандартних елементів оформлення і файли підтримки розширень Frontpage, залишаються прихованими.
- **Переходи** – редактор структури веб-вузла, що дає змогу у графічному режимі змінювати зв'язки і перебудовувати гіперпосилання. Майстер веб-вузла конструює посилання між сторінками за певним алгоритмом. За допомогою режиму перегляду Переходи можна побачити загальну схему вузла, додати або розірвати певні посилання, скоректувати правила розміщення посилань у панелях навігації веб-сторінок.
- **Гіперпосилання** – список веб-сторінок вузла і схема гіперпосилань виділеної сторінки (рис. 3).
- **Завдання** – список завдань, пов'язаних з певними файлами, які потрібно не забути виконати для завершення розробки вузла.

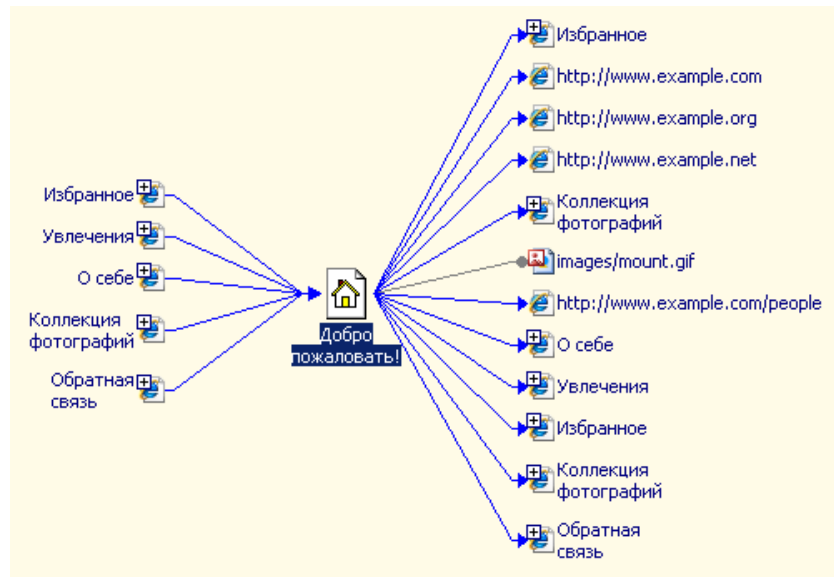


Рис. 3. Веб-вузол в режимі гіперпосилань.

У режимі Сторінка у нижній частині робочої ділянки, в якій розташовується редагована сторінка, є чотири вкладки, що дають змогу переглядати веб-сторінку у різних режимах

1. *Звичайний режим* роботи, при якому здійснюється створення сторінки за допомогою візуальних засобів. Сторінка створюється шляхом



розміщення у ній тексту, ліній, кнопок, посилань і інших об'єктів з використанням меню і панелей інструментів, при цьому не потрібно знання HTML-кодів.

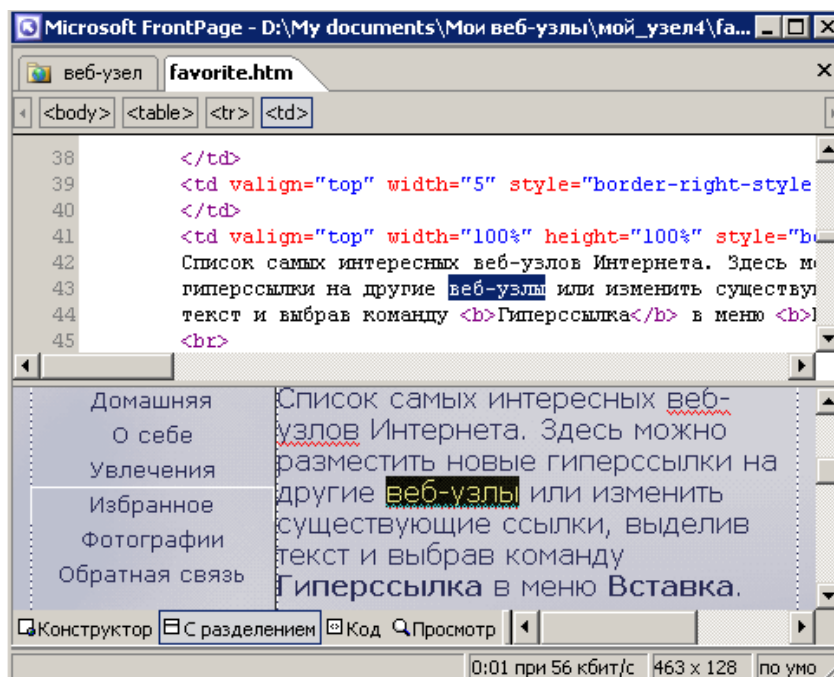


Рис. 4. Перегляд веб-сторінки в режимі з розділенням.

2. *Режим перегляду і створення сторінки в HTML-кодах.* Його можна використовувати як для перегляду HTML-кодів сторінки, створеної у звичайному режимі роботи, так і для створення сторінки за допомогою HTML-коду. При цьому частина сторінки може створюватися у звичайному режимі, а окремі елементи дописуватися в кодах.

3. *Режим з розділенням*, що одночасно відображає на екрані звичайний режим редагування сторінки та в режим перегляду HTML-кодів (рис. 4).

4. *Попередній перегляд сторінки* в тому вигляді, в якому вона буде представлена на сервері Інтернету.

### **Оформлення веб-сторінок**

За замовчуванням FrontPage пропонує для створюваних сторінок білий фон. Колір фону можна змінити, використовуючи пропоновані програмою кольори, що не позначиться на часі завантаження веб-вузла. В якості фону сторінки може виступати графічне зображення, однак це призведе до збільшення розміру файлу, а, відповідно, збільшується і час завантаження сторінки. Об'єднання групи веб-сторінок в єдину структуру вузла дає змогу програмі Frontpage відстежувати схему побудови посилань між сторінками

та забезпечувати однотипне оформлення всіх сторінок вузла.

Кожен веб-вузол будується на основі певної схеми, що задає однакове оформлення всім стандартним елементам сторінок. При розробці вузла у будь-який момент можна змінити схему. Для цього слід виконати такі кроки: обрати команду **Формат** ⇒ **Тема** ⇒ в діалоговому вікні обрати потрібну схему оформлення сторінки. У лівій нижній частині вікна діалогу є чотири прапорці, встановлення яких дає змогу виконувати такі дії:

- яскраві кольори – розфарбовує вибрану схему яскравішими кольорами;
- активні малюнки – ускладнює малюнок посилань, кнопок і маркерів, додаючи тіні і дрібні деталі;
- фоновий малюнок – розміщує на задньому плані сторінки фоновий малюнок;
- застосувати за допомогою CSS – призначає нове оформлення за допомогою каскадних таблиць стилів, що додаються в HTML-код. Каскадні таблиці зі стилями можна підключати до будь-якого HTML-файлу, формуючи на основі цих стилів оформлення елементів сторінки. Параметри стандартних схем можна за потреби модифікувати: створити власну колірну схему та графічне оформлення сторінки. Створену схему можна застосувати до всіх сторінок вузла, обравши у відповідному діалоговому вікні положення **До всіх сторінок** ⇒ ОК.

### **Текстова інформація**

У основі практично будь-якої веб-сторінки лежить текст. Правильне форматування тексту багато в чому визначає зручність сприйняття інформації.

Програма FrontPage надає в розпорядження розробника засоби щодо форматування заголовків, символів, абзаців, що охоплюють установку різних параметрів шрифту, інтервалів між символами, зсувів, відступів, створенню маркованих і нумерованих списків.

Для виділення заголовків, розділів тексту, для розділення інформації і просто для оформлення сторінок використовують горизонтальні лінії. Розміщення горизонтальної лінії на сторінці здійснюється командою **Вставка** ⇒ **Горизонтальна лінія**. Горизонтальна лінія – це єдиний графічний елемент, який можна «намалювати» на сторінці. Будь-які інші фігури (похилі лінії, кола, прямокутники) розміщуються тільки у вигляді файлів формату GIF або JPG, підготовлених у графічному редакторі або

за допомогою інструментів Frontpage.

### Графічні зображення

Графічні зображення, що розміщені на веб-сторінках, покращують сприйняття інформації, роблять сторінки яскравішими і такими, що запам'ятовуються.

При створенні веб-вузлів найчастіше застосовують графічні формати JPEG і GIF. Вибір формату визначається завданнями, що ставляться при розробці. Так, наприклад, GIF формат застосовується для зображень, що містять менше 256 кольорів, і використовується, як правило, для створення анімаційних ефектів. Якщо зображення містить більше 256 кольорів, то застосовується формат JPEG. Крім того, програма FrontPage дає змогу імпортувати файли, що мають інші формати, перетворюючи їх у формати JPEG і GIF.

У веб-сторінках всі малюнки прив'язуються до певної точки тексту, тому для розміщення малюнка у довільних точках сторінки розробники деколи вдаються до деяких особливих прийомів, наприклад розміщують графічні об'єкти в осередках таблиць з невидимими рамками.

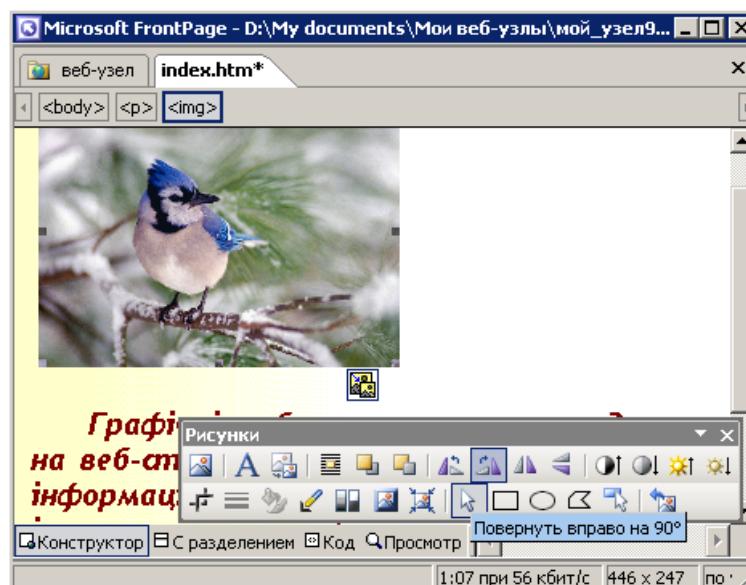


Рис 5. Редагування зображень за допомогою панелі інструментів малювання.

### Гіперпосилання та таблиці

Гіперпосилання призначені для переміщення між сторінками поточного веб-вузла, а також на інші вузли у пошуках потрібної інформації. При створенні веб-вузла за допомогою майстра додаток Frontpage автоматично розміщує на сторінках панелі навігації з гіперпосиланнями на різні веб-сторінки вузла. Посилання додаються і в текст сторінок. Однак для

забезпечення правильного функціонування веб-вузла розробникові часто доводиться вставляти додаткові посилання або змінювати параметри існуючих. Для додавання гіперпосилання можна скористатися кнопкою **Гіперпосилання** панелі інструментів **Стандартна**.

Залежно від тематики та інформаційного наповнення часто виникає потреба помістити на створювану сторінку деякі табличні дані. Крім того, документи формату HTML не підтримують розміщення тексту в декілька стовпців, тому веб-дизайнери часто використовують для цих цілей таблиці.

Для додавання таблиці до веб-сторінки потрібно встановити курсор в потрібне місце та вибрати команду **Вставити** ⇒ **Таблиця**, після чого вказати необхідну інформацію в діалоговому вікні, що відкриється.

Розробляючи веб-сторінки, слід пам'ятати, що їх проглядатимуть люди, які мають комп'ютери з різними дисплеями. Тому бажано задавати розміри елементів у відсотках від ширини вікна браузера. Якщо ви додасте на сторінку малюнок або таблицю шириною 1000 пікселів, то такий об'єкт просто не поміститься на екрані монітора шириною 800 пікселів. Указуючи розміри у відсотках, можна уникнути подібних неприємностей.

## ЗАВДАННЯ

1. Запустіть програму MS Frontpage на виконання. Запустіть майстер створення веб-вузлів та оберіть для вашого документу відповідний шаблон.

Для виконання завдання виконайте такі дії:

- Пуск ⇒ Програми ⇒ MS Office ⇒ MS Frontpage.
- Файл ⇒ Створити ⇒ Сторінка або веб-вузол ⇒ в ділянці завдань додатку відкриється вікно Створення веб-сторінок із списком шаблонів і майстрів ⇒ оберіть Односторінковий веб-вузол ⇒ у вікні, що відкрилося оберіть потрібний шаблон.
- У діалоговому вікні майстра створення веб-вузлів заповніть відповідні поля (не створюйте веб-вузол, що містить більше, ніж три сторінки, оскільки його редагування у такому випадку займе багато часу).
- Майстер згенерує новий веб-вузол і відкриє його в режимі перегляду завдань із списком дій, які необхідно виконати для отримання закінченого вузла.

2. Ввімкніть для зручності панель Список папок: Вигляд ⇒ Список папок або комбінацією клавіш Alt+F1.

3. Оберіть кольорове оформлення майбутнього сайту.

Для виконання завдання проробіть такі дії:

- Виберіть команду Формат ⇒ Тема ⇒ у діалоговому вікні Обрати тему, що відкриється, оберіть потрібну схему оформлення сторінки.
  - Збережіть створений документ.
4. Заповніть веб-сторінку необхідною текстовою інформацією. Використовуючи засоби щодо форматування заголовків, символів та абзаців, відформатуйте доданий текст якнайкраще.
5. Доповніть створений документ картинками та фотографіями; оберіть відповідний логотип.
6. Створіть гіперпосилання на інші сторінки веб-сайту та на об'єкти, що розміщені у мережі Інтернет.
7. Доповніть сторінку таблицею з відповідною інформацією.
- Встановіть курсор у потрібне місце та оберіть команду Таблиця ⇒ Вставити ⇒ Таблиця.
  - У діалоговому вікні, що відкрилося заповніть відповідні поля. Розміри таблиці (ширину та висоту) задайте у відсотках.
  - Заповніть таблицю відповідною інформацією.
  - Перейдіть у режим перегляду HTML-коду та знайдіть відповідні теги, які задають відображення таблиці.
8. Відредагуйте створений веб-вузол та збережіть його у своїй папці.
9. Продемонструйте веб-сайт викладачу та оформіть звіт з виконання цієї лабораторної роботи.

### **КОНТРОЛЬНІ ЗАПИТАННЯ**

1. Яке призначення програми MS Frontpage?
2. Для чого створюють власні веб-сайти?
3. На що варто звернути увагу, плануючи зміст веб-сайту?
4. Яким чином змінити заголовок веб-сторінки?
5. Як додати нову сторінку до веб-вузла?
6. Які об'єкти може містити веб-сторінка?
7. Як додати до веб-сторінки зображення?
8. Як вставити таблицю у веб-сторінку?
9. Як створити гіперпосилання на певний файл?
10. Як створити гіперпосилання на Інтернет-сторінку?

11. Як створити гіперпосилання на іншу сторінку веб-вузла?
12. Яким чином можна змінити кольорове оформлення сайту?
13. Як переглянути створений веб-сайт?
14. Як зберегти готовий сайт у форматі веб-сторінки?
15. Наведіть приклади серверів, що безкоштовно розміщують сайти.

## Лабораторна робота № 7

### **ТЕМА. Формування веб-сторінок засобами візуального редактора MS Frontpage. Робота з фреймами та формами. Динамічні ефекти**

**МЕТА:** продовжити роботу по створенню веб-документів за допомогою програми MS Frontpage; доповнити тематичну веб-сторінку відповідними формами та фреймами.

#### **ХІД РОБОТИ**

1. Створити тематичну веб-сторінку в програмі MS Frontpage.
2. Доповнити створений сайт відповідними формами та фреймами.
3. Доповнити документ динамічними ефектами.
4. Відредагувати створений документ.

#### **ТЕОРЕТИЧНІ ВІДОМОСТІ**

Редактор Frontpage містить великий набір шаблонів і майстрів для створення сайтів з різної тематики. Для форматування тексту можна використовувати всі можливості, передбачені в основному стандарті HTML, застосовувати спеціальні динамічні ефекти і анімацію. Редактор дає змогу легко розміщувати на сторінках різні мультимедіа-об'єкти: малюнки, відеофільми, анімацію, звукові фрагменти. Тісна інтеграція з пакетом MS Office уможливорює відображення на сторінках документів MS Word, таблиць і графіків MS Excel, дає змогу динамічно отримувати дані з MS Access, використовувати мову VBA, засоби перевірки орфографії і десятки готових тем для оформлення сторінок сайту.

У програмі Frontpage реалізована підтримка сучасних веб-технологій, таких, як каскадні таблиці стилів (CSS), динамічні ефекти (DHTML),

фрейми, активні сторінки (ASP), елементи ACTIVEX і Java-аплети. Програма Frontpage є не тільки редактором веб-сторінок, але і містить засоби управління створенням сайту, такі, як схема навігації на сторінках, аналіз сайту за допомогою різних звітів, колективна розробка, настроювання на певні браузері, завантаження сайту на веб-сервер за протоколами HTTP і FTP. Всі ці можливості дають змогу створювати за допомогою Frontpage повноцінні веб-сайти, такі, наприклад, як електронні магазини або ігрові сайти.

### **Ефекти динамічного HTML**

Розробникові веб-вузла, що працює з програмою FrontPage, надаються різні засоби, які дають змогу зробити веб-сторінки більш цікавими. До таких засобів насамперед належать **динамічні ефекти** (Dynamic HTML).

Frontpage дає змогу створювати динамічні сторінки, які не просто відображають дані, але і реагують на певні дії користувача, наприклад на переміщення покажчика миші. При оформленні веб-сторінок можна використовувати різні анімаційні ефекти, запозичені з програми PowerPoint. При створенні ефектів анімації використовується панель інструментів **Ефекти DHTML**. Комбінуючи значення всіх трьох списків цієї панелі інструментів, можна формувати найрізноманітніші ефекти. Наприклад, при завантаженні веб-сторінки текст влітатиме спіраллю або хвилеподібно, в'їжджатиме згори, знизу, справа, зліва або по діагоналі з різних кутів сторінки. Розмір розміщеного тексту можна збільшувати або зменшувати. Окремі слова тексту можуть падати зверху або випливати, рухаючись хвилеподібно.

Для розміщення на веб-сторінці динамічних HTML-об'єктів використовується діалогове вікно **Вставка компонента веб-вузла**, що пропонує такі компоненти: швидкий рядок, змінна кнопка (рис. 1), диспетчер оголошень тощо.

Є одне істотне обмеження, що перешкоджає широкому використанню динамічних ефектів при створенні веб-вузлів. Воно полягає в тому, що не всі оглядачі, за допомогою яких користувачі переглядають веб-сторінки, ці ефекти підтримують.

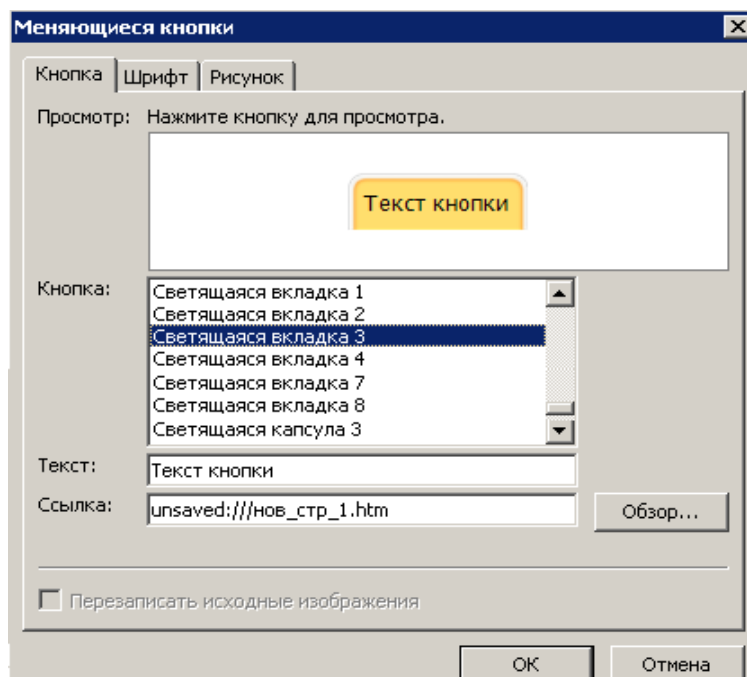


Рис 1. Діалогове вікно Змінна кнопка.

## Використання фреймів

Для того, щоб розбити веб-сторінку на окремі ділянки, можна використовувати таблиці. Набагато зручнішим, способом є фрейми, які дають змогу розбивати веб-сторінки на прямокутні ділянки, в кожній з яких можуть бути відображені окремі документи.

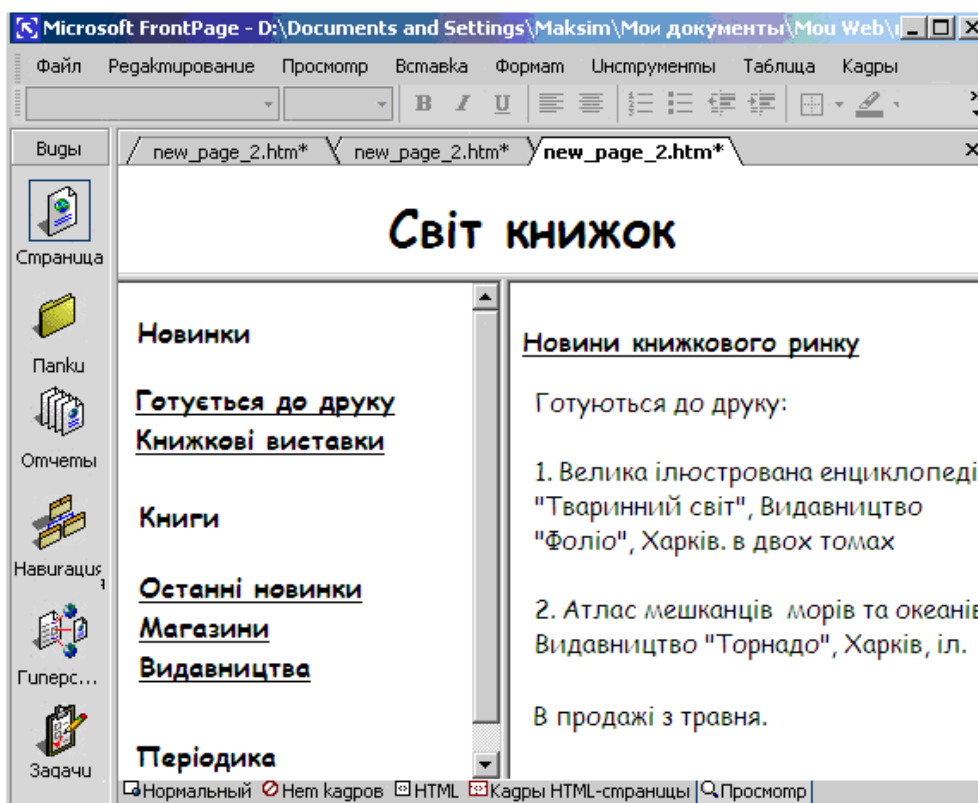


Рис. 2. Використання фреймів на веб-сторінці.



Фрейми зручно використовувати у тих випадках, якщо потрібно відобразити на веб-сторінці інформацію із складною системою рубрикації. У цьому випадку використовують фрейм, що ділить сторінку на дві ділянки. У одній ділянці розміщується зміст із посиланням на розділи рубрикатора, а в іншій – вибрана за посиланням веб-сторінка. На рис. 2 показана сторінка, яка розбита на три області завдяки використанню фрейму. У верхній області відображено заголовок компанії, в лівій – зміст веб-вузла, а в правій – зміст розділу, вибраного за допомогою змісту.

### **FrontPage-компоненти**

Компоненти, що надаються в розпорядження розробника програмою FrontPage, є готовими до вживання програмними модулями, для використання яких досить розмістити компоненти на веб-сторінці і налаштувати їхні властивості. Використання компонентів розширює функціональні можливості веб-вузла, прискорює розробку і позбавляє від необхідності програмування. Найбільш вживаними компонентами є:

- *Підстановка* (Substitution) – розміщує на веб-сторінці інформацію, що задається за допомогою змінної. При зміні значення змінної відбувається автоматичне оновлення даних в усіх місцях, де використана змінна.
- *Сторінка* (include Page) – вставляє зміст окремої сторінки з періодично оновлюваною інформацією у інші сторінки веб-вузла.
- *Сторінка з розкладом* (Scheduled Include Page) – вставляє зміст окремої сторінки в інші сторінки веб-вузла на певний інтервал часу.
- *Малюнок з розкладом* (Scheduled Picture) – вставляє графічне зображення на сторінку веб-вузла на певний інтервал часу.
- *Лічильник відвідувань* (Hit Counter) – розміщує на веб-сторінці компонент, що дають змогу отримати відомості про кількість користувачів, що переглянули веб-сторінку.
- *Форма пошуку* (Search Form) – дає змогу відвідувачам веб-вузла проводити пошук необхідної інформації.
- *Зміст* (Table of Contents) – розміщує на домашній сторінці зміст із посиланням на всі сторінки веб-вузла.
- *Електронна таблиця Office* (Office Spreadsheet) – додає на сторінку електронну таблицю, що є об'єктом Microsoft Office.
- *Діаграма Office* (Office Chart) – додає на сторінку діаграму, що є об'єктом Microsoft Office.

- *Зведена таблиця Office* (Office PilotTable) – додає на сторінку об'єкт Microsoft Office, що дає змогу під'єднатися до бази даних за допомогою OLE DB, проглянути заданий об'єкт і змінити його вміст.

Для розміщення на веб-сторінці компонентів використовується діалогове вікно Вставка компонента веб-вузла (рис. 3), що відкривається за допомогою команди **Вставка** ⇒ **Веб-компонент**.

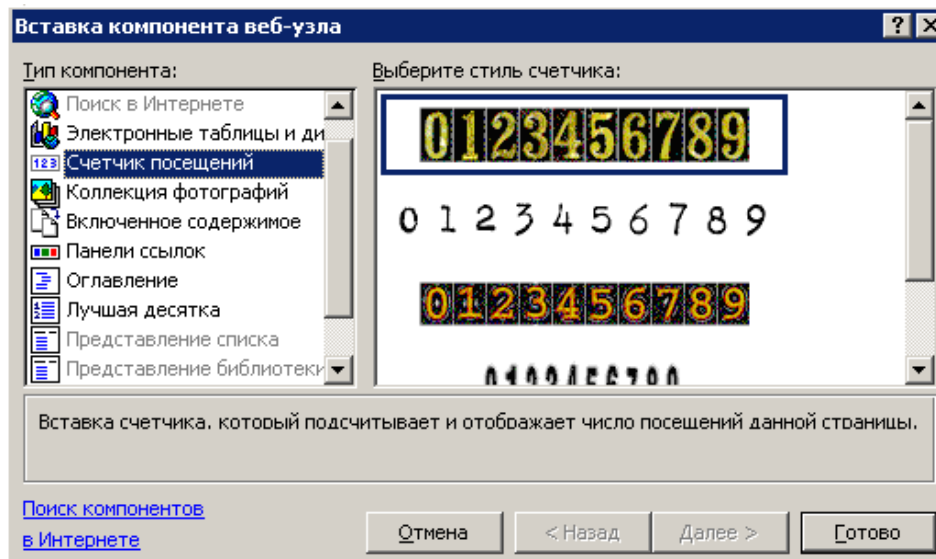


Рис. 3. Діалогове вікно, що використовується для вставки веб-компонентів.

## Форми

Форми містять об'єкти, які дають можливість відвідувачам вводити деяку інформацію. Це можуть бути поля, в які інформація вводиться вручну, списки, що містять варіанти відповідей для вибору відповідного варіанта, групи перемикачів, прапорці, які передбачають два варіанти відповіді тощо. Форми також можуть містити кнопки, що дають змогу виконувати певні дії, наприклад, переслати на сервер для подальшої обробки введену в поля інформацію або очистити поля введення форми.

Залежно від інформаційного наповнення форма може займати цілу веб-сторінку або її частину. Її розмір визначається об'ємом відомостей, які потрібно отримати від користувача веб-вузла. Поля форми на веб-сторінці виділяються контурною пунктирною рамкою. При розробці форми необхідно ретельно продумати, які саме об'єкти в ній доцільно використати. Основні вимоги, що пред'являються до форм – простота, стислість, зрозумілі конструкції для її заповнення.

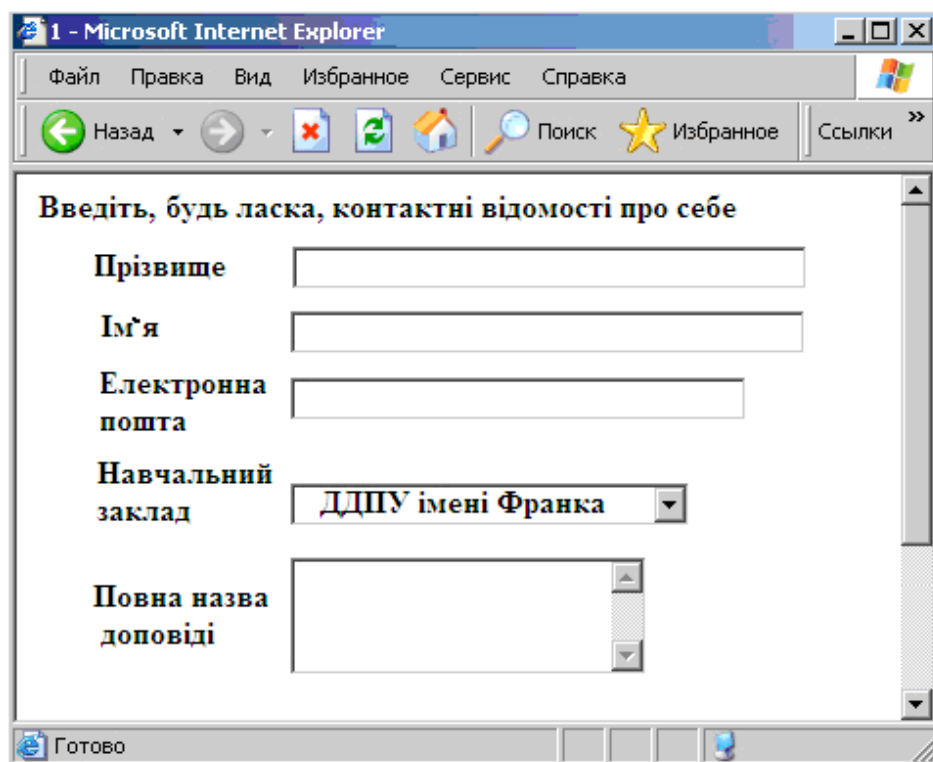
Для створення форм, так само як і для створення веб-сторінок,

програма FrontPage пропонує шаблони і майстра, що полегшують їхню розробку:

- Форма підтвердження – шаблон форми підтвердження про здобуття інформації;
- Форма зворотного зв'язку – шаблон форми для введення зауважень;
- Майстер сторінки форми – майстер створення форми для опитування відвідувачів веб-вузла;
- Гостьова книга – шаблон гостьової сторінки;
- Сторінка пошуку – шаблон сторінки у вигляді форми, використовуваної для пошуку слів;
- Реєстраційна форма – шаблон форми реєстрації користувачів.

Форми, створені за допомогою майстрів і шаблонів, можна редагувати, додаючи в них нові елементи керування. Крім того FrontPage надає засоби для самостійної розробки форм. Для розміщення об'єктів форм використовують команду **Вставка⇒Форма**.

Налаштування і редагування об'єктів форми здійснюються за допомогою вікна їхніх властивостей, яке можна відкрити подвійним кліканням мишки на об'єкті або командою контекстного меню **Властивості поля форми**. За допомогою вікна властивостей об'єктів форми можна задати порядок заповнення полів, зовнішній вигляд кнопок, доповнити поля інформацією певного типу тощо.



The image shows a screenshot of the Microsoft Internet Explorer browser window. The title bar reads "1 - Microsoft Internet Explorer". The menu bar includes "Файл", "Правка", "Вид", "Избранное", "Сервис", and "Справка". The toolbar contains icons for "Назад", "Поиск", and "Избранное". The main content area displays a form titled "Введіть, будь ласка, контактні відомості про себе". The form includes the following fields:

- Прізвище:
- Ім'я:
- Електронна пошта:
- Навчальний заклад:
- Повна назва доповіді:

The status bar at the bottom shows "Готово" and a small globe icon.

Рис. 4. Форма, створена в програмі FrontPage.

Після того, як форма створена, необхідно передбачити засоби для обробки даних, що вводяться в форму. Керування даними можна здійснювати декількома способами:

- Зберегти у файлі, що має формат HTML, звичайний текст або текст бази даних;
- Переслати електронною поштою;
- Передати для обробки в ASP- або CGI-сценарій;
- Помістити в дискусійну або реєстраційну форму.

Налаштування засобів обробки даних, отриманих за допомогою форми, здійснюється у вікні **Властивості форми**.

## ЗАВДАННЯ

1. Запустіть програму MS Frontpage на виконання. Запустіть майстер створення веб-вузлів та оберіть для вашого документу відповідний шаблон.

Для виконання завдання проробіть такі дії:

- Пуск ⇒ Програми ⇒ MS Office ⇒ MS Frontpage.
  - Файл ⇒ Створити ⇒ Сторінка або веб-вузол ⇒ в полі завдань додатку відкриється вікно Створення веб-сторінок із списком шаблонів і майстрів ⇒ оберіть Односторінковий веб-вузол ⇒ у вікні, що відкрилося оберіть потрібний шаблон.
  - У діалоговому вікні майстра створення веб-вузлів заповніть відповідні поля (не створюйте веб-вузол, що містить більше ніж три сторінки, оскільки його редагування в такому випадку займе багато часу).
  - Майстер згенерує новий веб-вузол і відкриє його у режимі перегляду завдань із списком дій, які необхідно виконати для отримання закінченого вузла.
2. Оберіть кольорове оформлення майбутнього сайту за допомогою команди Формат ⇒ Тема.
3. Заповніть веб-сторінку необхідною текстовою інформацією та картинками.
4. Створіть на сторінці форму для одержання певних даних від користувача та форму для пошуку даних.

- Встановіть курсор у потрібне місце та оберіть команду Вставка ⇒ Форма ⇒ оберіть зі списку, що розгорнувся, потрібний об'єкт.
  - Відредагуйте відповідні поля та надписи, використовуючи команди Властивості форми або Властивості поля форми з контекстного меню форми.
  - Перейдіть в режим перегляду та перегляньте створене поле форми.
  - Перейдіть в режим перегляду HTML-коду та знайдіть відповідні теги, які задають відображення форми.
5. Використайте на створених сторінках фрейми та доповніть вміст веб-сторінки динамічними ефектами за допомогою команд Вставка ⇒ Веб-компонент та Вигляд ⇒ Панелі інструментів ⇒ ефекти DHTML.
  6. Відредагуйте створений веб-вузол та збережіть його у власній папці.
  7. Продемонструйте веб-сайт викладачу та оформіть звіт з виконання цієї лабораторної роботи.

### **КОНТРОЛЬНІ ЗАПИТАННЯ**

1. Яке призначення програми MS Frontpage?
2. Для чого створюють власні веб-сайти?
3. На що варто звернути увагу, плануючи вміст веб-сайту?
4. З якою метою на веб-сторінках використовують форми?
5. Як додати до веб-сторінки форму?
6. Як вставити лічильник відвідувань у веб-сторінку?
9. Як створити гіперпосилання на певний інформаційний ресурс?
10. Як можна розбити веб-сторінку на окремі ділянки?
11. Які шаблони форм пропонує програма MS Frontpage?
12. Які динамічні ефекти пропонує MS Frontpage? Як доповнити ними вміст сторінки?
13. Як переглянути створений веб-сайт?
14. Як зберегти готовий сайт у форматі веб-сторінки?
15. Наведіть приклади серверів, що безкоштовно розміщують сайти.

## ТЕМА. Мова HTML. Основні елементи створення html-сторінок. Робота з тегами

**МЕТА:** ознайомитися з основами створення веб-сторінок за допомогою мови розмітки HTML; з основними тегами HTML та оволодіти навичками їхнього використання під час створення веб-сторінок. Використовуючи набуті знання та вміння, створити простий html-документ.

### ХІД РОБОТИ

1. Створити простий html-документ з заданою інформацією.
2. Доповнити створений документ різними тегами та інформацією про себе.
3. Переглянути створений документ за допомогою веб-браузера.
4. Переглянути початковий код сторінки HTML.
5. Відредагувати створений документ за власним уподобанням, використовуючи наведені у теоретичних відомостях теги.

### ТЕОРЕТИЧНІ ВІДОМОСТІ

Окремі документи, які складають веб-простір, називають **веб-сторінками**. Веб-сторінки зберігаються на жорстких дисках веб-серверів – спеціалізованих комп'ютерів з відповідним програмним забезпеченням. Групу сторінок, присвячену певній темі та розміщену у певному каталозі веб-сервера, називають веб-вузлом або **веб-сайтом**. Один фізичний веб-сервер може містити кілька веб-сайтів.

Веб-сторінки мають вигляд звичайних текстових документів, в які введено вказівки форматування. При відображенні таких документів браузером самі вказівки не відображаються, однак впливають на спосіб відображення решти частини документу. Згадані вказівки називаються дескрипторами або **тегами**. З їхньою допомогою текст можна робити кольоровим, використовувати шрифти різного розміру, вбудовувати мультиплікацію, відеофрагменти тощо.

Формат дескрипторів задається в описі спеціальної мови розмітки. Вона називається **мовою розмітки гіпертексту – HTML** (Hyper Text Markup Language). Документи, розмічені за допомогою цієї мови, називаються html-

документами. Html-документи мають розширення .html або .htm. Іноколи процес розробки веб-документів засобами мови HTML називають веб-програмуванням. Однак, HTML не є мовою програмування у звичайному розумінні, а є мовою розмітки.

#### **Переваги мови HTML:**

- документ, створений за допомогою деякої програми, наприклад текстового редактора, часто важко (а іноді і неможливо) використовувати в іншій програмі; HTML у цьому відношенні є універсальною;
- HTML – це відкритий стандарт;
- HTML не є власністю якої-небудь фірми;
- HTML надає можливість використання гіпертексту, підтримує мультимедіа.

Html-документи зберігаються у форматі ASCII. Це один з найпростіших способів зберігання текстової інформації, який не передбачає певного стильового форматування тексту. Однак, мова розмітки гіпертексту HTML передбачає стильове форматування у всій його потужності і повноті. HTML базується на наборі тегів, які призначені для визначення зовнішнього вигляду документа. Є теги, за допомогою яких можна підкреслити фрагмент тексту, зробити текст потовщеним, напівжирним, змінити його розмір тощо.

Для того щоб переглянути створений файл у форматі html (з розширенням «.html»), можна скористатися будь-яким оглядачем Інтернет, наприклад, Opera або Internet Explorer. Для цього потрібно запустити оглядач, вибрати з меню команду File => Open (*Файл => Відкрити*) та вказати потрібний файл. Розглянемо для прикладу документ, який виглядає так, як показано на рис.1.

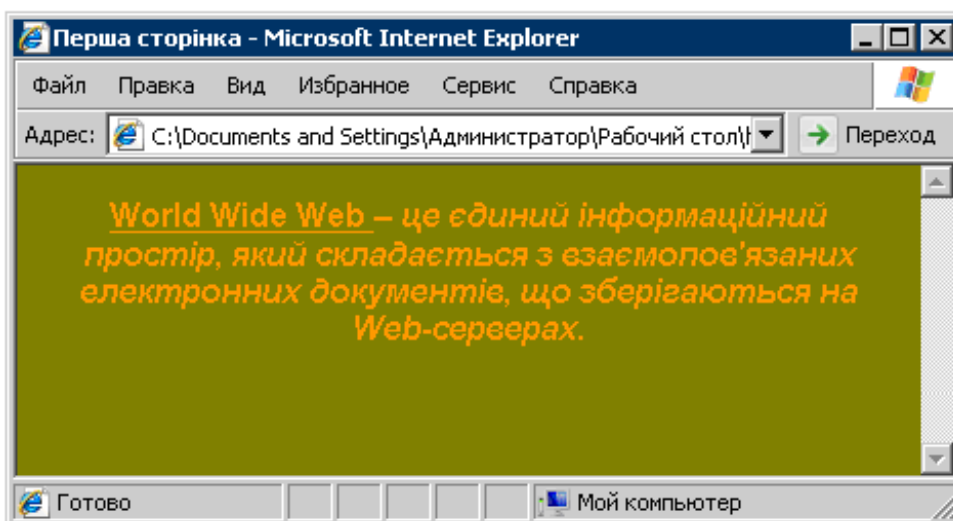


Рис. 1. Зовнішній вигляд простого документа HTML.

Розглянемо початковий HTML код сторінки. Для цього слід вибрати пункт меню *Вид =>Перегляд HTML кода*.

```
<HTML>
  <HEAD><TITLE>Перша сторінка </TITLE></HEAD>
  <BODY BGCOLOR='#808000'>
    <FONT FACE="Arial, Helvetica" COLOR='#FFA500'>
  <CENTER>
  <B><U> World Wide Web </U> </B> – <I> це єдиний інформаційний простір, який
складається з взаємопов'язаних електронних документів, що зберігаються на Web-
серверах.</I>
  </CENTER>
  </FONT>
</BODY> </HTML>
```

Будь-який html-документ починається з тега `<HTML>` і закривається ним же. Цей тег належить до групи тегів, який має відповідний йому закриваючий тег. Іноді вказівка закриваючого тега необов'язкова, але, в більшості випадків, це є необхідністю. Закриваючим тегом для тега `<HTML>` є тег `</HTML>`. Структура закриваючих тегів така, що вони практично повністю повторюють синтаксис відкриваючого тега, за винятком додавання символу `/` після першої кутової дужки. Після тега `<HTML>` йде, як правило, тег `<HEAD>`, який використовується для позначення заголовка документа. У заголовку розташовується ще один тег – `<TITLE>`, за допомогою якого задається назва документа. Будь-який текст, введений між тегами `<TITLE>` і `</TITLE>`, відобразиться в самому верхньому рядку вікна браузера Internet – в рядку назви документа.

Після заголовка документа йде тег `<BODY>`. Між тегами `<BODY>` і `</BODY>` розташовано зміст, або "тіло" документа. Тег `<BODY>` має певні параметри, основні з яких наведені нижче.

ALINK	Визначає колір активного посилання
BACKGROUND	Вказує на URL-адресу зображення, яке використовується в якості фонового
BOTTOMMARGIN	Встановлює межу нижнього поля документа в пікселях
BGCOLOR	Визначає колір фону документа
BGPROPERTIES	Якщо встановлено значення FIXED, фонове зображення не прокручується
LEFTMARGIN	Встановлює межу лівого поля документа в пікселях
LINK	Визначає колір ще не переглянутого посилання
RIGHTMARGIN	Встановлює межу правого поля документа в пікселях



SCROLL	Встановлює наявність (відсутність) смуг прокручування вікна браузера
TEXT	Визначає колір тексту
VLINK	Визначає колір вже переглянутого посилання

У розглядуваному документі для встановлення кольору фону документа використано параметр BGCOLOR. Значення цього параметра є стандартною комбінацією трьох кольорів – *Червоний-Зелений-Синій* (Red Green Blue). У цьому випадку значення параметра BGCOLOR задано за допомогою шістнадцяткового числа #808000, що відповідає оливковому кольору.

Далі в тіло документа поміщається певний текст. Для форматування тексту html-документів передбачена ціла група тегів, яку умовно можна поділити на теги **логічного** та **фізичного** форматування.

Теги **логічного** форматування визначають структурні типи своїх текстових фрагментів, причому ця розмітка не впливає на екранне представлення фрагмента браузером. Одним з цікавих тегів логічного форматування є тег <ACRONYM>, який зручно використовувати з параметром TITLE. Наприклад:

<ACRONYM TITLE="Дрогобицький державний педагогічний університет імені Івана Франка">

ДДПУ</ACRONYM> – один з провідних ВНЗ у нашому регіоні.

У цьому випадку візуальні браузери при наведенні курсору на текст, розмічений вказаним тегом, будуть видавати повне найменування навчального закладу у вигляді спливаючої підказки (рис. 2).

Деякі важливі теги логічного форматування:

- <DEL> – відмічає текст як віддалений, може використовуватися як елемент рівня тексту або рівня блоку;
- <EM> – використовується для виділення важливих фрагментів тексту (зазвичай такий текст відображається курсивом);
- <STRONG> – також використовується для виділення важливих фрагментів тексту (браузери відображають такий текст напівжирним шрифтом)

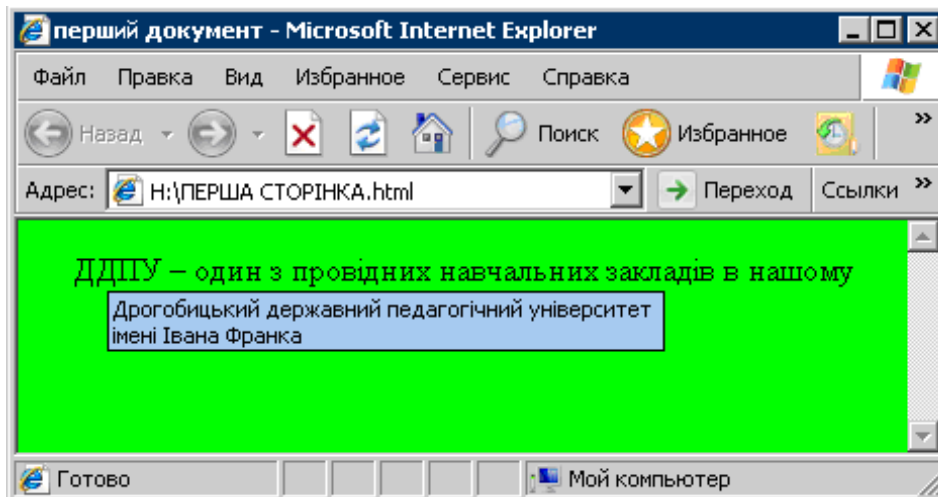


Рис. 2. Приклад використання тега <ACRONYM>.

Теги **фізичного** форматування визначають формат відображення вказаного в них фрагменту тексту у вікні браузера. Вигляд шрифту визначає тег <FONT>, для якого можуть задаватися параметри FACE, COLOR, SIZE. За допомогою параметра FACE встановлюється тип шрифту (в розглядуваному документі Arial, Helvetica). За допомогою параметра COLOR можна задати колір шрифту. Так, наприклад, в розглядуваному випадку значення цього параметра рівне #FFA500, що відповідає помаранчевому кольору (колір можна задавати і за допомогою стандартних імен: <FONT COLOR='Orange'>). Тег SIZE призначений для встановлення розміру шрифту в умовних одиницях від 1 до 7.

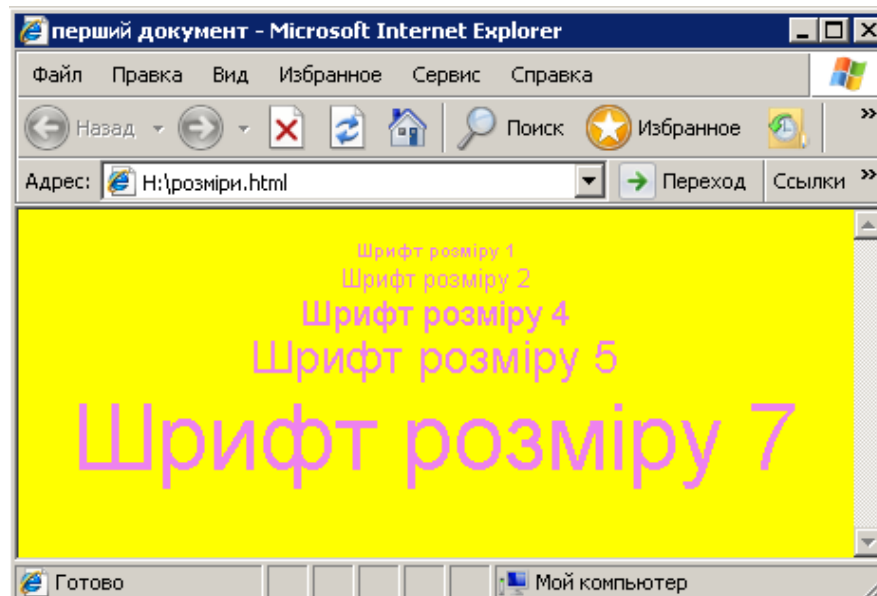


Рис. 3. Використання тегу <FONT SIZE =N>.

На рис. 3 зображений результат перегляду html-документа, в якому

використовуються ці теги. Крім того існує шість стандартних тегів, кожен з яких визначає розмір нашого тексту. Це : <H1> <H2> <H3> <H4> <H5> і <H6>.

Після тега <FONT> у початковому коді документа стоїть тег <CENTER>. За допомогою цього тега фрагмент тексту вирівнюється по центру щодо вікна браузера Internet. При зміні розмірів вікна текст автоматично вирівнюється так, щоб завжди бути посередині вікна. Для вирівнювання тексту по лівій межі вікна браузера використовується тег <LEFT>, по правій – тег <RIGHT>, по ширині – <JUSTIFY>.

Нижче приведено декілька поширених тегів, що відповідають за форматування тексту:

- <I> – відображає текст курсивом;
- <TT> – відображає текст моноширинним шрифтом;
- <SUB> – зсуває текст нижче рівня рядка;
- <SUP> – зсуває текст вище рівня рядка;
- <B> – жирний шрифт;
- <BIG> – великий шрифт;
- <SMALL> – маленький шрифт;
- <U> – підкреслює текст;
- <BLINK> – встановлює мигання тексту.

Для того, щоб розбити текст на абзаци, використовується тег <P>, який потрібно помістити перед початком кожного абзацу. Для примусового перенесення строки використовується тег <BR>.

### **Вставка зображень**

Для вставки зображень у html-документ (рис. 4) використовується така конструкція:

```
<IMG SRC="малюнок" BORDER="0" ALIGN="вирівнювання" WIDTH="ширина  
зображення" HEIGHT="висота зображення" HSPACE="відступ_1"  
VSPACE="відступ_2" ALT="підказка" NAME="ім'я" LOWSRC="малюнок_2">
```

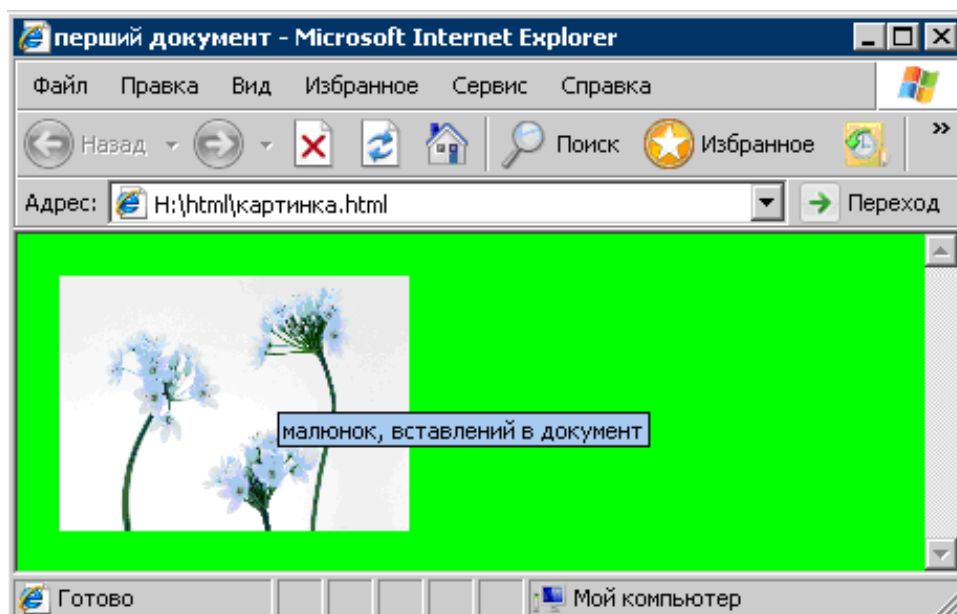


Рис. 4. Приклад вставки зображення в html-документ.

Нижче наведено опис кожного із вказаних параметрів:

<b>SRC</b>	Назва файлу, в якому є зображення (підтримуються *.jpg, *.gif, *.png). Якщо файл розташований в іншому каталозі, слід зазначити шлях до нього.
<b>ALIGN</b>	Вирівнювання зображень в документі: <b>left</b> – по лівому краю; <b>right</b> – по правому краю; <b>top</b> або <b>texttop</b> – вирівнюють верхню межу зображення з верхньою лінією поточної текстового рядка; <b>middle</b> – вирівнює базову лінію поточного текстового рядка з центром зображення; <b>absmiddle</b> – вирівнює центр поточного текстового рядка з центром зображення; <b>bottom</b> або <b>baseline</b> – вирівнює нижню межу зображення з базовою лінією поточного текстового рядка.
<b>BORDER</b>	Рамка навколо малюнку. Значення за замовчуванням – 0 (без рамки).
<b>WIDTH</b>	Ширина зображення в пікселях.
<b>HEIGHT</b>	Висота зображення в пікселях.
<b>HSPACE</b>	Горизонтальний відступ зображення в пікселях. Необов'язковий параметр.
<b>VSPACE</b>	Вертикальний відступ в пікселях. Необов'язковий параметр.
<b>ALT</b>	Вказує повідомлення, яке виводиться замість малюнка, якщо він не завантажився (не знайдений або користувач налаштував Оглядач так, що той не показує малюнки). Цей текст виводиться у вигляді підказки, якщо курсор миші навести на малюнок.
<b>NAME</b>	Визначає ім'я зображення. Необов'язковий параметр.

<b>LOWSRC</b>	Ім'я графічного файлу з розширенням з альтернативним зображенням нижчої якості (і меншого об'єму), ніж зазначене зображення. Оглядач, що підтримує цей параметр, спочатку завантажить малюнок з LOWSRC, а потім замінить малюнком SRC. Необов'язковий параметр.
---------------	---

Розміри зображення краще завжди задавати в параметрах HEIGHT і WIDTH – тим самим резервується місце у вікні веб-оглядача ще до завантаження цього зображення. Параметр ALT можна не вказувати, однак якщо малюнок з певних причин не завантажиться, користувач принаймні буде знати, що це за зображення.

## ЗАВДАННЯ

1. Створити у власній папці веб-сторінку із короткою інформацією про себе. Для виконання завдання проробіть такі дії:

- Створіть у власній папці текстовий документ з назвою «перша сторінка».
  - Заповнити створений документ інформацією про себе (прізвище, ім'я, по батькові), своє місто та навчальний заклад.
  - Використовуючи HTML-код документа, наведений в теоретичних відомостях, оформіть створений документ з інформацією про себе за правилами мови HTML.
  - Збережіть створений документ, після чого поміняйте розширення документа з .txt на .html.
2. Перегляньте створену веб-сторінку за допомогою будь-якого веб-оглядача.
3. Перегляньте html-код створеної сторінки за допомогою команди Вигляд ⇒ Перегляд HTML коду.
4. Доповнити створений документ малюнком.

Для виконання завдання проробіть такі дії:

- Перемістіть у власну папку потрібний графічний файл у форматі \*.jpg або \*.gif.
- У потрібному місці документа вставте тег <IMG> з такими атрибутами:  
<IMG SRC=" назва графічного файлу " BORDER="0" ALIGN="варіант вирівнювання" WIDTH="ширина зображення" HEIGHT="висота зображення" ALT="текст-підказка" NAME="ім'я зображення" >.
- Збережіть зміни у документі та перегляньте його за допомогою оглядача. При потребі поміняйте значення атрибутів тега <IMG>.

5. Відформатуйте створену веб-сторінку якнайкраще, використовуючи основні теги фізичного та логічного форматування мови гіпертекстової розмітки HTML, наведені у теоретичних відомостях.
6. Збережіть відредагований документ у своїй папці.
7. Продемонструйте створену сторінку викладачу та оформіть звіт з виконання цієї лабораторної роботи

## КОНТРОЛЬНІ ЗАПИТАННЯ

1. Що таке веб-сторінка?
2. Яке основне призначення дескрипторів (тегів)?
3. Яке розширення мають html-документи?
4. Як можна переглянути html-код створеної веб-сторінки?
5. В якому форматі зберігаються html-документи?
6. Як переглянути за допомогою оглядача html-файл?
7. З якого тегу починається будь-який html-документ?
8. За допомогою якого тега визначається назва документа? Де буде відображено його вміст?
9. Які параметри може мати тег <BODY>?
10. Як встановити колір фону створюваного документу?
11. Як задати потрібний колір тексту у всьому html-документу?
12. Які теги належать до тегів логічного форматування?
13. Які теги належать до тегів фізичного форматування?
14. Для чого використовується тег <ACRONYM>?
15. Для чого використовується тег <FONT>? Які параметри він може мати?
16. Як задати потрібний розмір шрифту для текстового фрагменту?
17. Яким тегом необхідно скористатися для примусового перенесення текстового рядка ?
18. Яке призначення мають теги <P> та <U>?
19. Як вставити в html-документ зображення?
20. Які атрибути може мати команда <IMG>?

## ТЕМА. Елементи створення html-сторінок. Створення посилань. Посилання на малюнок та фон сторінки. Карта. Створення таблиць

**МЕТА:** навчитися створювати у html-документі таблиці і гіперпосилання. Використовуючи набуті знання та вміння, створити тематичну веб-сторінку із посиланнями та таблицями та надати їй певний дизайнерський вигляд.

### ХІД РОБОТИ

1. Створити два простих html-документи із заданою інформацією.
2. Доповнити один з документів таблицею з певною інформацією.
3. Доповнити створені документи посиланнями.
4. Переглянути створені документи за допомогою веб-оглядача та надати їм певного оформлення, використовуючи теги форматування та малюнки.

### ТЕОРЕТИЧНІ ВІДОМОСТІ

#### Створення таблиць

Одним із способів впорядкування логічно зв'язаних інформаційних даних є їхня організація в таблиці. Таблиці є важливим засобом для створення веб-сторінок. До появи в мові HTML засобів для створення таблиць, не можна було навіть розмістити будь-який текст у колонках.

Для створення таблиці використовується така конструкція:

```
<TABLE >  
<CAPTION>Заголовок</CAPTION>  
<TR >  
<TD >  
...  
</TD>  
....  
</TR>  
</TABLE>
```

Кількість рядків або стовпців у таблиці може бути довільною. Тег <TABLE> починає опис таблиці і може мати такі параметри:

WIDTH="N", HEIGHT	Вказують розміри таблиці, якщо вони повинні бути чітко заданими комірками (можна задавати в пікселях, однак краще використовувати відсотки, оскільки в цьому випадку розмір таблиці буде змінюватися зі зміною вікна). За замовчуванням ширина таблиці визначається вмістом комірок.
BORDER="N"	Визначає товщину рамки таблиці. Якщо цей параметр не вказаний, його значення вважається нульовим (рамка не відображається).
BORDERCOLOR="White"	Встановлює колір окантовки.
BGCOLOR="White"	Визначає колір фонового зображення в таблиці.
BACKGROUND="image.gif"	Заповнює фон таблиці зображенням.
CELLSPACING="N"	Визначає відстань між рамками комірок в пікселях.
CELLPADDING="N"	Визначає відстань в пікселях між рамкою комірки та її змістом.
ALIGN=LEFT	Визначає розташування таблиці в документі. За замовчуванням таблиця розташована по лівому краю сторінки. Можливі значення атрибуту: LEFT (зліва); CENTER (по центру сторінки); RIGHT (справа).
FRAME="значення"	Керує зовнішніми межами таблиці, може приймати такі значення: VOID – межі відсутні (значення за замовчуванням); ABOVE – тільки межа згори; BELOW – тільки межа знизу; HSIDES – межі згори та знизу; VSIDES – межі зліва та справа; LHS – тільки ліва межа; RHS – тільки права межа; BOX, BORDER – всі чотири межі.
RULES="N"	Керує лініями, що розділяють комірками таблиці. Можливі значення: NONE – лінії відсутні (значення за замовчуванням); GROUPS – лінії лише між групами рядків; ROWS – тільки між рядками, COLS – тільки між колонками; ALL – між всіма рядками та колонками.
UNITS	Може приймати значення RELATIVE або PIXELS. Визначає одиниці вимірювання в інших параметрах. За замовчуванням, UNITS = PIXELS.

Для формування таблиці, що складається з декількох рядків, використовують команду <TR>, що розділяє рядки. Кожний новий рядок починається тегом <TR> і закінчується тегом </TR>. Між цими двома тегами можна визначити стовпці таблиці за допомогою тега <TD> і відповідного йому закриваючого тега </TD>. Дані, розміщені між тегами <TD> і </TD> і



розташовані в одному рядку (<TR> </TR>), визначають вміст окремого елемента таблиці.

Тег <TR>, що використовується для оформлення рядків таблиці, може мати такі параметри:

- ◆ ALIGN – використовується для завдання способу горизонтального форматування даних в середині комірок: вони можуть вирівнюватися по правій (RIGHT), по лівій (LEFT) межах або по центру (CENTER).
- ◆ VALIGN – використовується для завдання вертикального форматування даних в середині комірок: вони можуть вирівнюватися по верхній (TOP), нижній (BOTTOM) межах, по центру (MIDDLE) або мати загальну базову лінію (BASELINE).

Тег <TD >, який задає конкретні комірки, має такі параметри:

- ◆ WIDTH="N" – задає ширину комірки в пікселях;
- ◆ HEIGHT="N" – визначає висоту комірки в пікселях;
- ◆ COLSPAN="N" – визначає, скільки стовбців таблиці комірка буде перекривати (розтягає комірки по горизонталі; наприклад, <TD COLSPAN="2"> означає, що комірка буде розтягнута на дві колонки);
- ◆ ROWSPAN="N" – визначає, скільки рядків таблиці комірка буде перекривати;
- ◆ NOWRAP – якщо цей параметр вказаний, то зміст комірок не буде переноситися, для того, щоб влізти в ширину комірки;
- ◆ BGCOLOR – параметр вказує колір фону комірки у вигляді RGB-триплету або символічного імені;
- ◆ BACKGROUND="image.gif" – заповнює фон комірки зображенням.

Ще один тег для оформлення комірок таблиць – тег <TH>...</TH> – потрібен для завдання комірок-заголовків. Він в усьому співпадає з тегом <TD>, але на відміну від нього, зміст виділяється жирним шрифтом і розміщується по центру.

Якщо потрібно задати заголовок усієї таблиці, то використовується тег <CAPTION>. Він має бути в середині тегу <TABLE>, але поза описом комірок. У нього є лише один параметр ALIGN, що вказує положення заголовку: TOP – заголовок над таблицею по центру (значення за замовчуванням); LEFT– заголовок над таблицею зліва; RIGHT – заголовок над таблицею справа; BOTTOM – заголовок під таблицею по центру.

Якщо комірка порожня, то рамка навколо неї не малюється. За умови, якщо рамка таки потрібна, то в неї потрібно ввести символічний об'єкт

«&nbsp;» (non-breaking space – нерозриваючий пробіл). У цьому випадку комірка буде порожньою, однак рамка навколо неї буде (об’єкт &nbsp; обов’язково має закриватися крапкою з комою).

На рис. 1 зображений результат відкриття наступного html-документа, який містить таблицю:

```
<HTML>
<HEAD></HEAD>
<BODY BGCOLOR='Silver'>
<TABLE BORDER='3' ALIGN=CENTER BGCOLOR='Yellow'>
<CAPTION><H2><U> Студентський актив</U></H2></CAPTION>
<TH>ФІ-24</TH> <TH>ІН-25</TH><TH> ФМ-22 </TH>
<TR>
<TD>Петренко Я.</TD><TD>Найчук П.</TD> <TD>Антоник Л.</TD>
</TR>
<TR>
<TD>Собко М.</TD><TD>Мельник В.</TD><TD>Якимів А.</TD>
</TR>
</TABLE>
</BODY> </HTML>
```

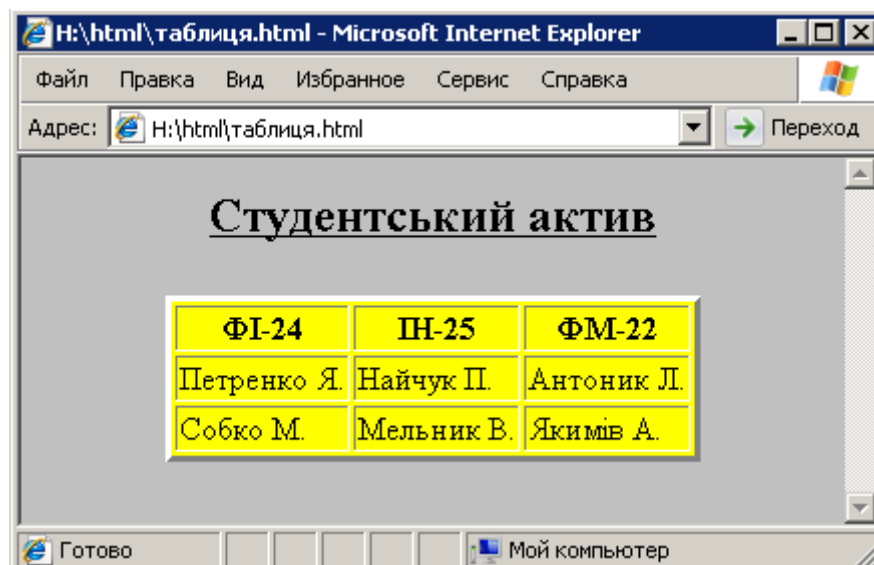


Рис.1. Приклад простої таблиці.

У цьому прикладі була створена таблиця з двома рядками і двома стовпцями. Тут було використано параметри: BORDER тега <TABLE>, який визначає товщину рамки таблиці, ALIGN=CENTER, що визначає спосіб розміщення таблиці на сторінці. Всередині однієї таблиці може бути розміщена інша таблиця, що дає можливість створювати складні структури.

### Створення посилань

Посилання на інші документи в HTML створюються за допомогою тега <A> або за допомогою навігаційних карт. Елемент <A> застосовують, якщо

в якості посилання буде використовуватися частина тексту або ціле зображення. Навігаційні карти доцільно застосовувати, якщо посиланням буде лише частина зображення.

Розглянемо в якості прикладу простий html-файл, який містить посилання на файл First.html:

```
<HTML><HEAD></HEAD>
<BODY>
<A HREF="First.html" TARGET="Вікно" TITLE="Підказка"> Назва посилання </A>
</BODY> </HTML>
```

Оформлення посилання починається із спеціального тега <A HREF='.....'>. У середині лапок необхідно вказати шлях до документа, перегляд якого передбачається при створенні посилання. Тег <A> має відповідний йому закриваючий тег </A>. Між цими двома тегами необхідно розмістити текст або зображення, яке буде фактично виконувати роль посилання. Якщо в якості гіперпосилання вибрати текст, він буде виділений підкресленням (рис. 2).

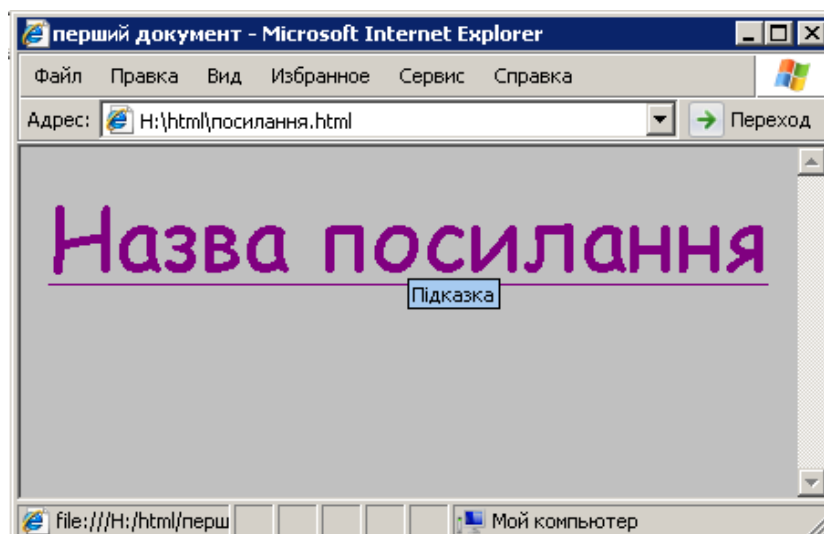


Рис. 2. Приклад найпростішого гіперпосилання.

### Параметри тегу <A>

1) **HREF** – визначає текст або зображення, що знаходиться між початковим та кінцевим тегамі як гіперпосилання на документ (або частину документа), вказаний у значенні цього атрибута.

Наприклад, лінк <A HREF="docs/title.html"> Документація </A> буде посилатись на файл title.html у підкаталозі docs. А для того, щоб після клацання на посиланні автоматично відкрилося вікно програми електронної пошти з пропозицією відправити лист за адресою abc@def.com, необхідно

додати в html-файл такий рядок:

```
<A HREF='mailto:abc@def.com'> відправити електронний лист до abc</A>
```

2) **NAME** – відмічає частину документа, що знаходиться між початковим та кінцевими тегамі, як можливий об'єкт для гіперпосилання. В якості значення потрібно латинськими літерами написати будь-яке слово-вказівник: 

```
<A NAME="part">Розділ</A>
```

. Після цього можна посилатися на відмічену частину простим вказуванням її назви після назви документа.

3) **TARGET** – визначає вікно, на яке вказує гіперпосилання. Він використовується лише разом з атрибутом HREF. В якості значення потрібно задати або назву одного з існуючих вікон або одне з наступних зарезервованих імен: self , parent, top, blank. Якщо в атрибуті TARGET вказати назву неіснуючого вікна, то створиться нове вікно з вказаною назвою.

Для того, щоб при наведенні курсору на посилання, воно змінювало свій колір в тег `<BODY>` потрібно додати такі параметри: text – колір тексту; link – колір посилання; vlink – колір переглянутого посилання; alink – колір активного посилання, при наведенні на нього курсору. Наприклад: 

```
<BODY text="black" link="blue" vlink="purple" alink="red">
```

.

Зазначимо, що тег `<A>` не може бути вкладеним в собі подібні, тобто неприпустимі є наступні конструкції:

```
<A HREF="link1.html">
```

Перший лінк

```
<A HREF="link2.html">Другий лінк</A>
```

Продовження першого лінку </A>

### **Створення гіперпосилань на зображення**

Гіперпосилання можна створити не лише на певну текстову інформацію, а й на будь-який графічний об'єкт. Для цього використовують тег `<IMG SRC>`, за допомогою якого можна вказати шлях до файлу зображення (формату .gif або .jpeg), яке потрібно розмістити на Web-сторінці.

Приклад створення такого гіперпосилання приведений нижче:

```
<HTML><HEAD></HEAD>
<BODY>
  <A HREF="URL" TARGET="Вікно">
    <IMG SRC="image.jpg " BORDER="0" WIDTH="100" HEIGHT="100"
TITLE="Підказка"> </A>
</BODY> </HTML>
```

У приведеному фрагменті image.jpg – ім'я файлу із зображенням, яке

буде використовуватися в якості посилання.

Нагадаємо, що зображення можна використовувати не лише в якості посилання, а й в якості фону створюваної веб-сторінки. Для цього слід скористатися параметром BACKGROUND тегу <BODY>:

```
<BODY BACKGROUND='image.jpg'>
```

Якщо розмір зображення менше розміру вікна браузера, зображення буде автоматично розмножено так, щоб повністю заповнити весь простір вікна. Вказаний графічний файл має бути розташований в одній папці з Вашим документом, інакше потрібно чітко зазначити шлях до нього.

### Зображення-карти

Карти – це спосіб зробити різні частини одного графічного зображення гіперпосиланнями. Вони дають змогу виділити окремі частини зображень та визначити для кожної з них свою дію. Щоб створити карту, потрібно вставити в тег <IMG SRC="..."> атрибут USEMAP="#name", де name – ім'я карти (значок # є обов'язковим). Наприклад:

```
<IMG SRC="image.jpg" BORDER="1" WIDTH="250" HEIGHT="250" ALT="Приклад зображення-карти" USEMAP="#image">
```

```
<MAP NAME="image">
```

```
<AREA SHAPE="rect" COORDS="11,11,70,24" TITLE="Посилання 1" HREF="URL">
```

```
<AREA SHAPE="CIRCLE" COORDS="70,72,83" TITLE="Посилання 2" HREF="URL">
```

```
<AREA SHAPE="rect" COORDS="190,136,128,149" TITLE="Посилання 3" HREF="URL">
```

Елемент <AREA> має такі атрибути:

- **SHAPE** – описує форму ділянки, що виділяється. Можливі значення: RECT – прямокутник, CIRCLE – круг, POLY – багатокутник, DEFAULT – весь малюнок стане посиланням.
- **COORDS** – координати, що визначають розміри та положення ділянки на малюнку. Всі координати відраховуються в пікселях від лівого верхнього кута зображення.
- **NOHREF** – визначає, що вказаній ділянці не відповідає ніяке посилання.
- **ALT** – альтернативний текст для виділеної ділянки, використовується невізуальними браузерами.
- **TITLE** – назва виділеної ділянки, яка з'являється при наведенні курсора на малюнок.

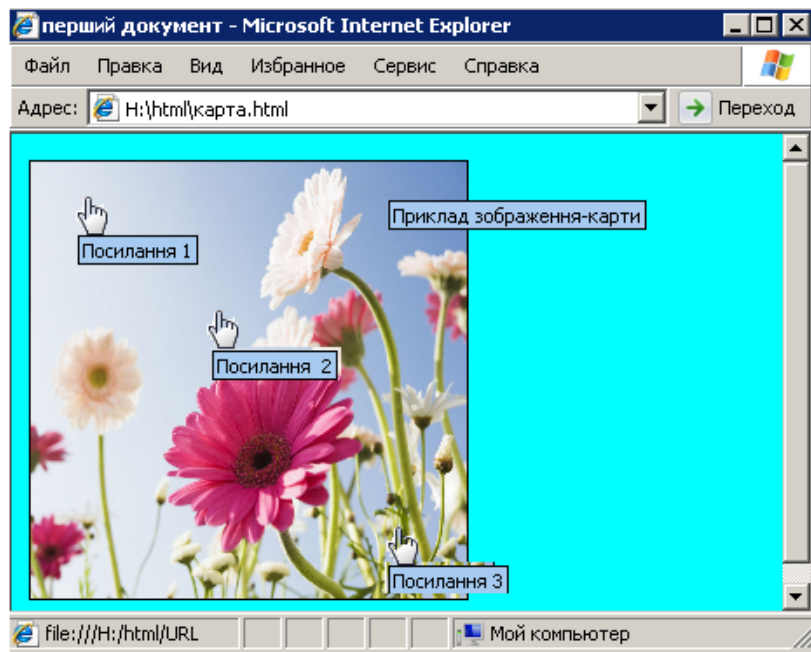


Рис. 3. Приклад зображення-карти з трьома посиланнями.

- **TARGET** – значення цього атрибута ("\_top", "\_blank", "\_self" або "\_parent") визначає, в якому вікні буде відкрито документ  
Для того, щоб точно розрахувати координати потрібної частини зображення існують спеціальні програми (MapEdit).

## ЗАВДАННЯ

1. Створити у своїй папці два простих html-документи з інформацією про вашу групу та ваших друзів.  
Для виконання завдання проробіть такі дії:
  - Створіть у власній папці два текстових документи з назвою «група» та «власне ім'я».
  - Заповніть створені документи відповідною інформацією та малюнками, використовуючи відповідні теги мови HTML.
  - Збережіть створені документи, після чого змініть розширення з .txt на .html.
2. Вставте в один зі створених документів таблицю і заповніть її потрібною інформацією.
3. Перегляньте створену веб-сторінку за допомогою будь-якого веб-оглядача. Перегляньте html-код створеної сторінки за допомогою команди Вигляд ⇒ Перегляд HTML-кода.
4. Доповнити створені документи посиланнями.

Для виконання завдання проробіть такі дії:

- Використовуючи тег `<A HREF='.....'>` створіть посилання з першого документа на другий. В якості гіперпосилання оберіть відповідний текстовий фрагмент.

Створіть посилання з другого документа на перший, використовуючи в якості посилання графічний об'єкт. Для цього у потрібному місці документа вставте наступний HTML-код:

```
<A HREF="URL" TARGET="Вікно">  
<IMG SRC="image.jpg " BORDER="0" WIDTH="100" HEIGHT="100"  
  TITLE="Підказка">  
</A>
```

- Створіть в одному з документів зображення-карту, що містить принаймні дві активні ділянки.

5. Збережіть зміни в документах та перегляньте їх за допомогою оглядача. Перевірте роботу створених посилань.
6. Відформатуйте створені веб-сторінки якнайкраще, максимально використовуючи теги та параметри мови гіпертекстової розмітки HTML, наведені в теоретичних відомостях.
7. Збережіть відредаговані документи у своїй папці.
8. Продемонструйте створені сторінки викладачу та оформіть звіт з виконання цієї лабораторної роботи.

## КОНТРОЛЬНІ ЗАПИТАННЯ

1. За допомогою яких команд створюються таблиці?
2. Які параметри має тег `<TABLE>`?
3. Яким чином задати потрібну товщину рамки таблиці?
4. Які значення може приймати атрибут `ALIGN`?
5. Як заповнити фон таблиці потрібним зображенням?
6. Для чого використовують атрибут `BGCOLOR` тегу `<TABLE>`? Які значення він може приймати?
7. Яке призначення тегу `<TD>`? Які параметри він може мати?
8. Які параметри може мати тег `<TR>`?
9. Для чого використовується тег `<CAPTION>`?
10. Як створити в таблиці порожню комірку?
11. За допомогою якого тега на мові HTML створюються гіперпосилання?

12. Які параметри може мати тег <A> ?
13. Які значення може набувати атрибут HREF тегу <A>?
14. Що визначає параметр TARGET тегу <A>?
15. Чи буде працювати гіперпосилання, якщо в атрибуті TARGET вказати назву неіснуючого вікна?
16. Які параметри потрібно вказати всередині тегу <BODY>, щоб при наведенні курсору на посилання, воно змінювало свій колір?
17. Що таке карти і коли їх доцільно використовувати?
18. Як створити гіперпосилання на зображення?
19. За допомогою якої команди можна встановити зображення в якості фону документа?
20. Як вказати назву ділянки на карті, яка б з'являлася при наведенні курсору на малюнок?

#### Лабораторна робота № 10.

### **ТЕМА. Елементи створення html-сторінок. Створення списків та фреймів в html-документах**

**МЕТА:** навчитися створювати у html-документі списки та фрейми. Використовуючи набуті знання та вміння, створити тематичні веб-сторінки, що містять списки і фрейми.

#### **ХІД РОБОТИ**

1. Створити html-документ, що містить фрейми.
2. Доповнити документ певною інформацією, використовуючи списки.
3. Переглянути створений документ за допомогою браузера та відредагувати його при потребі.
4. Надати створеній веб-сторінці відповідне оформлення.

#### **ТЕОРЕТИЧНІ ВІДОМОСТІ**

При створенні веб-сторінок часто виникає потреба розташувати певну інформацію у вигляді списків. Списки призначені для представлення інформації у впорядкованому вигляді. У html-документах використовується три види списків: неупорядковані списки, упорядковані списки, списки -



визначення.

**Впорядкований (пронумерований) список** в HTML складається з тегу-контейнеру списку, який визначає його тип, і стандартних тегів <LI>, що передують кожному пункту списку. Коли браузер зустрічає тег впорядкованого списку, він послідовно нумерує пункти списку. Впорядкований список відкривається тегом <OL>, а кожен його пункт починається стандартним тегом <LI>. Для створення заголовку списку використовується спеціальний тег <LH>. Список закривається тегом </OL>. Відкриваючий і закриваючий теги забезпечують переведення рядка до і після списку, відділяючи у такий спосіб список від решти тексту. Впорядковані списки можуть бути вкладеними один в один.

Тег <OL> має певні атрибути, що дають змогу встановлювати вид маркерів елементів списку та задавати початковий маркер списку. Наприклад:

- TYPE=A – встановлює маркер у вигляді прописних літер;
- TYPE=a – встановлює маркер у вигляді малих прописних літер;
- TYPE=I – маркер у вигляді великих римських цифр;
- TYPE=i – маркер у вигляді маленьких римських цифр;
- TYPE=1 – маркер у вигляді арабських цифр.

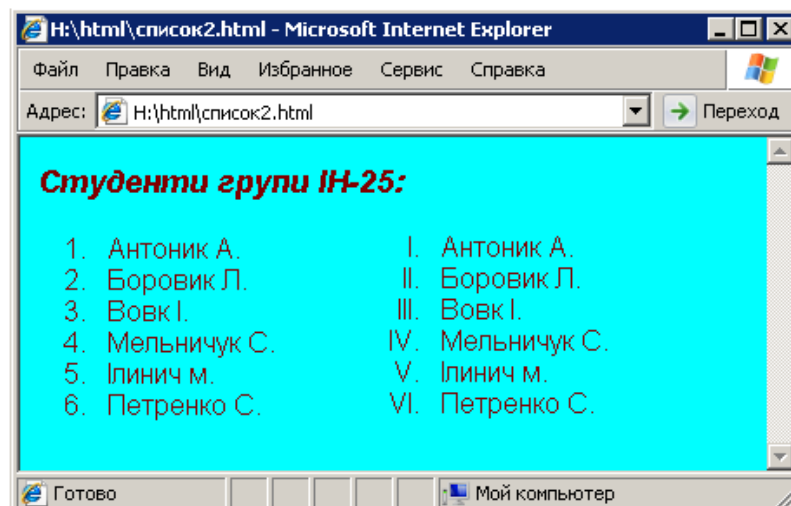


Рис. 1. Приклад впорядкованих списків з використанням атрибутів TYPE=1 та TYPE=I.

**Невпорядкований (маркований) список** – список, у якому відношення між пунктами невизначені. Маркований список замість числової нумерації передбачає використання різних символів, які називаються маркерами. Список розміщується всередині контейнера <UL>. Браузер створює автоматичний доступ для вкладених списків. Всі елементи списку,

помічені <LI> і розташовані усередині тегів <UL> і </UL>, виділяються в окрему групу.

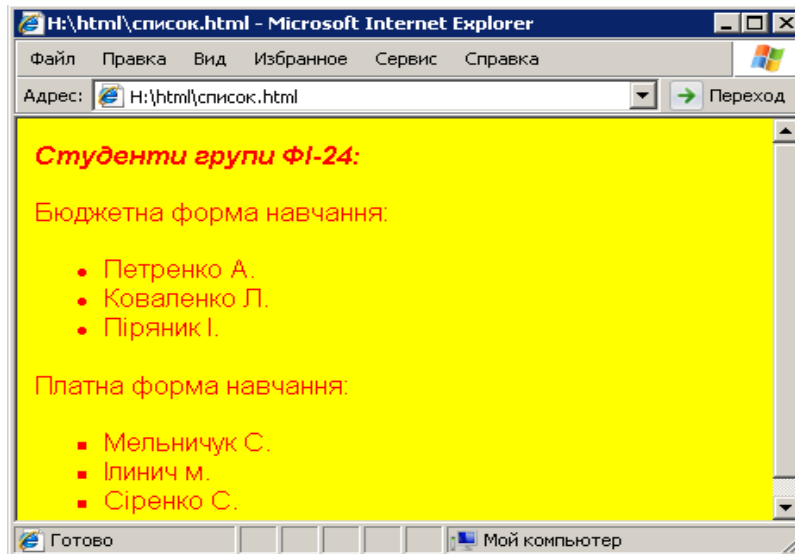


Рис. 2. Приклад невпорядкованого списку.

Вид маркерів у невпорядкованих списках можна встановити за допомогою атрибута TYPE, який визначає зовнішній вигляд символу і може мати такі значення: DISK – круга жирна точка, CIRCLE – коло, SQUARE – маленький чорний квадрат.

Тег <UL> має атрибут COMPACT, який дає змогу виводити список в більш компактному вигляді. На рисунку 2 представлено приклад невпорядкованого списку.

**Список визначень** – це особливий вид списків HTML. Вони подають текст у вигляді словникової статті, що складається з певного терміну та абзацу, який розкриває його значення. Елемент списку визначень DL є контейнером і забезпечує відокремлення списку від іншого тексту порожніми рядками. Всередині контейнеру тегом <DT> помічається термін, що визначається, а тегом <DD> – абзац з його визначенням:

```
<DL>  
<DT> Термін  
<DD> визначення терміну  
.....  
</DL>
```

Теги <DT> и <DD> не є контейнерами. Текст після тегу <DT> повинен міститися в одному рядку. Якщо ця вимога не виконана, або якщо рядок виходить за межі вікна браузера, то відбувається переведення рядка, але без відступу. Текст, що стоїть за тегом <DD> виводиться окремим абзацем з відступом вниз на один або два рядки.

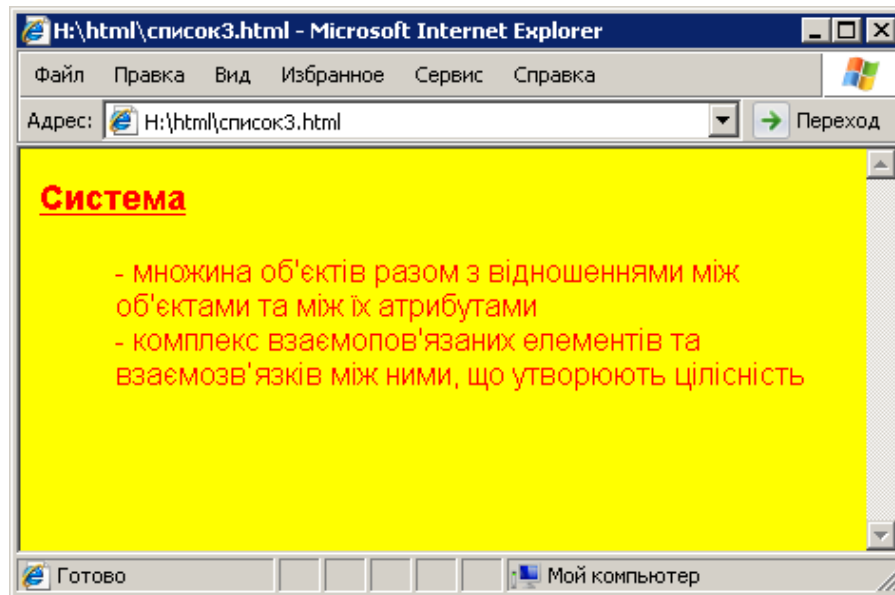


Рис. 3. Приклад списку визначень.

### Список типу <DIR>

Елемент типу DIR використовується для ідентифікації певної частини документу. Список контейнера, що починається тегом <DIR>, виводиться браузером аналогічно невпорядкованим спискам. Довжина кожного пункту цього типу списків обмежена 24 символами. Список потребує закриваючого тегу </DIR>.

### Робота з фреймами

Деколи з документами зручно працювати, коли вікно браузера розділено на декілька частин, в кожній з яких відображається окремий html-документ. Для того, щоб досягти подібного результату, слід скористатися одним з найпопулярніших засобів мови HTML – **фреймами**. Фрейми використовуються для того, щоб розділити вікно браузера на декілька ділянок і завантажити в кожную з них окрему веб-сторінку, що робить навігацію у html-документах більш зручною.

На екрані фрейми є прямокутниками. Для створення фреймової структури мова HTML має в своєму розпорядженні такі теги:

- <FRAMESET> – визначає спосіб розбиття на фрейми вікна браузера Internet, тобто склад і розміри кадрів на екрані;
- <FRAME> – визначає вміст кожного фрейму, тобто задає HTML-файл для кожного кадру;
- <NOFRAMES> – визначає, що показувати, якщо браузер не підтримує фрейми.

Всі зазначені елементи обов'язково мають бути розташовані поза тегами <BODY> ... </BODY>. Фрейми не мають ніякого відношення до тіла документу.

Всередині одного тега <FRAMESET> може розташовуватися ще один такий тег, що дає змогу створювати складну фреймову структуру.

### Формат тегу <FRAMESET>

Ця команда ділить ціле вікно або частину вікна на декілька вертикальних або горизонтальних кадрів. Кожний з цих кадрів може визначати html-файл, що відображається в ньому (за допомогою команди <FRAME>) або, відповідно, ділитися далі за тими ж правилами з вкладеним тегом <FRAMESET>.

Розглянемо коротко основні **атрибути** цієї команди:

**1. ROWS** – визначає кількість та розміри горизонтальних фреймів (фреймів-рядків) у вікні браузера. Значеннями цього атрибута є розміри фреймів, відокремлені комами. Є три способи визначення розмірів:

– у відсотках від висоти робочої області вікна браузера. Наприклад <FRAMESET ROWS="30%,30%,40%">;

– в пікселях. Наприклад: ROWS="75,\*";

– у вигляді знаку "\*" (зірочка), який вказує на те, що фрейм займає весь вільний простір вікна браузера, не зайнятий іншими фреймами з явно вказаними розмірами. Наприклад, зірочка в запису "25%,25%,\*" рівнозначна 50%; команда: <FRAMESET ROWS="100,\*"> створює 2 кадри: верхній становить 100 пікселів у висоту, нижній – простір, що залишився (рис. 4).

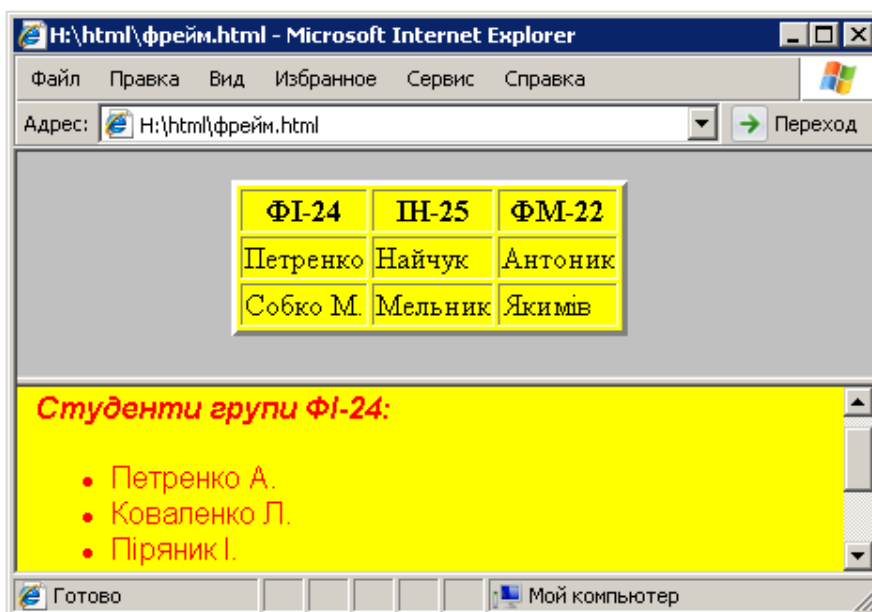


Рис. 4. Використання команди <FRAMESET ROWS='100,\*'>.

Зазначені способи можна поєднувати. Наприклад, команда `ROWS="25%,40,*"` розділить вікно документа на три горизонтальних фрейми, перший з яких матиме висоту в чверть вікна браузера, другий – в 40 пікселей, а третій займатиме всю ділянку, що залишилася.

**2. COLS** – визначає кількість і розміри вертикальних фреймів (фреймів-стовпців) у вікні браузера. Значеннями атрибуту є розміри фреймів, відокремлені комами. Розміри визначаються так само, як і для атрибуту `ROWS`.

Рядок `<FRAMESET COLS='60%, *'>` визначає розбиття вікна по вертикалі на 2 частини, перша з яких займає 60% від довжини вікна, а друга – весь простір, що залишився (рис. 5).

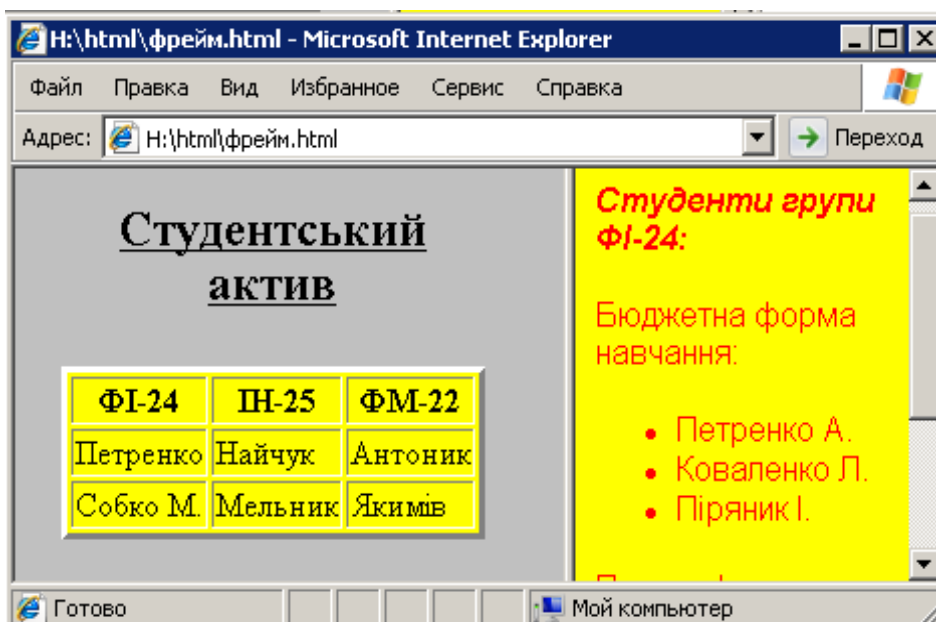


Рис. 5. Використання команди `<FRAMESET COLS='60%, *'>`.

**3. FRAMESPACING** – визначає відстань (так звану "сіру ділянку") між фреймами в пікселях. Зазначений атрибут необхідний для створення фреймів без рамок.

### Формат команди `FRAME`

Розглянемо спосіб визначення html-документа, який повинен відображатися в окремому фреймі. З цією метою необхідно використовувати тег `<FRAME>`. Дана команда існує тільки всередині блоку `<FRAMESET>` – `</FRAMESET>`. Її призначення – визначення функцій конкретного кадру. Файл з потрібним документом вказується за допомогою атрибута `SRC` тега `<FRAME>`. Для кожного фрейма повинен бути визначений один і лише один html-документ.

Основні **атрибути** команди FRAME:

1. **SRC** – обов'язковий атрибут, вказує адресу (URL) html-файлу, який відображається у цьому фреймі.

2. **NAME** – визначає ім'я цього фрейму, яке надалі використовується для посилання на нього з інших документів за допомогою атрибуту TARGET тегу <A>. В якості значення потрібно вказати будь-яке ім'я без пробілів, використовуючи латинські символи та цифри. Ім'я не повинно починатися із цифр і спеціальних символів.

3. **MARGINWIDTH** – визначає ширину (в пікселях) лівого і правого полів фрейму. Якщо атрибут не вказаний, браузер самостійно встановить оптимальний розмір відступу.

4. **MARGINHEIGHT** – визначає ширину (в пікселях) верхнього та нижнього полів фрейму. Якщо атрибут не вказаний, то, як і в попередньому випадку, браузер самостійно встановить оптимальний розмір відступу.

5. **SCROLLING** – визначає наявність смуг прокрутки вмісту фрейму. Можливі значення: Yes – відобразити смуги прокрутки, No – не відобразити смуги прокрутки, Auto – відобразити смуги прокрутки за потреби (якщо документ, вказаний в атрибуті SRC, не поміщається у фреймі).

Якщо параметр SCROLLING не заданий, то лінійки прокрутки створюються автоматично тільки тоді, коли розмір тексту, що відображається перевищує розмір кадру (це стосується як розміру по вертикалі, так і по горизонталі).

6. **NORESIZE** – не дає змоги змінювати розміри фрейму. Цей атрибут не потребує вказування значень.

7. **FRAMEBORDER** – визначає наявність рамок у фрейму. Можливі значення: Yes – відобразити рамки; No або 0 – не відобразити рамки. Деякі браузери не підтримують цей атрибут, тому для визначення ширини рамок використовують атрибут BORDER.

Для прикладу розглянемо наступний html-документ, що містить фрейми (рис. 6):

```
<HTML>
<HEAD></HEAD>
<FRAMESET FRAMEBORDER="1" FRAMESPACING="0" BORDER="1" COLS="165,*">
<FRAME SRC="Fr1.html" NAME="page">
<FRAMESET ROWS="55%,*">
  <FRAME SRC="Fr2.html" NAME="menu1" MARGINWIDTH="0">
  <FRAME SRC="Fr3.html" NAME="menu2" MARGINWIDTH="0">
```

```
</FRAMESET>  
<NOFRAMES> Ваш браузер не підтримує фрейми </NOFRAMES>  
</FRAMESET> </HTML>
```

У наступному прикладі вікно браузера розбивається на три частини. Ліву частину займає фрейм шириною 165 пікселей, в якому відображається файл Fr1.html. Права частина розбита на два горизонтальні фрейми, в першому з яких відображається документ Fr2.html, а другому, що займає всю частину документа, яка залишилася, – документ Fr3.html відповідно.

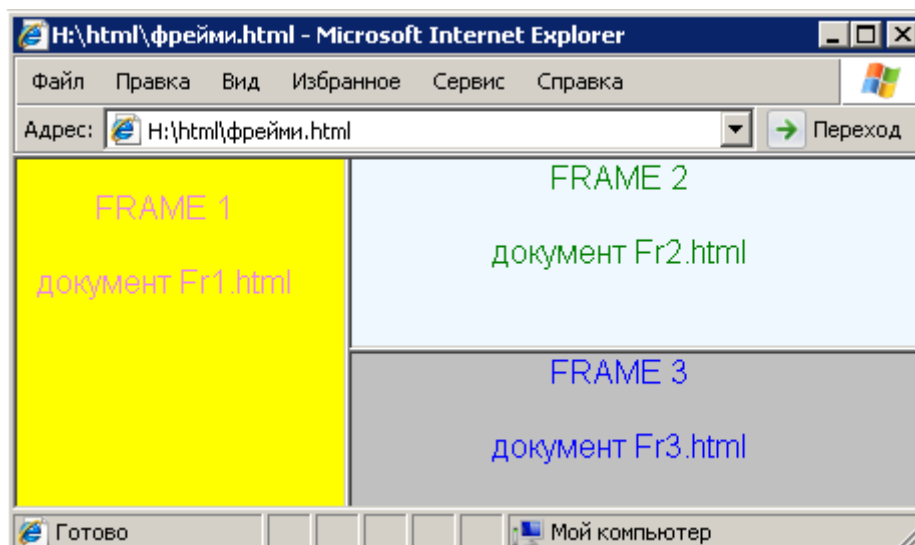


Рис. 6. Приклад розбиття html-документа на фрейми.

### Формат команди NOFRAME

У браузері, що обробляє кадри, вся інформація, що міститься між командами <NOFRAME> і </NOFRAME>, ігнорується. У браузері, що не обробляє кадри, ця інформація обробляється. Зазвичай, розробники сторінок, що використовують кадри, вставляють у тіло команди NOFRAME повідомлення про те, що цей браузер не обробляє кадри.

### Спеціальні значення параметра TARGET

При створенні гіперпосилань часто виникає потреба в тому, щоб новий документ відображався у тому самому вікні, в якому знаходиться посилання на нього. Для цього слід скористатися параметром TARGET тегу <A>, який, власне, і визначає вікно, на яке вказує гіперпосилання, тобто задає кадр, що виділяється за замовчанням для відображення гіпертексту. Якщо в якості значення задати ім'я одного з фреймів, попередньо розбивши вікно на фрейми, то потрібні документи, шлях до яких задають гіперпосилання, будуть відображатися у зазначеному кадрі, а вигляд решти вікна браузера змінюватися не буде. Наприклад, html-документ:

```

<HTML><HEAD></HEAD>
<FRAMESET FRAMEBORDER="1" BORDER="0" COLS="25%,*">
<FRAME SRC="fr1.html" NAME='menu1' >
<FRAME SRC="fr2.html" NAME='menu2'>
</FRAMESET> </HTML>

```

розбитий по вертикалі на два фрейми, один з яких містить документ fr1.html, а інший fr2.html. в першому документі створено чотири посилання на певні файли, вміст яких відображається у другому фреймі. Для цього в теги <A> документа fr1.html було введено параметр TARGET, в якості значення якого вказано назву другого фрейму:

```

<HTML>
<HEAD></HEAD>
<BODY BGCOLOR='Yellow'>
<CENTER>
</H2> Студенти ІФМІ </H2>
</CENTER>
<A HREF='s1.html' TARGET=' menu2' TITLE="інформація про студентів першого курсу"> <H3> 1 курс </H3> <P>
<A HREF='s2.html' TARGET='menu2' TITLE="інформація про студентів другого курсу">
<H3> 2 курс </H3> <P>
<A HREF='s3.html' TARGET=' menu2' TITLE="інформація про студентів третього курсу"> <H3> 3 курс </H3> <P>
<A HREF='s4.html' TARGET=' menu2' TITLE="інформація про студентів четвертого курсу"> <H3> 4 курс </H3> <P>
</BODY> </HTML>

```

Після клацання на будь-якому посиланні, розташованому у файлі fr1.html, відповідний цьому посиланню html-документ буде завантажений у другому фреймі (рис. 7).

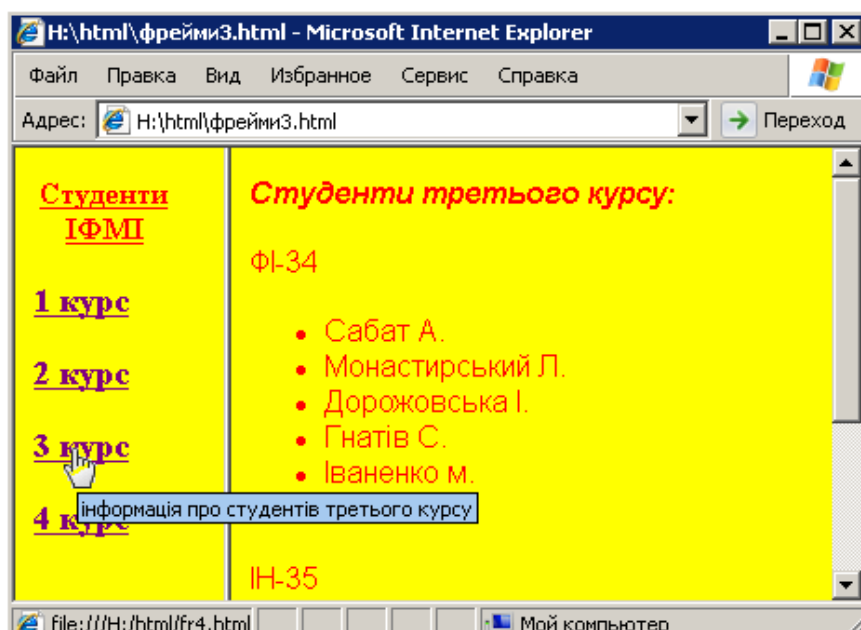
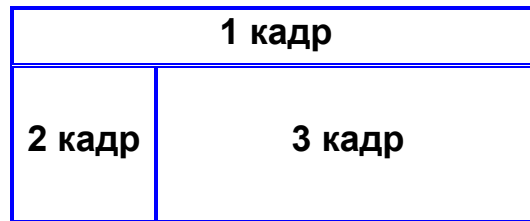


Рис. 7. Приклад html-документа з фреймами та посиланнями з використанням параметру TARGET.



## ЗАВДАННЯ

1. Створити у власній папці html-документ 1.html, розбитий на три фрейми: два горизонтальних, нижній з яких розбитий на два вертикальних кадри у такий спосіб:



2. Створити новий документ 2.html, що містить посилання на два створених Вами раніше тематичних документи. Використовуючи параметр TARGET тегу <A>, задати відображення цього документу в ділянці другого фрейму.

3. Задати відображення документів, на які створені гіперпосилання з файлу 2.htm, в ділянці третього кадру.

Для виконання завдання скористайтесь HTML-кодом документу fr1.html, наведеному у теоретичних відомостях (відображеного на рис. 7 в лівому фреймі).

4. Створити порожній документ 3.html, в якому в якості фону задати малюнок. Задати відображення цього документу в кадрі першого документу 1.html.

5. Доповнити будь-який з документів, що виводяться в ділянці третього фрейму маркованим і нумерованим списками.

6. Переглянути створену веб-сторінку за допомогою будь-якого веб-оглядача. Перевірте роботу створених Вами посилань. Перегляньте html-код створеної сторінки за допомогою команди Вигляд ⇒ Перегляд HTML-коду.

7. Відформатуйте створені веб-сторінки якнайкраще, максимально використовуючи теги та параметри мови гіпертекстової розмітки HTML.

8. Збережіть відредаговані документи у власній папці.

9. Продемонструйте створені сторінки викладачу та оформіть звіт з виконання цієї лабораторної роботи.

## КОНТРОЛЬНІ ЗАПИТАННЯ

1. Які види списків використовуються в html-документах?
2. Як на мові HTML створити впорядкований список?
3. Які значення може приймати атрибут TYPE тегу <OL>?
4. Що таке невпорядкований (маркований) список?
5. За допомогою якого тегу створюються марковані списки?
6. Як можна встановити вид маркерів в невпорядкованих списках?
7. Що таке список-визначення? Як створити список-визначення на мові HTML?
8. Для чого використовується тег <DIR>?
9. Яке призначення фреймів?
10. За допомогою яких тегів на мові HTML створюється фреймова структура?
11. Яке призначення тегу <FRAMESET>?
12. За допомогою якого тега визначається вміст кожного фрейму?
13. Яке призначення атрибуту ROWS тегу <FRAMESET>?
14. Які є способи визначення розмірів створюваних фреймів?
15. Яка команда визначає кількість і розміри вертикальних у вікні браузера?
16. За допомогою якої команди визначається відстань між фреймами в пікселях?
17. Які атрибути має тег <FRAME>?
18. Яке призначення атрибуту SRC тегу <FRAME>? Які значення він приймає?
19. За допомогою якої команди визначається наявність рамок у фреймів?
20. Яке призначення атрибуту SCROLLING тегу <FRAME>? Які значення він приймає?
21. З якою метою використовується команда NOFRAME?
22. Які значення може приймати атрибут TARGET?
23. Як створити фрейм з фіксованими розмірами, які не можна змінити?
24. Яке призначення атрибуту NAME тегу <FRAME>?
25. За допомогою якої команди можна розбити вікно документа на три вертикальних фрейми?

## ТЕМА. Елементи створення html-сторінок. Використання форм при створенні html-документів

**МЕТА:** ознайомитися з правилами створення форм на мові HTML та навчитися створювати форми в html-документах. Використовуючи набуті знання та вміння, створити тематичні веб-сторінки, що містять форми.

### ХІД РОБОТИ

1. Створити html-документ, що містить форми.
2. Доповнити цей документ певною інформацією.
3. Переглянути створений документ за допомогою браузера та відредагувати його при потребі.
4. Розбити створений документ на два документи. Створити посилання з першого документу на другий.
5. Надати створеним документам відповідне оформлення.

### ТЕОРЕТИЧНІ ВІДОМОСТІ

**Форма** – це засіб, що дає змогу організувати на сторінці діалог з її користувачем. Форми використовуються для створення інтерфейсів, що містять кнопки, списки, текстові рядки та текстові поля. Форми передають інформацію програмі обробки даних у вигляді пар: «ім'я поля» – «значення поля».

Для створення форм використовуються такі елементи:

- FORM – створює форму
- INPUT – створює поле в формі
- TEXTAREA – створює поле для введення декількох рядків тексту
- OPTION – створює в меню окремі пункти
- SELECT – створює меню в заповнюваній формі

Розглянемо їх детальніше.

#### 1. FORM

Будь-яка форма починається тегом <FORM> і закінчується тегом </FORM>. Всередині елемента FORM можна використовувати більшість елементів мови HTML. Розглянемо коротко основні **атрибути** тега:

► NAME – визначає ім'я форми, що є унікальним для документа. Цей атрибут потрібний, якщо в документі є декілька форм.

► ACTION – обов'язковий атрибут, який визначає URL-адресу, за якою буде відіслано вміст форми (тобто шлях до скрипта серверу, що обслуговує форму).

Усі дані, введені за допомогою цих елементів (введені безпосередньо в полях введення або обрані за допомогою прапорців або списків), відсилаються на обробку додаткові, зазначеному за допомогою цього атрибута. Після обробки даних додаток відсилає користувачеві результат обробки.

► METHOD – задає спосіб відправки вмісту форми (не обов'язковий). Можливі значення GET (за замовчуванням) та POST. Метод GET не дає змоги передати великий об'єм даних. Якщо передбачається, що користувач буде заповнювати дуже велику форму, вводити великі текстові дані або пересилати файл – потрібно використовувати METHOD="POST".

► ENCTYPE – визначає спосіб кодування вмісту форми при відправленні. За замовчуванням використовується "application/x-www-form-urlencoded".

► TARGET – задає ім'я вікна, в яке повертається результат обробки відправленої форми. Можливі значення: \_self, \_parent, \_top, \_blank або вказане ім'я вікна.

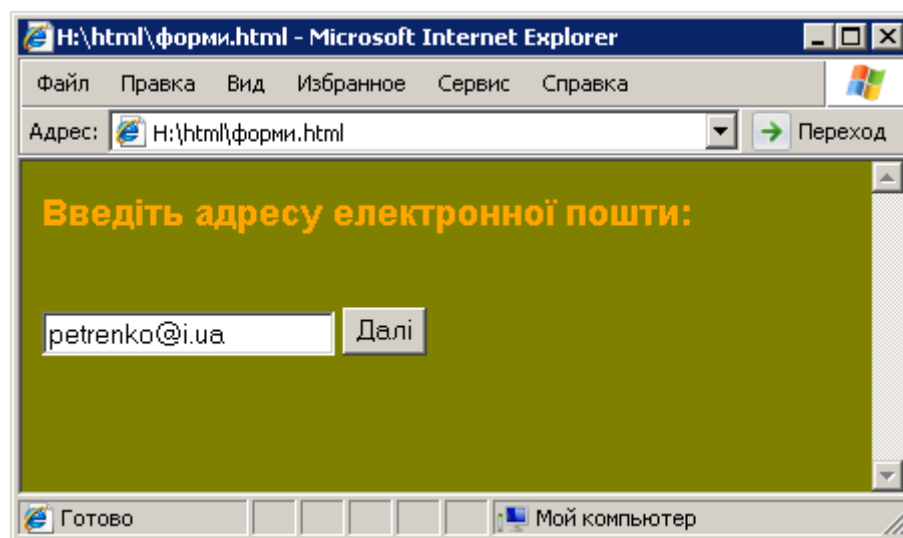


Рис. 1. Приклад найпростішої форми.

## 2. INPUT

Цей елемент створює поле форми (кнопку, поле введення і т. і.), вміст якого може бути змінено або активізовано користувачем. Елемент не має закриваючого тега і має розташовуватися всередині тегу FORM.

Основні атрибути:

► NAME – визначає ім'я, що використовується при передаванні вмісту

форми на сервер. Після відправлення всі дані форми передаються на обробку визначеному додаткові. Зазвичай використовується для ідентифікації поля або групи логічно пов'язаних полів.

► **TYPE** задає тип поля для введення даних. Значення за замовчуванням – "text". Можливі значення:

1) **text** – визначає вікно для введення текстового рядку. Може містити додаткові атрибути **SIZE** (визначає ширину вікна у символах) та **MAXLENGTH** (визначає максимально допустиму довжину рядка в символах, яке можна ввести в текстове поле). Значення атрибуту **MAXLENGTH** може бути більше, ніж кількість символів, вказаних в атрибуті **SIZE**. За замовчуванням кількість символів не обмежена. Наприклад:  
<INPUT TYPE=text SIZE=20 NAME=User VALUE="LENIN INC">

2) **password** створює поле вводу в один рядок, причому текст, який вводить користувач, відображається у вигляді символів "\*".

3) **radio** – визначає радіокнопку. Може містити додатковий атрибут **CHECKED** (показує, що кнопка помічена). У групі радіокнопок з однаковими іменами може бути лише одна помічена радіокнопка. Всі перемикачі, що мають однакове значення атрибута **NAME**, належать до однієї групи перемикачів. При надсиланні даних форми для обробки додатку передається лише значення встановленого перемикача.

Наприклад (рис. 2):

```
<INPUT TYPE="radio" NAME="sex" VALUE="Yes" CHECKED> Так<BR>
```

```
<INPUT TYPE="radio" NAME="sex" VALUE="No">Ні <BR>
```

```
<INPUT TYPE="radio" NAME="sex" VALUE="Possible">Не знаю <BR>
```

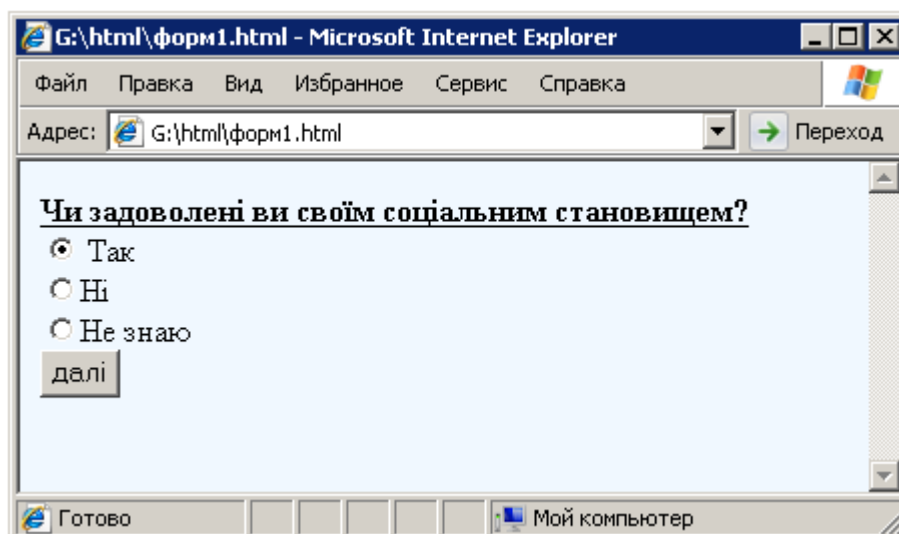


Рис. 2. Приклад використання команди TYPE="radio".

Створена група з трьох радіо кнопок, підписаних, Yes, No та Possible. Відміченою є перша з кнопок. Якщо користувач не обере іншу кнопку, буде передана змінна Question зі значенням Yes. Якщо користувач обере інший варіант, то, відповідно, буде передана змінна Question зі значенням No або Possible.

4) **checkbox** – створює поле вводу для атрибутів типу boolean ("так"/"ні") або для атрибутів, які можуть приймати декілька значень одночасно. Ці атрибути являють собою декілька полів checkbox, що можуть мати однакові імена. Кожне поле checkbox створює окрему пару name/value в інформації, яка відсилається на сервер, навіть якщо результатом є однакові імена.

Наприклад (рис. 3):

```
<FORM NAME="Form2" ACTION="http://www.igf.ru/cgi-bin/magazines.pl">  
<INPUT TYPE="checkbox" NAME="m1">Високий замок<br>  
<INPUT TYPE="checkbox" NAME="m2">Українська правда<BR>  
<INPUT TYPE="checkbox" NAME="m3" CHECKED>Кореспондент<BR>  
<INPUT TYPE="checkbox" NAME="m3" CHECKED>Експрес<BR>  
<INPUT TYPE="checkbox" NAME="m3" CHECKED>Влада грошей<BR>  
<U><B>Оформити передплату на обрані видання</B></U>:  
<INPUT TYPE="image" src=" image1.jpg" WIDTH="60" HEIGHT="40">  
</FORM>
```

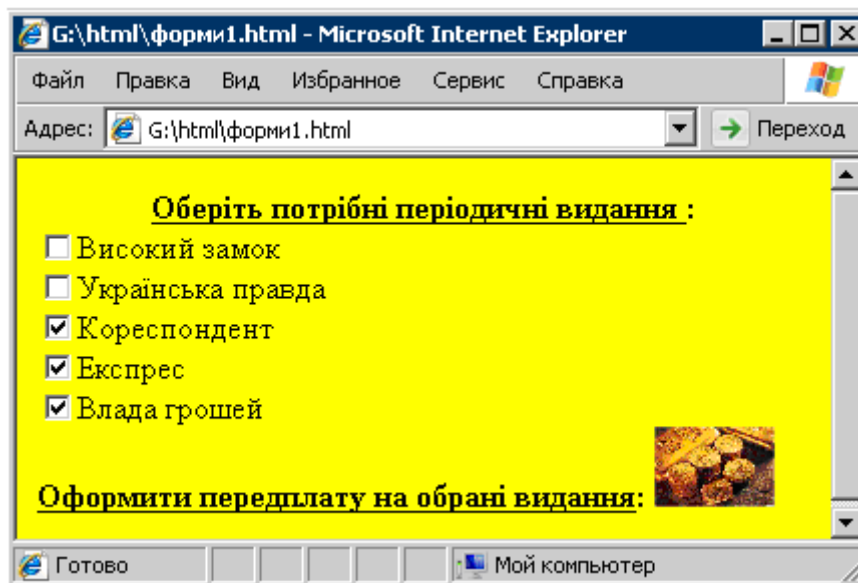


Рис. 3. Використання атрибуту TYPE="checkbox" команди <INPUT>.

5) **hidden** – визначає прихований елемент даних, який не бачить користувач під час заповнення форми, що передається на обробку без змін. Цей елемент корисно мати у формі, яка час від часу піддається обробці, для того, що розробник знав, з якою версією форми він має справу. Цей тип полів зручно використовувати для передачі даних від скрипта до скрипта

непомітно для користувача.

Наприклад: `<INPUT TYPE=hidden NAME=version VALUE="1.1">`

6) **image** – створює графічну кнопку-картинку для передачі даних на сервер. Ефект від клацання на такому зображенні повністю аналогічний ефекту від клацання на кнопці Submit, тобто після цього дані форми передаються на обробку додатку.

Розташування зображення на сторінці можна задати за допомогою атрибута SRC. При передачі даних серверу повідомляються координати X та Y точки зображення, на якій був здійснений клік клавішею миші. Координати відраховуються від верхнього лівого кута зображення. При цьому інформація про поле типу image записується у вигляді двох пар значень name/value. Значення name одержується шляхом додавання до назви відповідного поля суфіксів ".x" (абсциси), та ".y" (ординати).

Разом з цим атрибутом TYPE="image" може використовуватися параметр ALIGN – визначає спосіб вертикального вирівнювання для зображень. За замовчуванням має значення bottom. Наприклад (рис. 3):

`<INPUT type="image" src=" image1.jpg" WIDTH="60" HEIGHT="40">`

7) **submit** – визначає кнопку, при натисканні на яку запускається процес обробки даних з форми. Атрибут NAME дає змогу задати ім'я даній кнопці, яке може бути використано для певної функції в скрипті. Атрибут VALUE потрібний для задання тексту, який буде відображатися на кнопці в документі. Наприклад: `<INPUT TYPE=submit VALUE="Надіслати">`

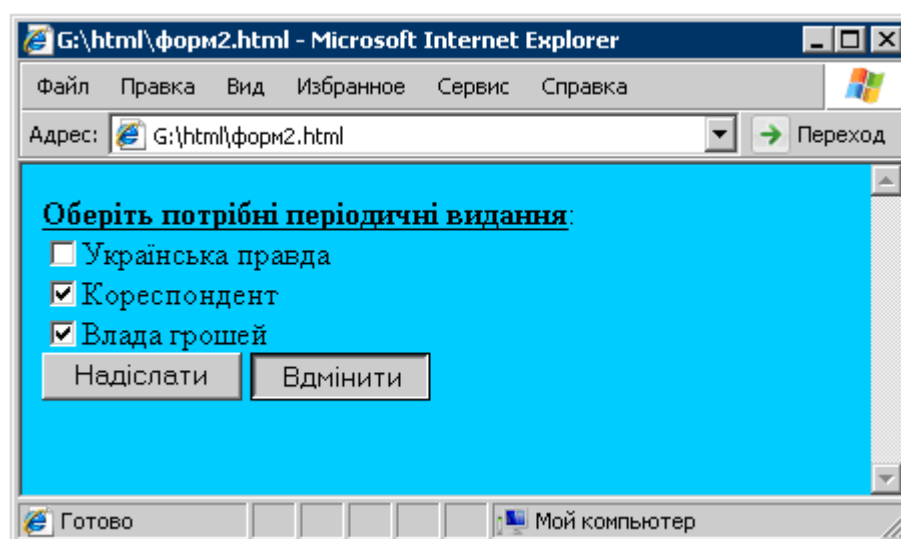


Рис. 4. Використання атрибутів TYPE=submit та TYPE=reset команди `<INPUT>`.

8) **reset** – визначає кнопку, при натисканні на яку поля форми очищуються і

встановлюються їх значення за замовчуванням. Оскільки при її використанні дані обробнику не передаються, кнопка цього типу може не мати атрибута NAME: Наприклад: `<INPUT TYPE=reset VALUE="Вдмінити">`

9) **file** – дає змогу користувачу приєднати до форми файл. Тип файлу конкретизується за допомогою параметра ACCEPT.

### 3. TEXTAREA

Цей атрибут створює поле вводу для тексту у декілька рядків. Наприклад:

```
<FORM ACTION="receive.html" METHOD=POST>
Ведіть, будь-ласка, коротку інформацію про себе. <BR>
<TEXTAREA NAME="inform" WRAP="virtual" COLS="40" ROWS="3">Інформація
про Вас ...</TEXTAREA><br>
<INPUT TYPE="submit" VALUE="OK">
</FORM>
```

Атрибут ROWS визначає кількість рядків, що відображаються, тобто фактично висоту прямокутної ділянки. Атрибут COLS, своєю чергою, визначає кількість стовпців, або, іншими словами, число символів, що відображаються в одному рядку тексту (довжину області). Для того щоб проглянути текст, що виходить за рамки створеного поля, можна скористатися смугами прокрутки. Будь-який текст, що перебуває між тегами `<TEXTAREA>` і `</TEXTAREA>`, буде відображений в цьому текстовому полі як текст за замовчуванням.

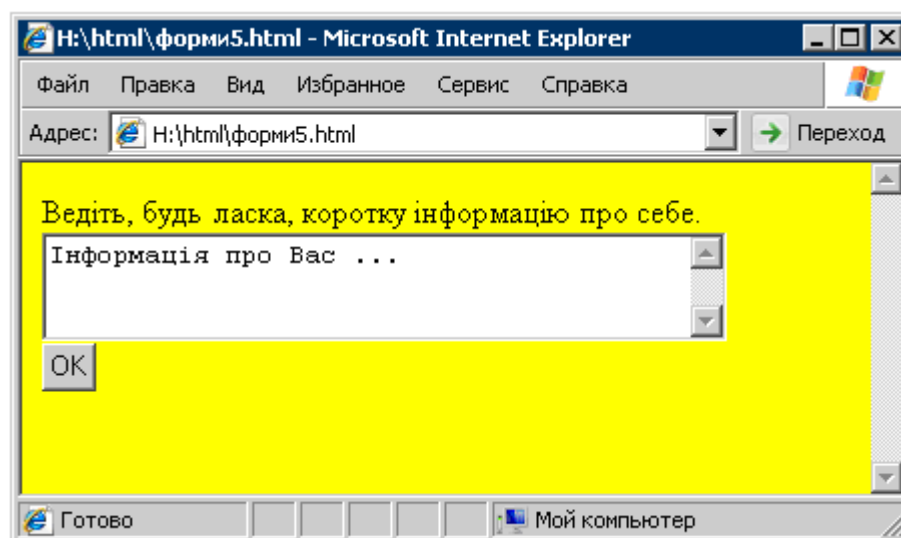


Рис. 5. Використання тегу `<TEXTAREA>`.

### 4. SELECT

Елемент SELECT створює у заповнюваній формі меню типу "Вибір одного пункту з багатьох" або "Вибір декількох пунктів з багатьох".



Зазначений елемент має розташовуватися всередині тегу <FORM>.

Атрибути:

- ▶ MULTIPLE – дає можливість вибору декількох пунктів меню при утримуванні клавіші Ctrl. За замовчуванням можна вибрати лише один пункт меню.
- ▶ NAME – визначає ім'я меню, унікальне для цієї форми, яке буде використовуватися при передачі даних на сервер.
- ▶ SIZE – визначає кількість пунктів у меню. Якщо значення цього атрибута більше одиниці, то результатом буде список пунктів. Наприклад (рис. 6):

```
<FORM ACTION="receive.cgi">  
<SELECT NAME="OS" MULTIPLE>  
  <OPTION VALUE="DOS">MS-DOS  
  <OPTION VALUE="WinXP">MS Windows98  
  <OPTION VALUE="Unix" SELECTED>UNIX  
  <OPTION VALUE="WinNT">MS Windows NT  
</SELECT>  
<INPUT TYPE="submit" VALUE="Надіслати">  
</FORM>
```

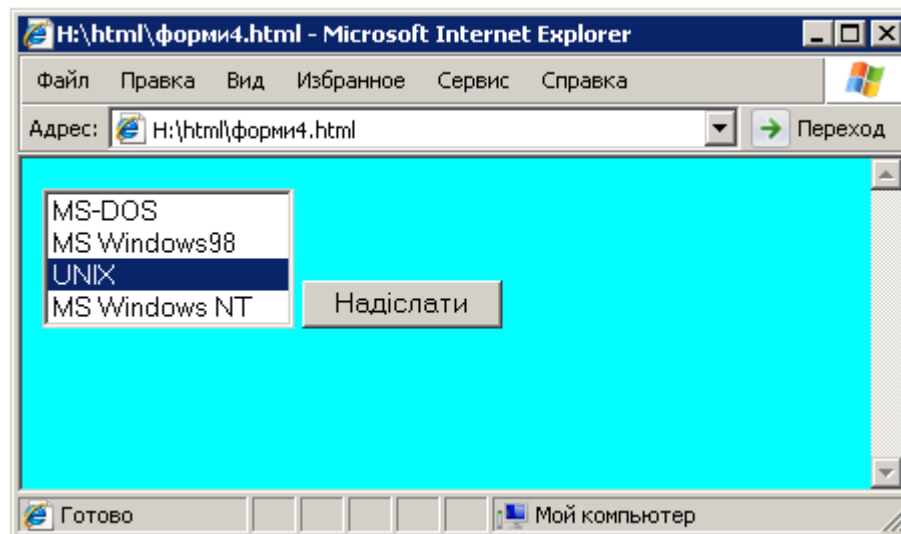


Рис. 6. Використання елемента SELECT.

## 5. OPTION

Елемент OPTION, який використовується лише з елементом SELECT, описує окремі пункти меню. Не потребує закриваючого тега. Атрибути:

- ▶ SELECTED – визначає пункт меню, який буде обраний спочатку при завантаженні документа. Якщо меню має тип "один з багатьох", то прапорцем SELECTED може бути позначений лише один пункт меню.
- ▶ VALUE – задає цьому пункту значення, яке буде використано поряд з іншими відомостями про вміст заповненої форми. При представленні інформації на сервер це значення буде об'єднано зі значенням атрибута

NAME в елементі SELECT.

Наприклад:

```
<SELECT NAME="gender">  
  <OPTION VALUE="male" SELECTED>Чоловік  
  <OPTION VALUE="female">Жінка  
  <OPTION VALUE="not_yet">Визначаюсь  
</SELECT>
```

На рис. 7 наведено тематичний html-документ, що містить різні типи форм.

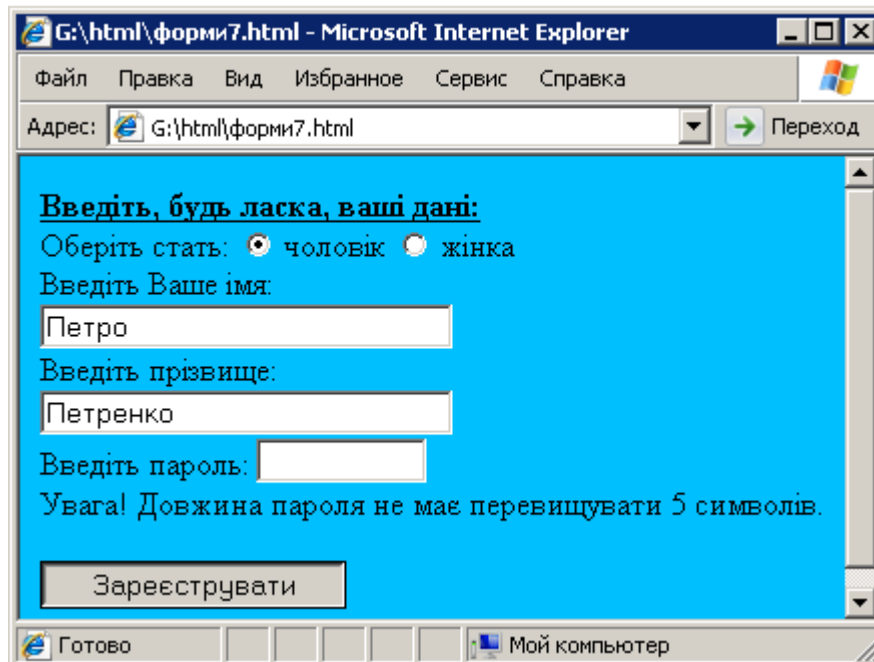


Рис. 7. Приклад html-документу, що містить форми

## ЗАВДАННЯ

1. Створити у власній папці html-документ анкета.html, для реєстрації анкетних даних абітурієнта.
2. Доповнити створений документ відповідною інформацією, використовуючи типи форм, описані у теоретичних відомостях.  
Для виконання завдання спочатку сплануйте інформаційне наповнення документа (які саме дані про абітурієнта потрібно запитати і за допомогою яких форм це краще реалізувати). Після чого наповніть створений документ відповідною інформацією, використовуючи HTML-коди, наведені в теоретичних відомостях.
3. Переглянути створену веб-сторінку за допомогою будь-якого веб-оглядача. При потребі внести відповідні корективи.

4. Розбити створений документ на два складові документи.  
Для цього створіть у своїй папці новий html-документ `анкета2.html`, в який помістіть частину інформації з документа `анкета.html`. Збережіть створені документи та перегляньте їх за допомогою оглядача.
5. З першого документу `анкета.html` створити гіперпосилання на другий документ `анкета2.html`, у такий спосіб, щоб користувач, заповнивши відповідні поля в першому документі, міг перейти до другого і продовжити вводити відповідні дані.
6. У кінці другого документу обов'язково помістити кнопку, при натисканні на яку запускається процес обробки даних з форми (`<INPUT TYPE=submit VALUE="Надіслати">` ) та кнопку, яка дає змогу очистити заповнені поля форми (`<INPUT TYPE=reset VALUE="Вдмінити">` ).
7. Перевірити роботу створених Вами документів. Перегляньте html-код створеної сторінки за допомогою команди Вигляд ⇒ Перегляд HTML-кода.
8. Відформатувати створені веб-сторінки якнайкраще, максимально використовуючи теги та параметри мови гіпертекстової розмітки HTML.
9. Збережіть відредаговані документи у власній папці.
10. Продемонструйте створені сторінки викладачу та оформіть звіт з виконання цієї лабораторної роботи.

### КОНТРОЛЬНІ ЗАПИТАННЯ

1. Що таке форми і з якою метою вони використовуються?
2. За допомогою яких команд на мові HTML створюються форми?
3. Яке призначення тегу `<FORM>`?
4. Для чого використовують атрибут `NAME` тегу `<FORM>`?
5. Який атрибут тегу `<FORM>` визначає URL-адресу, за якою буде відіслано вміст форми?
6. Яке призначення має атрибут `METHOD`?
7. За допомогою якої команди визначається спосіб кодування вмісту форми при відправці?
8. Яке призначення команди `INPUT`?
9. З якою метою використовують атрибут `NAME` тегу `<INPUT>`?
10. Які значення може приймати команда `TYPE`?

11. Що є результатом використання параметру TYPE=radio тегу <INPUT>?
12. Для чого всередині тегу <INPUT> вказують параметр CHECKED?
13. Як створити кнопку, при натисканні на яку запускається процес обробки даних з форми?
14. Яка команда дає змогу користувачу приєднати до створюваної форми файл?
15. Як створити графічну кнопку-картинку для передачі даних на сервер?
16. З якою метою використовується тег <TEXTAREA>?
17. Яке призначення команди SELECT?
18. Що визначає атрибут SIZE тегу <SELECT>?
19. Для чого використовується команда OPTION?
20. Як створити кнопку, при натисканні на яку поля заповнюваної форми очищуються?

## Лабораторна робота № 12

### ТЕМА. Каскадні таблиці стилів (CSS). Формат

**МЕТА:** ознайомитися з особливостями створення html-сторінок з динамічно змінним вмістом за допомогою каскадних таблиць стилів (CSS). Використовуючи набуті знання та вміння, створити власну таблицю стилів та застосувати її до html-документу.

### ХІД РОБОТИ

1. Створити новий html-документ.
2. Доповнити цей документ текстовою інформацією.
3. Створити свій стильовий файл, використовуючи селектори.
4. Застосувати створену таблицю стилів до html-документу.

### ТЕОРЕТИЧНІ ВІДОМОСТІ

**CSS** (Cascading Style Sheets) – це набір правил оформлення та форматування, які можуть бути застосовані до різноманітних елементів сторінки. У стандартному HTML для присвоєння певному елементу потрібних властивостей (колір, розмір, розташування на сторінці тощо)

потрібно кожного разу задавати ці властивості. Застосування таблиць стилів CSS дає можливість один раз визначити властивості елементів і визначити цей опис в якості стилю.

Каскадні таблиці стилів можна порівняти із стильовими файлами будь-якого текстового редактора. Для кожного елемента, що задається певним тегом HTML, можна визначити свій стиль відображення у вікні браузера (шрифти і кольори заголовків різних рівнів, шрифт і формат основного тексту і таке інше). Опис стилю можна зберегти в окремому файлі, що дасть змогу використати його на будь-якій кількості веб-сторінок. Отже, CSS є технологією створення та застосування стилів до html-документа.

Для керування форматуванням одного html-документа можна використовувати декілька таблиць стилів – браузер за певними правилами вибудовує пріоритетність їхнього застосування. Стиль задається за певними правилами, а таблиця стилів – це набір правил форматування елементів HTML. Будь-яке правило каскадних таблиць стилів має наступний синтаксис: ТЕГ { властивість: значення }.

Частина перед лівою фігурною дужкою – має назву **селектор**, а вираз в фігурних дужках – **декларація (визначення)**. Селектор є сполучною ланкою між документом HTML і стилями. Він визначає, які елементи залежать від цієї декларації. Селектором може бути будь-який тег HTML; визначення, своєю чергою, також складається з двох частин – властивості і її значення, розділених знаком двокрапки.

Наприклад, правило H1 {color:green; font-size:20pt} встановлює для всіх заголовків першого рівня у документі зелений колір та розмір шрифту 20 пунктів.

У розглянутому випадку селектором є елемент H1, а декларація, записана у фігурних дужках, задає значення двох властивостей: колір шрифту (властивість color, значення green) і розмір шрифту (властивість font-size, значення 20pt). В одному правилі можна задавати декілька визначень, розділених знаком «крапка з комою».

#### **Основні типи селекторів:**

– **клатвіатурні селектори** (type selectors) називаються так само, як і теги HTML. Наприклад: P – для звичайних абзаців, H1 – для заголовка першого рівня, LI – для елемента списку, TD – для елемента таблиці і т.і.

Застосування правила до клатвіатурного селектора приводить до того, що змінюється вигляд всіх елементів сторінки, які керуються відповідним

тегом HTML. Наприклад, правило TH { font-style: italic} застосовує курсив до всіх комірок заголовка таблиці, правило B {color: red;} робить весь напівжирний шрифт на сторінці червоним.

– **селектори класів** (Class selectors) не застосовуються автоматично до яких-небудь частин сторінки. Необхідно вказати, до яких саме ділянок сторінки потрібно їх застосувати. Клас дає змогу задати різні правила форматування для одного елемента певного типу або всіх елементів документа. Ім'я класу вказується у селекторі правила після імені тега і відділяється від нього крапкою. У тексті документа посилання на відповідний клас задається у параметрі class. Якщо клас повинен застосовуватися до всіх елементів документа, то в селекторі задається ім'я класу з крапкою без вказівки конкретного елемента.

У наступному прикладі (рис. 1) перший вираз створює клас s1, а другий – застосовує його до абзацу.

```
<STYLE type="text/css">
.s1 { color: white; background-color: Red;text-align: center; }
</STYLE> <BODY>
<p class=s1> CSS – каскадні таблиці стилів, які застосовуються для візуального
форматування документа в мовах розмітки. </p>
</BODY >
```

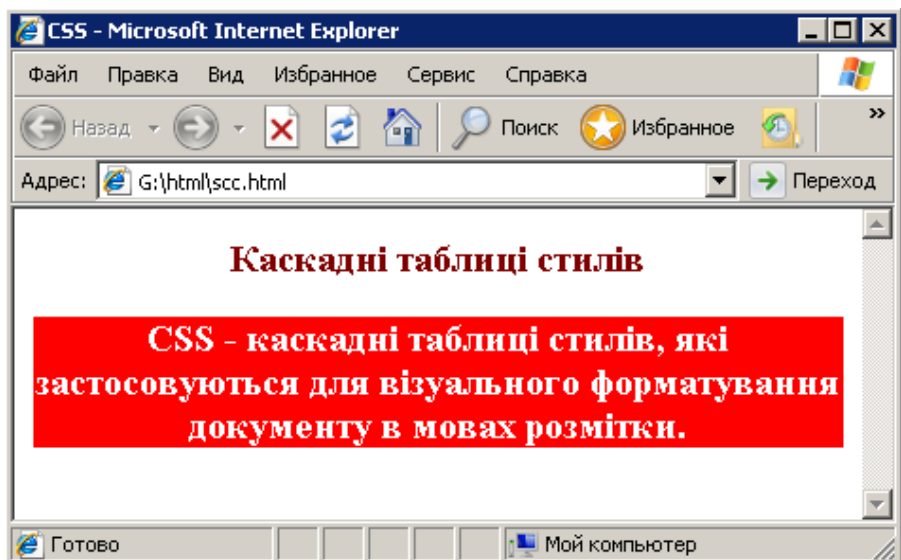


Рис. 1. Використання параметру class.

Таблицю стилів часто розміщують всередині тегу коментаря <!--.....-->.

Можна визначити декілька правил форматування для одного елемента і за допомогою параметра class відповідного тега застосовувати різні правила форматування у документі. Наприклад, можна визначити два класи відображення заголовка першого рівня:

```

<STYLE type="text/css">
H1.red { color: red; }
H1.blueBgrd { color: red; background-color: blue }
</STYLE>
<H1 class="red">Червоний шрифт</h1>
<H1 class="blueBgrd">Червоний шрифт на синьому фоні</h1>

```

– **селектори ідентифікаторів** (ID selectors) діють подібно до селекторів класів. Відмінність полягає в тому, що після створення стилю і прив'язки його до ідентифікатора можна вказати цей ідентифікатор для елемента сторінки. Наприклад, створюється стиль для всіх елементів з ідентифікатором bruce, а потім ідентифікатор bruce присвоюється абзацу. У результаті текст в абзаці повинен стати напівжирним:

```

#bruce { font-weight: bold} .....
<p id=bruce> Цей текст буде виділений напівжирним. </p>

```

Правила таблиць стилів регламентують використання унікального ідентифікаційного імені елемента як селектор, передуючи йому символом #. Параметр id задає унікальне ім'я елемента, яке використовується для посилань на нього. Цей параметр можна застосувати до будь-якого елемента документа.

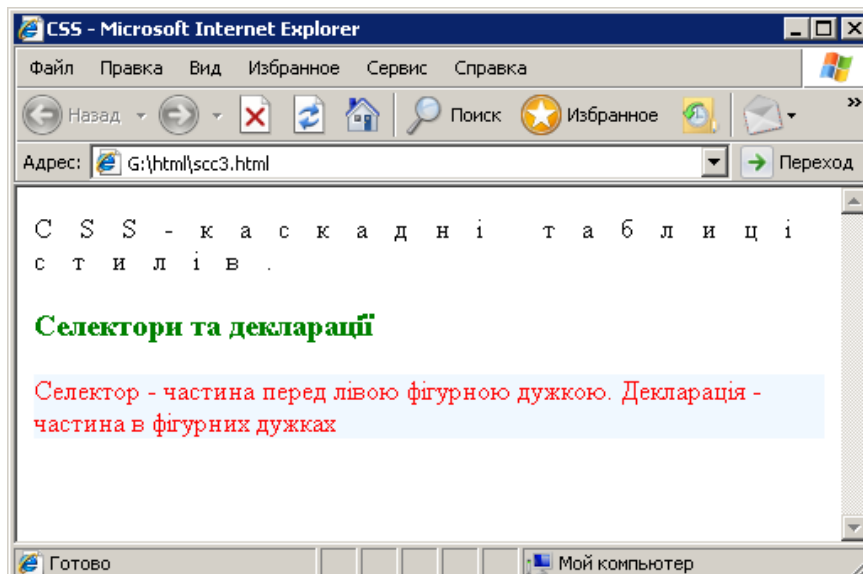


Рис. 2. Приклад використання селекторів ідентифікаторів.

Наприклад (рис. 2):

```

<STYLE>
  H2, H3 { color: green }
  #form { letter-spacing: 1em}
  #par1 {color: red; background-color: aliceblue}
</STYLE>
<BODY>
  <p id="form"> CSS - каскадні таблиці стилів. </P>

```

```
<H3>Селектори та декларації</H3>
<p id="par1"> Селектор - частина перед лівою фігурною дужкою. Декларація -
частина в фігурних дужках </P>
</BODY>
```

У цьому прикладі перший абзац ідентифікований ім'ям form в параметрі id, тому до нього застосовано правило з селектором #form, а друге правило у таблиці стилів застосовується до другого абзацу з ідентифікатором par1.

Не можна використовувати один і той же ідентифікатор більше, ніж для одного елемента на сторінці. Тому селектори ідентифікаторів, на відміну від селекторів класів, використовуються відносно рідко.

### **Вбудовування таблиць стилів у документ**

Щоб таблиця стилів могла впливати на зовнішнє представлення документа, її необхідно пов'язати з цим документом. Існує чотири способи скріплення документа і таблиці стилів:

- **Впровадження** – застосування всього документа таблиць стилів до html-документа за допомогою елемента стилю (style). Дає змогу задавати всі правила таблиці стилів безпосередньо у самому документі.
- **Вбудовування** в теги документа – застосування стилів до окремих елементів за допомогою атрибуту стилю. Дає змогу змінювати форматування конкретних елементів сторінки.
- **Скріплення** – поєднання зовнішньої таблиці стилів з HTML документом, використовуючи елемент посилання. Дає змогу використовувати одну таблицю стилів для форматування багатьох сторінок HTML.
- **Імпортування** – імпорт таблиць стилів, використовуючи CSS позначення @import. Дає змогу вбудовувати у документ таблицю стилів, розташовану на сервері.

Розглянемо детальніше зазначені способи.

При **впровадженні** таблиці стилів в документ, правила задаються в стильовому блоці, обмеженому тегами <STYLE type="text/css"> і </STYLE>, який повинен розміщуватися в розділі <HEAD> документа.

Наприклад (рис. 3):

```
<STYLE type="text/css">
<!--
H1 {color: Maroon; background-color: Yellow; font-size:12pt}
P {color: Teal; background-color: Aqua;text-align: center; }
B {text-transform: uppercase; }
-->
</STYLE>
```



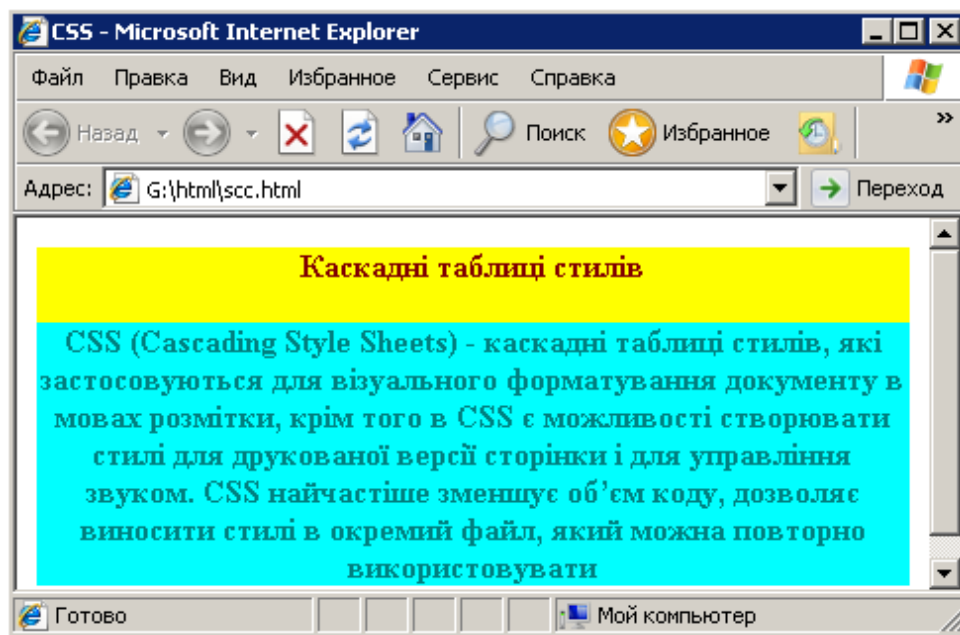


Рис.3. Приклад html-документа з вбудованою таблицею стилів.

Наступний спосіб завдання значень властивостей таблиці стилів призначений для оперативного форматування певного елемента документа і називається **вбудовування в теги**. Кожен тег HTML має параметр `style`, в якому можна задати значення його властивостей відповідно до синтаксису каскадних таблиць стилів. Наприклад, у наступному прикладі задається форматування заголовка першого рівня, визначальне його відображення шрифтом червоного кольору:

```
<H1 style="color: red">Текст заголовка</H1>
```

Вбудовування визначень стилів у конкретний тег впливає на відображення тільки елемента, що визначається цим тегом.

Спосіб **скріплення** полягає в наступному: таблиця стилів зберігається в окремому файлі і приєднується до документа за допомогою тега `<LINK>`, що задається в розділі `<HEAD>`:

```
<LINK rel="stylesheet" type="text/css" href="mystyles.css">
```

Скріплення дає змогу розробникові застосувати однаковий набір правил форматування до групи html-документів, тобто до всього сайту, що приводить до одноманітного відображення різних документів. Зв'язуваним файлом є звичайний текстовий файл з кодом таблиці стилів, що має розширення `.css`. Цей метод дає можливість посилатися на таблиці стилів, які розташовуються не на тому ж сервері, що і сама сторінка.

При **імпортуванні**, як і при скріпленні, вбудовується зовнішня таблиця стилів, але за допомогою властивості `@import` таблиці стилів: `@import: url`

(mystyles.css)

Його слід задавати на початку стильового блоку <STYLE> або зв'язуваної таблиці стилів перед завданням решти правил. Значенням властивості @import є URL-адрес файлу таблиці стилів.

Всі способи вбудовування таблиць стилів вільно поєднуються в одному документі.

### Групування селекторів

Головною метою створення мови CSS є стислість, тому передбачено чимало механізмів для зменшення розмірів таблиць стилів шляхом групування селекторів і декларацій. Наприклад, розглянемо такі правила:

```
H1 { font-weight: bold },  
H2 { font-weight: bold },  
H3 { font-weight: bold }.
```

Усі три правила встановлюють жирний шрифт. Оскільки всі три заяви є ідентичними, селектори можна згрупувати у такий спосіб: H1, H2, H3 {font-style: bold}

Селектор може мати більше однієї декларації (визначення). Визначення групуються аналогічно, тільки у списку вони розділяються крапками з комою. Наприклад:

```
H1 { font-weight: bold },  
H1 { font-size: 14pt },  
H1 { font-family: Arial },
```

можна записати у вигляді одного правила, згрупувавши визначення:

```
H1 { font-weight: bold; font-size: 14pt; font-family: Arial; }
```

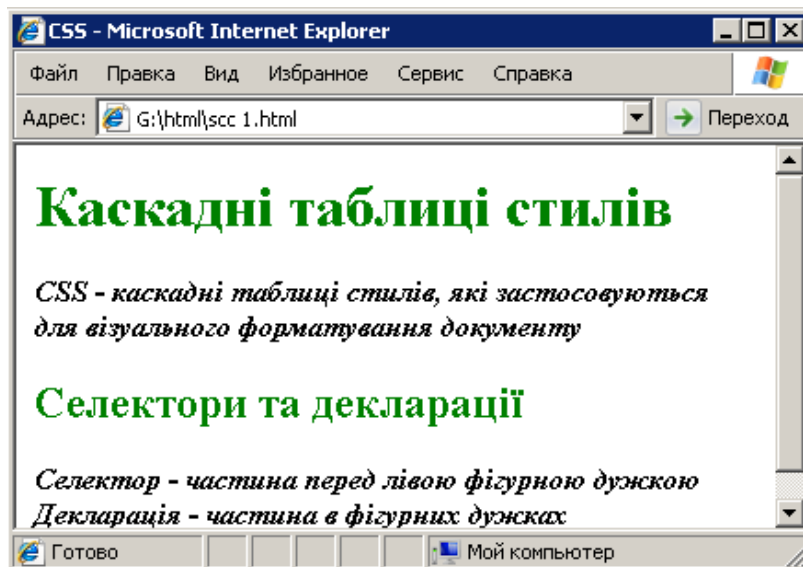


Рис. 4. Результат додання до таблиці стилів правила H1, H2 { color: green}.

При використанні властивості font попередній приклад запишеться так:

H1 (font: bold 14pt Arial). При завданні таблиці стилів можна вільно комбінувати всі три правила групування для зменшення її розмірів.

### Правила успадкування стилістичних властивостей

Якщо елемент розташований всередині деякого іншого елемента, то він успадковує стилістичні властивості «елемента-батька». Наприклад, для того, щоб встановити синій колір всім елементам у документі, можна перерахувати всі типи елементів у селекторі: H1, H2, P, LI { color: blue}.

Однак, більшість html-документів містять значну кількість селекторів, тому, замість того, щоб встановлювати стиль на кожен тип елемента, можна поставити його на елемент BODY: BODY {color: blue}. Оскільки інші елементи успадковують властивості від елемента BODY, всі вони будуть наслідувати синій колір. Якщо виникає потреба змінити спосіб відображення певного елемента, потрібно задати його властивості в стильовому файлі. Розглянемо такий приклад:

```
<STYLE TYPE="text/css">  
  BODY {color: blue }  
  H1 {color: green }  
</STYLE>
```

H1 є дочірнім для елемента BODY, тому успадковує його властивості. Отже, у наведеній таблиці стилів до нього застосовані одночасно два правила. Однак елементи H1 будуть відображатися зеленим кольором (рис. 5), оскільки друге правило є більш конкретним, ніж перше і впливає тільки на елементи H1 в документі.

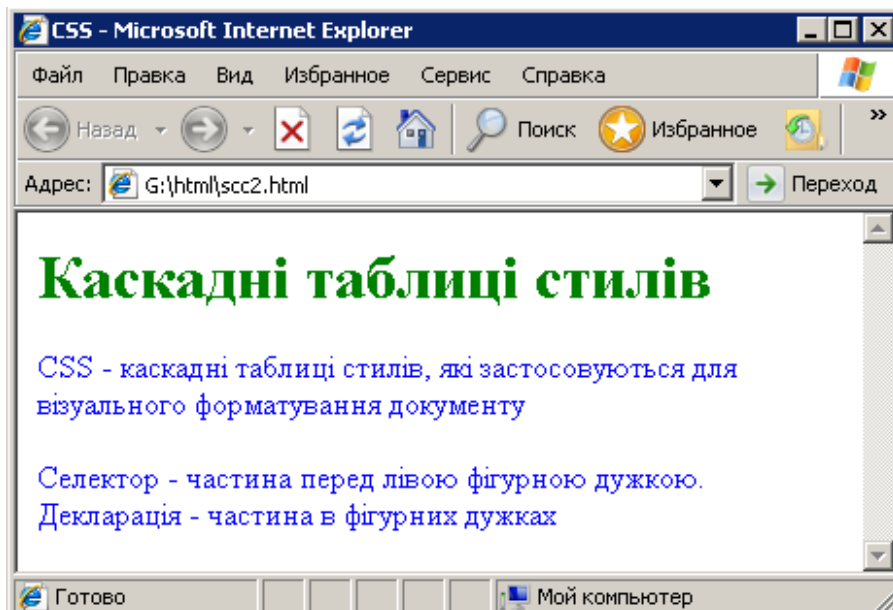


Рис. 5. Приклад успадкування стильових властивостей.

## Псевдокласи

У CSS є таке поняття як **псевдоклас**. На відміну від звичайного класу, дія псевдокласу розповсюджується не на весь текст, до якого застосований цей стиль, а лише на його частину і можливо лише у певному стані. Так, наприклад, CSS регламентує правила для відображення посилань через псевдокласи елемента А:

```
A:link {color: red} /*непереглянуте посилання*/
```

```
A:visited {color: blue} /*переглянуте посилання*/
```

```
A:active {color: green} /*активне посилання*/
```

```
A:hover {color: lime} /* посилання, вибране курсором миші*/
```

Будь-яке посилання у документі можна віднести до одного з перерахованих класів. Іншим приклад псевдокласу є визначення буквиці на початку абзацу: `p:first-letter {font-size: 200%; font-weight: bold}`

Зверніть увагу, що і в тому, і в іншому випадку дія стилю розповсюджується або на певний стан (користувач готується клацнути по посиланню), або на фрагмент тексту (змінюється тільки перша буква абзацу). У цьому і полягає сенс псевдокласів

## ЗАВДАННЯ

1. Створити новий html-документ з назвою «моє місто».

Для виконання завдання проробіть такі дії:

- Створіть у власній папці текстовий документ з назвою «моє місто».
- Збережіть створений документ, після чого поміняйте розширення документа з .txt на .html.

2. Доповнити його текстовою інформацією про своє рідне місто.

3. Створити у власній папці копію даного html-документу з ім'ям «моє місто\_1».

4. Надати документу «моє місто» певного оформлення, використовуючи основні теги фізичного форматування мови HTML (фон, вирівнювання, колір і розмір тощо)

5. Створити свій стильовий файл, використовуючи селектори. Задайте за замовчуванням такі параметри для всіх абзаців, перевизначивши тег `<p>` і псевдокласи тега `<p>`):

- вирівнювання абзацу;
- відступ червоного рядка;

- розмір і колір першої букви.
6. Задайте за замовчуванням такі властивості посилань для всіх сторінок:
    - колір і оформлення посилання;
    - колір і оформлення відвіданого посилання;
    - колір і оформлення активного посилання;
    - колір і оформлення посилання, у момент знаходження курсору миші над нею.
  7. Застосуйте створену таблицю стилів до html-документу «моє місто\_1». Збережіть відредаговані документи у власній папці.
  8. Перевірити роботу створених документів. Перегляньте html-код створеної сторінки за допомогою команди Вигляд ⇒ Перегляд HTML кода.
  9. Порівняти вигляд документів «моє місто» та «моє місто\_1».
  10. Продемонструвати створені документи викладачу та оформити звіт з виконання лабораторної роботи.

## **КОНТРОЛЬНІ ЗАПИТАННЯ**

1. Яке призначення каскадних таблиць стилів CSS ?
2. Що таке таблиці стилів?
3. Яку структуру мають правила CSS?
4. Що таке селектор? Приклади селекторів.
5. Наведіть приклади декларацій.
6. З яких частин складаються декларації?
7. Основні типи селекторів?
8. Які особливості мають клавіатурні селектори?
9. Наведіть приклади застосування селекторів класів.
10. Для чого застосовують селектори ідентифікатори?
11. Як можна зв'язувати створені таблиці стилів з html-документом?
12. Яке призначення має тег <STYLE>?
13. За допомогою яких тега відбувається приєднання до документа таблиці стилів?
14. Яке розширення має мати зв'язуваний текстовий файл з кодом таблиці

стилів?

15. Для чого та за якими правилами відбувається групування селекторів?
16. Як відбувається успадковування правил форматування?
17. Що таке активне посилання?
18. Яким чином способи відображення посилань у html-документах?
19. З якою метою створюються каскади таблиць стилів?
20. Якими є основні переваги використання каскадних таблиць стилів?

## Лабораторна робота № 13

### ТЕМА. Таблиці стилів CSS. Блочні об'єкти. Оформлення

**МЕТА:** ознайомитися з особливостями створення html-сторінок з динамічно змінним вмістом за допомогою каскадних таблиць стилів. Використовуючи набуті знання та вміння, навчитися створювати власні таблиці стилів з використанням блочних об'єктів та застосувати їх до html- документа.

#### ХІД РОБОТИ

1. Створити новий html-документ.
2. Доповнити цей документ текстовою інформацією на довільну тему.
3. Створити свій стильовий файл, використовуючи форматування та блочні об'єкти.
4. Застосувати створену таблицю стилів до html-документа.

#### ТЕОРЕТИЧНІ ВІДОМОСТІ

Модель форматування каскадних таблиць стилів орієнтована на представлення будь-якого елемента HTML в оточенні вкладених прямокутних блоками. Властивості таблиць стилів дають змогу встановлювати розміри і кольори всіх складових блоків: поле, межа, відступ, блок вмісту. Поле завжди є прозорим прямокутником, тому його колір успадковує колір елемента-батька (для абзацу це елемент <BODY>). Відступ завжди має колір фону самого елемента. Колір і ширину межі можна вказати окремо.

Всі блоки у сукупності складають блок форматування або блок відображення елемента, тобто видиме у вікні браузера зображення

елемента. Розміри блоку форматування/відображення елемента складаються з розмірів самого елемента і розмірів відступів, межі і полів.

У каскадних таблицях стилів, всі доступні властивості форматування елементів в html-документі розбиті на такі категорії: шрифт (встановлює властивості друкованих шрифтів), колір і фон (колір тексту і фону, а також зображення, що використовується в якості фону), текст (вирівнювання, форматування і розрядка тексту), блок (властивості форматування блоків елементів), візуальне форматування (властивості, пов'язані з блоками відображення елементів, їхнім позиціонуванням і відображенням списків), друк (специфікація розриву сторінки), фільтри і переходи (мультимедійні ефекти і перетворення графічних зображень), псевдокласи і інші властивості (властивості @import, cursor та important).

### Шрифти

Одним з важливих завдань при розробці html-документа є вибір відповідного шрифту для окремих частин документа. Шрифти розрізняються за своїм зовнішнім виглядом, за розміром, стилем та жирністю відображення. Каскадні таблиці стилів надають у розпорядження розробника набір властивостей для установки всіх перелічених параметрів шрифтів. Крім того, є можливість завантажувати відсутні на комп'ютері користувача шрифти безпосередньо з сервера, на якому розташований документ.

Властивість **font-family** дає змогу розробникові сторінки задати список шрифтів одного стилю і розміру, серед яких браузер може шукати необхідний символ. На відміну від інших властивостей каскадних таблиць стилів назви сімейств у списку відділяються комами: BODY {font-family: TIMESDL, "Times New", serif }.

При завантаженні html-сторінки браузер спочатку шукає на комп'ютері користувача шрифт TIMESDL. Якщо такий шрифт відсутній, то браузер намагається застосувати шрифт Times New, а якщо і він не знайдений, то використовується будь-який шрифт з сімейства шрифтів serif.

Поняття типових сімейств шрифтів введено у каскадні таблиці стилів з метою реалізації якнайгіршого варіанту відображення сторінки, якщо не знайдені спеціально використані автором шрифти. У будь-якій реалізації каскадних таблиць стилів повинно існувати п'ять типових сімейств шрифтів, які відповідають реальним шрифтам, що зазвичай встановлюються на більшості комп'ютерів:

- serif (наприклад, Times),
- sans-serif (наприклад, Helvetica),
- cursive (наприклад, Zapf-Chancery),
- fantasy (наприклад, Western),
- monospace (наприклад, Courier).

Властивість **font-style** визначає стиль шрифту з вибраного сімейства: нормальний (normal), курсивний (italic) або похилий (oblique).

Властивість **font-weight** вибирає із заданого сімейства шрифт певної жирності. У рекомендаціях регламентується 9 градацій жирності шрифту, що задаються числами 100, 200 і так далі до 900. Значення 100 відповідає "найблідішому" шрифту.

Для завдання нормального шрифту використовується ключове слово normal, що відповідає цифровому значенню 400. Значення bold застосовується для вибору загальноприйнятого напівжирного зображення і його цифровим еквівалентом є 700.

Властивість **font-size** визначає розмір шрифту. Його значення може бути абсолютним або відносним. Відносне значення можна задати одним з наступних ключових слів: xx-small, x-small, small, medium, large, x-large, xx-large, які є індексами в таблиці розмірів шрифтів, підтримуваних браузером. За умовчанням браузер використовує значення medium. Абсолютне значення можна задати і у вигляді абсолютного значення довжини, наприклад 10pt, але в цьому випадку висота шрифту не залежить від таблиці розмірів шрифтів браузера, що зберігається.

**Відносний** розмір шрифту можна задати також у відсотках до розміру шрифту батька або у відносних одиницях довжини:

```
P { font-size: 10pt },
EM { font-size: 120% },
```

Для встановлення різних параметрів тексту одночасно використовують властивість **font**, що дає змогу скоротити створені таблиці стилів. Всі значення перелічених властивостей задаються через пропуски в наступному порядку: font-style, font-variant, font-weight, font-size, font-height і font-family. Перші три властивості можуть не задаватися, що відповідає встановленню їхніх значень рівних normal. Розмір шрифту і висота рядка (властивість line-height) задаються через слеш, елементи списку сімейств шрифтів властивості font-family – через кому. Наприклад,

```
P { font: bold 18pt/20pt "Courier", fantasy }.
```



У наведеному прикладі для абзацу задається напівжирний шрифт Courier заввишки 18 пунктів, висота рядків – 20 пунктів (рис. 1). Якщо не знайдений шрифт Courier, то застосовується будь-який шрифт типового сімейства fantasy.

Властивість **@font-face** застосовується для завдання сімейства шрифту. Якщо вказаний шрифт відсутній на комп'ютері користувача, то він завантажується з мережі за заданою другим параметром URL-адресою:

```
@font-face { font-family: CoolFont; src:url(http://myserver.com/CoolFont.ect); }
```

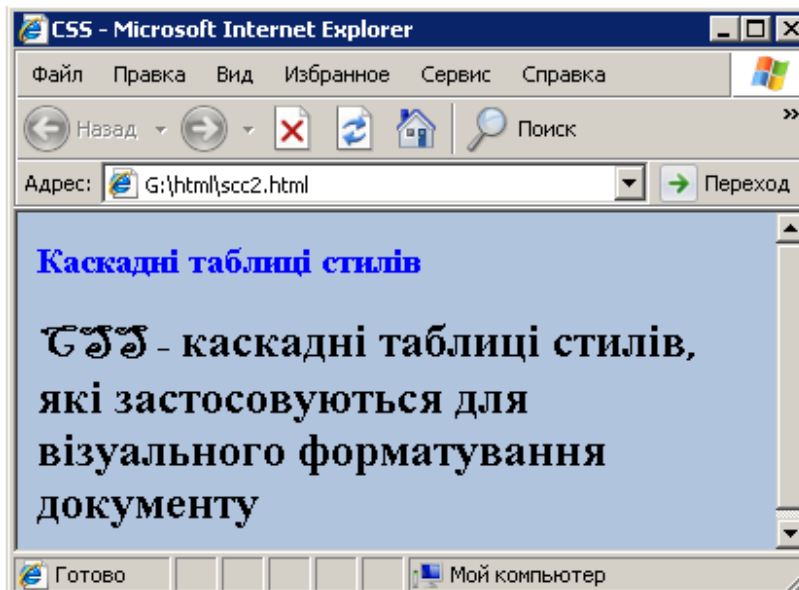


Рис. 1. Використання властивості font.

## Колір і фон

Для встановлення кольору тексту елемента існує єдина властивість color. Її значенням є колір, що задається за допомогою ключових слів або rgb-функції. Наступні правила встановлюють синій колір тексту відповідних елементів:

```
<P> { color: blue }, <EM> { color: rgb(0, 0, 255)}.
```

Колір фону визначається значенням властивості background-color, а зображення, використовуване як фон, задається властивістю background-image. Початковим значенням властивості background-color є transparent, яке визначає фон елемента як прозорий. Значенням властивості background-image є абсолютна або відносна адреса файлу зображення, використовуваного як фон. У наступному правилі задається адреса файлу зображення для фону тіла документа і відсутність фону для абзаців документа:

```
BODY { background-color: lightsteelblue; background-image: url(/image/image.gif)}
```

```
<P> { background-image: none }
```

Властивість `background-repeat` визначає повторюваність зображення фону і способи повторюваності. Допустимими значеннями є `repeat` (повторюваність і по вертикалі і по горизонталі), `repeat-x` і `repeat-y` (повторюваність відповідно по горизонталі або вертикалі) і `no-repeat` (зображення не повторюється). Наприклад,

```
BODY {background-image: url (butterfly.jpg); background-color: lightsteelblue;  
background-repeat: repeat-x; color: black}.
```

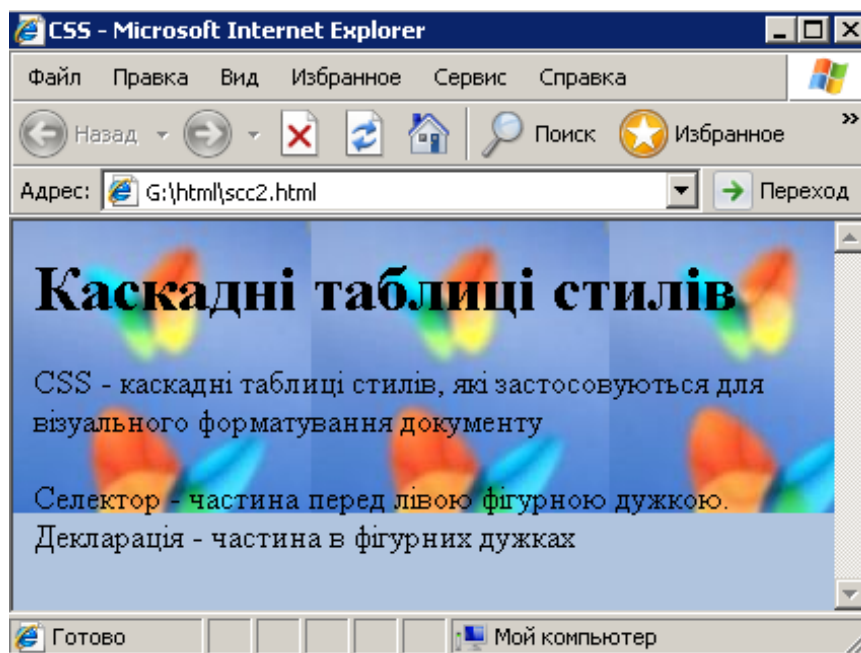


Рис. 2. Використання значення `repeat-x` властивості `background-repeat`.

Властивість `background-attachment` визначає, чи буде фон, на якому зображується документ, залишатися нерухомим (значення `fixed`) при прокрутці вмісту вікна браузера чи буде прокручуватися разом з документом (значення `scroll`). Приклад закріпленого у вікні браузера зображення фону представлений нижче:

```
BODY {background-image: url (butterfly.jpg); background-color: lightsteelblue;  
background-repeat: repeat-x; background-attachment: fixed}.
```

Початкове положення зображення, використовуваного як фон, у блоці вмісту елемента визначає властивість `background-position`. Значенням цієї властивості є координати прив'язки певних точок зображення і блоку вмісту. Їх можна задати у відсотках, в абсолютних одиницях довжини, а також з використання комбінацій ключових значень.

Властивість `background` дає змогу одночасно встановлювати значення властивостей `background-color`, `background-image`, `background-repeat` і

background-attachment. Всі допустимі значення індивідуальних властивостей задаються у вигляді списку, елементи якого відокремлені пробілами. Якщо значення якої-небудь властивості не задано, то для неї встановлюється початкове значення, визначене браузером:

```
BODY { background: lightsteelblue url(image.gif) center }.
```

Це правило встановлює колір і зображення фону, а також положення зображення у вікні браузера. Решту властивостей фону приймають початкові значення.

### Форматування тексту

Властивості даної категорії впливають на відображення символів, слів і абзаців. Вони визначають відстань між словами і буквами в словах, задають відступи і висоту рядків в абзацах.

Властивість **letter-spacing** впливає на відстань між символами при відображенні тексту. Його значення, що задається в одиницях довжини, визначає пропуск, що додається до встановленого за умовчанням пропуску між символами. Браузер збільшує відстані не тільки між символами слів, але і відстань між словами, оскільки пропуск – це також символ.

Каскадні таблиці стилів дають змогу перетворювати текст. Якщо значення властивості **text-transform** рівне `capitalize`, то всі слова відображаються з прописної букви. Значення `uppercase` і `lowercase` цієї властивості приводять, відповідно, до перетворення всіх букв в прописних або рядкових, незалежно від їхнього завдання у тексті документа HTML. Значення `none` знімає всі установки, набуті в результаті успадкування.

Властивість **text-decoration** задає підкреслення (`underline`), надкреслювання (`overline`) або перекреслювання (`line-through`) тексту.

Вирівнювання тексту в блоці вмісту елемента визначається значенням властивості **text-align**. Текст вирівнюється по лівому краю при значенні `left`, по правому краю – при значенні `right` і по центру – при значенні `center`. Відступ першого рядка елемента задається значенням властивості **text-indent**, яке визначає величину відступу в абсолютних або відносних одиницях довжини.

Властивість **vertical-align** визначає положення елемента по вертикалі щодо елемента-батька. Його значенням може бути будь-яке ключове слово з таблиці:

Значення	Результат
baseline	Вирівнювання базової лінії елемента (або низу, якщо елемент не має базової лінії) за базовою лінією батька
middle	Вирівнювання середньої точки елемента (зазвичай зображення) на рівні базової лінії батька плюс половина ширини блоку вмісту батька
sub	Елемент відображається у вигляді нижнього індексу
super	Елемент відображається у вигляді верхнього індексу
text-top	Вирівнювання верху елемента з верхом шрифту елемента-батька
text-bottom	Вирівнювання низу елемента з низом шрифту елемента-батька
top	Вирівнювання верху елемента з верхом найвищого елемента рядка
bottom	Вирівнювання низу елемента з елементом, розташованим нижче за всі елементи рядка

Відстань між базовими лініями двох сусідніх рядків (висота рядка) задається установкою значення властивості **line-height**. Числове значення цієї властивості визначає висоту рядка, що обчислюється множенням розміру шрифту поточного елемента на задане число.

### Блоки

CSS можна використовувати, щоб встановити, яким має бути простір навколо різних елементів. Серед всіх властивостей, що встановлюють параметри блокових елементів можна виділити три великі групи, що формують блоки **полів, меж і відступів**, причому групу, що працює з межею, можна підрозділити ще на чотири групи, що встановлюють значення кольору, стилю, ширини і одночасно всіх перерахованих властивостей межі. Всі ці групи відрізняє те, що у них зазначені властивості для завдання параметрів верхніх, нижніх, правих, лівих і одночасно всіх чотирьох частин відповідних блоків форматування елемента.

У властивості `margin` можна одночасно встановити значення всіх чотирьох параметрів поля елемента. Наприклад, наступний `blockquote` елемент визначає розташування цитати (рис. 3):

```
BLOCKQUOTE {margin: 0.5em 0em 0.5em 0em; font: bold oblique}.
```

Якщо у властивості `margin` визначено тільки одне значення, то воно застосовується до всіх параметрів поля елемента. При завданні двох або трьох значень, не вказані значення беруться з установок протилежних боків.

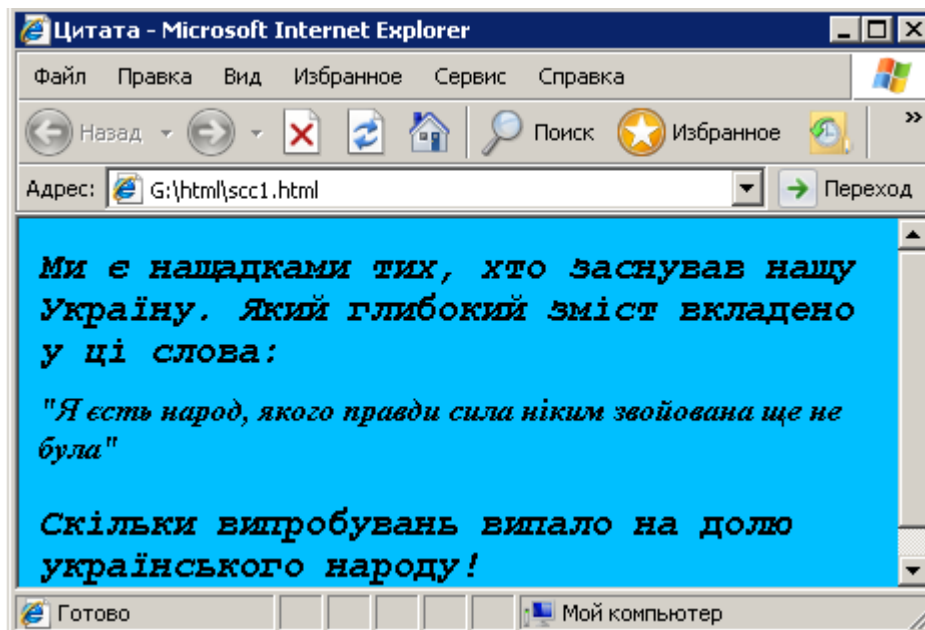


Рис. 3. Використання властивості margin.

Властивість **padding** дає змогу одночасно встановити значення всіх чотирьох відступів елемента. Значення властивості **border-width** визначає ширину межі елемента для всіх перерахованих її частин. Все, що було сказано про завдання значень для одночасної установки полів, відноситься і до цієї властивості.

Значеннями цих властивостей можуть бути ключові параметри *thin*, *medium* і *thick* або значення довжини. Ширина межі, визначувана ключовими параметрами залежить від браузера. Єдине, що можна гарантувати – це те, що ширина *thin* не більше ширини *medium*, яка, своєю чергою, не більше ширини *thick*.

Властивість **border-color** визначає кольори всіх частин межі. Чотири параметри кольору підкоряються все тим же правилам, описаним при завданні полів елемента. Якщо заданий тип межі, але не заданий її колір, то за умовчанням використовується колір самого елемента.

Всі попередні установки властивостей межі не матимуть ніякої дії на відображення елемента, якщо не встановлений тип межі, оскільки за замовчуванням тип межі не визначений і вона не відображається. Властивість **border-style** визначає одночасно типи всіх частин межі. Значеннями можуть бути ключові параметри *none*, *solid*, *double*, *groove*, *ridge*, *inset*, *outset*. Опис типів меж, відповідаючи всім перерахованим значенням, наведені у таблиці.

Ключовий параметр	Тип межі
none	Межа не відображається
solid	Межа відображається суцільною лінією
double	Межа відображається подвійною лінією (сума товщини двох ліній і проміжку між ними рівна значенню властивості border-width)
groove	Межа відображається, неначе вона втиснула в лист
ridge	Межа відображається, неначе вона виступає над листом
dotted	Межа відображається, пунктирною точковою лінією
inset	Весь блок елемента відображається, неначе він втиснутий в лист
outset	Весь блок елемента відображається, неначе він виступає над листом

### Позиціонування

Елементи HTML відображаються браузером послідовно, в тому порядку, як вони визначені в тексті html-файла з урахуванням їхнього положення в структурі документа і розміщення попередньо відображених елементів і елементів-контейнерів, в яких вони можуть міститися. При компонуванні сторінки використовуються налаштування браузера для визначення положення кожного елемента. Наприклад, два послідовні абзаци слідує один за одним, причому кожен починається з нового рядка.

Властивість **position** елемента дає змогу визначити спосіб його позиціонування на сторінці: статичний, відносний або абсолютний. Відносний спосіб визначає зсув елемента щодо його природного положення у потоці відображення елементів. Абсолютний спосіб видаляє елемент з природного потоку позиціонування і дає змогу розмістити його на сторінці абсолютно довільно. Статичний спосіб, що є замовчуваним способом позиціонування елементів, припускає природний потік відображення елементів сторінки у вікні браузера відповідно до ієрархії об'єктів документа.

Значення *static*, *relative* і *absolute* властивості **position** визначає відповідний спосіб позиціонування елемента, який складається із значення вказаної властивості елемента, його положення в ієрархічній структурі документа, місцем його визначення в початковому файлі HTML і значенням його властивостей *top* і *left*. Ці останні властивості визначають зсув вниз і управо лівого верхнього кута блоку відображення елемента.

За допомогою позиціонування блоків можна створювати 2,5-вимірні зображення. При цьому використовується додаткова властивість *z-index*,

яка визначає порядок перекриття блоків. На рисунку 4 наведено приклад перекриття блоків, що створює ефект 2,5-вимірного зображення. Код цього прикладу наведено нижче:

```
<div style="position: relative; width: 200; height: 200; z-index: 0; background-color: #FFFF00"> Блок – 1 </div>  
<div style="position: relative; width: 200; height: 200; left: 100; z-index: 1; top: -100; background-color: #00FFFF"> Блок – 2 </div>  
<div style="position: relative; left: 220; top: -400; width: 200; height: 200; z-index: 2; background-color: #00FF00"> Блок – 3 </div>  
<div style="position: relative; width: 750; height: 20; z-index: 1; top: -580; background-color: #FE76AF"> Блок – 4 </div>  
<div style="position: relative; top: -600; z-index: 3; left: 100"> Майже 3D </div>
```



Рис. 4. 2,5-вимірне позиціонування.

За допомогою блоків можна керувати переповненням і видимістю окремих елементів, зокрема тексту. За замовчування уся інформація, яка не помістилася у блок, виходить за його межі. При заданні властивості **overflow:hidden** все, що виходить за межі блоку, буде приховано. При переповненні блоку із визначеною властивістю **overflow:auto** блок не збільшуватиметься, а інформацію можна переглянути за допомогою смуг прокрутки

## ЗАВДАННЯ

1. Створити новий html-документ.

Для виконання завдання проробіть такі дії:

- Створіть у власній папці текстовий документ.
  - Збережіть створений документ, після чого поміняйте розширення документа з .txt на .html.
2. Доповнити його текстовою інформацією на довільну тему, яку логічно можна подати у табличній формі.
  3. Оформити сторінку за допомогою стильових блоків, використовуючи форматування, позиціонування та обрамлення тексту.
  4. Зробити колаж з картинок або фотографій (не менше 5-ти) застосувавши 2,5-мірне позиціонування. Для виконання цього завдання використати приклад, наведений у теоретичних відомостях.
  5. Додати на сторінку мітки, застосувавши властивість overflow:
    - visible – для першого розділу,
    - auto – для другого розділу,
    - hidden – для третього розділу,
    - scroll – для четвертого розділу,
    - auto – для п'ятого розділу.

Задати розміри, колір фону або бордюр блоків.

6. Створити власний стильовий файл.
7. Застосуйте створену таблицю стилів до html-документу. Збережіть відредагований документ у своїй папці.
8. Перевірити роботу створеного документу. Перегляньте html-код створеної сторінки за допомогою команди Вигляд ⇒ Перегляд HTML кода.
9. Продемонструвати створений документ викладачу та оформити звіт з виконання лабораторної роботи.

## **КОНТРОЛЬНІ ЗАПИТАННЯ**

1. Які значення може мати властивість display, і що вони визначають?
2. За допомогою яких властивостей визначається горизонтальне форматування елементів?
3. Вкажіть основні одиниці вимірювання довжини які використовуються в CSS для завдання значень властивостей, що визначають розміри об'єктів.
4. Для чого використовується властивість font-family?



5. Що визначає властивість font-style? Які значення вона може приймати?
6. Як можна встановити напівжирний шрифт?
7. З якою метою використовують властивість font-size?
8. Як встановити потрібний розмір шрифту?
9. За допомогою якої властивості встановлюється колір тексту елемента?
10. Як за допомогою CSS встановити зображення в якості фону?
11. Які значення приймає властивість background-repeat?
12. Як реалізується ефект переміщення вмісту вікна над нерухомим малюнком?
13. Що визначає властивість background-position? Які значення вона приймає?
14. Як змінити відстань між символами при відображенні тексту?
15. Якою властивістю визначається вирівнювання тексту у блоці вмісту елемента?
16. Як визначити колір та тип межі?
17. З якою метою використовується властивість position елемента?
18. Як задати довільне розміщення елемента на сторінці?
19. Для чого використовується властивість z-index?
20. Для чого використовується властивість overflow?

## Лабораторна робота № 14

### ТЕМА. Основи JavaScript. Змінні, операції, стрічки. Масиви

**МЕТА:** ознайомитися з особливостями створення html-сторінок з динамічно змінним вмістом. Використовуючи набуті знання та вміння, навчитися працювати із змінними та масивами в JavaScript.

#### ХІД РОБОТИ

1. Створити новий html-документ.
2. Доповнити цей документ текстовою інформацією на довільну тему.
3. Створити свій власний скрипт з використанням змінних та масивів.
4. Застосувати створений скрипт до html-документа.

## ТЕОРЕТИЧНІ ВІДОМОСТІ

Для написання динамічних веб-сторінок використовуються фрагменти коду, написані на мові JavaScript (або іншій мові сценаріїв), яка має синтаксис, відмінний від HTML. Фрагменти, написані на мові JavaScript і інших подібних мовах, що інтерпретуються, називають **сценаріями**.

Як правило, скрипти вставляються безпосередньо у тіло html-документа за допомогою спеціального тега `<SCRIPT>...</SCRIPT>`. Браузер аналізує інформацію, що перебуває між цими тегами і для виконання сценарію приводить у дію той або інший інтерпретатор. Альтернативним і у багатьох випадках корисним варіантом є розміщення скриптів в окремих файлах. Цей підхід найбільш прийнятний при використанні скриптів досить великого об'єму, а також для зберігання службових процедур.

Скрипти, збережені в окремому файлі з розширенням `.js`, включаються в код сторінки так:

```
<SCRIPT src="file.js"></SCRIPT> або  
<SCRIPT src="script-l.js" language JavaScript 1.1 ></SCRIPT>
```

Якщо браузер користувача не має засобів роботи зі скриптами, то корисно використовувати тег `<NOSCRIPT>...</NOSCRIPT>` для виводу відповідного цьому випадку варіанту вмісту сторінки. Наприклад:

```
<NOSCRIPT>  
!!! Браузер не підтримує скрипти !!!  
</NOSCRIPT>
```

Тег `<SCRIPT>`, так само як і інші теги, може мати свої атрибути. Це `LANGUAGE=(мова)` і `TYPE=(тип)`. У першому вказується мова, на якій написаний сценарій. За замовчуванням, значенням цього атрибуту є JavaScript, тому, для звичайного JavaScript-сценарію цей атрибут можна не вказувати. В атрибуті `TYPE` можна вказати тип сценарію. У більшості випадків це `text/javascript` (або `text/vbscript` для сценаріїв, написаних на VBScript).

Розглянемо приклад найпростішої html-сторінки з використанням скриптів (рис.1).

```
<HTML>  
<HEAD></HEAD>  
<BODY>  
<SCRIPT language JavaScript 1.3 >  
document.write("Ця сторінка написана на JavaScript!");  
</SCRIPT>  
<NOSCRIPT>
```

!!! Браузер не підтримує скрипти !!!  
</NOSCRIPT> </BODY> </HTML>

Якщо у наведеному прикладі текст сценарію замінити на наступний  
<SCRIPT> window.status = "Ця сторінка написана на JavaScript!"; </SCRIPT>  
то вікно браузера буде порожнім, оскільки ніякого тексту не виводилося.  
Проте, зазначений напис буде відображено у рядку стану.

А при використанні сценарію

```
<SCRIPT> window.status = " Ця сторінка написана на JavaScript!";  
setTimeout("window.status = 'Ваш браузер підтримує скрипти '",2000); </SCRIPT>
```

сторінка у момент завантаження виглядатиме так само, як і в попередньому прикладі, однак через дві секунди вміст рядка стану зміниться на фразу "Ваш браузер підтримує скрипти", оскільки функція setTimeout(), здійснює дію, визначену усередині неї, з деякою затримкою (яка обчислюється в мілісекундах, тому значення 2000 відповідає затримці в 2 секунди).

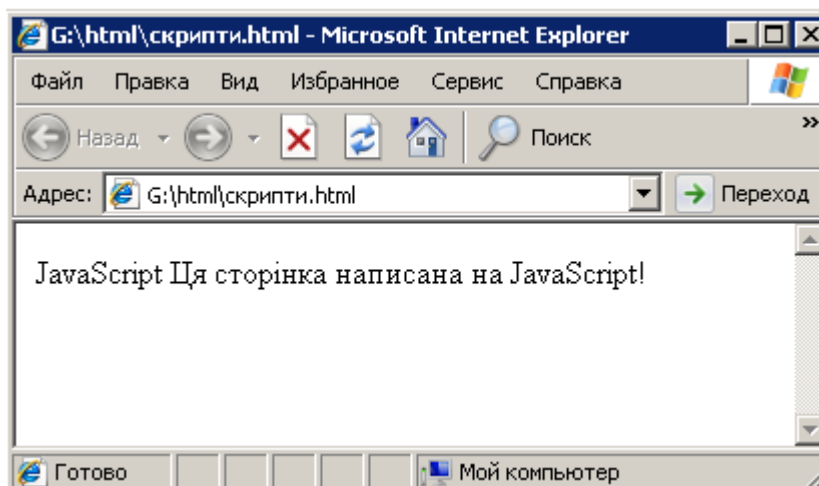


Рис. 1. Просте використання JavaScript.

У JavaScript (як і в HTML) потрібно уважно стежити, щоб всі лапки у потрібному місці закривалися. Наприклад, не можна використовувати наступний запис:

```
setTimeout("window.status = ""Ваш браузер підтримує скрипти """, 2000);
```

оскільки тоді браузер "вирішив би", що рядок закінчився після знаку рівності, а далі, не зустрівши коми, поскаржився на помилку. Якщо браузер зустрине помилку в коді JavaScript, то буде видано повідомлення про помилку, причому сценарій не буде виконаний.

Також необхідно дотримувати регістр символів, оскільки прописні і рядкові букви розрізняються. Якщо замість setTimeout() написати SetTimeout() або settimeout(), то буде видано повідомлення про помилку.



самим задається розмірність і тип елементів). Наприклад: `m3 = new Array ("фізика", "інформатика", "математика");` – масив з трьох строкових елементів.

Звернення до елементів масиву відбувається шляхом вказівки імені масиву і індексу елемента. При цьому вказівка індексу елемента більшого, ніж вказано при оголошенні масиву веде до збільшення розмірності масиву, а не до помилки, як в інших мовах програмування.

JavaScript надає декілька цікавих методів роботи з масивами:

- метод **Reverse** міняє порядок елементів масиву на зворотний: `mas1.reverse;`
- метод **sort** – сортує елементи масиву: `mas3 = mas1.sort;`
- метод **join** – об'єднує елементи масиву в рядок, використовуючи при цьому вказаний роздільник: `str = m3.join(" -> ");`

### **Введення/виведення в JavaScript**

**Виведення** даних на екран у JavaScript може відбуватися різними способами. Найбільш простим є застосування оператора **Alert()**. Результатом виконання оператора `Alert` є вивід на екран діалогового вікна, вмістом якого є значення виразу аргументу. Поки користувач не натисне кнопку ОК, робота сценарію не буде продовжена.

Вивід даних за допомогою вікна оператора `Alert` зручно використовувати для контролю значень змінних на тому або іншому етапі виконання програми, тобто при відладці. Приклад скрипта, в якому вивід на екран здійснюється за допомогою вікна оператора `Alert` (рис. 2):

```
<SCRIPT>
mas2 = new Array ("фізика", "інформатика", "математика" );
dd = mas2.join ("->");
alert(dd);
</SCRIPT>
```

Слід зазначити, що функція `Alert` є методом об'єкту `Window`, який описує поточне вікно браузера. Тому синтаксично коректніше викликати цю функцію так: `window.alert ("Текст повідомлення")`.

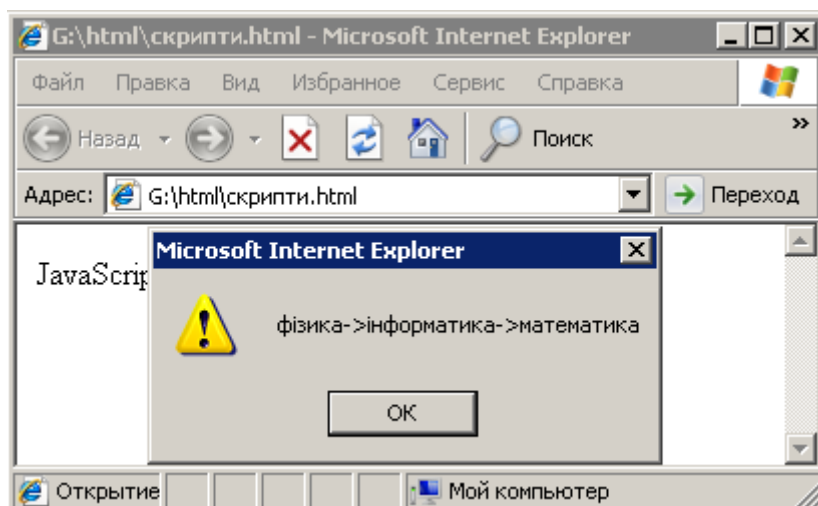


Рис. 2. Діалогове вікно оператора Alert.

Ще одним способом виведення інформації на екран є вивід в тіло документа, що організовується за допомогою оператора **write**. Вираз, який є аргументом оператора виводу, може містити будь-яку рядкову константу, а також містити різні теги HTML. Наприклад (рис. 3):

```
<HTML>
<BODY bgcolor= "deepskyblue" >
<H3> Виведення тексту засобами HTML </H3><BR>
<SCRIPT>
document.writeln (" <FONT color=#FFFF00> " +
"<center> Виведення тексту за допомогою JavaScript </center> <BR> ");
document.writeln ("<center> <img src=2.jpg > /center>");
</SCRIPT> </BODY> </HTML>
```

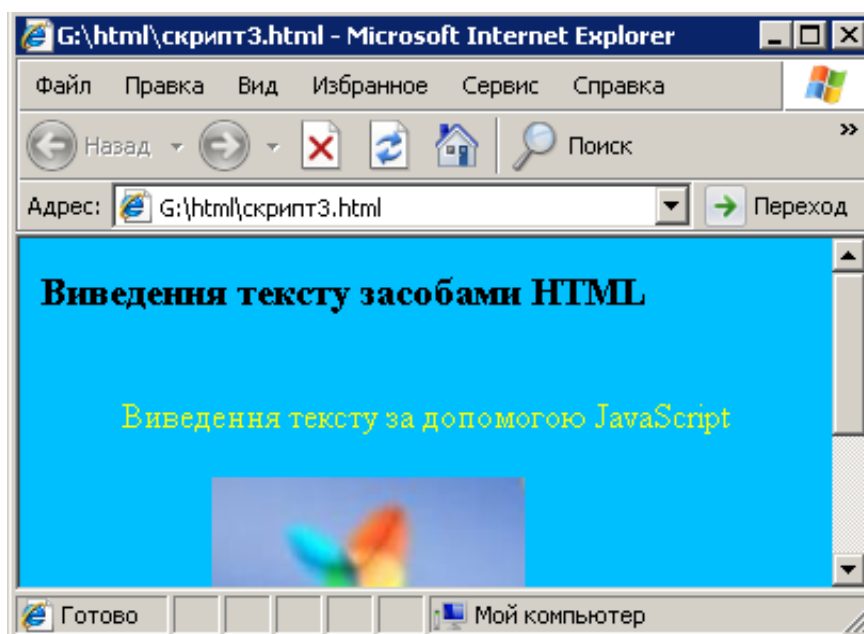


Рис. 3. Вивід на екран засобами JavaScript.

Оператор `document.writeln()` відрізняється від оператора

document.write() тим, що переносить позицію виводу на новий рядок. На основі цього способу виведення у вікно браузера можна побудувати багато корисних скриптів. Розглянемо, наприклад, скрипт, який при завантаженні сторінки виводить на екран малюнок (фотографію), вибраний випадковим чином з існуючого масиву.

Створюємо масив, в який поміщаємо варіанти малюнків. Щоб уникнути частих повторів зображень, необхідно достатньо багато варіантів, тому об'єм масиву буде достатньо великим, порівняним з розміром типового html-документа. Тому оголошення масиву варто розмістити в окремому файлі скрипта, наприклад, f1.js. У такому разі достатньо підключити до документа створений файл:

```
<SCRIPT src=" f1.js"> </SCRIPT>
```

і вставити наступний скрипт у потрібному місці:

```
<SCRIPT language=javascript!> var i = 0;  
i = Math, round (Math, random ())* (a. length – lib-document, write ("Фото: ... " + a[ i ]);  
</SCRIPT>
```

Так можна отримати цікавий різновид фотоальбому. Замість зображень можна виводити іншу потрібну інформацію (наприклад текст).

Розглянемо оператори **введення**. JavaScript надає декілька способів організації введення. Перший – використання методу Prompt об'єкту window. Він має такий синтаксис:  $d = \text{window.prompt}(\text{"Текст повідомлення"}, \text{"Значення за замовчуванням"})$ .

У результаті виконання такої команди на екрані з'явиться вікно запиту, де користувачеві буде виведено запрошення на введення, що міститься у виразі "Текст повідомлення". Після введення значення привласнюється змінній  $d$ . Якщо користувач не ввів нічого, то  $d$  буде привласнене значення виразу «Значення за замовчуванням».

Приклад введення за допомогою оператора Prompt (рис. 4):

```
<SCRIPT>  
var d = " – мова сценаріїв";  
c=prompt("Введіть потрібне слово", "JavaScript ");  
s =d+c;  
alert (s) ;  
</SCRIPT>
```

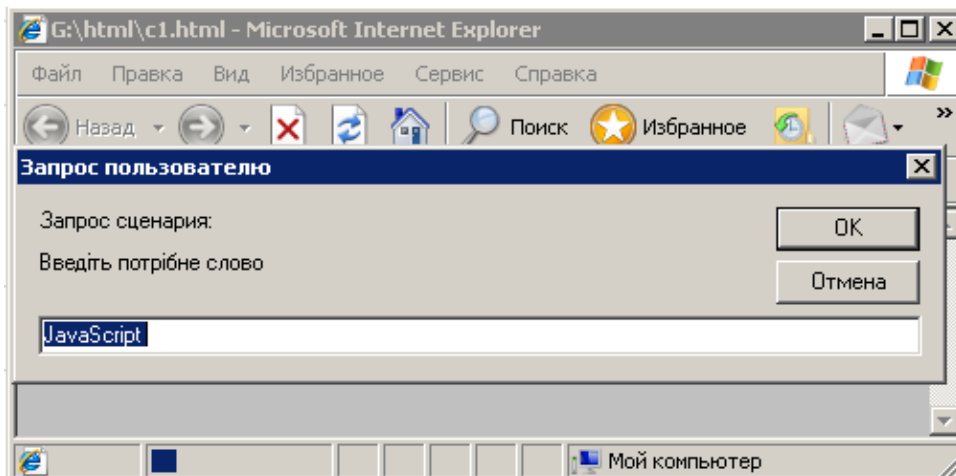


Рис. 4. Введення даних в JavaScript.

Введення значень булевого типу найзручніше здійснювати за допомогою оператора `window.confirm`, що має синтаксис:

```
b=confirm ("Питання");
```

У результаті виконання такої команди на екрані з'явиться вікно з поставленим питанням і двома кнопками. Залежно від натискання користувачем тієї або іншої кнопки змінна *b* набуде або значення `true`, або `false` (кнопка `Отмена`) (рис. 5).

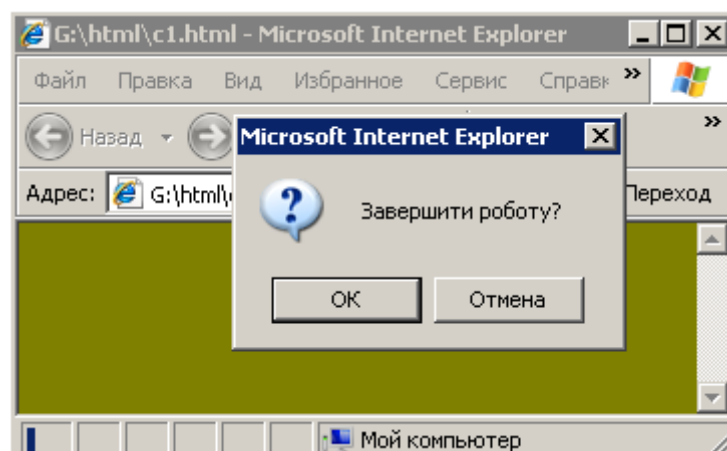


Рис.5. Вікно оператора Confirm.

Розглянемо для прикладу використання скриптів для створення сторінки, керованої за допомогою миші. JavaScript дає можливість змінити колір звичайного тексту при наведенні на нього миші. Для цього у лапки потрібно помістити ту дію, яку він повинен виконати – тобто змінити колір цього блоку `<DIV>`, наприклад, на червоний. Доступ до властивостей поточного елемента здійснюється за допомогою ключового слова `this`:

```
<DIV onmouseover="this.style.colors'red"> Цей текст змінить колір при наведенні вказівника миші! </DIV>
```



Якщо тепер відкрити цю сторінку в браузері, то при наведенні покажчика миші на другий рядок тексту, колір рядка дійсно зміниться на червоний. Щоб при відведенні покажчика миші з рядка колір змінився назад на чорний, потрібно додати обробник подій, що реагує на відведення показника – onmouseout (див. додаток 3):

```
<DIV onMouseOver="this.style.color='red' " onMouseOut="this.style.color='black'">
```

Цей текст змінить колір при наведенні вказівника миші! </DIV>

Тепер при наведенні вказівника миші на цей рядок, його колір зміниться на червоний, а при відведенні вказівника – назад на чорний. Використання скриптів значно спрощує цей процес. Вводимо такий сценарій:

```
<SCRIPT LANGUAGE="JavaScript">  
function change() { document, all. text1. style. color="red";} .function change2()  
{ document.all.text1.style.color="black"; } //-->  
</SCRIPT>
```

Після цього у тілі документа зазначаємо лише імена використаних функцій:

```
<DIV ID="text1" onMouseOver="change()" onMouseOut="change2()">Цей текст  
змінить колір при наведенні вказівника миші! </DIV>
```

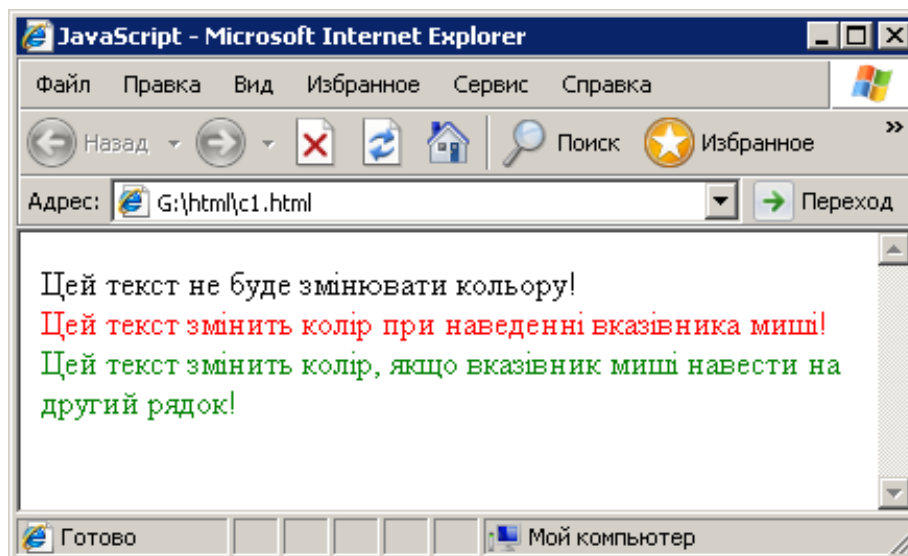


Рис. 6. Приклад сторінки, керованої за допомогою миші.

Таким способом можна змінити не тільки той блок тексту, на який наведено покажчик миші, але і будь-які інші елементи, якщо тільки привласнити їм ім'я. На рисунку 6 наведено приклад документа, в якому при наведенні миші на другий рядок її колір змінюватиметься на червоний, а колір третього рядка – на зелений.

## ЗАВДАННЯ

1. Створити новий html-документ та доповнити його текстовою інформацією на довільну тему.
2. Створити на основі розглянутих прикладів власний скрипт для веб-сторінки з використанням змінних та масивів.
3. Використайте різні способи виведення даних в документі: за допомогою функції `window.alert()` та оператора `document.write()`.
4. Передбачити у створюваному скрипті введення будь-яких даних, використовуючи для цього оператори введення `window.confirm` або `window.prompt`, синтаксис яких наведений у теоретичних відомостях.
5. Створити сценарій, за допомогою якого у даному html-документі буде реалізована можливість зміни кольору деякого тестового блоку при наведенні на нього курсору мишки.
6. Застосувати створений скрипт до html-документа.
7. Переглянути створену веб-сторінку за допомогою будь-якого веб-оглядача. Перевірити роботу скриптів. Перегляньте html-код створеної сторінки за допомогою команди Вигляд ⇒ Перегляд HTML-коду.
8. Продемонструвати створений документ викладачу та оформити звіт з виконання лабораторної роботи.

## КОНТРОЛЬНІ ЗАПИТАННЯ

1. Що таке сценарії?
2. З якою метою використовують мову JavaScript?
3. Які є способи впровадження створених скриптів у документ?
4. Для чого використовується тег `<NOSCRIPT>`?
5. Яке призначення функції `setTimeout()`?
6. Що буде результатом включення у документ сценарію `window.status="JavaScript!"`?
7. Як на мові JavaScript оголошуються змінні?
8. Для чого використовується спеціальний символ `\ n`?
9. За допомогою якого оператора задаються масиви?
10. Які методи роботи з масивами Вам відомі?

11. За допомогою яких операторів відбувається виведення даних на екран?
12. Основне призначення оператора Promt?
13. Для чого використовують оператор window.confirm? Який він має синтаксис ?
14. Яке призначення функції function change()?
15. Як засобами JavaScript вивести на екран картинку ?

## Лабораторна робота № 15

### ТЕМА. Основи JavaScript. Розгалуження, цикли, форми

**МЕТА:** ознайомитися з особливостями створення html-сторінок з динамічно змінним вмістом. Використовуючи набуті знання та вміння, навчитися працювати із розгалуженнями, циклами та формами в JavaScript.

#### ХІД РОБОТИ

1. Створити новий html-документ.
2. Створити свій власний скрипт з використанням розгалужень та циклів.
3. Застосувати створений скрипт до html-документу.
4. Створити html-документ, що містить просту форму та застосувати до неї скрипт для перевірки введеної інформації.

#### ТЕОРЕТИЧНІ ВІДОМОСТІ

JavaScript має певні оператори, що реалізують основні алгоритми управління потоком обчислень: оператор розгалуження, оператор циклу з кінцевим числом повторень, оператор циклу while. Розглянемо їх детальніше.

**Оператор розгалуження** реалізує вибір тієї або іншої послідовності дій залежно від умови (умовний оператор):

```
if (умова) {Послідовність 1}
else{Послідовність 2}
```

Вираз "умова" – це, як правило, комбінація операторів відношення або результат дії оператора confirm.

**Оператор циклу** з кінцевим числом повторень повторює певну послідовність дій задане число разів: for ("вираз", "умова",

"операція"){Послідовність дій}.

Тут необхідне використання цілочисельної змінної, початковим значенням якої буде "вираз". Цикл повторюватиметься, поки буде істинною "умова". При цьому при кожній ітерації циклу над змінною-лічильником виконуватиметься дія "операція". Наприклад: `for (i=0; i<5; i++){s+=mas[i];}` – обчислення суми елементів деякого масиву.

**Цикл while** повторює деяку послідовність дій до тих пір, поки виконується деяка умова. Синтаксис циклу наступний: `while ("умова"){Послідовність дій}`

З огляду на те, що перевірка умови передує виконанню послідовності дій, цикл `while` отримав назву циклу з передумовою. Для примусового виходу з циклів використовується команда **break**. Для переходу до наступної ітерації циклу (дострокового виконання послідовності дій усередині циклу) використовується оператор **continue**.

Розглянемо достатньо простий приклад, який демонструє, використання оператора циклу `for`. Припустимо, нам потрібно вивести на екран таблицю множення. Звичайно, можна вручну написати кожен її рядок:

```
<TABLE>
<TR>
<TD>2&times;2=4</TD>
<TD>3&times;2=6</TD>
<TD>4&times;2=8</TD>
.....
```

Це спосіб достатньо довгий і нудний, крім того, легко можна допустити випадкову помилку. Використаємо для цього засоби JavaScript. Тег `<TABLE>` можна винести за межі сценарію. Далі потрібно сформулювати деяку кількість рядків (традиційно рівна кількості варіантів другого множника, який зазвичай приймає значення від 2 до 10). Можна цей множник занести в змінну і написати:

```
for (i=2; i<=10; i++) { document.write ("<TR>"); document.write ("</TR>") ; }
```

Вираз у дужках після оператора циклу `for` означає наступне: початкове значення змінної – 2; умова виконання циклу – змінна повинна бути менше або рівна 10; на кожному кроці змінна збільшується на 1 (позначення “++” означає збільшення на одиницю, а “–” зменшення на одиницю.)

Якщо зараз запустити цей цикл, то у вікні браузера нічого не відобразиться, оскільки немає тегів елементів таблиці ( `<TD>` ). Оскільки у кожному рядку повинно бути стільки осередків, скільки значень приймає

перший множник (занесемо його в змінну "j"), організуємо між записом тегів <TR> і </TR> ще один цикл:

```
for (j=2; j<10; j++) document.write("<TD>"+j+"&times;"+i+"="+i*j+"</TD>") ;
```

Тут умовою виходу з циклу є  $j < 10$ , а не  $j \leq 10$ , оскільки традиційно перший множник в таблиці множення не перевищує 9.

Зверніть увагу на рядок методу document.write. Тут у лапках вказане те, що потрібно безпосередньо помістити на сторінку. Змінні вказані поза лапками, щоб в документ записувалися їхнє значення. Весь рядок з'єднується знаками "+".

Щоб отримати результат множення змінної i змінну j, використаний запис "i\*j". Значення i\*j в нашому прикладі поміщене в дужки, щоб виключити можливість неправильної інтерпретації браузером, хоча це не обов'язково.

Тепер оголошуємо змінні i та j на початку сценарію, використовуючи ключове слово var: var i,j; Крім того, для покращення сприйняття, можна "розграфити" таблицю, відокремивши стовпці один від одного. Для цього потрібно використати атрибут RULES= тега <TABLE>:

```
<TABLE BORDER="1" CELLSPACING="0" CELLPADDING="2" RULES="cols">
```

Остаточно маємо (рис. 1):

```
<HTML>
<HEAD>
<TITLE>Таблиця множення</TITLE>
</HEAD>
<BODY>
<TABLE BORDER='3'ALIGN=CENTER BGCOLOR='aliceblue' RULES="cols">
<SCRIPT >
var i, j ; for (i=2; i<=10; i++) { document.write ("<TR>"); for (j=2; j<10; j++)
document.write("<TD>"+j+"&times;"+i+"="+i*j+"</TD>");
document.write ("</TR>") ; }
</SCRIPT>
</TABLE>
</BODY> </HTML>
```

Замість кропіткого запису вручну всіх елементів таблиці, ми обмежилися шістьма рядками коду. Крім того, якщо виникне потреба розширити таблицю, наприклад організувати виведення значень аж до 20x20, це можна зробити, просто замінивши два числа в коді .

### **Керування відображенням документа**

Засоби контролю за відображенням сторінок в JavaScript доповнені командами, що дають змогу керувати вікнами браузера та їхнім вмістом.

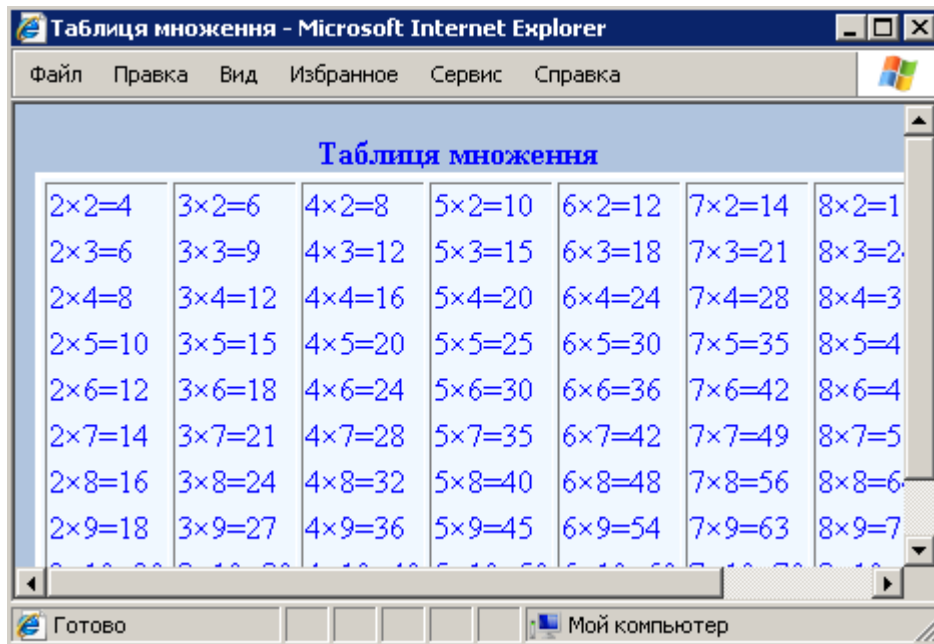


Рис.1. Використання оператора циклу for.

- Команда `document. clear` очищає поточне вікно. Це може стати в нагоді при виведенні даних у тіло документа.
- Команда `window.close` закриває поточне вікно браузера.
- Команда `window.open (url, місцеположення, атрибути)` відкриває документ за адресою `url` у вікні або фреймі, заданому у виразі "місцеположення". Параметри вікна описані у виразі "Атрибути". Ця команда широко використовується для відкриття супутніх основному вікну вікон з рекламою.
- Команда `location.href = "url"` проводить перехід у поточному вікні на нову адресу, вказану в `url`. Властивість `location` можуть мати об'єкти `window`, а також фрейми. Це дає змогу здійснювати завантаження документа у потрібному вікні або фреймі.
- Команда `setTimeout("window.local ion.href = 'url'", час)` завантажує вказану сторінку через деякий час.

### Призначені для користувача функції

При створенні динамічних веб-сторінок нерідко доводиться проробляти одні і ті ж дії з різними елементами, тому для скорочення розміру коду можна записати їх один раз на самому початку, поставивши попереду ключове слово `function`, тобто, визначивши їх як функцію.

Функція повинна приймати якийсь аргумент і повертати результат. При визначенні функції аргумент потрібно вказати в дужках після її назви. Наприклад:

function Name ("список аргументів"){Послідовність операторів}

Ім'ям функції може бути будь-яке не зарезервоване поєднання символів. «Тіло» функції повинне бути поміщене у фігурні дужки.

Для повернення результату функції використовується оператор return. При цьому відбувається припинення подальшого виконання коду функції, і управління передається операторові, наступному за точкою виклику.

```
function truncate (a)
{return a – a % 1;}
```

Ця функція відсікає дробову частину числа. Викликається вона так:

```
var a = 7.3;
b = Trunc( a);
alert(b);
```

Призначені для користувача функції можуть розташовуватися або в скрипті, який здійснює їхній виклик, або в іншому скрипті, включеному у цю ж сторінку. Часто зустрічається ситуація, коли призначені для користувача функції, що часто вживаються, розташовуються в окремому файлі скрипта, який підключається до потрібних сторінок. Ці функції мають різноманітне застосування. Наприклад, використання наступного скрипта приводить до закривання поточного вікна при натисканні кнопки «Закрити» (рис. 2):

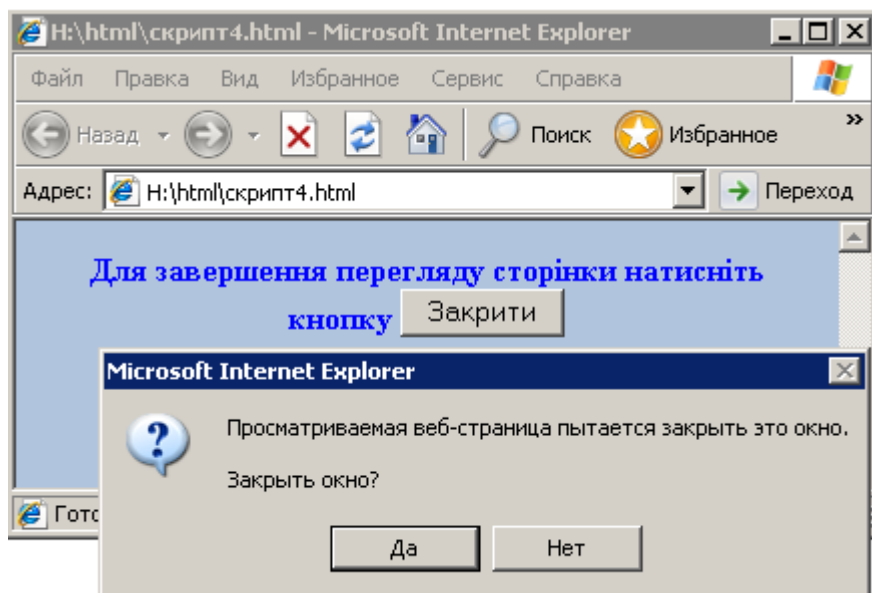


Рис. 2. Використання команди function kill().

```
<SCRIPT> function kill(){window.close ();} </SCRIPT>
```

```
Для завершення перегляду сторінки натисніть кнопку <input type=button name = text  
value = "Закрити" onClick=kill(>
```

За допомогою функції function perehod\_a () можна організувати

відкриття двох документів при натисканні на одне посилання.

Розглянемо наступний приклад: створення на веб-сторінці зображення, що з'являється поступово, збільшуючись в розмірах. Спочатку просто додамо малюнок на сторінку:

```
<IMG SRC="3.jpg" WIDTH="151" HEIGHT="10" BORDER="0" ALT="малюнок">
```

Реальний фізичний розмір малюнка є зменшений. Тепер звернемося до нього з сценарію JavaScript. Доступ до всіх малюнків на сторінці можна отримати, написавши метод `document.images` і вказавши у квадратних дужках номер малюнка на сторінці.

Нумерація картинок починається з нуля. Зазначений малюнок, як перший на сторінці, матиме номер 0. Отже, для звернення до нього з сценарію слід використовувати метод `document.images[0]`. Якби ми помістили на сторінку ще одну картинку, то до цієї другої картинки можна було б звернутися `document.images[1]`.

Напишемо функцію, яка перевірятиме, чи не досягнув малюнок потрібного розміру, і якщо ні, то збільшувати його ширину і висоту. Перевірити розміри можна так:

```
if (document.images[0].width<451)
```

Якщо після номера малюнку поставити крапку і написати будь-який наявний його html-атрибут, то можна зі сценарію дізнатися його значення і, за необхідності, змінити його: `document.images[0].width+=2;` `document.images[0].height+=2;` – висота і ширина малюнка змінюється на 2 пікселя.

Щоб “зациклити” функцію, можна викликати її ж після деякої затримки. Наприклад, якщо наша функція називатиметься `sizer()`, то останнім її рядком може бути: `setTimeout("sizer()", 20);`

Такий виклик функції з самої себе називається рекурсивним викликом і досить часто уживається в JavaScript. Тепер достатньо один раз викликати цю функцію, і далі вона весь час викликатиме себе сама.

Остаточо маємо (рис. 3):

```
<HTML> <HEAD>
<TITLE>Картинка, що змінює розміри</TITLE>
<SCRIPT LANGUAGE"JavaScript">
function sizer () { if (document.images[0].width<451)
{ document.images[0].width+=2; document.images[0].height+=2;
setTimeout("sizer()", 20); } } //--> </SCRIPT>
</HEAD>
<BODY>
```



```

<IMG SRC="3.jpg" WIDTH="151" HEIGHT="10" BORDER="0"ALT="малюнок">
<SCRIPT LANGUAGE="JavaScript"> sizer();
</SCRIPT>
</BODY> </HTML>

```

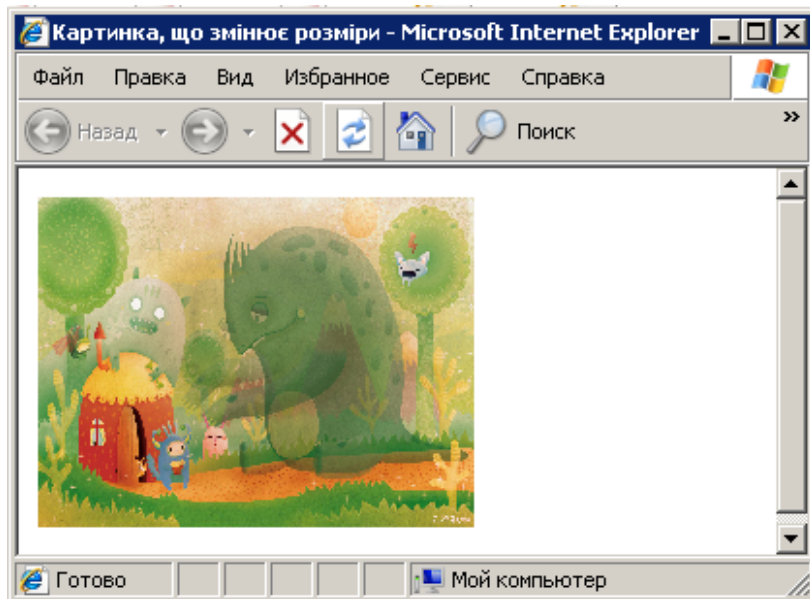


Рис. 3. Приклад малюнку, що поступово змінює розміри.

У результаті ми отримуємо картинку, що поступово збільшується в розмірах. Так само можна керувати ще деякими елементами: посиланнями і елементами тега `<AREA>` через метод `document.links`, формами через метод `document.forms` і `document.forms[номер_форми].elements` і т.і. (див. додаток 4).

### Використання форм в JavaScript

Форми є невід'ємною частиною дизайну багатьох сторінок. Часто при заповненні форми користувачем виникає потреба перевірки коректності введення інформації. У JavaScript, на відміну від HTML, є засоби контролю вмісту форм.

Кожен об'єкт `document` має властивість `document.forms`, яке є масивом об'єктів типу `form`. Доступ до властивостей тієї або іншої форми можна отримати або з цього масиву (`document.forms[0]` – перша форма документа), або безпосередньо на ім'я форм, яке задане в описі (`document.forma_1` – звернення до форми з ім'ям `forma_1`).

У таблиці наведені основні властивості об'єкту `form`.

Властивість	Опис
<b>Elements [ ]</b>	Містить масив елементів, використовуваних у формі
<b>action</b>	Встановлює дію, яку потрібно провести при передачі даних форми на сервер

<b>target</b>	Містить ім'я вікна, в якому повинен відобразитися результат заповнення форми
<b>encoding</b>	Містить спосіб кодування даних форми. Відповідає атрибуту ENCTYPE елементу FORM
<b>method</b>	Визначає спосіб передачі даних форми на сервер
<b>submit ( )</b>	Відправляє форму на сервер

Елементи введення у формі також є об'єктами. Інформація про них зберігається в масиві `elements []` об'єкту форм. У той же час, можливо діставати доступ до властивостей полів безпосередньо за допомогою вказівки їхніх імен (`forma_l.edit_l` – звернення до поля `edit_l` форми `forma_l`). Розглянемо також основні властивості об'єкту `element`:

Властивість	Опис
<b>checked</b>	Містить стан елемента
<b>defaultChecked</b>	Містить стан елемента за замовчуванням
<b>defaultValue</b>	Містить значення елемента за замовчуванням
<b>form</b>	Містить об'єкт форми, в якій знаходиться елемент
<b>length</b>	Містить кількість елементів у списку
<b>name</b>	Містить ім'я елемента
<b>selectedIndex</b>	Визначає номер вибраного за замовчуванням елемента списку
<b>value</b>	Задає значення елемента

та основні події:

Подія	Опис
<b>onBlur</b>	Генерується при втраті елементом фокусу введення
<b>onChange</b>	Генерується при зміні елемента
<b>onClick</b>	Генерується при клацанні мишею на елементі
<b>onFocus</b>	Генерується при отриманні елементом фокусу введення
<b>onSelect</b>	Генерується при виборі вмісту елемента

Розглянемо, як можна перевірити коректність заповнення полів форми перед відправкою даних на перевірку. Маємо деяку анкету, в яку треба ввести ініціали, e-mail користувача та вік користувача. Поле введення прізвища повинне бути не порожнім. Коректність введення e-mail визначатимемо наявністю у вмісті поля введення знаку @. Вік має бути в межах від 18 до 65 років. (рис. 4).

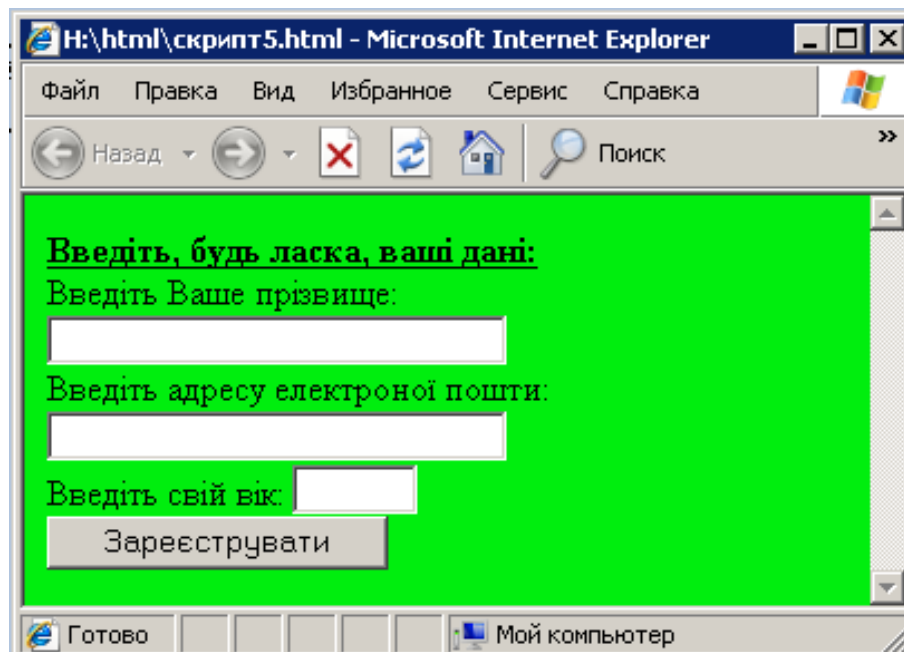


Рис .4. Приклад форми, яка потребує перевірки перед відправкою.

```
<FORM NAME="Form1" onSubmit=usercheck(>
<B><U>Введіть, будь ласка, ваші дані:</U></B><br>
Введіть Ваше прізвище:<BR>
  <INPUT TYPE="text" NAME="textfield" SIZE="30"><BR>
Введіть адресу електронної пошти:<BR>
  <INPUT TYPE="text" NAME="e-mail" SIZE="30" ><BR>
Введіть свій вік:
<INPUT TYPE="text" NAME="p1" size= 5 ><BR>
  <INPUT TYPE="submit" VALUE="Зареєструвати">
</FORM>
```

Щоб перевірити коректність заповнення даної форми потрібно використати наступний скрипт:

```
<SCRIPT>
function usercheck()
{em = document.form1.e-mail.value;
u1=(document.form1.textfield.value != "");
u2=(em.indexOf("@") !=-!);
u3=((document.form1.p1.value > 18)&(document.form1.p1.value <65));}
if ((u1)&(u2)&(u3))
{document.form1.submit;
Else {alert("Помилка ! \n Перевірте правильність введення !")}
}
</SCRIPT>
```

## ЗАВДАННЯ

1. Створити новий html-документ та доповнити його текстовою інформацією на довільну тему.

2. Створити на основі розглянутих прикладів власний скрипт для веб-сторінки з використанням циклів та розгалужень.
3. Доповнити створений документ малюнком. Використовуючи метод `document.images[ ]` та рекурсію, реалізувати наступну зміну розмірів малюнка: картинка поступово збільшується до деякого граничного розміру, після чого починає аналогічно зменшуватися. Для того, щоб організувати зменшення розмірів малюнка в прикладі наведеному у теоретичних відомостях, проробіть такі дії:
  - поміняйте граничні розміри зображення
  - замініть фрагмент коду  

```
document.images[0].width+=2; document.images[0].height+=2
```

на наступний:  

```
document.images[0].width-=2; document.images[0].height-=2.
```
4. Застосувати створений скрипт до html-документу.
5. Створити html-документ з назвою «форма», що містить просту форму та застосувати до неї скрипт для перевірки введеної інформації. При виконанні цього завдання можна використати відповідний сценарій, наведений у теоретичних відомостях.
6. Перевірити роботу створених документів. Перегляньте html-код створеної сторінки за допомогою команди Вигляд ⇒ Перегляд HTML-кода.
7. Продемонструвати створені документи викладачу та внести за потреби рекомендовані корективи.
8. Оформити звіт з виконання лабораторної роботи.

## КОНТРОЛЬНІ ЗАПИТАННЯ

1. Які оператори використовують в JavaScript для реалізації алгоритмів керування потоком обчислень ?
2. З якою метою використовується оператор циклу з кінцевим числом повторень?
3. Як діє оператор розгалуження? Який синтаксис він має?
4. Як діє оператор циклу `while`? Який синтаксис він має?
5. Яка команда використовується для примусового виходу з циклів?
6. Яке призначення має оператор `continue`?
7. Що є результатом виконання команди `document. Clear`?

8. З якою метою використовується команда `window.close`?
9. Яке призначення має команда `Window.open ( )`?
10. Для чого використовується оператор `return`? Які особливості він має?
11. Як завантажити потрібну сторінку через деякий, заздалегідь визначений час?
12. Які функції потрібно використовувати, щоб створити на веб-сторінці зображення, що поступово зменшується у розмірах?
13. Які властивості має об'єкт `form`? Яке призначення вони мають?
14. Як, використовуючи сценарії JavaScript, перевірити коректність заповнення користувачем поля «e-mail»?
15. Як, використовуючи сценарії JavaScript, перевірити коректність заповнення користувачем поля «пароль», що має містити від 5 до 12 символів?

### ДОДАТОК 1. Перетворення IP-адреси з двійкового формату в десятиковий

У двійковому форматі кожному біту в октеті відповідає певне десяткове число. Максимальне десяткове значення октету дорівнює 255 (бере участь кожен біт). Кожен октет перетвориться у число незалежно від інших.

Біт, встановлений в 0, завжди відповідає нульовому значенню. Біт, встановлений в 1, може бути перетворений у десяткове число. Молодший біт октету є десятковим числом 1, а старший – 128. Максимальне значення октету (255) досягається, коли кожен його біт дорівнює 1.



У таблиці показано, як біти одного октету перетворюються у десяткове число.

Двійковий запис	Значення бітів	Десяткове число
00000000	0	0
00000001	1	1
00000011	1+2	3
00000111	1+2+4	7
00001111	1+2+4+8	15
00011111	1+2+4+8+16	31
00111111	1+2+4+8+16+32	63
01111111	1+2+4+8+16+32+64	127
11111111	1+2+4+8+16+32+64+128	255

### ДОДАТОК 2. Оформлення математичних виразів для публікацій в мережі Internet

До найпростіших математичних виразів належать ті, написання яких не потребує використання додаткових (спеціальних) тегів мови (X)HTML. Розглянемо формулу скороченого множення:  $a^2 - b^2 = (a - b)(a + b)$ . HTML-розмітка цієї формули така:  $a^{<sup>2</sup> - b^{<sup>2</sup> = (a - b)(a + b)$ .

Для запису символів, яких немає на клавіатурі (грецькі літери, спеціальні математичні знаки тощо) слід використовувати &-символи. У таблиці 1 подано найбільш вживані математичні символи.

Таблиця 1. Позначення деяких спеціальних математичних символів в HTML.

Вигляд	Ім'я	Опис символу
°	&deg;	градус
<sup>1</sup>	&sup1;	верхній індекс – 'один'
<sup>2</sup>	&sup2;	верхній індекс – 'два' (квадрат)
½	&frac12;	дріб – одна друга
¾	&frac34;	дріб – три четвертих
–	&minus;	мінус
±	&plusmn;	плюс-мінус
×	&times;	знак множення
÷	&divide;	знак ділення
<i>f</i>	&fnof;	знак функції
...	&hellip;	три крапки
'	&prime;	одичний штрих
⇒	&rArr;	подвійна стрілка вправо
⇐	&lArr;	подвійна стрілка вліво-вправо
∂	&part;	частковий диференціал
∈	&isin;	належить
∏	&prod;	послідовність множників
∑	&sum;	сума послідовності
∠	&ang;	кут
√	&radic;	квадратний корінь
∫	&int;	інтеграл
∞	&infin;	нескінченість
∩	&cap;	перетин
∪	&cup;	об'єднання
≈	&asymp;	приблизно рівне
≠	&ne;	нерівне
<	&lt;	знак 'менше'
>	&gt;	знак 'більше'
≤	&le;	менше рівне
≥	&ge;	більше рівне
⊥	&perp;	перпендикулярно

Нижче наведено таблицю з літерами грецького алфавіту. Для їхнього запису в HTML-кодi слід використовувати шаблон `&англійська_назва_літери;`. Якщо назва літери грецького алфавіту починається з малої літери, то у браузері (веб-оглядачі) буде відображатися мала грецька літера, а якщо велика літера, то – велика (наприклад,  $\Gamma$ ,  $\gamma$  – `&Gamma`; та `&gamma`; відповідно).

Таблиця 2. Грецький алфавіт

Друковані літери	Укр. назва літери	Англ. назва літери
Α α	альфа	alpha
Β β	бета	beta
Γ γ	гама	gamma
Δ δ	дельта	delta
Ε ε	епсилон	epsilon
Ζ ζ	дзета	zeta
Η η	ета	eta
Θ θ	тета	theta
Ι ι	йота	iota
Κ κ	капа	kappa
Λ λ	лямбда	lambda
Μ μ	мю	mu
Ν ν	ню	nu
Ξ ξ	ксі	xi
Ο ο	омікрон	omicron
Π π	пі	pi
Ρ ρ	ро	rho
Σ σ	сігма	sigma
Τ τ	тау	tau
Υ υ	іпсілон	upsilon
Φ φ	фі	phi
Χ χ	хі	chi
Ψ ψ	псі	psi
Ω ω	омега	omega

Зауважимо, що записувати ці слова треба строго дотримуючись реєстру літер. Бо, наприклад, запис `&ALPHA;` не дасть бажаного результату.



### ДОДАТОК 3. Основні події мови JavaScript

Обробник події	Підтримуючі елементи	Опис
onAbort	IMG	Переривання завантаження зображення
onBlur	A, AREA, BUTTON, INPUT, LABEL, SELECT, TEXTAREA	Втрата фокусу поточним елементом. Виникає при кліканні мишкою зовні елемента або натисканні клавіші табуляції
onChange	INPUT, SELECT, TEXTAREA	Зміна значень елементів форми.
onClick	Майже всі елементи	Одинарний клік (натиснута і відпущена кнопка мишки)
onDbClick	Майже всі елементи	Подвійний клік
onError	IMG, WINDOW	Виникнення помилки при виконанні сценарію
onFocus	A, AREA, BUTTON, INPUT, LABEL, SELECT, TEXTAREA	Отримання елементом фокусу (клік мишкою на елементі або натискання клавіші табуляції )
onKeyDown	Майже всі елементи	Натиснута клавіша на клавіатурі
onKeyPress	Майже всі елементи	Натиснута і відпущена клавіша на клавіатурі
onKeyUp	Майже всі елементи	Відпущена клавіша на клавіатурі
onLoad	BODY, FRAMESET	Завершене завантаження елемента
onMouseDown	Майже всі елементи	Натиснута кнопка мишки в межах поточного документа
onMouseMove	Майже всі елементи	Переміщення курсору мишки в межах поточного документа
onMouseOut	Майже всі елементи	Курсор мишки виведено за межі поточного елемента
onMouseOver	Майже всі елементи	Курсор мишки наведено на поточний елемент
onMouseUp	Майже всі елементи	Відпущена кнопка мишки в межах поточного елемента
onMove	WINDOW	Переміщення вікна
onReset	FORM	Скидання даних форми ( клік по кнопці <code>&lt;input type="reset"&gt;</code> )
onResize	WINDOW	Зміна розмірів вікна
onSelect	INPUT, TEXTAREA	Виділення тексту в поточному елементі
onSubmit	FORM	Відправка даних форми (клік по кнопці <code>&lt;input type="submit"&gt;</code> )

## ДОДАТОК 4. Приклади програм на JavaScript

**Приклад 1.** При наведенні вказівника миші на текст, змінюється фон, на якому цей текст розташований.

```
<p onmouseover="this.style.backgroundColor='blue'"  
onmouseout="this.style.backgroundColor='#EEEEEE'">
```

Наведіть, будь ласка, курсор миші. </p>

**Приклад 2.** Створення кнопки, що дає змогу повернутися на декілька позицій назад (кнопка "Back" / "Назад").

На одну позицію:

```
<a href="javascript:history.back()">Назад</a>
```

На декілька позицій одразу:

```
<a href="javascript:history.go(-3)">Назад на три позиції</a>
```

**Приклад 3.** Створення годинника в рядку стану.

```
<script language="JavaScript">  
function clock() {  
  today=new Date();  
  clock_status=today.getHours()+":"+today.getMinutes()+":"+today.getSeconds();  
  status=clock_status;  
  setTimeout("clock()",100);}  
</script>
```

Тіло документа починаємо з тега <body onLoad="clock()">

**Приклад 4.** Зміна вигляду курсора (вказівника миші) за допомогою JavaScript.

```
<script>  
function changeCursor(obj,i)  
{  
  t=i;  
  if(i==0)t="DEFAULT"  
  else if(i==1)t="CROSSHAIR"  
  else if(i==2)t="HAND"  
  else if(i==3)t="MOVE"  
  else if(i==4)t="TEXT"  
  else if(i==5)t="WAIT"  
  obj.style.cursor=t;  
}  
</script>
```

Припустимо нам потрібно змінити вигляд курсору над лінком:

```
<a OnMouseOver="changeCursor(this,2)" href="filename.htm">Лінк</a>
```

**Приклад 5.** Створення наступного ефекту: при наведенні вказівника миші на комірку таблиці змінюється колір фону цієї комірки.

```
<table>  
<tr>  
<td onmouseover="this.bgColor='#eeeeee'" onmouseout="this.bgColor='#ffffff'">
```

```
Комірка_1
</td>
<td>
Комірка_2
</td>
</tr>
</table>
```

**Приклад 6.** Створення малюнка, який з'являється, поступово зменшуючись у розмірах.

```
<SCRIPT LANGUAGE"JavaScript">
function sizer ()
{ if (document.images[0].width<451)
{ document.images[0].width+=-1;
document.images[0].height+=-1;
setTimeout("sizer()",100); }
</SCRIPT>
```

У тіло документа включаємо тег:

```
<IMG SRC="a.jpg " WIDTH="151" HEIGHT=""10" BORDER="0" ALT="a">
```

Та наступний скрипт

```
<SCRIPT > sizer(); </SCRIPT>
```

**Приклад 7.** Кожного місяця в один і той же день, наприклад, першого числа, відбувається зміна малюнка (кожному місяцю відповідає свій малюнок).

```
<body>

<script language="JavaScript">
d=new Date();
document.monthImg.src=d.getMonth()+".gif";
</script>
```

## **ЗРАЗОК ОФОРМЛЕННЯ ЗВІТУ**

### **«Основи Інтернет-технологій»**

Звіт до лабораторної роботи №2

на тему: «Веб-браузери»

студента групи ІН-14 Петренка Петра

**Мета:** набуття навичок налаштування властивостей браузерів, перегляду, збереження і друку веб-сторінок, створення закладок для швидкого завантаження відвідуваних веб-сайтів

#### **Хід роботи**

1. Налаштував режим автономної роботи браузера Internet Explorer.  
Зробив порожню сторінку домашньою.

##### ***Порядок виконання завдання 1:***

1. Завантажив браузер: Пуск (Start) ⇒ Програми (Programs) ⇒ Internet Explorer;
2. Для встановлення автономної роботи браузера після його запуску вибрав команди меню Файл ⇒ Працювати автономно;
3. Встановив нову домашню сторінку у меню Сервіс: Властивості браузера ⇒ закладка Загальні ⇒ клацнув під полем адреси домашньої сторінки на кнопці З порожньої ⇒ Ок.

2. Створив канал з погодою на робочому столі комп'ютера.

##### ***Порядок виконання завдання 2:***

1. Клікнув правою клавішею миші на вільному місці робочого столу комп'ютера. У контекстному меню вибрав команду Властивості ⇒ Екран ⇒ закладка Робочий стіл ⇒ Налаштування робочого столу ⇒ Елементи робочого столу ⇒ закладка Веб ⇒ Створити ⇒ Галерея.
2. На комп'ютер завантажилася веб-сторінка фірми Microsoft, на якій запропоновано створити низку каналів: розділ Weather Map from MSNBC ⇒ Додати на робочий стіл. Канал створено.
3. Створив на робочому столі комп'ютера ярлик для веб-порталу Yahoo!

**Висновок:....**

## ЛІТЕРАТУРА

1. Гусев В.С. Internet: учеба, работа, полезные ресурсы. Краткое руководство. – К. : Діалектика. – 2005. – 256 с.
2. Інформатика. Комп'ютерна техніка. Комп'ютерні технології. Посіб./ за ред. О.І. Пушкаря – К. : Видавничий центр "Академія", 2001. – 696 с.
3. Хелеби С, Мак-Ферсон Д. Принципи маршрутизації в Internet. – М. : Вільямс. – 2001. – 448 с.
4. Морзе Н.В. Основи комп'ютерних мереж та Інтернету – К. : Видавнича група ВНУ. – 2006. – 256 с.
5. Гаевський А.Ю., Романовський В.А. Самоучитель по созданию Web-страниц: HTML, JavaScript и Dynamic HTML. – К:А.С.К., – 2002. – 472 с.
6. Дідре Х. HTML и XHTML. – К. : Знання. – 2005. – 266 с.
7. Ландэ Д.В. Поиск знаний в Internet. Профессиональная работа. – К. : Діалектика, – 2005. – 272 с.
8. Мак-Федрис П. Использование JavaScript. Вильямс. – 2002. – 895 с.
9. Пігур-Пастернак. Інтернет-служби: Лабораторний практикум з курсу “Локальні мережі та Інтернет” для студентів вищих навчальних закладів – Дрогобич: РВВ ДДПУ ім. Івана Франка. – 2009. – 81 с.
10. Фок Б. Internet с самого начала: пер. с англ. – СПб: Питер. – 1995. – 56 с.
11. Гольський В.Б., Шаклеїна І.О. Нові інформаційно-комунікативні технології та ТЗН: методичні рекомендації до виконання лабораторних робіт. – Дрогобич: ДДПУ, – 2010. – 121 с.
12. Кузнецов И. Анимация для интернета: краткий курс. – СПб. : Питер, – 2001. – 288 с.
13. Гончаров А. Самоучитель HTML. – Питер. – 2002. – 242 с.
14. Джексон Крейфорд Тиге. DHTML и CSS для Интернет. – NT Press. Москва. – 2005. – 520 с.
15. Дунаев В. Самоучитель Java Csript. – Питер. – 2005. – 395 с.
16. Олифер В., Олифер А. Новые технологии и оборудование IP-сетей. – СПб.: БХВ. – 2000. – 512 с.
17. Кирсанов Д. Веб-дизайн – СПб. : “Символ-Плюс”, – 2001. – 376 с.
18. Самойлов Е.Э. Web-дизайн для начинающих: Практическое руководство. – М. : «Триумф», – 2009. – 192 с.
19. Колесников А. Internet для пользователя – К., – 2000. – 304 с.

20. Крамер Э. HTML: наглядный курс Web-дизайна – М. : “Вильямс”, – 2001. – 304 с.
21. Олифер В. Г., Олифер Н. А. Компьютерные сети: принципы, технологии, протоколы. – СПб. : Питер, – 2001. – 256 с.
22. Дронов В.А. Macromedia DreamWeaver 8. – СПб.: «БХВ – Петербург», – 2006. – 319 с.
23. Глинський Я.М. Практикум з інформатики. навчальний посібник, 4-е доп. вид. – Львів, ДЕОЛ, – 2001. – 224 с.
24. Толстых В.К. Современные Internet-технологии: конспект (слайды) лекций. – Режим доступа: <http://www.tolstykh.com/edu>
25. «Основы интернет», онлайн-підручник – Режим доступа: <http://psbatishev.narod.ru/internet>

**Для нотаток**

Навчальне видання

**Ірина Шаклеїна, Надія Ших**

# **ОСНОВИ ІНТЕРНЕТ-ТЕХНОЛОГІЙ**

**Методичні рекомендації  
до виконання лабораторних робіт**

**Редакційно-видавничий відділ  
Дрогобицького державного педагогічного університету  
імені Івана Франка**

**Головний редактор  
*Ірина Невмержицька***

**Редактор  
*Леся Грабинська***

**Технічний редактор  
*Наталя Кізима***

**Коректор  
*Наталя Кізима***

Здано до набору 10.04.2012 р. Підписано до друку 11.05.2012 р. Формат 60x90/16.  
Папір офсетний. Гарнітура Arial. Наклад 330 прим. Ум. друк. арк. 9,05. Зам 190.

Редакційно-видавничий відділ Дрогобицького державного педагогічного університету імені Івана Франка (Свідоцтво про внесення суб'єкта видавничої справи до державного реєстру видавців, виготівників і розповсюджувачів видавничої продукції ДК №2155 від 12.04.2005 р.) 82100, Дрогобич, вул. І.Франка, 24, к. 42, тел. 2-23-78.