

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДНІПРОВСЬКИЙ ДЕРЖАВНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

МЕТОДИЧНІ ВКАЗІВКИ
до лабораторних робіт
з дисципліни «Програмування Інтернет»

для здобувачів вищої освіти першого (бакалаврського) рівня зі спеціальності
121 «Інженерія програмного забезпечення»
за освітньо-професійною програмою «Інженерія програмного забезпечення»

ЗАТВЕРДЖЕНО:
редакційно-видавничою секцією
науково-методичної ради ДДТУ
16.05. 2019 р., протокол № 5

Кам'янське
2019

*Розповсюдження і тиражування без офіційного дозволу Дніпровського державного технічного університету **заборонено.***

Методичні вказівки до лабораторних робіт з дисципліни «Програмування Інтернет» для здобувачів вищої освіти першого (бакалаврського) рівня зі спеціальності 121 «Інженерія програмного забезпечення»/

Укладач О.О. Шумейко. – Кам'янське: ДДТУ, 2019. – 76с.

Укладач:

Доктор технічних наук, професор
Шумейко О.О.

Рецензент:

завідувач кафедри прикладної
математики
доктор технічних наук, професор
Самохвалов С.Є.

Затверджено на засіданні кафедри програмного забезпечення систем
(протокол 4 від 03 квітня 2019 р.)

Коротка анотація видання. Містить короткі теоретичні відомості та практичні задачі щодо основних конструкцій мови гіпертекстової розмітки HTML, а також каскадних таблиць стилю, викладені основні принципи верстки Web-сторінок, що є необхідною умовою для роботи з інтернет додатками.

Зміст

Основні вимоги з охорони праці	4
ВСТУП	5
Лабораторна робота №1	7
Створення Web-сторінки за допомогою HTM	
Лабораторна робота №2	13
Застосування каскадних таблиць стилів для оформлення Web-сторінки	
Лабораторна робота №3	17
Форматування Web-сторінки- списки, таблиці , фрейми.	
Лабораторна робота №4	23
Створення макету Web-сторінки	
Лабораторна робота №5	38
Робота з HTML-формами.	
Лабораторна робота №6	42
Фреймворк Bootstrap.	
РЕКОМЕНДОВАНА ЛІТЕРАТУРА	71
Додаток А	72
Додаток Б	73

Основні вимоги з охорони праці

При виконанні лабораторних робіт студент повинен дотримуватися нижче зазначених правил техніки безпеки:

1. Починати роботу тільки після проходження інструктажу з техніки безпеки.
2. Не вмикати/вимикати самостійно рубильник в комп'ютерному залі.
3. При виконанні лабораторних робіт за комп'ютером може знаходитись не більше ніж два студента.
4. Не вводити власних паролів або іншу конфіденційну
5. інформацію у комп'ютер
6. Не змінювати налаштувань комп'ютера.
7. Не встановлювати та не видаляти жодних програм без дозволу викладача.
8. Закінчивши виконання лабораторної роботи студент повинен скопіювати результати на носій , закрити всі програми, з якими працював без збереження своїх результатів і повідомити викладача про закінчення роботи
9. Зареєструватись у журналі користувачів комп'ютерного залу

ВСТУП

Дисципліна «Програмування Інтернет» для здобувачів вищої освіти першого (бакалаврського) рівня зі спеціальності 121 «Інженерія програмного забезпечення», викладається упродовж одного навчального семестру і складається з двох змістових модулів (частин).

На даний момент Web-дизайн і Web-програмування є модним напрямком. Взагалі-то кажучи, написати Web-сторінку зовсім не складно. Набагато складніше зробити цю сторіночку гарною, щоб на неї заходили і радили зайти друзям. Для цього перш за все, має бути цікаве наповнення сторінки.

З іншого боку, як каже народна мудрість, - по одягу зустрічають ... Іншими словами, від оформлення теж багато що залежить.

Перш за все, найбільш важлива інформація повинна бути на початку сторінки. Назва сайту має бути на самому видному місці - в лівому верхньому кутку і бути присутнім на кожній сторінці у вигляді посилання на головну сторінку.

Сторінка не повинна бути перевантажена графікою і завантажуваними об'єктами. Для того, щоб відповідь системи сприймалася як миттєва, граничне значення реагування системи повинне не перевищувати 0.1 сек. Щоб сприйняття системи не порушувалось і хід думки клієнта не переключався, час реагування повинен не перевищувати 1 сек. Граничний час, протягом якого користувач сфокусований на діалозі не більше 10 сек. Ці показники вироблені Joint Computer Conference в 1968 році і за ті роки, які пройшли з моменту прийняття цих рекомендацій, вони стали лише більш актуальними.

Всьому цьому присвячені дисципліни, орієнтовані на роботу з Інтернет, першою з яких є «Програмування Інтернет».

1. Загальні положення

Метою лабораторних робіт з дисципліни «Програмування Інтернет» є набуття базових знань, вмінь та навичок, необхідних для створення WEB-додатків, знайомство з технологією створення клієнтських програмних засобів.

Методичні вказівки складено у відповідності до робочої програми з дисципліни «Програмування Інтернет» для здобувачів вищої освіти першого (бакалаврського) рівня зі спеціальності 121 «Інженерія програмного забезпечення».

Лабораторні роботи виконується на лабораторних заняттях з дисципліни. Підготовка до лабораторних занять здійснюється студентами в часи самостійної роботи.

Для студентів очної форми навчання виконання всіх лабораторних робіт даного навчального видання є обов'язковим. Студенти заочної форми

навчання виконують лише роботи, передбачені робочою програмою навчальної дисципліни. Перелік і кількість запропонованих до розв'язку задач визначається викладачем, який веде лабораторні роботи, відповідно до робочої програми дисципліни.

Усі лабораторні роботи складено за єдиним планом – на початку кожної роботи наведено її тему та мету, далі викладено короткі теоретичні відомості щодо тематики роботи з розглядом прикладів розв'язку типових задач, після чого містяться завдання та запитання для самоперевірки.

Наприкінці методичних вказівок наведено порядок виконання і захисту робіт та перелік посилань на використану укладачами видання та рекомендовану для виконання роботи літературу.

Після виконання кожної роботи студенти складають звіт, котрий захищають. За результатами виконання та захисту роботи виставляються бали за спеціальною шкалою оцінювання, наведеною у робочій програмі. Бали, отримані за окремі роботи, формують загальну суму балів за практикум, яка враховується у підсумкову оцінку за модуль та семестр.

Лабораторна робота №1

Створення Web-сторінки за допомогою HTML

Мета роботи: ознайомлення з синтаксисом мови HTML та її особливостями.

Теоретичні основи.

HTML (HyperText Markup Language) мова опису структури сторінки, що надає можливість створення посилань на пов'язані сторінки. Інструкції форматування в HTML називаються тегами (tags). Іноді теги містять параметри, наприклад,

 Наприклад,

Більшість тегів представлено у вигляді пар

<tag> ... </tag>

Теги не чутливі до регістру, але за традицією їх прийнято писати прописними літерами.

Умовно теги можна класифікувати за трьома категоріями:

- теги коментарів, а також теги оголошення, що інформують браузер про те, що документ є HTML документ.
- теги заголовків.
- теги тіла HTML документа.

Теги коментарів

Текст між парним тегом <COMMENT> або всередині тегу <! ...> є коментарем і не виводяться у вікні браузера.

Парний тег <HTML> замикає вміст всієї сторінки (за винятком, можливо, коментарів).

Теги заголовка

Парний тег <HEAD> дозволяє створити назву <TITLE>, співвідношення між декількома документами <BASE>, метод посилки повідомлень програмам пошуку <META>.

Парний тег <TITLE> визначає текст назви документа, який відображається у вікні браузера.

Тег <BASE> містить повний URL документа. Цей тег має один обов'язковий атрибут href задає повний URL документа - Web-адресу (Uniform Resource Locators).

Тег <META> надає інформацію про документ для браузера, пошукових машин і інших додатків. Синтаксис:

```
<META
  content = description
  http-equiv = response
  name = text
  title = text
  url = url>
```

Якщо значення http-equiv одно "refresh", то документ перевантажується через значення атрибута content число секунд. Значення url містить адреса

документа, що завантажується через значення content секунд. Наприклад, content = "2; url = example.html".

Для полегшення роботи пошукової машині можна створити два тега <META>, у першому задати значення name = "keywords", а іншого name = "description" (короткий опис).

<META name = "keywords" content = "Лінія. Площина">

<META name = "description" content = "Матеріали по графіку">

Для поновлення сторінки не з кешу, а з сервера слід використовувати конструкцію

<META http-equiv = "expires" content = "0">

Тег тіла документа

Відразу після заголовка йде тіло документа обмежене парним тегом <BODY>.

Синтаксис

<BODY

alink = color

background = url

bgcolor = color

bgproperties = fixed

bottommargin = pixels

class = classname

id = value

lang = language

language = javascript | jscript | vbscript

leftmargin = pixels

link = color

rightmargin = pixels

scroll = yes | no

style = properties

text = color

title = string

topmargin = n

vlink = color

event = script

>

Опишемо атрибути, які найбільш часто використовуються.

- background це URL графічного файлу, який використовується в якості фонового зображення.
- bgproperties значення fixed говорить про те, що фоновий малюнок не прокручується.
- bgcolor колір фону в шістнадцятирічному RGB (наприклад, olive- # 808000).
- text колір тексту в шістнадцятирічному RGB.

- leftmargin, rightmargin відстань лівим і правим краями тексту і вікна броузера.
- link, alink, vlink колір для гіперпосилань в шістнадцятирічному RGB для зумовленої назви, активного і вже відвіданого гіперпосилань.

Теги форматування документа

Теги заголовків різного рівня від <H1> до <H6> є парними тегами з наступними атрибутами

Синтаксис

```
<H1
  align = center | left | right
  class = classname
  id = value
  lang = language
  language = javascript | jscript | vbscript
  style = properties
  title = string
  event = script>
```

З них найбільш популярним є тег вирівнювання align.

<P> не обов'язково парний тег обмежує параграф. Перед параграфом автоматично ставиться порожній рядок. Єдиним атрибутом є тег вирівнювання align.

<CENTER> парний тег центрування.

 Кінець рядка.

<HR> горизонтальна лінія.

Синтаксис

```
<HR
  align = center | left | right
  color = color
  id = value
  lang = language
  language = javascript | jscript | vbscript
  noshade
  size = n
  style = properties
  title = string
  width = n
  event = script
```

>

Тут noshade лінія без тіні, size товщина лінії в пікселях і width довжина лінії або в пікселях або у відсотках по відношенню до ширини екрану.

<HR align = center color = green size = 3 width = 50%>

Далі парні теги задають тип шрифту - напівжирний, <I> -курсив, <U> - підкреслений, <STRIKE> -закреслений, <TT> - фіксованої ширини. <BIG> і <SMALL> -великий і менший розміри, <SUB>, <SUP> - верхній і нижній

індекси, <CITE> - для виділення цитат, <Code> і <PRE> для виведення невеликих і великих фрагментів коду.

Парний тег <ADDRESS> служить для ідентифікації автора та останнього оновлення сторінки. Розташовується або на початку, або наприкінці документа.

```
<ADDRESS>
  Created by POVT
  Last Modified 01.01.01
</ADDRESS>
```

Парний тег задає шрифт.

```
<FONT
  class = classname
  color = color
  face = font
  id = value
  lang = language
  language = javascript | jscript | vbscript
  size = n
  style = properties
  title = string
  event = script>
```

Основні атрибути size розмір шрифту від 1 до 7 (найбільший). Якщо число використовується зі знаком + або -, то це означає відносний розмір порівняно з основним шрифтом певним тегом <BASEFONT>. Face ім'я шрифту, наприклад, "Times New Roman". <BASEFONT> задає стандартний шрифт (список атрибутів той же).

Наприклад,

```
<B> <Font size = 5 color = green> Текст </Font> </B>
```

Теги зображень

Для вставки зображення служить парний тег

Синтаксис

```
<IMG
  align = absbottom | absmiddle | baseline | bottom | left | middle | right |
  texttop | top
  alt = text
  border = n
  class = classname
  datafld = colname
  datasrc = # id
  dynsrc = url
  height = n
  hspace = n
  id = value
  ismap
```

```

lang = language
language = javascript | jscript | vbscript
loop = n
lowscr = url
name = name
src = url
style = properties
title = string
usemap = url
vspace = n
width = n
event = script>

```

Опишемо найбільш часто використовувані атрибути.

- alt -текст альтернативного опису зображення.
- border ціле число, що визначає товщину рамки зображення.
- Controls відображає керуючі елементи відеокліпів.
- Dynsrc URL-адреса відеокліпу.
- Height, width висота і ширина зображення. При цьому зображення стискується або розтягується під розміри виділеної області.
- Hspace, vspace товщина бічних сторін і верхньої та нижньої частин рамок зображення. Ці атрибути подібні border, але при цьому рамка невидима.
- Ismap вказує, що зображення є картою і відповідним чином реагує на клацання миші.
- Loop- число повторень відеокліпу, якщо одно -1 або infinite, то повторюється по циклу.
- Src - URL зображення.
- Start -подія, при якому почне програватися відеокліп.
- Usemap -фрагменти ідентифікатора для сайту клієнта з картою образу.

Покажчиком гіперпосилання може бути слово, група слів, зображення. Покажчик гіперпосилань створюються за допомогою парного тега <A> (anchor-якір).

Синтаксис

```

<A
    accesskey = key
    class = classname
    datafld = colname
    datasrc = # id
    dynsrc = url
    href = url
    id = value
    lang = language
    language = javascript | jscript | vbscript

```

```

methods = http_methods
name = name
rel = "stylesheet"
style = properties
tabindex = n
target = window_name | _blank | _parent | _self | _top
title = string
urn = urn
event = script>

```

Опишемо найбільш часто вживані атрибути.

Href URL адреса - href = "type url", де в якості параметра type використовується

- http: для переходу на іншу сторінку,
- file: для завантаження і відкриття файлу,
- mailto: для відправки повідомлення електронною поштою за вказаною адресою.
- Target визначає місце завантаження на яке вказує посилання.

Параметри

_blank відкриває нове вікно браузера,
 _parent завантаження документа проводиться в батьківський фрейм поточного вікна,
 _self завантаження в поточне вікно або фрейм,
 _top завантаження в усі вікно браузера, а не у фрейм.

Атрибут NAME створює закладки в будь-якому місці документа, перехід на який здійснюється по імені.

Зручність закладок в тому, що можна на кожній сторінці на початку створити зміст. Це дуже полегшує навігацію і дозволяє без прокрутки відразу перейти на потрібний рядок або заголовок. Для того щоб створити цю закладку треба вставити, наприклад, рядок: Для використання цих закладок на посиланні вказується назва закладки після значка # (грати) мітка Якщо закладка розташовується в іншому файлі, можна використати посилання

 Закладка в документі Test

В якості посилань можна використовувати зображення. Для цього всередині <A> ... потрібно вказати .

Наприклад,

Small sky

```

<A Href=http://www.sky.com/> <img src = Images / sky.gif alt = "Small sky"
width = 100 height = 100> </A>

```

У цьому випадку зображення буде обводиться рамкою в 2 пікселя, і якщо це заважає, то слід для зображень використовувати атрибут border = 0.

Наведемо приклад посилання на електронну пошту

```

<A href="mailto:mail@ukr.net"> посилання </A>

```

Завдання.

Створити дві html-сторінки, на першій ваше ім'я синім кольором на жовтому фоні, а на другій – прізвище жовтим кольором на синьому, при чому сторінки змінюють одна одну кожну одну секунду.

Контрольні питання.

1. На які категорії діляться теги html?
2. Як відрізняються заголовки <h6> від <h7>?
3. Який тег використовується, щоб зберегти первинне форматування?
4. Чим відрізняється атрибут href від src?

Лабораторна робота №2

Застосування каскадних таблиць стилів для оформлення Web-сторінки

Мета роботи: використання стилів для оформлення зовнішнього вигляду документу.

Теоретичні основи.**CSS -Cascading Style Sheets**

Стилі за замовчуванням наведені на <http://www.w3.org/TR/CSS/>. Стилі можна міняти, використовуючи наступні методи.

Методи визначення таблиці стилів у документі HTML

- Вкладення (inline)
- Вбудовування (embedding)
- Зв'язування (linking)
- Імпорт (import)

color:red; - декларація

background:#aaaaaa; - властивість: значення

Вкладення

<p style="color:red; background:#aaaaaa">Метод вкладення стилів</p>

Вбудовування

В <head> вкладаємо опис типу - селектор{правило}

<style type="text/css">

p{color:red;

background:#aaaaaa"

```
}
```

```
</style>
```

Зв'язування

```
<link rel="stylesheet" type="text/css" href="style.css">
```

Селектори тегу й класу

```
тег ( div{color:red}
```

```
тег + клас ( div.green{color:green}
```

```
клас ( .blue{color:blue}
```

```
<div> Звичайний div </div>
```

```
<div class="green"> div з класом green </div>
```

```
<p class="green"> p з класом green </p>
```

```
<p class="blue"> p з класом blue </p>
```

```
<div class="blue"> div з класом blue </div>
```

Інші селектори

```
id ( #back{color:red}
```

```
тег + id ( div#back{color:black} особливо безглуздо, тому що id унікальний
```

```
контекстні ( div b{color:green}
```

```
селектори ( td td td{color:blue}
```

```
<div id="back">div c id=back</div>
```

```
<div> div c <b>контекстним</b> селектором</div>
```

```
<table><tr><td>
```

```
<table><tr><td>
```

```
<table><tr><td>
```

Третій рівень вкладеності

```
</td></tr></table>
```

```
</td></tr></table>
```

```
</td></tr></table>
```

Селектор атрибута

```
p[align] {color:red}
```

```
input[type="radio"] {color:green} - Строга рівність (не як слово, а як рядок)
```

```
p[title~="Дніпро"] {color:green} слово цілком
```

```
p[title*="Дніпро"] {color:green} наявний підрядок (Дніпродзержинськ)
```

Універсальний селектор

```
*{color:black}
```

Сусідні селектори

```
b + i {color:red}
```

Дочірні селектори

```
div > p {color:red}
```

Псевдокласи та псевдоелементи

```
:link
```

```
:visited
```

```
:active
```

```
:hover
```

```
:focus
```

```
:first-child
```

```
:first-line
```

```
: first-letter
```

```
:after
```

```
:before
```

```
<head>
```

```
<style type ="text/css">
```

```
:first-child{color:#0f0;}
```

```
</style>
```

```
a:link{color:blue}
```

```
a: visited{color:gold}
```

```
a:active {color:red}
```

```
a:hover {color:green}
```

```
input:focus {color:red}
```

```
p:first-child{color:blue}
```

```
p :first-line{color:red}
```

```
p:first-letter{color:green}
```

```
p:after{content:"new"}
```

```
p:before{content:"att."}
```

```
</head>
```

```
<body>
```

```
<p>Перший параграф у body</p>
```

```
<p>Другий параграф у body</p>
```

```
<div>
```

```
<p>Перший параграф у div</p>
```

```
<p>Перший параграф у div</p>
```

```
</div>
```

```
<p>Це <a href="#"> гіперпосилання </a></p>
```

```
</body>
```

Комбінації селекторів

Контекстний селектор - предки-нащадки (приклад- у strong серед предків є div)

```
div strong{color:#0af;}
```

Дочірній селектор - батько-дитина (приклад - в strong батько div)

```
div > strong{color:#0af;}
```

Сусідній селектор - найближчий сусід ліворуч (приклад - для p, у якого сусід ліворуч (раніше по контексту) h1)

```
h1 + p{color:#00f;}
```

Сусідній селектор - будь-який сусід ліворуч

```
h1 ~ p{color:#00f;}
```

Угруповання селекторів

```
h1,h2,h3 {color:#f00;}
```

Пріоритети правил CSS

	I	II	III
!important	#id	.class	tads
style		:pseudo-class	:pseudo-elements
		[attributes	

Сполучення селекторів може бути довільним. У даному прикладі їхнє сполучення має однакова вага. При всіх рівному - хто останніх, той і прав:


```
div.text #in{color:#f00;}

#top p.news{color:#0f0;}

<div id ="top" class ="text">

    <p id ="in" class = "news">Блок</p>

</div>
```

Значення !important використовується для підвищення пріоритетності

```
p{

    color:$aaa !important;

}
```

У ВСІХ параграфів буде встановлений сірий колір.

Іншим методом буде використання атрибута style

```
<p style="color:#abc;">Блок</p>
```

Завдання.

Використовуючи CSS написати html-сторінку, на якій використані можливості псевдокласів та псевдоелементів, при чому

- Реалізувати стилі по класам, тегам та по id.
- Реалізувати різне відображення курсору миші на різних об'єктах.
- Зробити власний курсор миші.

Контрольні питання.

1. Які існують методи підключення стилів?
2. Яка різниця між класом та псевдокласом?
3. В чому різниця методами задання стилів html та CSS?
4. Як встановити пріоритет стилю?

Лабораторна робота №3

Форматування Web-сторінки- списки, таблиці , фрейми.

Мета роботи: Реалізація та форматування списків, таблиць та фреймів.

Теоретичні основи.

Списки

Списки і таблиці є дуже зручним засобом форматування тексту. Розглянемо спочатку списки - маркований і нумерований.

Парний тег (list item) визначає елемент у маркірованому і нумерованому списках. Використовується разом з парними тегами і .

Синтаксис

```
<LI
  class = classname
  id = value
  lang = language
  language = javascript | jscript | vbscript
  style = properties
  title = string
  type = 1 | A | a | I | i
  value = value
  event = script>
```

Атрибут type задає тип нумерації впорядкованого списку і тип маркера.

Допустимі значення

- A (прописні літери),
- a (рядкові),
- I (прописні римські),
- i (рядкові римські),
- 1 (арабські числа).

Value - початковий номер.

Текст поміщений між парними тегами ... </ UL> утворює маркований список, а між ... </ OL> - нумерований. Елементи списку лежать всередині тегів ... </ LI>. Синтаксис аналогічний.

- Європа
 - i. Німеччина
 - ii. Україна

Азія

- iii. Китай
- iv. Індія

 Європа

<LI type = i> Німеччина </ LI>

<LI type = i> Україна </ LI>

</ OL>

</ LI>

 Азія

<LI value = 3 type = i> Китай </ LI>

<LI type = i> Індія </ LI>

```
</ OL>
</ LI>
</ UL>
```

Оформлення точкою можна змінити задаючи інше відображення атрибутом type, який може приймати значення disc, square і circle.

- Європа
 1. Німеччина
 2. Україна
- Азія
 - c. Китай
 - d. Індія

```
<UL>
<LI type = circle> Європа
  <OL>
    <LI type = 1> Німеччина </ LI>
    <LI type = 1> Україна </ LI>
  </ OL>
</ LI>
<LI type = square> Азія
  <OL>
    <LI value = 3 type = a> Китай </ LI>
    <LI type = a> Індія </ LI>
  </ OL>
</ LI>
</ UL>
```

Списки визначень відрізняються від попередніх за структурою. Парний тег <DL> створює список визначень, тег (не обов'язково парний) <DT> позначає певний термін, а тег (теж не обов'язково парний) <DD> позначає визначення терміну заданого тегом <DT>.

В якості маркера може бути використано графічне зображення. Для цього достатньо для тега визначити значення властивості list-style-image атрибуту style.

Таблиці

Парний тег таблиць називається <TABLE>. Параметрами цього тега задається загальний вигляд таблиці, її колір, товщина рамки і багато іншого. Усередині цього тега розташовуються наступні парні теги - <TR> рядок таблиці, <TH> комірка заголовку. У ній вирівнюються по центру і виділений напівжирним шрифтом, <TD> комірка, <CAPTION> заголовок таблиці.

Синтаксис

```
<Table
  align = center | left | right
  background = url
```

```

bgcolor = color
border = n
bordercolor = color
bordercolordark = color
bordercolorlight = color
cellpadding = n
cellspacing = n
class = classname
cols = n
datapagesize = n
datasrc = # id
frame = above | below | border | box | insides | lhs | rhs | void | vsides
height = n
id = value
lang = language
language = javascript | jscript | vbscript
rules = all | cols | groups | none | rows
style = properties
title = string
width = n
event = script>

```

Align як і завжди цей параметр задає вирівнювання таблиці (за умовчанням - по лівому краю).

Bgcolor, bordercolor, bordercolordark, bordercolorlight задають кольори фону, рамки, тіні рамки і її підсвічування.

border використовується для вказівки ширини межі таблиці в пікселях. За замовчуванням має розмір рівний 0, і при цьому межі таблиці не відображаються, тобто виходить прозора таблиця.

width, height задає ширину і висоту таблиці в пікселях або відсотках. Як правило браузер підбирає розмір таблиці таким чином, щоб там все гарненько вмістилося, але якщо потрібен конкретний розмір, то можна його вказати в пікселях або у відсотках від вільного простору.

Кожну клітинку браузер обводить своєї власної рамкою, і параметр cellspacing задає ширину простору між ними.

Параметр cellpadding теж задає ширину простору, але тільки вже між рамкою комірки таблиці і її змістом всередині.

- Cols -число стовпців,
- frame - задає ті частини рамки таблиці, які будуть відображатися на екрані. Допустимі значення
- border (відображається вся рамка. Використовується за замовчуванням),
- void рамка без зовнішніх меж,
- above -немає рамки в нижній частині таблиці,
- below немає рамки у верхній частині таблиці,

- hside - немає рамки по краях,
- lhs - немає рамки з правого краю,
- rhs - немає рамки з лівого краю,
- vsides немає рамки в підставі і у верхній частині таблиці,
- box - відображається вся рамка.
- rules - задає внутрішні розділові лінії. Може приймати значення
 - none - лінії не відображаються,
 - groups - відображаються горизонтальні лінії між групами елементів,
 - rows - відображаються горизонтальні лінії,
 - cols - відображаються вертикальні лінії,
 - all - відображаються всі лінії.

Синтаксис

```
<Caption
  align = bottom | center | left | right | top
  class = classname
  id = value
  lang = language
  language = javascript | jscript | vbscript
  style = properties
  title = string
  event = script>
```

Приклад таблиці:

a _{1,1}	a _{1,2}
a _{2,1}	a _{2,2}

<Caption align = bottom>

Приклад таблиці:

```
</ Caption>
<table border = 1>
<tr> <td> a <sub> 1,1 </ sub> <td> a <sub> 1,2 </ sub>
<tr> <td> a <sub> 2,1 </ sub> <td> a <sub> 2,2 </ sub>
</ table>
```

Тег <tr> має кілька атрибутів.

Атрибут align задає вирівнювання всередині всіх елементів таблиці. Якщо, наприклад вказати align = center, то вміст кожної комірки буде вирівняний по центру.

Атрибут valign також задає вирівнювання, але вже по вертикалі. Може приймати значення top - вирівнювання по верху і middle - по центру, ну і bottom по низу.

Атрибути bgcolor і background задають колір або зображення фону одного рядка таблиці.

Стовпчики таблиці більш багаті своїми параметрами. У них входять всі параметри рядків таблиці описані вище, ну і ще парочка: `rowspan` об'єднує вказану кількість осередків в одну по вертикалі і `colspan` поєднує комірки по горизонталі. Природно, `width` і `height` задають рекомендовані розміри комірки по горизонталі і вертикалі. Якщо комірка порожня, тобто `<td> </td>` то вона не відображається браузером, хоча для цього треба користуватися `<td colspan=`

Приклад:

a	b	c	d	e
f		g		h
		i	j	
z				

`<Caption align = top>`

Приклад:

`</Caption>`

`<table border = 3>`

`<tr> <td> a <td> b <td> c <td> d <td> e`

`<tr> <td background = images / sky.gif align = middle colspan = 2 rowspan = 2> f`

`<td colspan = 2> g <td width = 10% rowspan = 2> h`

`<tr> <td> i <td> j`

`<tr> <th bgcolor = navy align = center colspan = 5> z `

`</table>`

Відзначимо, що в комірці можна зробити вкладену таблицю, а в ній ще одну і т.д. Атрибут `<th>` також задає клітинку, але тільки відрізняється тим що текст всередині такого осередку відображається жирним шрифтом.

Завдання.

Використовуючи CSS написати html-сторінку з таблицею

А	α	Рухомий рядок (горизонтальний)	
	червоний	Замощений фон	Рухомий рядок (вертикальний)
зображення	жовтий	Html-сторінка з першого завдання	
	зелений		

Контрольні питання.

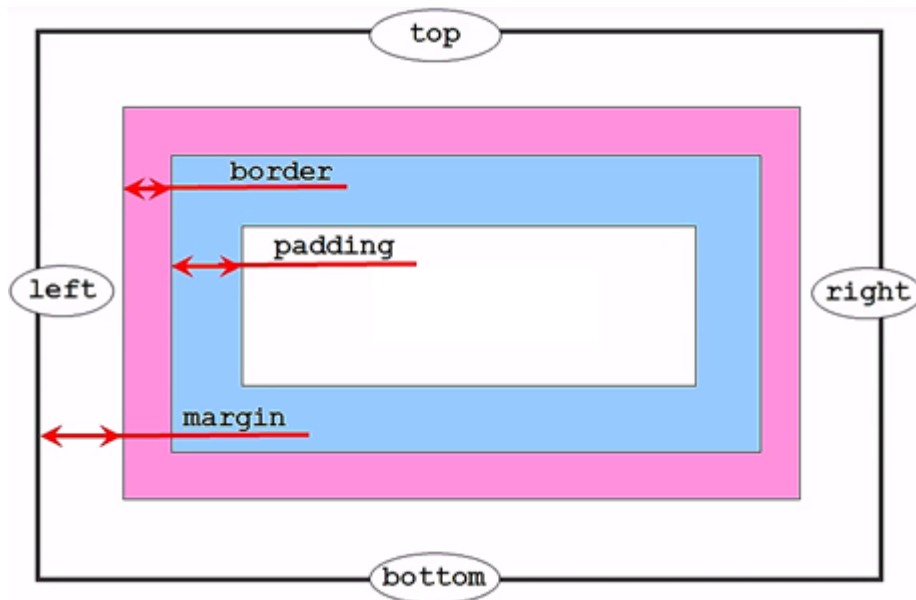
1. Як об'єднати рядки таблиці?
2. Як об'єднати стовбці таблиці?
3. Які бувають списки?
4. Як зробити власний маркер?

Лабораторна робота №4

Створення макету Web-сторінки

Мета роботи: Застосування позиціювання для розміщення блоків.
Теоретичні основи.

Відступи й рамки



Відступи зовні

`margin-top: auto|величина|%`

`margin-right: auto|величина|%`

`margin-bottom: auto|величина|%`

`margin-left: auto|величина|%`

`margin: margin-top margin-right margin-bottom margin-left`

Послідовність строго визначена. Якщо горизонтальні величини рівні (або вертикальні), то кожне з них можна скоротити до однієї. Може бути, що всі однакові.

`margin: 10px 20px 20px 30px;`

`margin: 15px;`

`margin-left: 10px; margin-right: 5px;`

`margin` по вертикалі завжди скидається в 0, якщо `margin` у відсотках указати елементу контенту (наприклад `div`), то цей відсоток береться від ширини області, у якій лежить елемент (контейнер).

Якщо `margin` від'ємний, то дозволено іншим елементам проводити перекриття, наплив одного елемента на інший.

Відступи зсередини

padding-top: величина|%

padding-right: величина|%

padding-bottom: величина|%

padding-left: величина|%

padding: padding-top padding-right padding-bottom padding-left

padding: 10px 20px 20px 30px;

padding: 15px;

Тут відсотки краще не використовувати. Від'ємних значень ні, за замовчуванням -0

Рамки

border-width: величина|%(thin|medium|thick)

border-color: колір

border-style: none|dotted|dashed|solid|double|groove|ridge|inset|outset

без рамки|точками|розривна лінія|неперервна|імітація

опуклості|елемент втиснутий|елемент опуклий

border: border-width border-style border-color

border: 1px solid black;

Рамка для кожної сторони

border-top-(width|color|style)

border-right-(width|color|style)

border-bottom-(width|color|style)

border-left-(width|color|style)

Але краще кожну сторону визначати так

border-left: 3px solid black;

border-right: 1px dotted #ccc;

Приклад

<html>

<head>

<meta http-equiv="content-type" content="text/html; charset=windows-1251">

<title>Приклад рішення завдання</title>

<style type="text/css">


```

h3{
margin:0 50px -50px; //тут залазимо на іншу область
padding:50px;
border:10px solid #369;
background:#69c;
}
p{
margin:0 150px 50px;

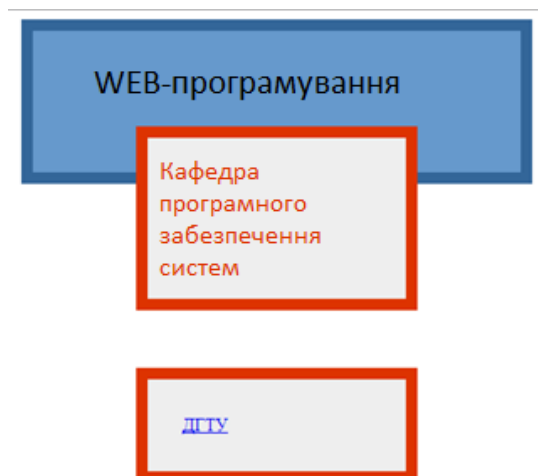
```

якщо напишемо `margin:30px 150px 50px;` то нічого не відбудеться - урахується ТІЛЬКИ найбільший по модулю (по вертикалі) `margin`. Якщо значення різні за знаком, то вони підсумуються

```

padding:30px;
border:10px solid #d30;
background:#eee;
}
</style>
</head>
<body>
<h3> Web-програмування</h3>
<p>
Кафедра програмного забезпечення систем
<p><a href="www.dstu.dp.ua">ДГТУ</a>
</body>
</html>
<html>

```



Робота з контентом

Параметри блоку

width: величина|%
width: 100px;
height: величина|%
height: 100px;
float: none|left|right
float: left;

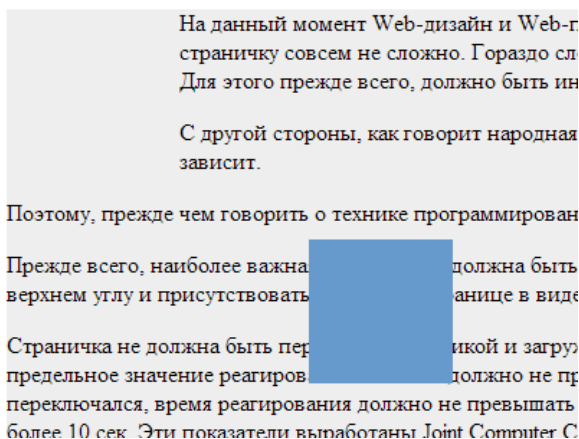
плаваючий елемент свій розмір визначає по своєму контенту. Причому блокові елементи це не враховують.

clear: none|left|right|both
clear: both;

ураховувати плаваючий елемент і блокові елементи із вказівкою де стоїть плаваючий елемент. Приклад

Параметри прошарку

position: static|absolute|relative
position: absolute;
top: auto|величина|%
top: 100px;
left: auto|величина|%



```
left: 100px;
<style type="text/css">
#left{
    margin:10px;
    background:#69c;
    width:100px;
    height:100px;
    float:left;
    position:relative;
    left:200px;
```

```
        top:150px;
    }
    #text{
        background:#eee;
    }
</style>
```

Всі елементи, вважають, що плаваюча область стоїть на своєму місці, а вона зміщена на left top **щодо свого** первісного розташування.

Абсолютно розташований елемент стає блоковим і з жодним іншим елементом його не реагує. Зсув в абсолютно розташованому елементі ведеться від батьківського елемента від кореневого, тобто лівого верхнього кута. Приклад використання - поява меню або підказки (position: absolute;) при наведенні на батьківський елемент (position: relative).

```
<div style="position:absolute; top:100px; left:50px">
```

Цей прошарок розташований абсолютно.

```
<div style="position:relative; top:10px; left:10px">
```

Цей прошарок розташований відносно.

```
</div>
```

```
</div>
```

Ще один метод розташування fixed. У цьому випадку при зсуві сторінки, наприклад, при прокручуванні, цей елемент стоїть на тому самому місці, незалежно, куди і як ми прокручуємо сторінку. При цьому ніякого пошуку батьків, як у попередньому випадку немає.

Візуальні властивості

display: none|block|inline|list-item

display: block;

Це правило є в будь-якого елементу - означення його типу.

block -блоковий елемент, inline-строковий і list-item -список, є й інші.

Значення none указує на те, що **елемент не буде промальовуватися**, немов його й немає взагалі.

visibility: hidden|visible|inherit

visibility: hidden;

Видимість об'єкта. За замовчуванням - visible. Якщо властивість приймає значення hidden, то об'єкт не промальовується, але МІСЦЕ під нього резервується (на відміну від display: none).

overflow: auto|scroll|visible|hidden

overflow: auto;

Переповнення контенту. За замовчуванням - visible. При переповненні області контент виходить за її межі. Елементи сторінки це не бачать. При

установці hidden, все що виходить за межі блоку - ховається. Значення scroll установлює прокручування НА ЕЛЕМЕНТІ, дозволяючи переглядати контент. Мінус - смуга прокручування є завжди - потрібна вона чи ні. Щоб позбутися від цього, використовуємо auto.

```
clip: auto|rect(top right bottom left)
clip:rect(10px 20px 30px 10px);
```

Визначає ту частину елемента (прямокутну), що будемо показувати (обрізка). Всі значення щодо лівого верхнього кута контейнера вихідного елемента.

```
z-index: auto|величина|inherit
z-index:3;
```

Значення z-index (номер прошарку) ведеться від 0 (самий нижній, невидимий при нашаруванні). Може бути від'ємним.

Значення inherit. Примусове спадкування властивостей.

Приклади створення різного дизайну для однієї й тієї ж сторінки (html код той самий), наведені на ресурсі <http://www.csszengarden.com/>

Властивості курсору

```
cursor: auto
```

- crosshair
- default
- pointer
- move
- text
- wait
- e-resize, ne-resize, nw-resize, n-resize, se-resize, sw-resize, s-resize, w-resize
- url("зображення")

Загальна структура елементів

Рядкові

inline

inline-block

- img, iframe, object, embed
- -audio, video, canvas
- -input, textarea, select, button

inline-table

Блокові

block

list-item(li)

table(...)

На жаль, існує ряд елементів, для яких розроблювачі браузерів не прийшли до угоди. У цьому випадку потрібно передбачати всі варіанти, наприклад, при використанні закруглення кутів блоку, наприклад,

```
border-radius:30px;
-webkit-border-radius:30px;
-moz-border-radius:30px;
-o-border-radius:30px;
-ms-border-radius:30px;
```

Надалі перейдемо до методів гумової верстки.

Створюємо контейнер `<div id="container">`, що відповідає за розташування - по середині, притиснутим до низу або якось по іншому.

Усередину цього службового блока покладемо два – один відповідає за головний контент, іншої – footer. У блок головного змісту вкладемо `<div id="all">`, де безпосередньо буде лежати контент.

```
<div id="container">
  <div id="main">

    <div id="all"></div>

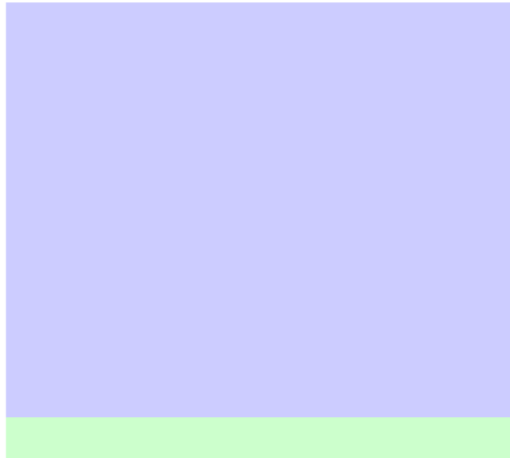
  </div>
```

```
<div id="footer"></div>
</div>
```

Стилі, що формують основний вид виглядають так

```
/*схема із притиснутим до низу footer*/
```

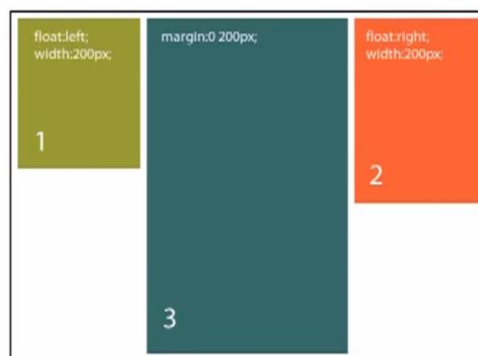
```
html,body, #container{
    height:100%;
}
html{/*це так, про всякий випадок, раптом системні налаштування
не стандартні, колір фону, шрифту, до речі, за замовчуванням фон -
прозорий*/
background:#fff;
color:#333;
}
#container{
min-width:800px;
max-width:800px;
margin:auto; /* центрування по вертикалі скинеться в нуль, по
горизонталі - вирівнюється*/
}
#main{
min-height:100%; /* тому що height не встановлено, а в контейнера
100%, то висота встановлюється по мінімуму або по контенту (якщо
більше мінімальної висоти)*/
margin:0 0 -100px; /* останній - на висоту нижнього блоку, щоб
footer було видно */
background:#ccf;
}
#all{
overflow:hidden; /* щоб уміст плаваючих блоків не провалювався*/
padding:0 0 100px; /* щоб контент не залазив під нижній блок*/
}
#footer{
height:100px; /* раніше обраний розмір підвалу*/
background:#cfc;
}
```



Перейдемо до створення колонок.



Тому що розмір може мінятися, то крайні колонки будуть плаваючими.



margin потрібний, щоб колонки віджати від країв, без цього цей блок ліг би під плаваючими областями, а текст залишився б обтікати. Порядок розміщення - очевидний, спочатку плаваючі області (не важливо в якому порядку).

Додаємо в основний файл

```
<div id="all">
  <div id="left">Лівий</div>
```

```

<div id="right">Правий</div>
<div id="center">Центр</div>
</div>

```

і в стильовий файл додаємо

```

#left{
float:left;
width:200px; /*абсолютні значення, щоб колонка не плавала*/
}
#right{
float:right;
width:200px; /*абсолютні значення, щоб колонка не плавала*/
}
#center{
margin:0 200px;
background:#cfc;
}

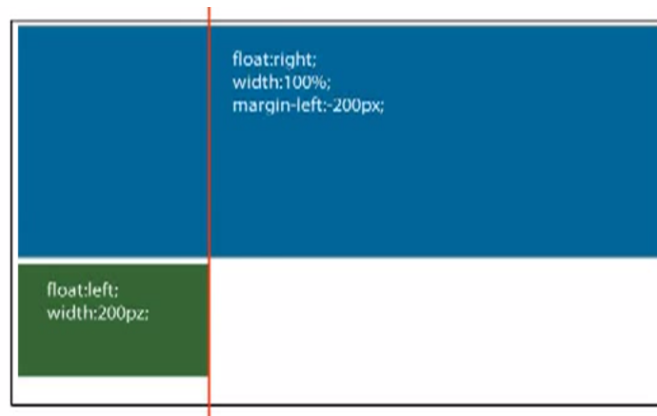
```

Підфарбувавши крайні колонки, одержимо



Не симпатично, хотілися б незаповнені місця колонок зафарбувати по максимуму. Для цього зробимо два контейнери, заліємо їхнім потрібним кольором і покладемо туди колонки.

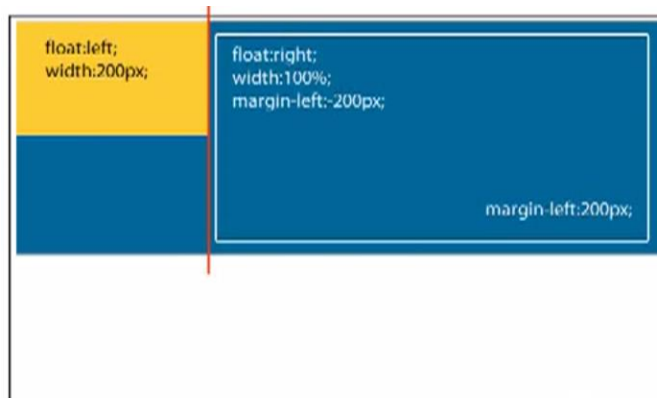
Розглянемо ще один підхід, більше універсальний, не зав'язаний на порядок завантаження колонок. На плаваючий елемент фіксованої ширини (ліворуч) накотимо інший плаваючий елемент (синій).



Установлюючи від'ємне значення margin, рівне ширині елемента, ми дозволяємо на цю ширину залізти. У результаті одержуємо



А щоб вкласти правий стовпець, уміст синього елемента вкладемо ще в один елемент, у який вкладемо аналогічно попередньому



Структура сторінки буде мати вигляд

```
<div id="container">
  <div id="main">
    <div id="all">
      <div id="out">
        <div id="in">
```

```

        <div id="center">Центр</div>
    </div>
    <div id="right">right</div>
</div>
<div id="left">left</div>
</div>
</div>
<div id="footer"></div>
</div>

```

а в стилі додаємо

```

#out{
float:right;
width:100%;
margin-left:-200px;
}
#in{
float:left;
width:100%;
margin-right:-200px;
}

```

На перший погляд нічого не змінилося. Проведемо заключний штрих - розфарбування колонок до кінця.

У стилі додаємо

```

#main{
background:url(..Images/right.png) repeat-y 100% 0; /* 0 по правому
краї*/
}

```

якщо в all покласти

```
background:url(..Images/left.png) repeat-y;
```

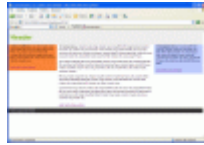
те ліва колонка зафарбується на висоту контенту + padding. Не те. А от, якщо покласти в container, то все вийде. Але. Якщо контенту в лівій колонці багато, то буде переповнення, що приведе до виходу тексту із зафарбованого блоку. А all, саме регулюється висотою контенту. Додавши в обидва блоки, одержуємо рішення.

Завдання.

Зробити верстку сторінки з номером, який відповідає номеру в журналі групи



Три колонки
гумові
процентної
ширини
(N.01)



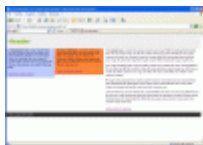
Три колонки
гумові
процентної
ширини
(N.02)



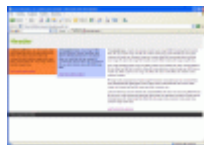
Три колонки
гумові
процентної
ширини
(N.03)



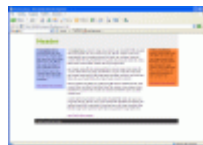
Три колонки
гумові
процентної
ширини
(N.04)



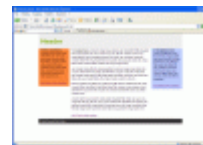
Три колонки
гумові
процентної
ширини
(N.05)



Три колонки
гумові
процентної
ширини
(N.06)



Три колонки
фіксованої
ширини
(N.07)



Три колонки
фіксованої
ширини
(N.08)



Три колонки
фіксованої
ширини
(N.09)



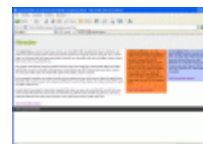
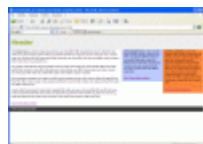
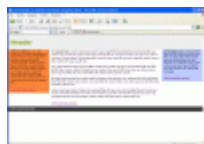
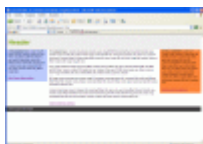
Три колонки
фіксованої
ширини
(N.10)



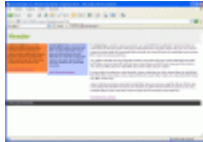
Три колонки
фіксованої
ширини
(N.11)



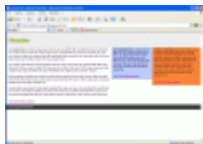
Три колонки
фіксованої
ширини
(N.12)



Гумовий, дві
колонки
фіксованої
ширини
(N.13)



Гумовий, дві
колонки
фіксованої
ширини
(N.17)

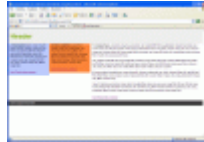


Гумовий,
одна колонка
фіксованої
ширини (N.1)

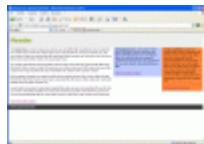


Гумовий, дві
колонки
процентної

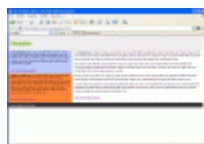
Гумовий, дві
колонки
фіксованої
ширини
(N.14)



Гумовий, дві
колонки
фіксованої
ширини
(N.18)

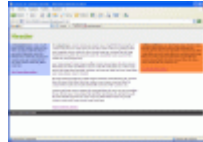


Гумовий,
одна колонка
фіксованої
ширини (N.2)

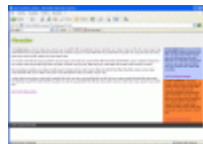


Гумовий, дві
колонки
процентної

Гумовий, дві
колонки
фіксованої
ширини
(N.15)



Гумовий, одна
колонка
фіксованої
ширини
(N.19)

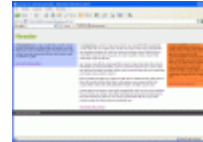


Гумовий, дві
колонки
фіксованої
ширини (N.3)



Гумовий, одна
колонка на
всю ширину и

Гумовий, дві
колонки
фіксованої
ширини
(N.16)



Гумовий, одна
колонка
фіксованої
ширини
(N.20)



Гумовий, дві
колонки
фіксованої
ширини (N.4)



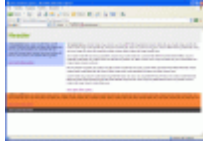
Гумовий, одна
колонка на
всю ширину и
дві процентні

ширини (N.5)

ширини (N.6)

дві процентні
(N.7)

(N.8)



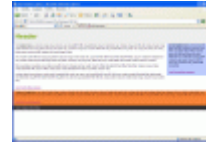
Гумовий,
одна колонка
на всю
ширину и дві
процентні
(N.9)



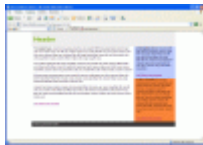
Гумовий,
одна колонка
на всю
ширину и дві
процентні
(N.10)



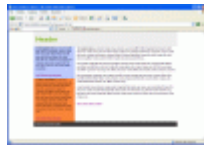
Дві колонки
гумові, одна
на всю
ширину и
одна
фіксовані
(N.11)



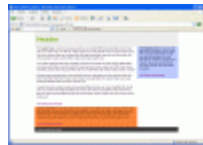
Дві колонки
гумові, одна
на всю
ширину и
одна
фіксовані
(N.12)



Три колонки
фіксованої
ширини
(N.13)



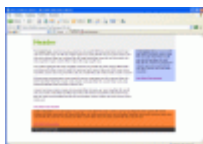
Три колонки
фіксованої
ширини
(N.14)



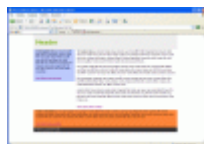
Три колонки
фіксованої
ширини
(N.15)



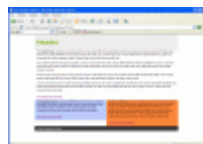
Три колонки
фіксованої
ширини
(N.16)



Три колонки
фіксованої
ширини
(N.17)



Три колонки
фіксованої
ширини
(N.18)



Фіксований
одна колонка
на всю
ширину і дві
на половину



Фіксований
одна колонка
на всю
ширину і дві
на половину

		(N.19)	(N.20)
--	--	--------	--------

Контрольні питання.

1. Чим відрізняється таблична верстка від гумової?
2. Що виконує атрибут float?
3. Як реалізувати наплив одного блоку на інший?
4. Як реалізувати наплив елементу span на блок?

Лабораторна робота №5

Робота з HTML-формами.

Мета роботи: Робота з основними елементами HTML-форм.**Теоретичні основи.**

Web-документ повинен володіти можливістю динамічної зміни в часі і зворотним зв'язком між користувачем і сервером. Одним із засобів взаємообміну інформацією є HTML-форми, CGI сценарії (Common Gateway Interface- загальний інтерфейс шлюзів), JavaScript-сценарії та ін. HTML-форма служить для введення даних і відправки сценарію серверу.

Синтаксис

```

<FORM
  action = url
  class = classname
  enctype = encoding
  id = value
  lang = language
  language = javascript | jscript | vbscript
  method = get | post
  name = name
  style = properties
  target = windows_name | _blank | _parent | _self | _top
  title = string
  event = script
>

```

Тут name- ім'я форми для наступних посилань на неї, action - метод передачі даних від клієнта до сервера. Може приймати значення get - використовується за умовчанням і призводить до додавання вмісту форми до URL і обробляється як аргумент командного рядка, post - при використанні цього параметра вміст форми пересилається не як частина URL, а у вигляді окремої частини HTTP-заголовку. Потім сервер передає цю інформацію в CGI з якого зчитуються дані в міру необхідності.

Усередині тегів <Form> ... </ Form> може міститися будь-яка кількість тегів, що задають елементи управління - <INPUT>, <SELECT>, <TEXTAREA>.

Тегом <INPUT> створюються елементи управління - поле введення, кнопка, прапорець, перемикач і деякі інші.

Синтаксис

<INPUT

accesskey = key

align = left | center | right

alt = text

class = classname

disabled

dynsrc = url

id = value

lang = language

language = javascript | jscript | vbscript

lowsrc = url

maxlength = n

name = name

readonly

size = n

style = properties

tabindex = n

title = string

type = button | checkbox | file | hidden | image | password | radio | reset |

submit | text

value = value

event = script

>

Опишемо докладніше атрибути.

- align - вирівнювання тексту стоїть за елементом управління.
- checked-прапорець або перемикач встановлений при завантаженні сторінки.
- disabled-елемент керування не досяжний для користувача.
- maxlength = максимальне число символів, які можуть бути введені в поле введення. За замовчуванням - необмежена.
- name-ім'я елемента управління.
- size- розмір поля введення в символах. За замовчуванням -20.
- src- адреса зображення, якщо значення атрибута type одно image.
- type-тип елемента керування.
- button-кнопка, атрибут value задає текст на кнопці
- checkbox- прапорець. Атрибут value задає стан прапорця - on (встановлений) і off (знятий)
- hidden- приховане значення. Цей елемент дозволяє розмістити дані невидимі користувачеві. Атрибут value визначає значення елемента.

- hfssword- поле введення пароля, тобто поле введення, у якого вводяться символи замінюються зірочками. Атрибут value визначає значення за замовчуванням.
- image-малюнок.
- radio-перемикач. Атрибут value визначає стан перемикача. Допустимі значення on, off.
- reset-кнопка, що встановлює у всіх інтерфейсних елементах значення використовувани за замовчуванням.
- submit- кнопка, дія якої призводить до відправки вмісту форми на сервер.
- title-спливаюча підказка.
- event- одна з подій
- onblur-генерується при втраті елементом управління фокусу.
- onclick- генерується при натисканні на елементі управління.
- onchange-генерується при зміні вмісту об'єкта, наприклад, тексту у вікні введення.
- ondblclick-генерується при подвійному натисканні на елементі управління.
- onfocus-генерується при отриманні фокусу елементом управління.
- onhelp- генерується при натиску клавіші F1 або команди Довідка у вікні браузера.
- onkeydown-генерується коли клавіша натиснута.
- onkeypress-генерується при натисканні клавіші.
- onkeyup-генерується, коли клавіша відпущена.
- onmousedown-генерується при натисканні миші по елементу управління.
- onmousemove-генерується при переміщенні покажчика миші над елементом управління.
- onmouseout-генерується при покиданні покажчика миші елемента керування.
- onmouseover-генерується коли покажчик миші знаходиться над елементом управління.
- onmouseup-генерується при відпуску клавіші миші.
- onresize-генерується при зміні розмірів елемента керування.
- onselect-генерується при зміні поточного вибору, наприклад, виборі нового елемента в списку.

Приклад

```
<center>
<form name formBlank>
  <table border = 18 bgColor = gray>
    <tr>
      <td> П.І.Б. </ td>
```



```

<td colspan = 3> <input name = text1 size = 40
  style = "height: 22px; width: 289px"> </ td>
<td> </ td>
  <td> </ td>
</ tr>
<tr>
<td> Адреса </ td>
<td colspan = 3> <input name = text2 size = 40
  style = "height: 22px; width: 289px"> </ td>
<td> </ td>
  <td> </ td>
</ tr>
<tr>
<td> Місто </ td>
<td> <input name = text3 size = 22> </ td>
<td> Код </ td>
  <td> <input name = text4 size = 8
    style = "height: 22px; width: 79px"> </ td>
</ tr>
</ table>
</ form>
</ center>

```

Завдання.

Написати html-форму реєстрації, на якій повинні бути обов'язкові елементи

- поле «Прізвище»
- поле «Login»
- поле «Password»
- поле «Вік»
- пункт «Стать»
- пункт «Згоден»
- поле «Коментар»

Контрольні питання.

1. Чим відрізняється елемент Text від елементу Password?
2. Чим відрізняється елемент Text від елементу Number?
3. Чим відрізняється елемент Button від елементу Submit?

Лабораторна робота №6

Фреймворк Bootstrap.

Мета роботи: ознайомлення з фреймворком Bootstrap.

Теоретичні основи.

Bootstrap - бібліотека шаблонів CSS, вільно поширювана командою сайту MaxCDN, розроблений як фреймворк для забезпечення однаковості внутрішніх інструментів Twitter.

Для використання Bootstrap сторінки повинні бути зверстані за новітніми стандартами. Вони повинні включати і використовувати HTML5 doctype і viewport meta tag для правильного «чуйної» поведінки сторінок. Bootstrap розроблявся як *mobile first*, тобто його налаштування насамперед оптимізовані під мобільні пристрої, а вже потім за допомогою медіа-запитів підганяємо масштаб компонентів як необхідно на інших пристроях. Ось як повинен виглядати «верхній», корінний код сторінки:

```
<!doctype html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-
to-fit=no">
    <!-- Bootstrap CSS -->
    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">
    <title>Hello, world!</title>
  </head>
  <body>
    <h1>Hello, world!</h1>
  </body>
</html>
```

Bootstrap підтримують всі сучасні, стабільні релізи браузерів і платформ.

Контейнер.

Це базовий елемент в Bootstrap і вони необхідні при використанні стандартної системи сітки. Вибираючи контейнер з фіксованою шириною (що означає, що max-width змінюється на кожному брейкпойнті) або контейнер з плаваючою шириною (width == 100%).

Контейнери *можуть* мати вкладені елементи, але в більшості випадків можна обійтися без них.

```
<div class="container">
```

```
<!-- Content here -->
</div>
```

Використовуючи `.container-fluid` маємо контейнер, що займає 100% зони перегляду.

```
<div class="container-fluid">
...
</div>
```

Система сіток Bootstrap використовує контейнери, ряди і колонки, щоб зручно розташовувати вміст. Приклад

```
<div class="container">
  <div class="row">
    <div class="col-sm">
      Одна із трьох колонок
    </div>
    <div class="col-sm">
      Одна із трьох колонок
    </div>
    <div class="col-sm">
      Одна із трьох колонок
    </div>
  </div>
</div>
```

Одна з трьох колонок	Одна з трьох колонок	Одна з трьох колонок
----------------------	----------------------	----------------------

У даному прикладі за допомогою наших встановлених класів сіток були створені 3 колонки рівної ширини для маленьких, середніх і великих девайсів. Ці колонки вирівняні за допомогою батьківського класу `.container`.

Ось невелика пояснення роботи Bootstrap:






- Інструмент для центрування контенту сайту. Використовуйте клас `.container` для фіксованої ширини або `.container-fluid` для 100% -ї ширини.
- Ряди - це обгортка для колонок. Кожна колонка має горизонтальний padding (званий gutter) для контролю простору між колонками. Цей padding (паддінг) впливає на ряди з негативним margin. В цьому випадку весь вміст ваших колонок буде візуально центровано внизу з лівого боку.
- Вміст має бути розташоване в колонках, і тільки колонки можуть бути розташовані в рядках.
- Завдяки флексбоксу колонки сітки без встановленого атрибуту «ширина» автоматично отримують рівну ширину. Наприклад, чотири

екземпляра класу `.col-sm` автоматично отримають ширину однієї колонки = 25%.

- Цифри в найменуванні класів колонок показують, скільки колонок з 12-ти можливих в ряду ви б хотіли використовувати. Так, якщо ви хочете використовувати три колонки однієї ширини, використовуйте `.col-sm-4`.
- Ширина колонок `width` задана у відсотках, що дозволяє колонкам бути гнучкими і змінювати розмір щодо їх батьківського елемента.
- Колонки мають горизонтальний `padding` для створення відступів між окремими колонками, але ви можете видалити `margin` і `padding` з колонок, додавши клас `.no-gutters` в `.row`.
- Є 5 «ярусів» сіток, по одному для кожного «брейкпойнта»: всі контрольні точки (екстра маленький), маленький, середній, великий і екстра великий.
- Яруси сітки засновані на мінімальній широті, тобто вони підходять для кожного вищого ярусу (тобто, `.col-sm-4` підходить для маленьких, середніх, великих і XL девайсів).
- Ви можете використовувати зумовлені класи сітки (наприклад `.col-4`) або препроцесори Sass для створення своєї розмітки.

Бутстрап використовує `em` і `rem` для завдання більшості розмірів, а пікселі `px` для «брейкпойнтів» сітки і ширини контейнерів. Це відбувається тому, що ширина зони видимості на кожному пристрої вимірюється в пікселях і не змінюється з розміром шрифту.

Подивимося, як діють деякі аспекти системи сіток Bootstrap на різних пристроях.

					
	Extra small <576px	Small ≥576px	Medium ≥768px	Large ≥992px	Extra large ≥1200px
Максимальна ширина контейнера	None (auto)	540px	720px	960px	1140px
префікс класу	<code>.col-</code>	<code>.col-sm-</code>	<code>.col-md-</code>	<code>.col-lg-</code>	<code>.col-xl-</code>
число колонок	12				

Ширина відступу	30px (15px з кожного боку стовпчика)
Може бути вкладеним	Так
упорядкування колонок	Так

Автоматичне розташування за допомогою колонок

Наприклад, тут ми бачимо дві сітки, які підійдуть до будь-якого пристрою і зоні видимості, від xs до xl. Додавайте будь-яку кількість простих класів для кожного брейкпойнта, і кожна колонка буде однакової ширини.

1 з 2	2 з 2	
1 з 3	2 з 3	3 з 3

```
<div class="container">
  <div class="row">
    <div class="col"> 1 з 2 </div>
    <div class="col"> 2 з 2 </div>
  </div>
  <div class="row">
    <div class="col"> 1 з 3 </div>
    <div class="col"> 2 з 3 </div>
    <div class="col"> 3 з 3 </div>
  </div>
</div>
```

Авто-розташування колонок в сітці флексбокса також дає можливість встановити ширину однієї колонки, при цьому інші «родинні» колонки автоматично змінять свій розмір навколо неї. Зауважимо, що інші колонки будуть змінювати розмір незалежно від ширини центральної колонки.

1 з 3	2 з 3 (широка)	3 з 3
1 з 3	2 з 3 (широка)	3 з 3

```
<div class="container">
  <div class="row">
    <div class="col"> 1 з 3 </div>
    <div class="col-6"> 2 з 3 (широка)</div>
```

```

    <div class="col"> 3 з 3 </div>
  </div>
  <div class="row">
    <div class="col"> 1 з 3 </div>
    <div class="col-5"> 2 з 3 (широка) </div>
    <div class="col"> 3 з 3 </div>
  </div>
</div>

```

Вміст адаптивної ширини

Використовуйте класи `col-{breakpoint}-auto` для створення колонок, що змінюють свою ширину по ширині вмісту.

1 з 3	Вміст адаптивної ширини	3 з 3
1 з 3	Вміст адаптивної ширини	3 з 3

```

<div class="container">
  <div class="row justify-content-md-center">
    <div class="col col-lg-2">
      1 з 3
    </div>
    <div class="col-md-auto">
      Вміст адаптивної ширини
    </div>
    <div class="col col-lg-2">
      3 з 3
    </div>
  </div>
  <div class="row">
    <div class="col">
      1 з 3
    </div>
    <div class="col-md-auto">
      Вміст адаптивної ширини
    </div>
    <div class="col col-lg-2">
      3 з 3
    </div>
  </div>
</div>

```

Сітка Bootstrap включає 5 «ярусів» визначених класів, використовуваних для побудови складного адаптивного контенту. Адаптуйте розміри своїх колонок для правильного їх відображення на всіх видах і розмірах пристроїв.

Для сіток, які виглядають і розташовуються однаково на всіх пристроях будь-якого розміру, використовуйте класи `.col` та `.col-*`. Визначте іменований клас з цифрою, коли вам потрібна колонка певного розміру, у всіх інших випадках вільно користуйтеся `.col`.

col	col	col	col
col-8			col-4

```
<div class="container">
  <div class="row">
    <div class="col">col</div>
    <div class="col">col</div>
    <div class="col">col</div>
    <div class="col">col</div>
  </div>
  <div class="row">
    <div class="col-8">col-8</div>
    <div class="col-4">col-4</div>
  </div>
</div>
```

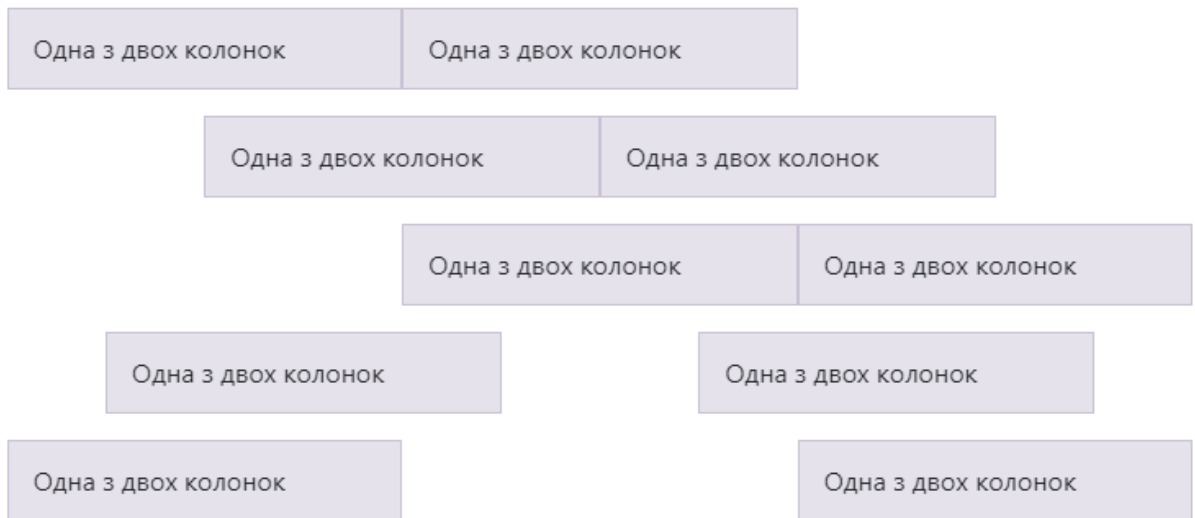
Використовуючи один набір з класів `.col-sm-*`, можете створити базову сітку, яка спочатку складена по вертикалі, а потім її колонки стають горизонтальними (на екстремалих девайсах).

col-sm-8		col-sm-4
col-sm	col-sm	col-sm

```
<div class="container">
  <div class="row">
    <div class="col-sm-8">col-sm-8</div>
    <div class="col-sm-4">col-sm-4</div>
  </div>
  <div class="row">
    <div class="col-sm">col-sm</div>
    <div class="col-sm">col-sm</div>
    <div class="col-sm">col-sm</div>
  </div>
</div>
```

</div>

Горизонтальне вирівнювання



```
<div class="container">
  <div class="row justify-content-start">
    <div class="col-4">
      Одна з двох колонок
    </div>
    <div class="col-4">
      Одна з двох колонок
    </div>
  </div>
  <div class="row justify-content-center">
    <div class="col-4">
      Одна з двох колонок
    </div>
    <div class="col-4">
      Одна з двох колонок
    </div>
  </div>
  <div class="row justify-content-end">
    <div class="col-4">
      Одна з двох колонок
    </div>
    <div class="col-4">
      Одна з двох колонок
    </div>
  </div>
  <div class="row justify-content-around">
    <div class="col-4">
```



```

    Одна з двох колонок
  </div>
  <div class="col-4">
    Одна з двох колонок
  </div>
</div>
<div class="row justify-content-between">
  <div class="col-4">
    Одна з двох колонок
  </div>
  <div class="col-4">
    Одна з двох колонок
  </div>
</div>
</div>

```

Таблиці

Basic Table

Країна	Country	Валюта
Україна	Ukraine	Гривня (₴)
Європейська унія	European Union	Євро (€)
Велика Британія	United Kingdom	Фунт стерлінгів (£)

```

<div class="container">
  <h2>Basic Table</h2>
  <table class="table">
    <thead>
      <tr>
        <th>Країна</th>
        <th>Country</th>
        <th>Валюта</th>
      </tr>
    </thead>
    <tbody>
      <tr>
        <td>Україна</td>

```

```

    <td>Ukraine</td>
    <td>Гривня (&#8372;)</td>
</tr>

<tr>

    <td>Європейська унія</td>
    <td>European Union</td>
    <td>Євро (&euro;)</td>
</tr>

<tr>

    <td>Велика Британія</td>
    <td>United Kingdom</td>
    <td>Фунт стерлінгів (&pound;)</td>
</tr>

</tbody>
</table>
</div>

```

Striped Rows

.table-striped class додає у таблицю ефект "зебри":

Країна	Country	Валюта
Україна	Ukraine	Гривня (₴)
Європейська унія	European Union	Євро (€)
Велика Британія	United Kingdom	Фунт стерлінгів (£)

```

<div class="container">
  <h2>Striped Rows</h2>
  <p>.table-striped class додає у таблицю ефект "зебри":</p>
  <table class="table table-striped">
    <thead>
      <tr>
        <th>Країна</th>

```

```

    <th>Country</th>
    <th>Валюта</th>
  </tr>
</thead>
<tbody>
  <tr>
    <td>Україна</td>
    <td>Ukraine</td>
    <td>Гривня (&#8372;)</td>
  </tr>
  <tr>
    <td>Європейська унія</td>
    <td>European Union</td>
    <td>Євро (&euro;)</td>
  </tr>
  <tr>
    <td>Велика Британія</td>
    <td>United Kingdom</td>
    <td>Фунт стерлінгів (&pound;)</td>
  </tr>
</tbody> </table></div>

```

Bordered Table

.table-bordered class додає у таблицю зовнішній бордюр:

Країна	Country	Валюта
Україна	Ukraine	Гривня (₴)
Європейська унія	European Union	Євро (€)
Велика Британія	United Kingdom	Фунт стерлінгів (£)

```

<div class="container">
  <h2>Bordered Table</h2>

```

<p>.table-bordered class додає у таблицю зовнішній бордюр:</p>

```
<table class="table table-bordered">
```

```
  <thead>
```

```
    <tr>
```

```
      <th>Країна</th>
```

```
      <th>Country</th>
```

```
      <th>Валюта</th>
```

```
    </tr>
```

```
  </thead>
```

```
  <tbody>
```

```
    <tr>
```

```
      <td>Україна</td>
```

```
      <td>Ukraine</td>
```

```
      <td>Гривня (&#8372;)</td>
```

```
    </tr>
```

```
    <tr>
```

```
      <td>Європейська унія</td>
```

```
      <td>European Union</td>
```

```
      <td>Євро (&euro;)</td>
```

```
    </tr>
```

```
    <tr>
```

```
      <td>Велика Британія</td>
```

```
      <td>United Kingdom</td>
```

```
      <td>Фунт стерлінгів (&pound;)</td>
```

```
    </tr>
```

```
</tbody> </table></div>
```

Hover Rows

.table-hover class дає можливість виділяти курсором миші:

Країна	Country	Валюта
Україна	Ukraine	Гривня (₴)
Європейська унія	European Union	Євро (€)
Велика Британія	United Kingdom	Фунт стерлінгів (£)

```
<div class="container">
```

```
<h2>Hover Rows</h2>
```

```
<p> .table-hover class дає можливість виділяти курсором миші:</p>
```

```
<table class="table table-hover">
```

```
<thead>
```

```
<tr>
```

```
<th>Країна</th>
```

```
<th>Country</th>
```

```
<th>Валюта</th>
```

```
</tr>
```

```
</thead>
```

```
<tbody>
```

```
<tr>
```

```
<td>Україна</td>
```

```
<td>Ukraine</td>
```

```
<td>Гривня (&#8372;)</td>
```

```
</tr>
```

```
<tr>
```

```
<td>Європейська унія</td>
```

```
<td>European Union</td>
```

```
<td>Євро (&euro;)</td>
```

```
</tr>
```

```
<tr>
```

```

<td>Велика Британія</td>
<td>United Kingdom</td>
<td>Фунт стерлінгів (&pound;)</td>
</tr>
</tbody> </table></div>

```

Condensed Table

.table-condensed class зменшує розмір таблиці за рахунок стиску у двічі cell padding:

Країна	Country	Валюта
Україна	Ukraine	Гривня (₴)
Європейська унія	European Union	Євро (€)
Велика Британія	United Kingdom	Фунт стерлінгів (£)

```

<div class="container">
  <h2>Condensed Table</h2>
  <p> .table-condensed class зменшує розмір таблиці за рахунок стиску у двічі
  cell padding:</p>
  <table class="table table-condensed">
    <thead>
      <tr>
        <th>Країна</th>
        <th>Country</th>
        <th>Валюта</th>
      </tr>
    </thead>
    <tbody>
      <tr>
        <td>Україна</td>
        <td>Ukraine</td>
        <td>Гривня (&#8372;)</td>
      </tr>
      <tr>

```

```

<td>Європейська унія</td>
<td>European Union</td>
<td>Євро (&euro;)</td>
</tr>
<tr>
<td>Велика Британія</td>
<td>United Kingdom</td>
<td>Фунт стерлінгів (&pound;)</td>
</tr>
</tbody> </table></div>

```

Contextual Classes

Контекстні класи можуть бути використані для розфарбовки рядків або елементів таблиці

Країна	Country	Валюта
Україна	Ukraine	Гривня (₴)
Європейська унія	European Union	Євро (€)
Велика Британія	United Kingdom	Фунт стерлінгів (£)

```

<div class="container">
  <h2>Contextual Classes</h2>
  <p>Контекстні класи можуть бути використані для розфарбовки рядків або
елементів таблиці</p>
  <table class="table">
    <thead>
      <tr>
        <th>Країна</th>
        <th>Country</th>
        <th>Валюта</th>
      </tr>
    </thead>
    <tbody>
      <tr class="danger">

```

```

<td>Україна</td>
<td>Ukraine</td>
<td>Гривня (&#8372;)</td>
</tr>
<tr class="success">
<td>Європейська унія</td>
<td>European Union</td>
<td>Євро (&euro;)</td>
</tr>
<tr class="info">
<td>Велика Британія</td>
<td>United Kingdom</td>
<td>Фунт стерлінгів (&pound;)</td>
</tr>
</tbody> </table></div>

```

Контекстні класи, які можуть бути використані, є:

Клас	Опис
<code>.active</code>	Застосовується колір при наведенні курсору на рядок або елемент таблиці
<code>.success</code>	Вказує на успішне завершення дії
<code>.info</code>	Вказує на нейтральне інформаційне повідомлення
<code>.warning</code>	Вказує на попередження, або на що треба звернути увагу
<code>.danger</code>	Вказує на небезпеку або потенційно негативну дію

<p>Контекстні класи, які можуть бути використані, є:</p>

```

<table class="table table-striped">
<thead>
<tr> <th>Клас</th>
<th>Опис</th></tr>
</thead><tbody>
<tr>

```



```

<td><code>.active</code></td>

<td>Застосовується колір при наведенні курсору на рядок або елемент
таблиці</td>

</tr>

<tr>

<td><code>.success</code></td>

<td>Вказує на успішне завершення дії</td>

</tr>

<tr>

<td><code>.info</code></td>

<td>Вказує на нейтральне інформаційне повідомлення</td>

</tr>

<tr>

<td><code>.warning</code></td>

<td>Вказує на попередження, або на що треба звернути увагу</td>

</tr>

<tr>

<td><code>.danger</code></td> <td>Вказує на небезпеку або потенційно
негативну дію</td>

</tr></tbody></table>

```

Елементи форми

Button Styles



```

<div class="container">

<h2>Button Styles</h2>

<button type="button" class="btn btn-default">Default</button>

<button type="button" class="btn btn-primary">Primary</button>

```

```

<button type="button" class="btn btn-success">Success</button>
<button type="button" class="btn btn-info">Info</button>
<button type="button" class="btn btn-warning">Warning</button>
<button type="button" class="btn btn-danger">Danger</button>
<button type="button" class="btn btn-link">Link</button>
</div>

```

Горизонтальна група кнопок



```

<div class="container">
  <h2>Горизонтальна група кнопок</h2>
  <div class="btn-group">
    <button type="button" class="btn btn-primary">HTML</button>
    <button type="button" class="btn btn-primary">CSS</button>
    <button type="button" class="btn btn-primary">JavaScript</button>
  </div>

```

Великі:

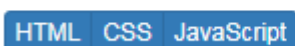


```

<h3>Великі:</h3>
<div class="btn-group btn-group-lg">
  <button type="button" class="btn btn-primary">HTML</button>
  <button type="button" class="btn btn-primary">CSS</button>
  <button type="button" class="btn btn-primary">JavaScript</button>
</div>

```

Малі:



```
<h3>Малі:</h3>
```

```
<div class="btn-group btn-group-xs">
```

```
  <button type="button" class="btn btn-primary">HTML</button>
```

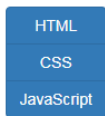
```
  <button type="button" class="btn btn-primary">CSS</button>
```

```
  <button type="button" class="btn btn-primary">JavaScript</button>
```

```
</div>
```

```
</div>
```

Вертикальна група кнопок



```
<div class="container">
```

```
  <h2>Вертикальна група кнопок</h2>
```

```
  <div class="btn-group-vertical">
```

```
    <button type="button" class="btn btn-primary">HTML</button>
```

```
    <button type="button" class="btn btn-primary">CSS</button>
```

```
    <button type="button" class="btn btn-primary">JavaScript</button>
```

```
  </div>
```

```
</div>
```

Розтягнута група кнопок



```
<div class="container">
```

```
  <h2>Розтягнута група кнопок</h2>
```

```
  <div class="btn-group btn-group-justified">
```

```
    <div class="btn-group">
```

```
      <button type="button" class="btn btn-primary">HTML</button>
```

```
    </div>
```

```
  <div class="btn-group">
```

```
    <button type="button" class="btn btn-primary">CSS</button>
```

```

</div>

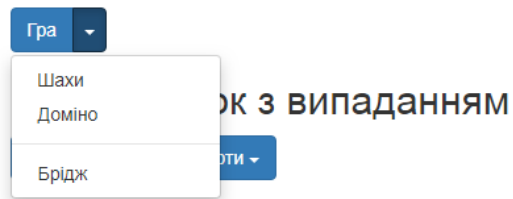
<div class="btn-group">
  <button type="button" class="btn btn-primary">JavaScript</button>
</div>

</div>

</div>

```

Split Buttons



```

<div class="container">

  <h2>Split Buttons</h2>

  <div class="btn-group">

    <button type="button" class="btn btn-primary">Гра</button>

    <button type="button" class="btn btn-primary dropdown-toggle" data-
toggle="dropdown">

      <span class="caret"></span>

    </button>

    <ul class="dropdown-menu" role="menu">

      <li><a href="#">Шахи</a></li>

      <li><a href="#">Доміно</a></li>

      <li class="divider"></li>

      <li><a href="#">Брідж</a></li>

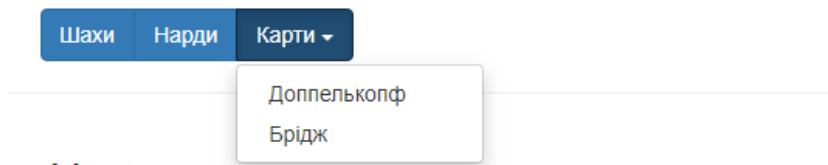
    </ul>

  </div>

</div>

```

Група кнопок з випаданням



```
<div class="container">
  <h2>Група кнопок з випаданням</h2>
  <div class="btn-group">
    <button type="button" class="btn btn-primary">Шахи</button>
    <button type="button" class="btn btn-primary">Нарди</button>
    <div class="btn-group">
      <button type="button" class="btn btn-primary dropdown-toggle" data-
toggle="dropdown">
        Карти <span class="caret"></span></button>
      <ul class="dropdown-menu" role="menu">
        <li><a href="#">Доппелькопф</a></li>
        <li><a href="#">Брідж</a></li>
      </ul>
    </div>
  </div>
</div>
</div>
```

Меню

Таблетки



```
<div class="container">
  <h2>Таблетки</h2>
  <ul style="margin:19px 0 15px 0;" class="test2 nav nav-pills">
    <li class="active"><a data-toggle="pill" href="#">Home</a></li>
    <li><a data-toggle="pill" href="#">Menu 1</a></li>
```

```

<li><a data-toggle="pill" href="#">Menu 2</a></li>
<li><a data-toggle="pill" href="#">Menu 3</a></li>
</ul></div>

```

Dynamic таблетки

Home Menu 1 Menu 2 Menu 3

HOME

Зміст домашньої сторінки

```

<div class="container">
  <h2>Dynamic таблетки</h2>
  <ul class="nav nav-pills">
    <li class="active"><a data-toggle="pill" href="#homePill">Home</a></li>
    <li><a data-toggle="pill" href="#menuPill1">Menu 1</a></li>
    <li><a data-toggle="pill" href="#menuPill2">Menu 2</a></li>
    <li><a data-toggle="pill" href="#menuPill3">Menu 3</a></li>
  </ul>
  <div class="tab-content">
    <div id="homePill" class="tab-pane fade in active">
      <h3>HOME</h3>
      <p>Зміст домашньої сторінки</p>
    </div>
    <div id="menuPill1" class="tab-pane fade">
      <h3>Menu 1</h3>
      <p>Зміст першого розділу.</p>
    </div>
    <div id="menuPill2" class="tab-pane fade">
      <h3>Menu 2</h3>
      <p>Зміст другого розділу.</p>
    </div>
    <div id="menuPill3" class="tab-pane fade">

```

```
<h3>Menu 3</h3>
```

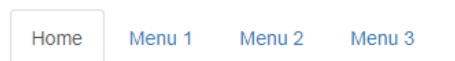
```
<p>Зміст третього розділу.</p>
```

```
</div>
```

```
</div>
```

```
</div>
```

Закладки



```
<div class="container">
```

```
<h3>Закладки</h3>
```

```
<ul style="margin:20px 0 22px 0;" class="test1 nav nav-tabs">
```

```
<li class="active"><a data-toggle="tab" href="#">Home</a></li>
```

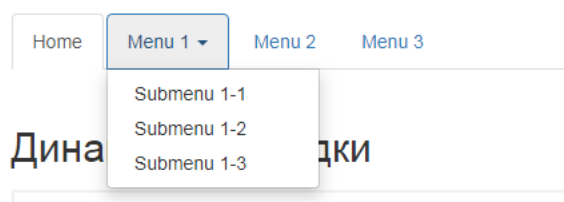
```
<li><a data-toggle="tab" href="#">Menu 1</a></li>
```

```
<li><a data-toggle="tab" href="#">Menu 2</a></li>
```

```
<li><a data-toggle="tab" href="#">Menu 3</a></li>
```

```
</ul>
```

Закладки з випаданням



```
<h3>Закладки з випаданням</h3>
```

```
<ul style="margin:20px 0 22px 0;" class="test1 nav nav-tabs">
```

```
<li data-toggle="tab" class="active"><a href="#">Home</a></li>
```

```
<li data-toggle="tab" class="dropdown">
```

```
<a class="dropdown-toggle" data-toggle="dropdown" href="#">Menu 1
```

```
<span class="caret"></span></a>
```

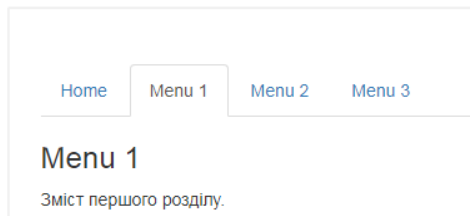
```
<ul class="dropdown-menu">
```

```

<li><a href="#">Submenu 1-1</a></li>
<li><a href="#">Submenu 1-2</a></li>
<li><a href="#">Submenu 1-3</a></li>
</ul>
</li>
<li data-toggle="tab"><a href="#">Menu 2</a></li>
<li data-toggle="tab"><a href="#">Menu 3</a></li>
</ul></div>

```

Динамічні закладки



```

<div class="container">
<h2>Динамічні закладки</h2>
<div style="margin-top:15px;margin-bottom:18px;border:2px solid
#f1f1f1;border-radius:2px;padding:25px;">
<ul style="margin:19px 0 18px 0;" class="nav nav-tabs test2">
  <li class="active"><a data-toggle="tab" href="#home">Home</a></li>
  <li><a data-toggle="tab" href="#menu1">Menu 1</a></li>
  <li><a data-toggle="tab" href="#menu2">Menu 2</a></li>
  <li><a data-toggle="tab" href="#menu3">Menu 3</a></li>
</ul>
<div class="tab-content">
  <div id="home" class="tab-pane fade in active">
    <h3>HOME</h3>
    <p>Зміст домашньої сторінки</p>
  </div>
  <div id="menu1" class="tab-pane fade">

```



```

<h3>Menu 1</h3>

<p>Зміст першого розділу.</p>
</div>

<div id="menu2" class="tab-pane fade">

  <h3>Menu 2</h3>

  <p>Зміст другого розділу.</p>
</div>

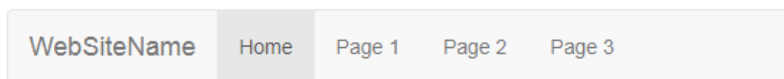
<div id="menu3" class="tab-pane fade">

  <h3>Menu 3</h3>

  <p>Зміст третього розділу.</p>
</div></div></div>

```

Горизонтальне меню



```

<h1>Горизонтальне меню</h1>

<nav class="navbar navbar-default">

  <div class="container-fluid">

    <div class="navbar-header">

      <a class="navbar-brand" href="#">WebSiteName</a>

    </div>

    <ul class="nav navbar-nav">

      <li class="active"><a href="#">Home</a></li>

      <li><a href="#">Page 1</a></li>

      <li><a href="#">Page 2</a></li>

      <li><a href="#">Page 3</a></li>

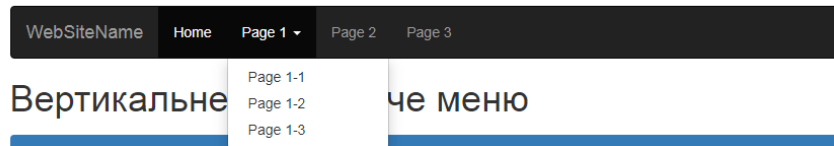
    </ul>

  </div>

</nav>

```

Горизонтальне випадаюче меню (інверсне)



```
<h1>Горизонтальне випадаюче меню (інверсне)</h1>
```

```
<nav class="navbar navbar-inverse">
```

```
  <div class="container-fluid">
```

```
    <div class="navbar-header">
```

```
      <a class="navbar-brand" href="#">WebSiteName</a>
```

```
    </div>
```

```
    <ul class="nav navbar-nav">
```

```
      <li class="active"><a href="#">Home</a></li>
```

```
      <li class="dropdown"><a class="dropdown-toggle" data-toggle="dropdown"
href="#">Page 1 <span class="caret"></span></a>
```

```
        <ul class="dropdown-menu">
```

```
          <li><a href="#">Page 1-1</a></li>
```

```
          <li><a href="#">Page 1-2</a></li>
```

```
          <li><a href="#">Page 1-3</a></li>
```

```
        </ul>
```

```
      </li>
```

```
      <li><a href="#">Page 2</a></li>
```

```
      <li><a href="#">Page 3</a></li>
```

```
    </ul>
```

```
  </div>
```

```
</nav>
```

Вертикальне випадаюче меню



```

<h1>Вертикальне випадаюче меню </h1>
<ul style="margin:16px 0 15px 0;" class="test2 nav nav-pills nav-stacked">
  <li data-toggle="pill" class="active"><a href="#">Home</a></li>
  <li data-toggle="pill" class="dropdown">
    <a class="dropdown-toggle" data-toggle="dropdown" href="#">Menu 1
    <span class="caret"></span></a>
    <ul class="dropdown-menu">
      <li><a href="#">Submenu 1-1</a></li>
      <li><a href="#">Submenu 1-2</a></li>
      <li><a href="#">Submenu 1-3</a></li>
    </ul>
  </li>
  <li data-toggle="pill"><a href="#">Menu 2</a></li>
  <li data-toggle="pill"><a href="#">Menu 3</a></li>
</ul></div>

```

Зображення

Закруглені вугли

.img-rounded class надає закругленість вуглів зображення:



```

<div class="container">
  <h2>Закруглені вугли</h2>
  <p>.img-rounded class надає закругленість вуглів зображення:</p>
  
</div>

```

Коло

.img-circle class обмежує зображення еліпсом:



```
<div class="container">
```

```
<h2>Коло</h2>
```

```
<p>.img-circle class обмежує зображення еліпсом:</p>
```

```

```

```
</div>
```

```
<div class="container">
```

Слайд

.img-thumbnail class створює слайд зображення:



```
<h2>Слайд</h2>
```

```
<p>.img-thumbnail class створює слайд зображення:</p>
```

```

```

```
</div>
```

Масштабування зображення під розмір вікна браузера

`.img-responsive` class дозволяє масштабувати зображення відповідно батьківського елемента (для ілюстрації треба змінити розмір вікна браузера):



```
<div class="container">
```

```
<h2>Масштабування зображення під розмір вікна браузера</h2>
```

```
<p>.img-responsive class дозволяє масштабувати зображення відповідно  
батьківського елемента (для ілюстрації треба змінити розмір вікна  
браузера):</p>
```

```

```

```
</div>
```

Іконки

На <http://getbootstrap.com/components/> лежить перелік іконок

Існує ресурс fontawesome.io безкоштовних іконок (їх треба завантажити та прописати відповідний стиль) при чому `fa-spin` дозволяє обертати будь-яку іконку.

Для анімації можна використати `fa-pulse` (повертає за 8 кроків), а також `fa-spinner`, `fa-refresh`, та `fa-cog`. А також можна збільшити розмір.



```
<p><i class="fa fa-spinner fa-spin"></i>
```







```
<i class="fa fa-circle-o-notch fa-spin fa-lg"></i>
```

```
<i class="fa fa-refresh fa-spin fa-2x"></i>
```

```
<i class="fa fa-cog fa-spin fa-3x"></i>
```

```
<i class="fa fa-spinner fa-pulse fa-4x"></i></p>
```

Іконку можна повернути та відзеркалити

 normal
 fa-rotate-90
 fa-rotate-180
 fa-rotate-270
 fa-flip-horizontal
 fa-flip-vertical

`<i class="fa fa-hand-pointer-o"></i> normal
`

`<i class="fa fa-hand-pointer-o fa-rotate-90"></i> fa-rotate-90
`

`<i class="fa fa-hand-pointer-o fa-rotate-180"></i> fa-rotate-180
`

`<i class="fa fa-hand-pointer-o fa-rotate-270"></i> fa-rotate-270
`

`<i class="fa fa-hand-pointer-o fa-flip-horizontal"></i> fa-flip-horizontal
`

`<i class="fa fa-hand-pointer-o fa-flip-vertical"></i> fa-flip-vertical
`

Завдання.

Написати html-форму, на якій повинні бути обов'язкові елементи

	Email address		Password
-----------------------------------------------------------------------------------	---------------	-----------------------------------------------------------------------------------	----------

та меню


 Home

 Library

 Applications

 Settings

Контрольні питання.

1. Чим відрізняється `class="container-fluid"` від `class="container"`?
2. Яка різниця між `class="col-lg-8"`, `class="col-md-8"` та `class="col-sm-6"`?
3. Як створити іконку  ?

РЕКОМЕНДОВАНА ЛІТЕРАТУРА

Базова

1. Айзенменгер, Р. HTML 3.2/4.0. Справочник, М.: Бином, 1998, 368 с.
2. Дебольт, Вирджиния. HTML и CSS. Совместное использование. НТ Пресс, 2006, 512 с.
3. Дунаев, В.В. HTML, скрипты и стили. СПб: БХВ, 2006, 832 с.
4. Матросов, А.В., Сергеев, А.О., Чаунин, М.П. Html 4.0, СПб: БХВ, 2005, 672 с.
5. Пфаффенбергер, Брайан; Шафер, Стивен; Уайт, Чак и др. HTML, XHTML и CSS. Библия пользователя. М.: Вильямс; Издание 3-е, 2007, 752 с.
6. Штайнер, Г. HTML/XML/CSS. Лаборатория Базовых Знаний; Издание 2-е, перераб. 2005, 510 с.

Допоміжна

7. Спортак Марк Ф. и др. Компьютерные сети. "ДиаСофт", 1999, 430 с.
8. Нильсен Якоб. Веб-дизайн. Символ-плюс, СПб, 2002, 512 с.

ІНФОРМАЦІЙНІ РЕСУРСИ

9. <http://www.dstu.dp.ua>
10. <http://www.codecademy.com/>
11. <https://www.codeschool.com/>

МЕТОДИЧНЕ ЗАБЕЗПЕЧЕННЯ

12. Шумейко А.А. Web-программирование
<http://178.219.93.18:8080/Portal/Data/3/19/NeT/index.html>

Додаток А

Порядок виконання та захисту лабораторних робіт

1. Лабораторні роботи виконуються студентами на лабораторних заняттях з дисципліни в обчислювальному залі з використанням ПК та в часи самостійної роботи.
2. Для студентів очної форми навчання виконання всіх лабораторних робіт даного навчального видання є обов'язковим. Студенти заочної форми навчання виконують лише роботи, передбачені робочою програмою навчальної дисципліни.
3. Перелік і кількість запропонованих до розв'язку задач визначається викладачем, який веде лабораторні заняття, відповідно до робочої програми дисципліни.
4. Перед виконанням роботи студент зобов'язаний опрацювати та стисло законспектувати теоретичні основи, відповідні розділи рекомендованої літератури (список додається) і конспекту лекцій, а також ознайомитися з прикладами розв'язку типових задач за темою лабораторної роботи.
5. Підготовлена лабораторна робота подається викладачу на перевірку у вигляді звіту, обов'язковими розділами якого є: тема роботи, стислий конспект теоретичних основ за темою роботи, умова задачі (або задач), розв'язок задачі у вигляді блок-схеми (за необхідністю), роздрукованого на принтері тексту програми та протоколу її реалізації.
6. Звіт бажано оформлювати на папері формату А4 (297x210мм) з однієї сторони. Якщо звіт виконується у зошиті у рукописному вигляді, то тексти програм та протоколи їх реалізації мають бути роздруковані на принтері та вклеєні у зошит.
7. Необхідно також подати до перевірки рекомендований викладачем носій інформації з текстами налагоджених програм та їх виконавчими файлами.
8. Робота, в якій виявлено суттєві помилки або недоробки, повертається студенту для доопрацювання та усунення письмово зазначених викладачем зауважень. Доопрацьована або перероблена робота подається для повторної перевірки без видалення раніше зазначених зауважень.
9. Захист роботи здійснюється під час лабораторних занять і консультацій. За результатами захисту за кожну роботу виставляється оцінка за спеціальною шкалою оцінювання, наведеною у робочій програмі з дисципліни.
10. Бали, отримані за окремі роботи, формують загальну суму балів за практикум, яка вказується на титульному аркуші звіту за підписом викладача та враховується у підсумкову оцінку за модуль та семестр.

Додаток Б

Порядок виконання та захисту домашніх контрольних робіт (для студентів-заочної форми навчання)

1. Домашня контрольна робота (ДКР) виконується студентами самостійно на основі засвоєного теоретичного матеріалу та набуття практичних навичок з навчальної дисципліни.
2. Формування індивідуальних завдань до ДКР здійснюється відповідальним за дисципліну викладачем на базі даного навчального видання. Завдання містять теоретичні запитання та практичні задачі.
3. Завдання до ДКР студент отримує особисто від викладача під час установчої сесії або на кафедрі (у разі відсутності під час сесії). Консультування з приводу виконання ДКР здійснюється у міжсесійний період у відповідності до графіку консультацій. Приклади розв'язку типових задач, а також перелік допоміжної літератури до виконання ДКР містяться у даному навчальному виданні.
4. Виконана за своїм варіантом ДКР оформлюється у вигляді звіту та подається на кафедру для перевірки разом з електронним носієм, де зберігаються виконані на комп'ютері задачі. Звіт бажано оформлювати на папері формату А4 (297х210мм) з однієї сторони. Якщо звіт виконується у зошиті у рукописному вигляді, то тексти програм та протоколи їх реалізації мають бути роздруковані на принтері та вклеєні у зошит. Рекомендований викладачем електронний носій інформації повинен містити файли з текстами налагоджених програм та їх виконавчі файли.
5. Робота, в якій виявлено суттєві помилки або недоробки, повертається студенту для доопрацювання та усунення письмово зазначених викладачем зауважень. Доопрацьована або перероблена робота подається для повторної перевірки без видалення раніше зазначених зауважень.
6. Зарахування правильно виконаної ДКР здійснюється шляхом її захисту за комп'ютером. За результатом захисту ДКР студент отримує допуск до семестрового контролю.

НАВЧАЛЬНЕ ВИДАННЯ

Методичні вказівки до лабораторних робіт з дисципліни «Програмування Інтернет» для здобувачів вищої освіти першого (бакалаврського) рівня зі спеціальності 121 «Інженерія програмного забезпечення»

Укладач:
проф. Шумейко Олександр Олексійович

Підписано до друку 16.05. 2019 р.
Формат __А5__
Обсяг 3,3 др. арк.
Наклад ____15____ прим. Замовлення 96
51918, м. Кам'янське, вул. Дніпробудівська, 2

