Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

**Лабораторна робота № 1**

з дисципліни «Основи клієнтської розробки»

Тема: «Системи контролю версій. Git. Проєкт. Структура проєкту.»

|  |  |
| --- | --- |
| Виконав:  студент групи ІА-24  Тильна Марія.  Дата здачі 20.02.2024  Захищено з балом \_\_\_\_\_\_\_\_\_\_\_ | Перевірила:  ст. вик. кафедри ІСТ  Хмелюк Марина Сергіївна |

**Лабораторна робота №1**

**1. Тема** Системи контролю версій. Git. Проєкт. Структура проєкту.

**2. Завдання**

1) Встановити Git на комп’ютер. Створити каталог для проєкту.Ініціалізувати Git директорію у своєму проєкті (git init).

2) Створити 4 сторінки відповідно до обраної тематики з розширенням

**.html.** Перша сторінка має назву - **index.html**. В вказати автора документа. Кожна сторінка повинна мати назву <title></title>. Кожна сторінка повинна мати заголовок <head></head>.

1. Додати створені сторінки під контроль Git на комп’ютері.
2. Зареєструватись на **GitHub**. Створити репозиторій. Завантажити

проєкт на віддалений репозиторій (на **GitHub**).

**3. Теоретичні відомості**

Системи контролю версіями (Version Control Systems, VCS) важливий аспект розробки сучасного ПЗ.

VCS надає такі можливості:

- підтримку зберігання файлів у репозиторії;

- підтримку історії версій файлів у репозиторії;

- знаходження конфліктів при зміні вихідного коду та забезпечення

синхронізації при роботі в багатокористувацькому середовищі розробки;

- відстеження змін авторів.

Звичайний цикл роботи розробника з СКВ виглядає так:

- **Оновлення робочої копії.** Розробник виконує операцію оновлення

робочої копії з сервера на скільки можливо

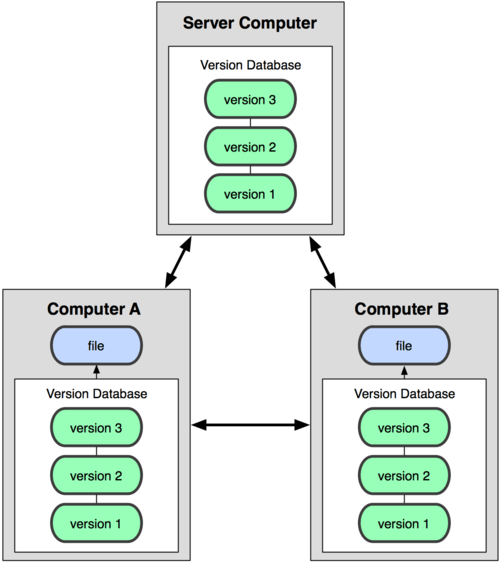
- **Модифікація проекту.** Розробник локально модифікує проект,

змінюючи файли, що входять до нього, в робочій копії.

- **Фіксація змін.** Завершивши черговий етап роботи, розробник фіксує

свої зміни, передаючи їх на сервер. Перед використанням VCS може вимагати від розробника оновлення.

**Розподілені системи контролю версій**



**Розподілені системи контролю версій (Git)**

***Переваги:***

- Так як кожного разу, коли клієнт забирає свіжу версію файлів, він

створює собі повну копію всіх даних, то у разі збоїв на сервері, через який йшла робота, будь-який **клієнтський репозиторій може бути скопійовано назад на сервер, щоб відновити базу даних**.

- Можливість **працювати з кількома віддаленими репозиторіями.**

Таким чином, можна одночасно працювати по-різному з різними групами людей у рамках одного проекту.

**Git** – це розподілена система контролю версій, яка виникла внаслідок

розробки ядра Linux.

Git моделює свої дані як набір знімків, зберігаючи посилання на них.

Git має цілісність, все перевіряється за контрольною сумою перед

збереженням і з цього моменту ідентифікується за цією контрольною сумою.

Git має три основні стани, в яких можуть знаходитися ваші файли:

зафіксовано, змінено і підготовлено.

- **Зафіксовано (Підтверджено)**: дані безпечно зберігаються локально.

- **Змінено**: ще не підтверджено.

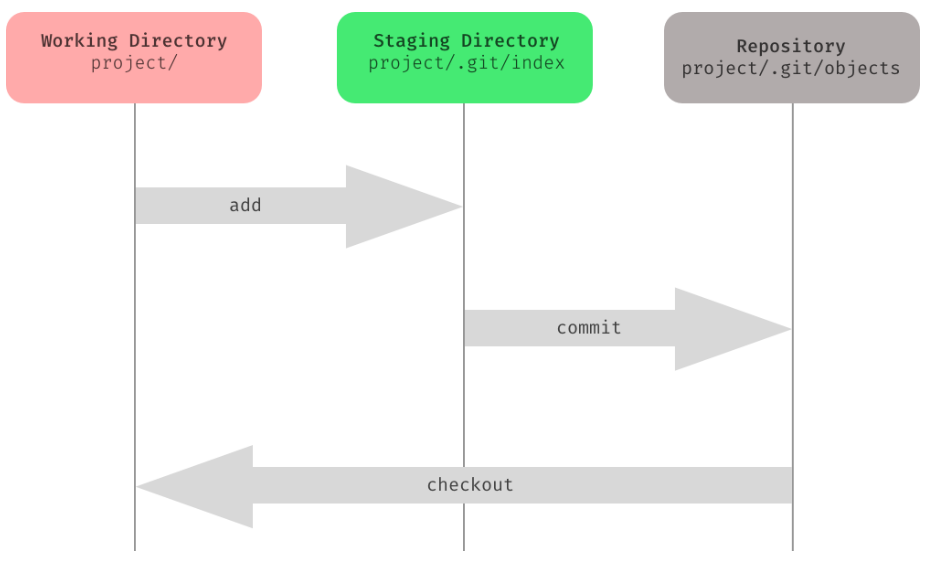
- **Підготовлено**: позначений для переходу до наступного коміту.

Ці стани приводять до трьох розділів:

**Робочий каталог**: це копія версії проекту.

**Staging Area**: простий файл, в якому зберігається інформація про те, що буде у наступному коміті.

**Каталог Git**: зберігає метадані та об'єкти бази даних для вашого проекту

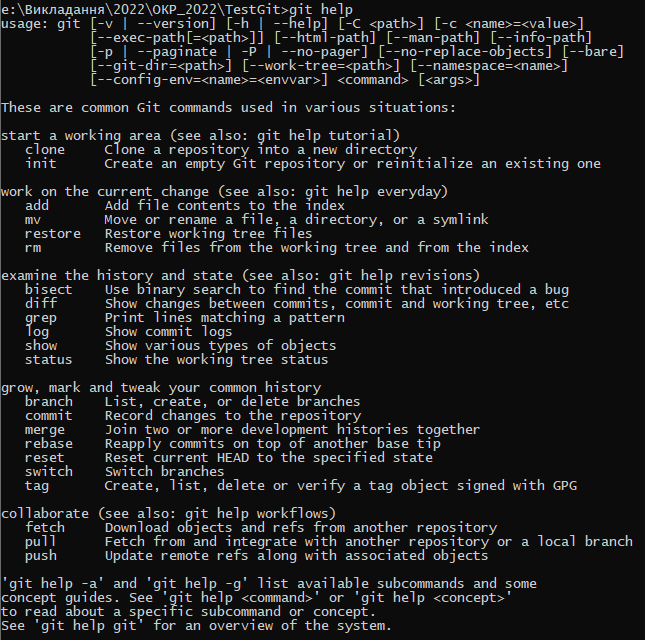


## Установка Git

Необхідно встановити Git. можна пройти за цим посиланням на офіційне джерело <https://git-scm.com/downloads>



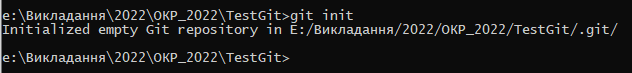
Найголовніша команда Git help - виводить список усіх команд із коротким описом.

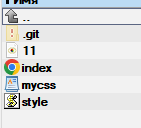


## Структура директорії .git/

робота з Git буде починається з того, що потрібно проініціалізувати Git директорію у своєму проєкті. Це робиться за допомогою команди:

## git init

Її необхідно ввести в корені вашого проєкту. Це створить у поточному каталозі новий підкаталог .git



з наступним вмістом:



У цій директорії буде вся конфігурація Git та історія проєкту. За бажанням можна редагувати ці файли вручну, вносячи необхідні зміни в історію проекту.

## config

Цей файл містить налаштування Git репозиторію. тут можна зберігати email та ім'я користувача.

## description

Цей файл призначений для GitWeb - це веб-інтерфейс, написаний для перегляду Git-репозиторія використовуючи веб-браузер і містить інформацію про проект (назва проекту та його опис).

## hooks

У цьому каталозі Git надає набір скриптів, які можуть автоматично запускатись під час виконання git команд. Наприклад, можна написати скрипт, який редагуватиме повідомлення комміту відповідно до ваших вимог.

## info

Каталог info містить файл exclude, в якому можна вказувати будь-які файли, і Git не додаватиме їх у свою історію.

## refs

Каталог refs зберігає копію посилань на об'єкти коммітів у локальних і віддалених гілках.

## logs

Каталог logs зберігає історію проекту для всіх гілок у вашому проекті.

## objects

Каталог objects зберігає у собі BLOB об'єкти, кожен із яких проіндексований унікальним SHA (Secure Hash Algorithm).

## index

Проміжна область з метаданими, такими як тимчасові мітки, імена файлів, а також файли SHA, які вже упаковані Git. У цю область потрапляють файли, над якими ви працювали, під час виконання команди git add (додавання файлів).

## head

Файл містить посилання на поточну гілку, в якій ви працюєте

## orig\_head

Щоразу під час злиття до цього файлу потрапляє SHA гілка, з якою проводилося злиття.

## fetch\_head

Файл зберігає посилання у вигляді SHA на гілки, які брали участь у git fetch (завантаження комітів, файлів, посилань з віддаленого репозиторія в локальний)

## merge\_head

Файл зберігає посилання у вигляді SHA на гілки, які брали участь у git merge (злиття)

## commit\_editmsg

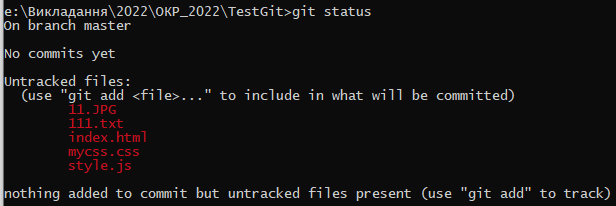
Файл містить останнє введене вами повідомлення комміту.

Робота над проектом з використанням СКВ обмежена певними діями:

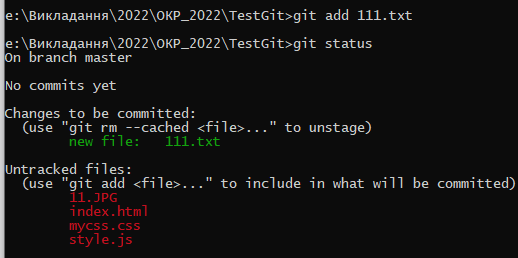
1. Внести зміни до проекту;
2. Додати зміни до індексу (staging area) - git add (ви повідомляєте Git, які саме зміни повинні бути занесені в історію);
3. Закомітити зміни - git commit (зберегти зміни до історії проекту);
4. Запушити - git push (надіслати результати роботи на віддалений сервер, щоб інші розробники теж мали до них доступ).

Створимо новий файл 111.txt і перевіримо стан директорії .

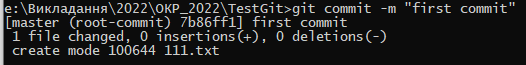
Команда **git status** відображає стан директорії та індексу (staging area). Це дозволяє визначити, які файли в проекті відстежуються Git, а також які зміни будуть включені до наступного коміту.



**git add** 111.txt



Файл додано до індексу. Тепер можна закомітити внесені зміни та додати коментар. **git commit -m "first commit"**



Відредагуємо файл і перевіримо статус

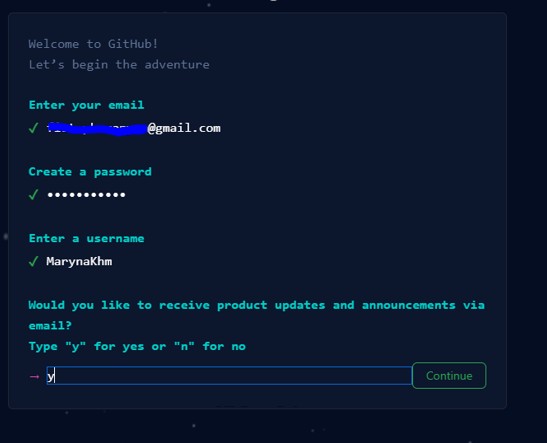


Git повідомляє, що ми маємо змінений файл 111.txt. І тепер нам потрібно додати його в індекс і потім закомітити.

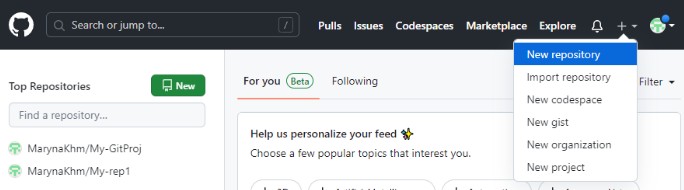
Ми все записуЄмо та зберігаємо зміни на своїй машині, тепер нам потрібно відправити версію нашої історії на віддалений сервер.

Можна скористатись репозиторієм на **GitHub**.

## Реєстрація на [GitHUB.com](https://github.com/)

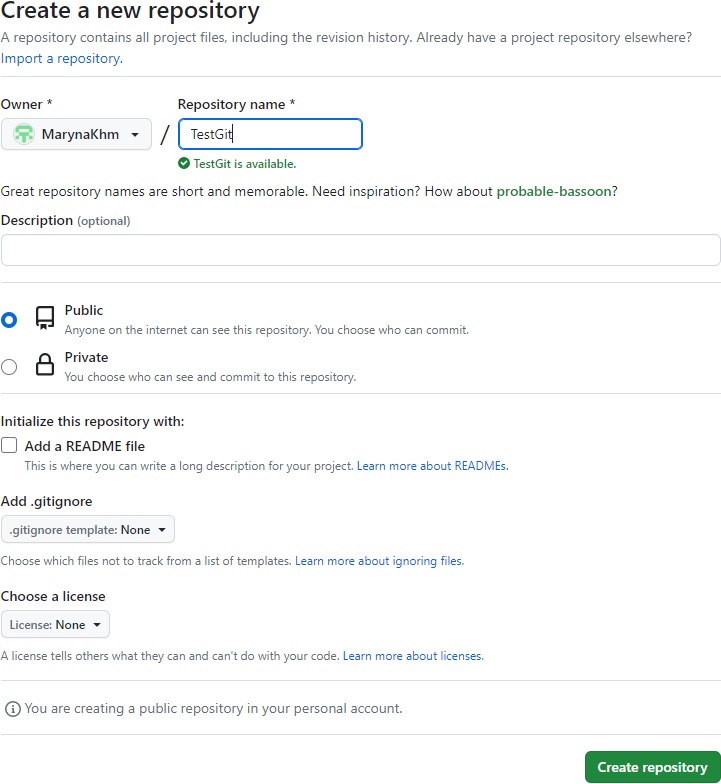


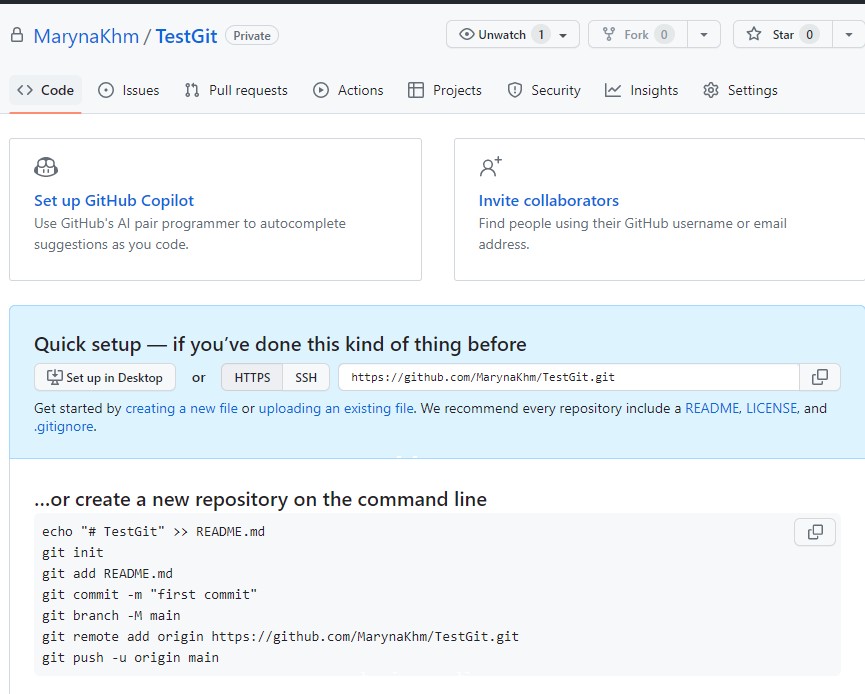
**Створення репозиторія**



Ввести коротку назву, видимість сховища, опис (опціонально), натиснути

## Create repository





Далі необхідно додати віддалений репозиторій у ваш Git:

**git remote add <remote\_name> <remote\_repo\_url>** - команда створює віддалене сховище з посиланням на локальний репозиторій. З цього моменту можна звернутися до видаленого репозиторію через це посилання.

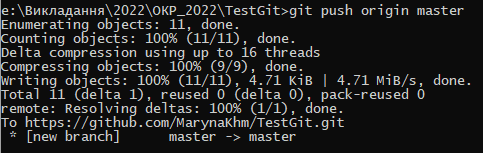


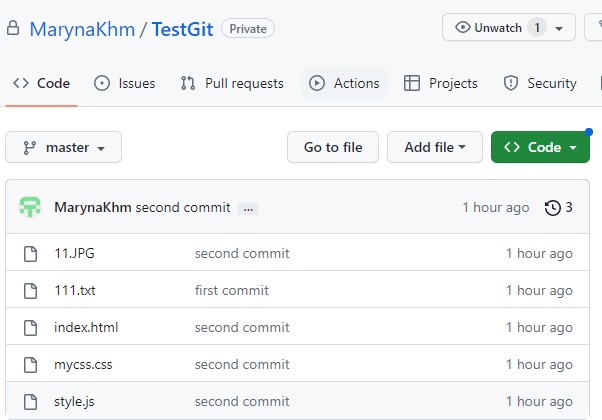
«origin» є коротким іменем для віддаленого репозиторію, на який він буде посилатись (може бути будь-яке ім’я).

відправимо результат нашої роботи в репозиторій.

**git push** origin master

Тепер історія змін вашого проєкту буде зберігатися у віддаленому репозиторії.

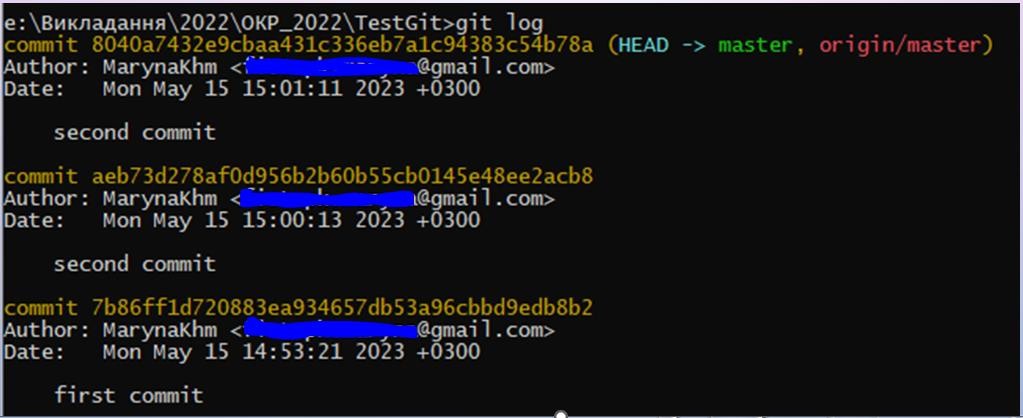




**Гілка в Git** - це простий переміщуваний покажчик на один з коммітів. За замовчуванням, ім'я основної гілки в Git — master. Як тільки ви почнете створювати комміти, гілка master завжди вказуватиме на останній комміт.

Для перегляду історії в Git є ряд команд:

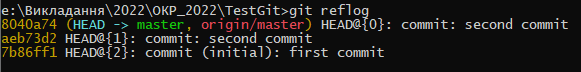
**git log** - призначена для відображення всієї історії вашого проєкту, Дозволяє дізнатися, які зміни ви внесли раніше.



**git show** - використовується для відображення повної інформації про будь- який об’єкт у Git, комміт або гілку. За замовчуванням git show відображає інформацію коміту, на який у даний момент часу вказує HEAD.



**git reflog** - виводить упорядкований список комітів, на який вказує HEAD (відображає історію всіх ваших переміщень по проєкту). Основна перевага цієї команди полягає в тому, що якщо ви випадково видалили частину історії або відкотилися назад, ви зможете переглянути момент втрати потрібної вам інформації та відкотитись назад. git reflog зберігає свою інформацію на вашій машині окремо від комітів, тому при видаленні будь-чого в історії можна знайти її в git reflog.



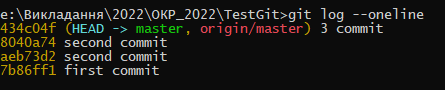
**git reset** - дозволяє відкотити проєкт до визначеної точки. Цю команду можна використовувати з трьома параметрами:

**git reset --soft <commit>** - вміст вашого індексу, а також робочої директорії, залишаються незмінними. якщо ми відкотимося назад на пару комітів, ми змінимо посилання вказівника HEAD на вказаний комміт і всі зміни, які були до цього внесені, покажуться в індексі.

**git reset --mixed <commit>** - змінимо посилання вказівника HEAD, але всі попередні зміни в індекс не попадуть, а будуть відображатися як не занесені в індекс. Це дає можливість внести в індекс тільки ті зміни, які нам необхідні.

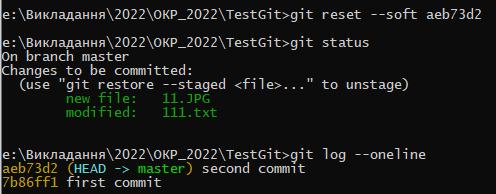
**git reset --hard <commit>** - знову змінимо посилання вказівника HEAD, але всі попередні зміни не попадуть ні в індекс, ні в зону шуканих файлів, ми повністю зітремо всі зміни, які внесли раніше.

Переглянемо історію



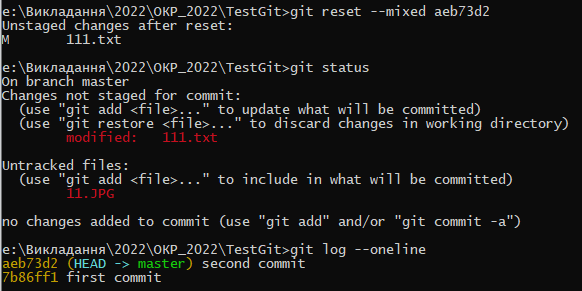
Зроблено 4 коміти

Відкотимось до 2 коміта **git reset --soft**



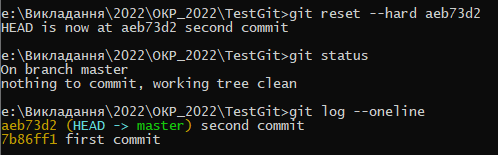
Вказівник HEAD перемістився на другий комміт, а стан індексу залишився незмінним.

Відкотимось до 2 коміта **git reset --mixed**



У цьому випадку вказівник знову перемістився на другий комміт, але попередні зміни потрапили в зону відстежуваних файлів. Це означає, що тепер ми можемо вирішити - залишити ці зміни, додавши їх в індекс, або позбутися них.

Відкотимось до 2 коміта **git reset --hard**



Вказівник HEAD перемістився на другий комміт, а всі попередні зміни були стерті, що видно по порожньому індексу та зоні відстежуваних файлів.

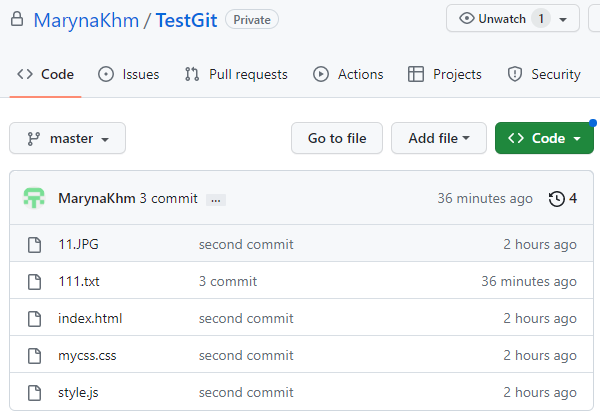
## Гілкування в Git

Гілкування означає, що у вас є можливість працювати над різними версіями проєкту. Тобто, якщо раніше історія вашої розробки була прямою послідовністю комітів, то тепер вона може розійтися в певних точках. Це дуже корисна функція з багатьох причин, наприклад для взаємодії кількох розробників.

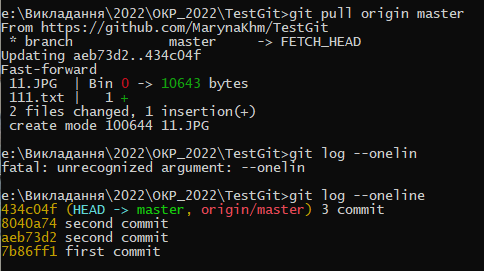
Стан робочої директорії (локальний репозиторій)



У віддаленому репозиторії



Команда **git pull** використовується для вилучення та завантаження вмісту з віддаленого репозиторію та негайного оновлення локального репозиторію цим вмістом.

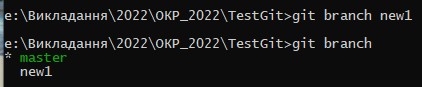


Git за замовчуванням під час ініціалізації створює гілку master і вже веде свою роботу в ній. Перевірити можна **git branch**:



Для створення нової гілки, її потрібно створити

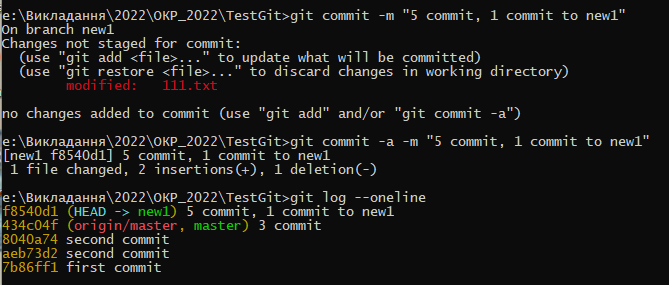
## git branch <branch\_name>



\* вказує на поточну гілку, в якій ми працюємо. Для того, щоб перейти на іншу гілку, є команда **git checkout <branch\_name>**



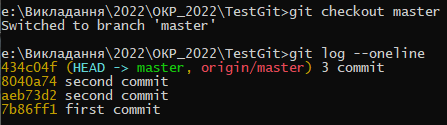
Зробимо зміни в файлі 111.txt, зробимо коміт



Перейдемо в гілку master

## git checkout master

Переглянемо історію **git log --oneline**, і переконаємось, що все залишилося без змін.

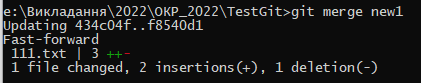


Окрім розділеної історії в Git (гілкування), ми також можемо об’єднати одночасно два потоки розробки. Це означає, що нашу роботу в новій гілці ми можемо злити в master. Такий процес злиття можна виконати за допомогою команди

## git merge <branch\_name>

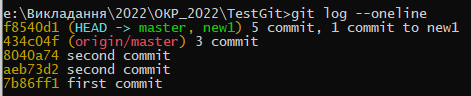
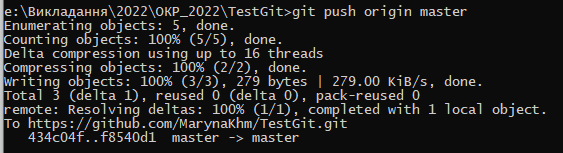
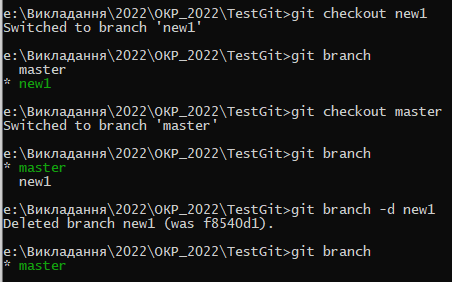
якщо ми хочемо злити зміни з гілки “new1” у гілку “master”, нам необхідно перейти на гілку “master” і в ній виконати:

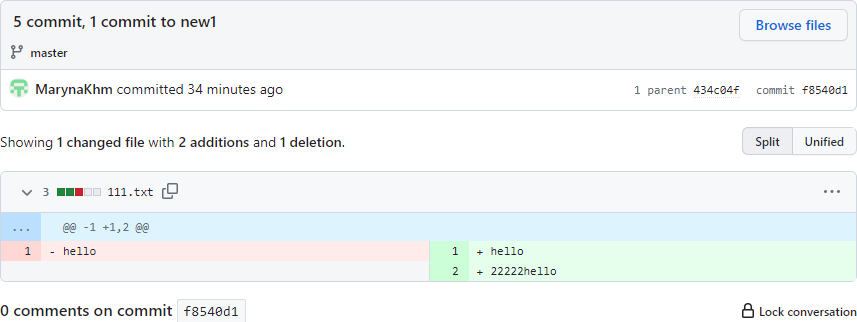
## git merge new1



Тепер непотрібної гілки можна позбутися і видалити її за допомогою команди

## git branch -d <branch\_name>.



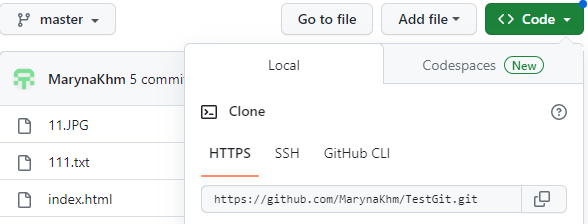


коду. собі:

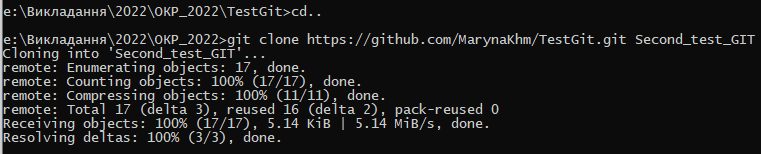
Що потрібно зробити, щоб інша людина отримала всі ваші зміни?

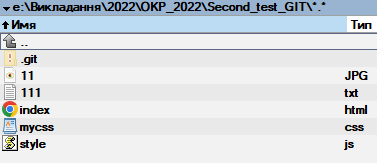
Для цього знадобиться GitHub або будь-який інший сервіс для зберігання Якщо у людини раніше не було проєкту, то їй доведеться його «клонувати»

## git clone <URL repo>



Адресу репозиторію на GitHub можна отримати, натиснувши на зелену кнопку Code.

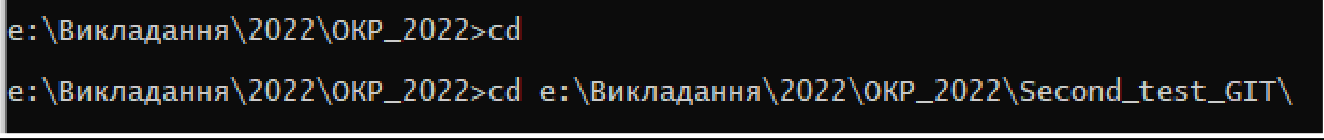


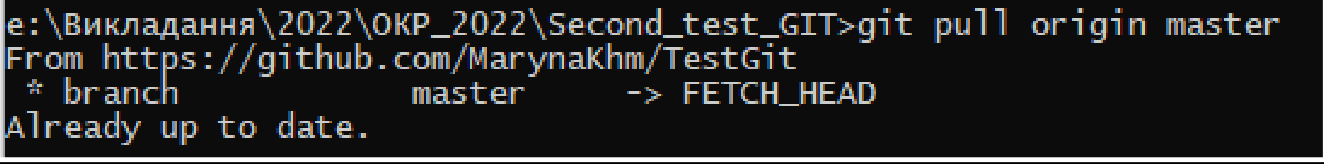
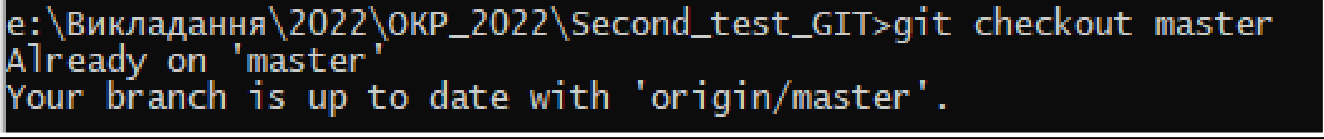


Перед тим, як створити новий функціонал і нову гілку, варто оновити master.

Для цього потрібно знаходитися в цій гілці і виконати наступну команду:

## git checkout master git pull origin master





Які проблеми можуть виникнути при злитті?

Git старається автоматично зливати зміни, однак це не завжди можливо. Іноді виникають конфлікти.

Наприклад, коли в двох гілках були зміни в одному і тому ж рядку коду. Якщо таке сталося, то необхідно вирішити конфлікт вручну.

## git checkout my1 git merge master

Якщо конфлікт, то буде відповідне повідомлення

## Auto-merging 111.txt

**CONFLICT (content): Merge conflict in 111.txt**

## Automatic merge failed; fix conflicts and then commit the result.

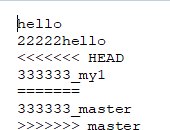
Для перегляду вмісту файлу можна скористатись командою **cat.**

У файлі можна побачити наступні рядки

<<<<<<< HEAD

=======

>>>>>>> master



Ці нові рядки можна розглядати як «роздільники конфлікту». Рядок ======= є «центром» конфлікту.

Весь вміст між цим центром і рядком <<<<<<< HEAD знаходиться в поточній гілці my1, на яку посилається покажчик HEAD.

А весь вміст між центром і рядком >>>>>>> master є вмістом гілки, звідки потрібно зливати.

Після внесення потрібних змін потрібно додати файл через **git add <file name>** як змінений і створити новий commit: **git add 111.txt**

## git commit -m "fixed conflict“

Переглянути зміни відносно двох гілок можна командою: **git diff <source branch> <target branch>**

**HTML** - це мова розмітки, який представляє прості правила оформлення і компактний набір структурних і семантичних елементів розмітки (тегів).

HTML дозволяє описувати спосіб представлення логічних частин документа (заголовки, абзаци, списки і т.д.) і створювати веб-сторінки різної складності.

Не використовуйте кириличні назви файлів і папок - давайте їм англійські назви! Перша сторінка завжди носить назву index.html.

Кожен тег призначений для вирішення певної задачі: роботи з текстом, посиланнями, графікою, таблицями і т.д.

Структура html-документа:



HTML-документ складається з тексту, який являє собою інформаційний вміст і спеціальних засобів мови HTML - тегів розмітки, які визначають структуру і зовнішній вигляд документа при його відображенні браузером. Структура HTML- документа досить проста:

1. Опис документа починається з вказівки його типу (секція DOCTYPE).
2. Текст документа полягає в тег <html>. Текст документа складається з заголовка і тіла, які виділяються відповідно тегами <head> і <body>.

* У заголовку (<head>) вказують назву HTML-документа і інші параметри, які браузер буде використовувати при відображенні документа.
* Тіло документа (<body>) - це та частина, в яку поміщається власне вміст HTML-документа. Тіло включає призначений для відображення текст і керуючу розмітку документа (теги), які використовуються браузером.

Наявність секції DOCTYPE дозволяє вказати браузеру, який тип документа йому належить розбирати, тобто, які вимоги потрібно виконувати при обробці

гіпертексту.

Приклади DOCTYPE:

* <! DOCTYPE HTML PUBLIC "- // W3C // DTD HTML 4.01 Frameset // EN"

["http:/](http://www.w3.org/TR/html4/frameset.dtd)/[www.w3.org/TR/html4/frameset.dtd](http://www.w3.org/TR/html4/frameset.dtd)">

Гіпертекстовий документ в форматі HTML 4.01, що містить фрейми.

* <! DOCTYPE HTML PUBLIC "- // W3C // DTD HTML 4.01 // EN"

["http:/](http://www.w3.org/TR/html4/strict.dtd)/[www.w3.org/TR/html4/strict.dtd](http://www.w3.org/TR/html4/strict.dtd)">

Гіпертекстовий документ в форматі HTML 4.01 із суворим синтаксисом (тобто не використані застарілі і не рекомендовані теги).

* <! DOCTYPE HTML PUBLIC "- // W3C // DTD HTML 4.01 Transitional // EN" "<http://www.w3.org/TR/html4/loose.dtd>">

Гіпертекстовий документ в форматі HTML 4.01 з нестрогим ( «перехідним») синтаксисом (тобто використані застарілі або які не рекомендовані теги і атрибути).

* <! DOCTYPE HTML> оголошення для документів HTML5.

Стандарт вимагає, щоб секція DOCTYPE була присутня в документі, тому що це дозволяє прискорити і поліпшити обробку гіпертексту. Це досягається за рахунок того, що браузер може не робити припущень про те, як інтерпретувати теги, а звіритися зі стандартним визначенням (файлом .dtd).

Заголовок призначений для розміщення метаінформації, яка описує веб- документ як такий.

Мета-тег HTML - це елемент розмітки html, що описує властивості документа як такого (метадані). Призначення мета-тега визначається набором його атрибутів, які задаються в тезі <meta>.

Мета-теги розміщують в блоці <head> ... </ head> веб-сторінки. Вони не є обов'язковими елементами, але можуть бути дуже корисні.

Приклад опису метаданих:

<head>

<meta name = "author" content = "рядок"> - автор веб-документа

<meta name = "date" content = "дата"> - дата останнього зміни веб-сторінки

<meta name = "copyright" content = "рядок"> - авторські права

<meta name = "keywords" content = "рядок"> - список ключових слів

<meta name = "description" content = "рядок"> - короткий опис (реферат)

<meta name = "ROBOTS" content = "NOINDEX, NOFOLLOW"> - заборона на індексування

<meta http-equiv = "content-type" content = "text / html; charset = UTF-8"> - тип і кодування

<meta http-equiv = "expires" content = "число"> - управління кешуванням

<meta http-equiv = "refresh" content = "число; URL = адреса"> - перенаправлення

</ head>

Блок <body> містить те, що потрібно показати користувачеві: текст, зображення, впроваджені об'єкти тощо.

<html> ... </ html> - контейнер гіпертексту

<head> ... </ head> - контейнер заголовка документа

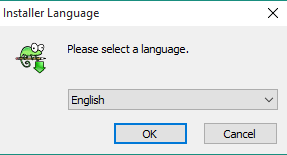
<title> ... </ title> - назва документа (те, що відображається в заголовку вікна браузера)

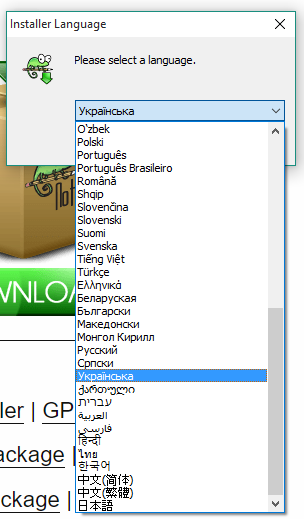
<body> ... </ body> - контейнер тіла документа

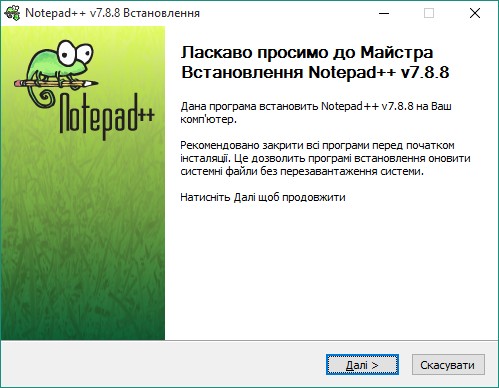
## Середовище розробки

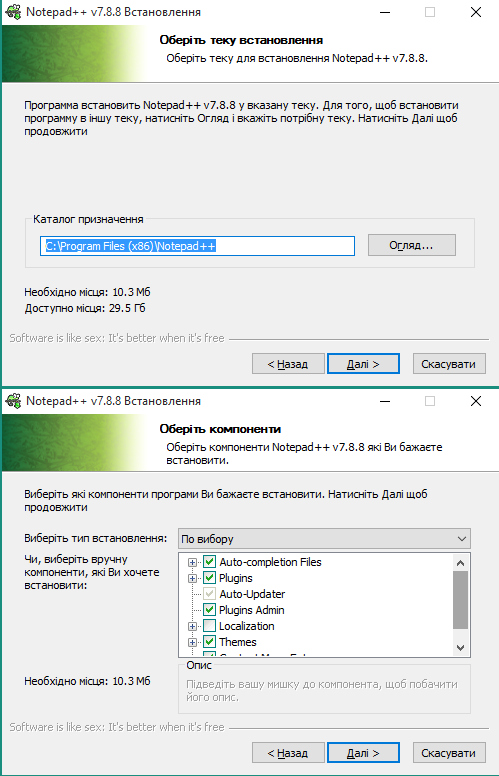
Створити html-документ можна в будь-якому текстовому редакторі, зберегти файл з розширенням .html

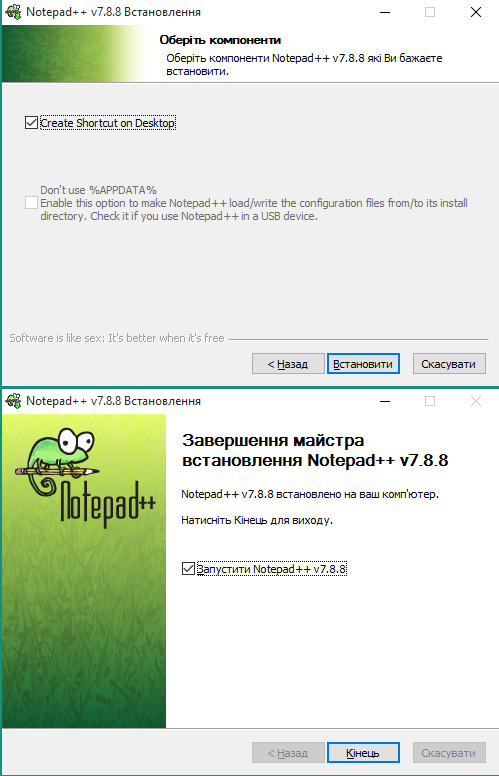
Установка та налаштування **Notepad++**

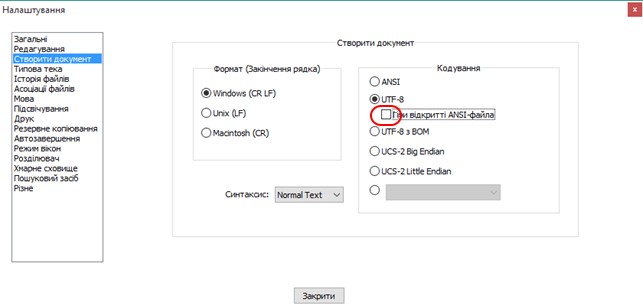


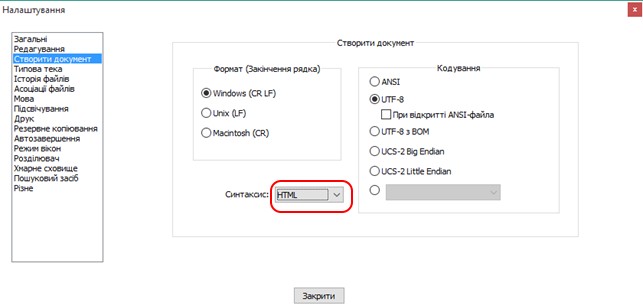


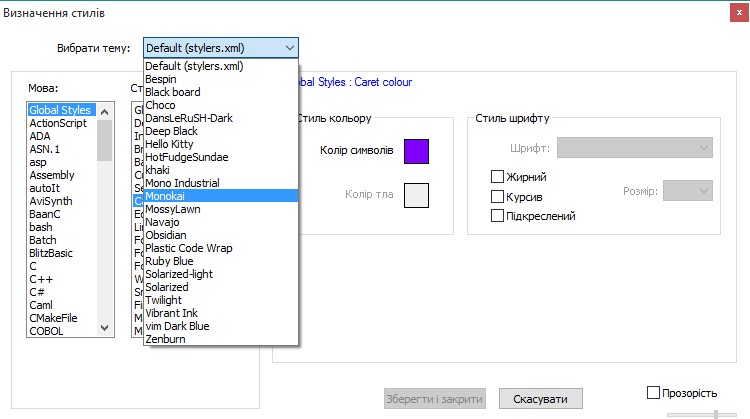


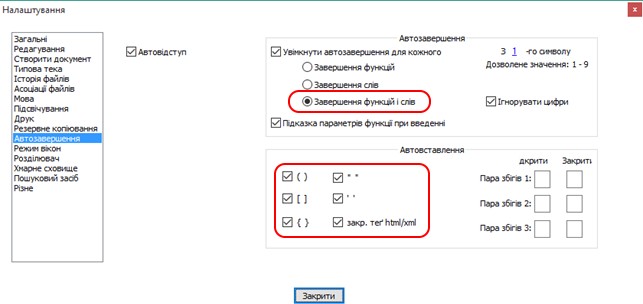












**4. Результати**

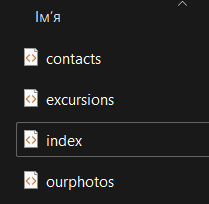
1) Встановила Git на комп’ютер.



Створила каталог для проєкту та ініціалізувала Git директорію у своєму проєкті (git init).



1. Створила 4 сторінки відповідно до обраної тематики “Екскурсії” з розширенням **.html.**



<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Головна</title>

    <meta name="author" content="Tylna Mariia">

</head>

<body>

    <h1>Ласкаво просимо</h1>

    <!-- Вміст головної сторінки -->

</body>

</html>

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Екскурсії</title>

</head>

<body>

    <h1>Наші екскурсії</h1>

    <!-- Вміст сторінки екскурсій -->

</body>

</html>

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Наші фото</title>

</head>

<body>

    <h1>Сторінка наших фото</h1>

    <!-- Вміст сторінки наших фото -->

</body>

</html>

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Контакти</title>

</head>

<body>

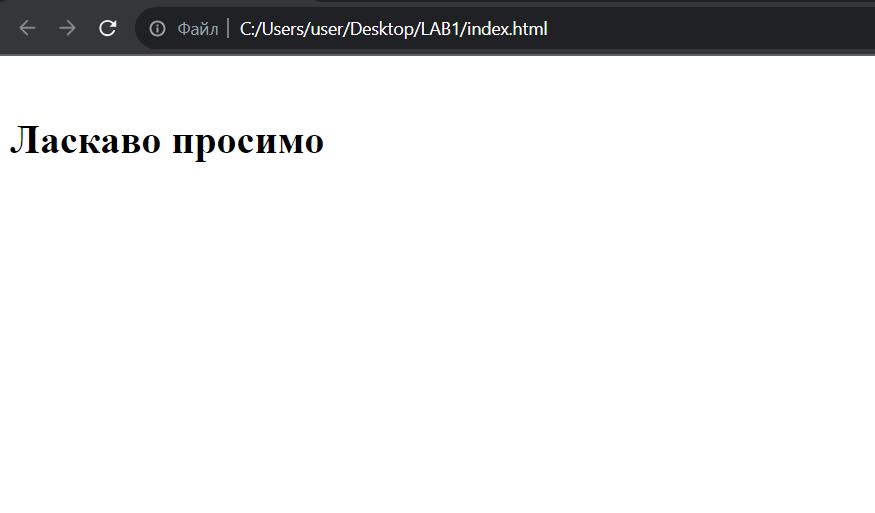
    <h1>Наші контакти</h1>

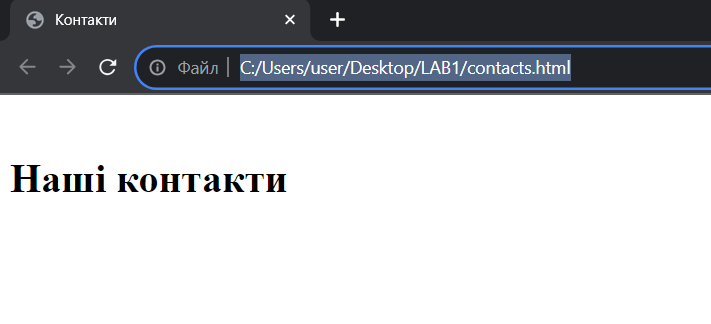
    <!-- Вміст сторінки контактів -->

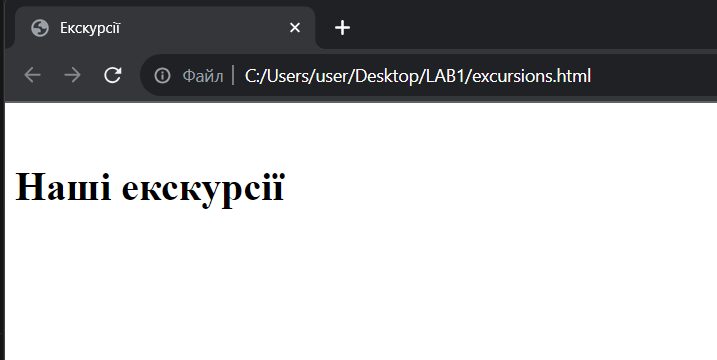
</body>

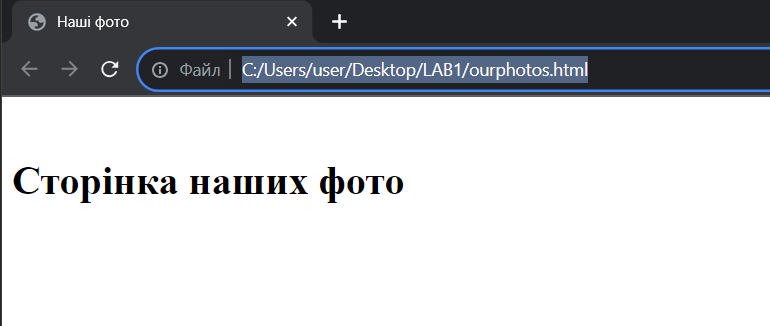
</html>

**Результат:**

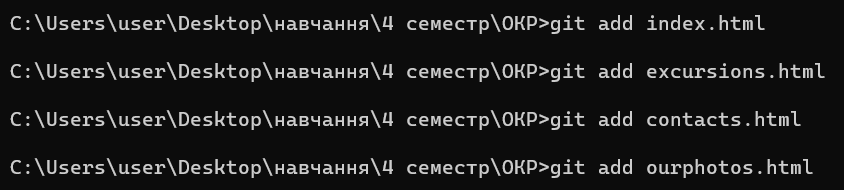








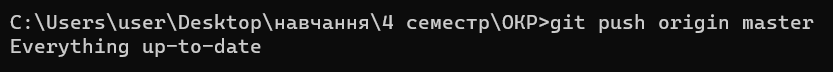
1. Додала створені сторінки під контроль Git на комп’ютері.

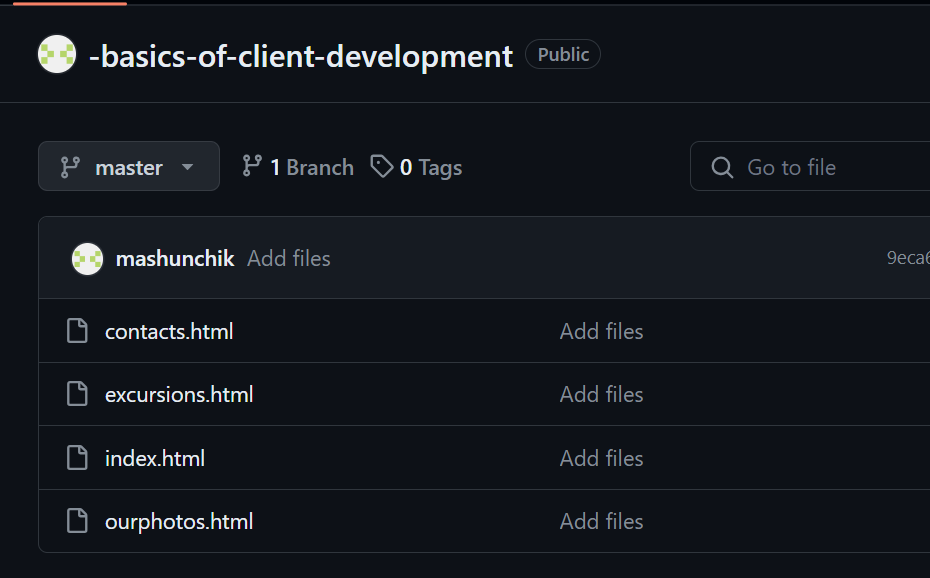


1. Створила репозиторій на GitHub



Завантажила проєкт на віддалений репозиторій (на GitHub).





**Висновок:** У ході виконання лабораторної роботи я успішно встановила Git на комп'ютері та створила каталог для проєкту, ініціалізувавши Git директорію. Було створено 4 сторінки відповідно до обраної тематики з розширенням .html, кожна з них має відповідну структуру з вказаними заголовками та назвами. Всі сторінки були успішно додані під контроль Git на комп'ютері, а також зареєстровано на GitHub, де створено віддалений репозиторій та завантажено наш проєкт. Виконані завдання дозволили мені ефективно використовувати Git для контролю версій та забезпечити збереження мого проєкту на віддаленому сервері.

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

**Лабораторна робота № 2**

з дисципліни «Основи клієнтської розробки»

Тема: «[Лабораторна робота 2. HTML. Структура документа.](#_bookmark3) [Заголовки. Гіперпосилання. Форматування тексту.](#_bookmark3) [Кольори. Списки. Зображення. Фон. Таблиці, фрейми](#_bookmark3)»

|  |  |
| --- | --- |
| Виконав:  студент групи ІА-24  Тильна Марія.  Дата здачі 20.02.2024  Захищено з балом \_\_\_\_\_\_\_\_\_\_\_ | Перевірила:  ст. вик. кафедри ІСТ  Хмелюк Марина Сергіївна |

**Лабораторна робота №2**

**1. Тема:** HTML. Структура документа. Заголовки. Гіперпосилання. Форматування тексту. Кольори. Списки. Зображення. Фон. Таблиці, фрейми.

**2. Завдання**:

Використовувати тему, обрану в 1 роботі.

1)В тілі сторінок додати по декілька абзаців тексту (відповідно до тематики) використовуючи тег <p></p>. Використовуючи гіперпосилання, реалізувати переходи між сторінками, на певний рядок поточною сторінки, певний рядок іншої сторінки, на сторінку в інтернеті). Застосувати атрибут, який задає колір гіперпосилань.

2)Відформатувати текст сторінок (виділення курсивом, напівжирним, шрифт, розмір, колір, вирівнювання), використовуючи тег <font></ font > і відповідні атрибути.

Додати декілька картинок <img>. За допомогою атрибутів width і height зменшити і збільшити розмір зображення в 2 рази. Зробити картинку гіперпосиланням: при натисканні на картинку повинен відкриватися повнорозмірний варіант в новому вікні.

Додати список, використовуючи різні типи форматування (нумерація <ul></ul>, маркування <ol></ol>, визначення <dl></dl>). Застосувати різні типи маркерів type.

Додати графічний фон на сторінки (картинку або колір).

3)На сторінці додати таблицею 3 \* 4. Залити кольором шапку з заголовками колонок. Додати заголовок до таблиці. В комірки першого рядка вставити картинки, другого рядка - гіперпосилання, 3 рядка - текст.

4)Створити складну таблицю, застосувавши різні види вирівнювання для різних рядків, об'єднання комірок (rawspan, calspan), групувння комірок( colgroup, сol), використати відступи (сellspacing, сellpadding).

5)Вставити плаваючий фрейм

6)Завантажити проєкт на віддалений репозиторій (на GitHub).

**3.Теоретичні відомості**

## Теги

Тег (html-тег, тег розмітки) - керуюча символьна послідовність, яка задає спосіб відображення гіпертекстової інформації.

HTML-тег складається з імені, за яким може слідувати необов'язковий список атрибутів. Весь тег (разом з атрибутами) полягає в кутові дужки <>:

<Імя\_тега [атрибути]>

Як правило, теги є парними і складаються з початкового та кінцевого тегів, між якими і можна відслідковувати. Ім'я кінцевого тега збігається з ім'ям початкового, але перед ім'ям кінцевого тега ставиться коса риска / (<html> ... </ html>). Кінцеві теги ніколи не містять атрибутів. Деякі теги не мають кінцевого елемента, наприклад тег <img>. Регістр символів для тегів не має значення.

Приклади часто використовуваних тегів HTML:

<html> ... </ html> - контейнер гіпертексту

<head> ... </ head> - контейнер заголовка документа

<title> ... </ title> - назва документа (те, що відображається в заголовку вікна браузера)

<body> ... </ body> - контейнер тіла документа

<div> ... </ div> - контейнер загального призначення (структурний блок)

<hN> ... </ hN> - заголовок N-ного рівня (N = 1 ... 6)

<p> ... </ p> - основний текст

<a> ... </a> - гіперпосилання

<ol> ... </ ol> - нумерований список

<ul> ... </ ul> - маркований список

<li> ... </ li> - елемент списку

<table> ... </ table> - контейнер таблиці

<tr> ... </ tr> - рядок таблиці

<td> ... </ td> - елемент таблиці

<img> - зображення

<form> ... </ form> - форма

<i> ... </ i> - відображення тексту курсивом

<b> ... </ b> - відображення тексту напівжирним шрифтом

<em> ... </ em> - виділення (курсивом)

<strong> ... </ strong> - посилення (напівжирним шрифтом)

<br> - примусовий розрив рядка

## Атрибути

Атрибути - це пари виду «властивість = значення», уточнюючі уявлення відповідного тега:

<Тег атрибут = "значення"> ... </ тег>

Атрибути вказують в початковому тегу, кілька атрибутів поділяють одним або декількома пропусками, табуляцією або символами кінця рядка. Значення атрибута, якщо таке є, слід за знаком рівності, хто стоїть після імені атрибута. Порядок запису атрибутів у тегу не важливий. Якщо значення атрибута - одне слово або число, то його можна просто вказати після знака рівності, не виділяючи додатково. Всі інші значення необхідно брати в лапки, особливо якщо вони містять кілька розділених пробілами слів.

Примітка: Не дивлячись на необов'язковість лапок, їх все ж варто завжди використовувати.

Атрибути можуть бути обов'язковими і необов'язковими. Необов'язкові атрибути можуть бути опущені, тоді для тега застосовується значення цього

атрибута за замовчуванням. Якщо не вказано обов'язковий атрибут, то вміст тега швидше за все буде відображено неправильно.

Короткий список деяких часто використовуваних атрибутів і їх можливих значень:

style = "опис\_стилів - локальні стилі

src = "адреса" - адреса (URI) джерела даних (наприклад картинки або скрипта) align = "left | center | right | justify" - вирівнювання, за замовчуванням left (по

лівому краю)

width = "число" - ширина елемента (в пікселях, піках, поінтах і ін.) height = "число" - висота елемента (в пікселях, піках, поінтах і ін.)

href = "адреса" - гіперпосилання, адреса (URI) на який буде виконаний перехід

name = "ім'я" - ім'я елемента

id = "ідентифікатор" - унікальний (в межах веб-сторінки) ідентифікатор елемента

size = "число" - розмір елемента

class = "ім'я\_класу" - ім'я класу у вбудованій або пов'язаної таблиці стилів title = "рядок" - назва елемента

alt = "рядок" - альтернативний текст

## Гіперпосилання

Гіперпосилання - це особливим чином позначений фрагмент веб-сторінки (текст, зображення та ін.), Який пов'язаний з іншим документом. Для вказівки гіперпосилань використовується тег <a>. Гіперпосилання дозволяють переміщатися між пов'язаними веб-сторінками.

Гіперпосилання умовно можна розділити на наступні види:

* Внутрішні -зв'язуючими документи всередині одного і того ж вузла;
* Зовнішні - зв'язуючі Web-сторінку з документами, що не належать даному вузлу;
* Гіперпосилання на поштову адресу;
* Мітки-якоря - дозволяють переходити відвідувачеві на певні розділи документа.

Перехід за посиланнями можна виконувати як на цілі документи, так і на спеціальним чином помічені (іменовані) фрагменти тексту:

<a name="якір"> Прив'язка до фрагменту тексту </a>

<a href="#якір"> Посилання на якір </a>

<а href = "адреса посилання"> текст для клацання миші </а>

<а href = " адреса посилання "> <IMG src = "посилання на малюнок"> </а>

Усередині тега <BODY> використовується атрибут, що задає колір гіперпосилань

link - задає колір вихідних посилань vlink - задає колір відвіданих посилань

alink - задає колір активних посилань (колір при натисканні миші)



Якщо потрібно зробити посилання на документ, який відкривається в новому вікні браузера, використовується атрибут

target="\_blank" тега <a>.

<a href=" pag2.html " target="\_blank"> <IMG src=log.gif> </a>

<a href="pag2.html" target="\_blank"> сторінка відкриється в новому вікні </a>

Для вказівки електронної пошти і запуску електронної програми використовується посилання:

<а href =[“ma](mailto:vvv@mail.ru)i[lto:vvv@mail.ru](mailto:vvv@mail.ru)”> Іванов Іван</а>

<а href="#new"> Нові надходження </а> - перехід до рядка тієї ж сторінки з позначкою тегом <а name="new">

<а href="pag2.htm#new1"> примітки </а> - перехід на сторінку сайту pag2 до рядка з позначкою тегом

<а name="new1">

<p> подробиці читайте <a href="pag2.htm"> друга сторінка </a> </ p> - посилання на іншу сторінку того ж сайту

<p> <a href="pag2.htm"> IMG src = log.gif alt = "про нашу фірму" </a> </p> - посилання на іншу сторінку того ж сайту, але посиланням є малюнок

<a href="myfile.exe" title=" файл 10 мегабайт"> Завантажити програму </a> - Посилання з підказкою title

<a Href="<http://home.ifmo.ru/index.html>"> тест </a> - зовнішнє посилання

Посилання можуть бути абсолютними і відносними.

*Абсолютні* посилання вказують, як правило, на зовнішній ресурс. Для них потрібно вказувати повний шлях:

<a href=["ht](http://example.com/page.html)t[p://example.com/page.html](http://example.com/page.html)"> Абсолютне посилання </a>

<a href="[http://example.com/images/figure1.gif">](http://example.com/images/figure1.gif) Посилання на сторінку в каталозі </a>

*Відносні* посилання, навпаки, використовують для переходу на внутрішні сторінки сайту. Для них потрібно вказувати шлях щодо посилається:

<a href="/index.html"> Посилання на сторінку в кореневому каталозі </a>

<a href="page.html#seg1"> Посилання на фрагмент сторінки в поточному каталозі </a>

<a href="images/figure1.gif"> Посилання на сторінку в підкаталозі поточного каталогу </a>

<a href="/docs/manual.html"> Посилання на сторінку в підкаталозі кореневого каталогу </a>

<a href="../files/index.html"> Посилання на сторінку в вищележачому каталозі </a>.

Колір в HTML

Колір в HTML може бути заданий ключовими словами - назвами квітів на англійській мові:

FF0000 - яскраво-червоний (red) 00FF00 - яскраво-зелений (green) 0000FF - яскраво

синій (blue)

FFFF00 - жовтий (yellow) - суміш червоного і зеленого 000000 - чорний (black)

FFFFFF - білий (white)

|  |  |
| --- | --- |
| Black = "#000000" | Green = "#008000" |
| Silver = "#C0C0C0" | Lime = "#00FF00" |
| Gray = "#808080" | Olive = "#808000" |
| White = "#FFFFFF" | Yellow = "#FFFF00" |
| Maroon = "#800000" | Navy = "#000080" |
| Red = "#FF0000" | Blue = "#0000FF" |
| Purple = "#800080" | Teal = "#008080" |
| Fuchsia = "#FF00FF" | Aqua = "#00FFFF" |

Значення кольору вказується в тезі після символу решітки (#). Наприклад для тексту:

<font color = "# 808080"> сірий текст </ font>

Для фону всієї сторінки в тезі body атрибут bgcolor:

<body bgcolor = "# FFFF00"> фон </ body>

Форматування тексту

Форматувати текст можна традиційними способами: виділяти курсивом, напівжирним, вибирати шрифт, розмір, колір, вирівнювати текстові фрагменти.

HTML дозволяє управляти відображенням тексту на сторінці.

<b> ... </ b> - виділення тексту жирним

<i> ... </ i> - виділення тексту курсивом

<u> ... </ u> - підкреслення тексту

<sub> ... </ sub> - форматувати текст як підрядковий індекс

<sup> ... </ sup> - форматувати текст як надрядковий індекс

<center> ... </ center> - вирівнювання тексту по центру

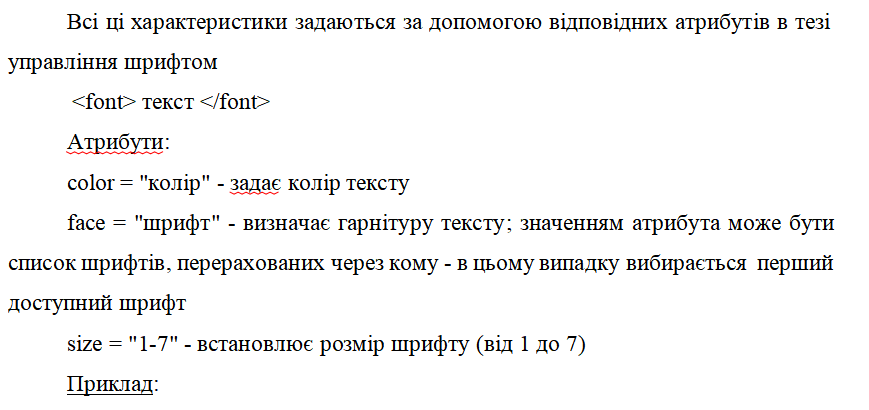
<font> ... </ font> - встановлює розмір, колір і гарнітуру тексту Приклад:

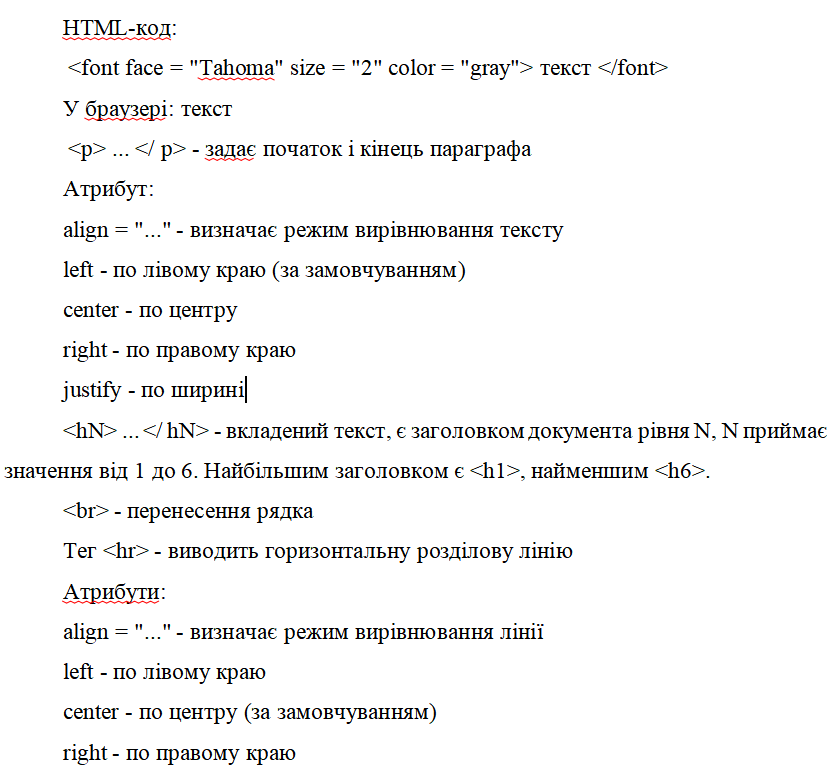
HTML-код: 101 <sub> 2 </ sub> = 5

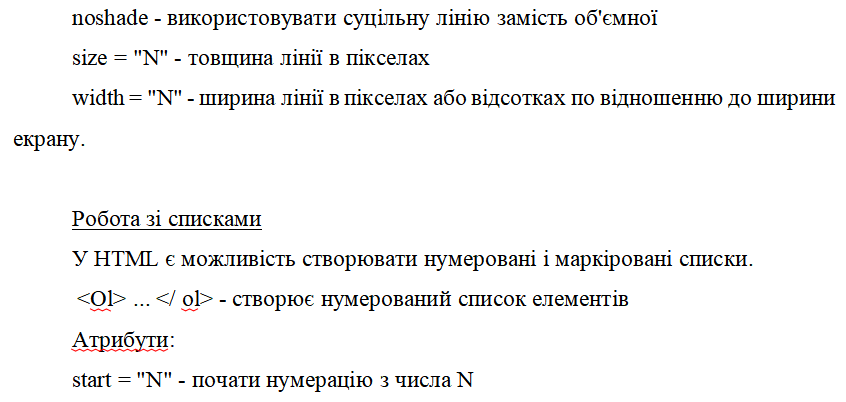
У браузері: 1012 = 5

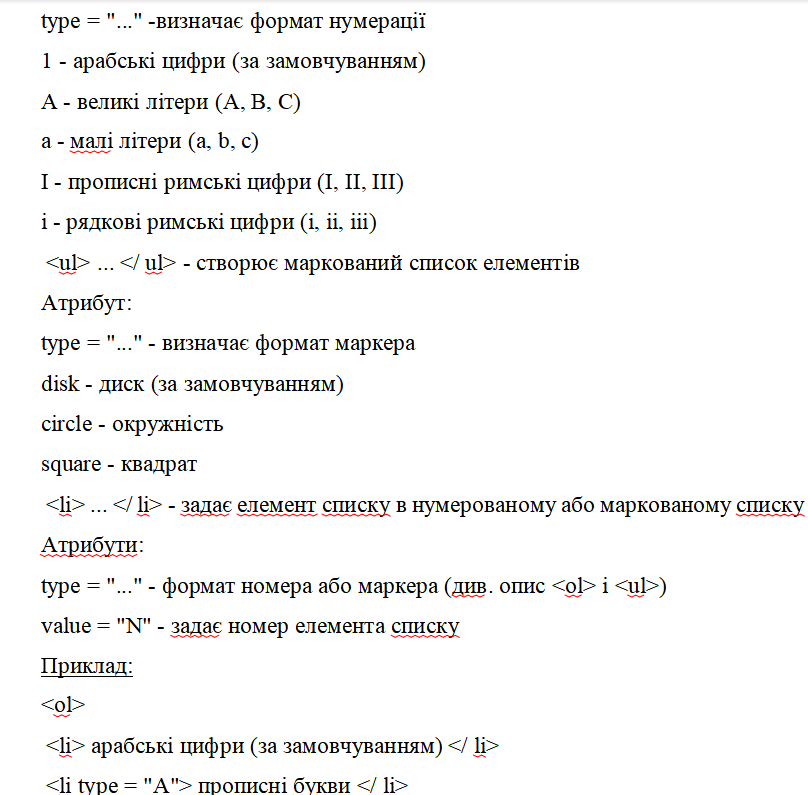
HTML-код: 2 <sup> 8 </ sup> = 256

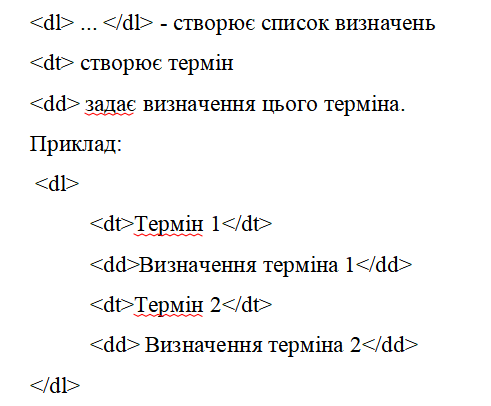
У браузері: 28 = 256











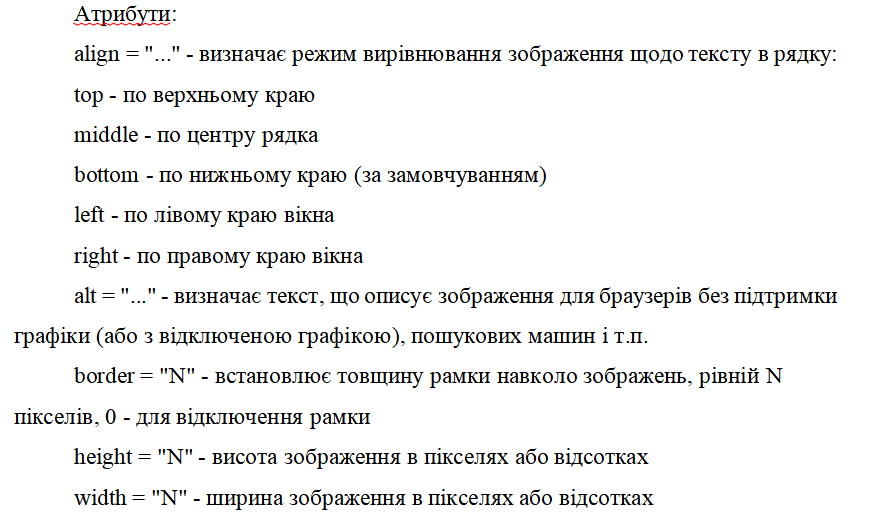
Зображення

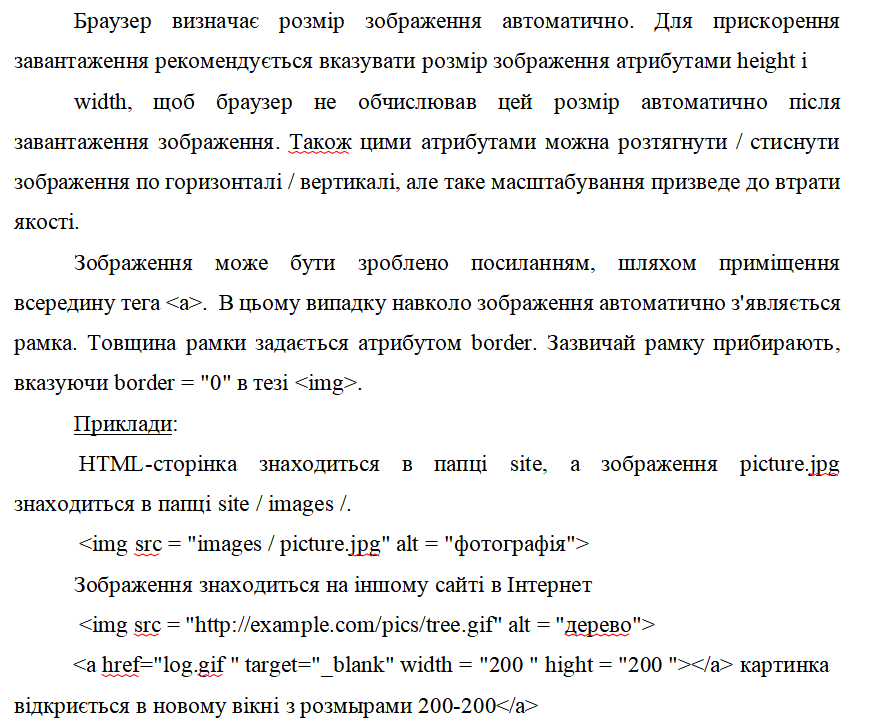
Вставка зображень на сторінці здійснюється непарним тегом <img>.

Обов'язковий атрибут src вказує абсолютний або відносний URL зображення.

Стандартними форматами зображень є GIF, PNG и JPEG.

Уникайте використання інших форматів зображень (наприклад, BMP або TIFF), тому що вони можуть не підтримуватися окремими типами браузерів.





Фонове зображення сторінки

Можна задавати адресу фонового зображення для сторінки в атрибуті background тега <body>. Фонове зображення відображається в натуральну величину. Якщо розмір зображення менше розміру вікна браузера, то малюнок повторюється по горизонталі вправо і по вертикалі вниз.

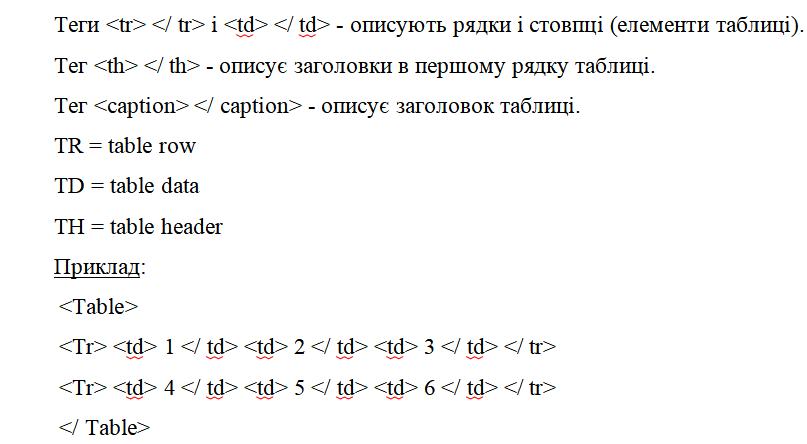
<body background = "bg1.jpg"> </ body>

**Таблиця в HTML** - це сукупність даних, розташованих і пов'язаних між собою за допомогою комірок, що розміщуються в рядках і колонках. Таблиця

заповнюється даними через підрядник. Для вставки таблиць визначено 3 основних тега.

Вміст комірок поміщається в теги <td> ... </td>, які, в свою чергу, поміщаються в теги рядків <tr> ... </tr>, а вони вже - в тег <table> ... </table>.

Всі інші елементи таблиці - текст, малюнки, списки - повинні бути вкладеними в нього. Допускається також вкладення таблиць одна в іншу, тобто вмістом комірки може бути інша таблиця.



Кількість тегів <tr> ... </ tr> визначає кількість рядків. У кожному тезі рядки повинно бути одне і те ж число тегів <td> ... </ td>, яке дорівнює кількості стовпців, інакше таблиця відобразиться неправильно. Можна створювати вкладені таблиці: вкладати таблицю в комірку іншої таблиці.

<table> ... </ table> - визначає початок і кінець коду таблиці, містить в собі теги рядків і комірок.

Атрибути:

align = "..." - визначає режим вирівнювання таблиці щодо тексту в рядку left - по лівому краю

right - по правому краю

valign = "..." - вирівнює текст в таблиці по вертикалі. Значення: top, bottom, middle, baseline

Приклад:

*<table border = "1">*

*<tr align = "center" valign = "top">*

*<td width = 120 height = 100> по центру, по верхній межі </ td>*

*<td align = "right" valign = "middle" width = 200> за правій межі, по середині*

*</ td>*

*</ tr>*

*</ table>*

background = "URL" - задає фоновий малюнок у таблиці bgcolor = "колір" - колір фону таблиці

border = "N" - встановлює товщину меж таблиці, рівну N пікселів (0 для відключення)

bordercolor = "колір" - колір рамки

bordercolorlight = колір - колір рамки зліва і зверху bordercolordark = колір - колір рамки праворуч і знизу title = "Текст" - підказка

width = число - ширина таблиці у відсотках або пікселях

*<table width = "500">*

*<table width = "100%">*

frame = "..." - описує параметри зовнішньої рамки. box - відображає всі частини рамки навколо таблиці void - видаляє всі рамки навколо таблиці

above - рамка тільки зверху belov - рамка тільки знизу lhs - рамка тільки зліва

rhs - рамка тільки справа

vsides - рамка тільки зліва і справа

Приклад:

*cellpadding = "0" cellpadding = "15"*

cellspacing = "N" - розмір вільного простору між коміркми Приклад:

*cellspacing = "0" cellspacing = "15"*

<tr> ... </ tr> - визначає рядок елементів таблиці Атрибути:

align = "..." - визначає режим вирівнювання вмісту комірок рядка left - по лівому краю

center - по центру

right - по правому краю justify - по ширині

background = "URL" - URL зображення, яке заповнить фон комірок рядка bgcolor = "колір" - колір фону комірок рядка

valign = "..." - визначає режим вирівнювання вмісту комірок рядка по вертикалі

top - по верхньому краю

middle - по середині (за замовчуванням) bottom - по нижньому краю

<td> ... </ td> - визначає коміру даних таблиці атрибути:

align = "..." - визначає режим вирівнювання вмісту комірки left - по лівому краю

center - по центру

right - по правому краю

background = "URL" - URL зображення, яке заповнить фон комірки bgcolor = "колір" - колір фону комірки

valign = "..." - визначає режим вирівнювання вмісту комірки по вертикалі top - по верхньому краю

middle - по середині (за замовчуванням) bottom - по нижньому краю

height = "N" - висота комірки в пікселях

width = "N" - ширина комірки в пікселях або відсотках від ширини таблиці.

<col> и <colgroup>- задає ширину і інші характеристики однієї або декількох стовпців таблиці.





Атрибути для <col> і <colgroup>:

align - Встановлює вирівнювання вмісту колонки по краю. span - Кількість колонок, до яких слід застосовувати атрибути. valign - Задає вертикальне вирівнювання вмісту колонки. width - Ширина колонок.

Приклад:

*<table>*

*<colgroup>*

*<col style="width:100px" /><col />*

*</colgroup>*

*<thead>*

*<tr><th>Column 1</th><th>Column 2</th></tr>*

*</thead>*

*<tfoot>*

*<tr><td>Footer 1</td><td>Footer 2</td></tr>*

*</tfoot>*

*<tbody>*

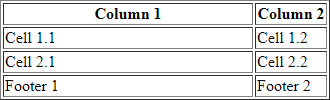
*<tr><td>Cell 1.1</td><td>Cell 1.2</td></tr>*

*<tr><td>Cell 2.1</td><td>Cell 2.2</td></tr>*

*</tbody>*

*</table>*

Результат:



Рядки HTML таблиці можна розділити на три семантичні секції: header, body і footer (заголовок, тіло і нижній колонтитул)

<thead> - означає загловок таблиці і містить теги <th> ... </ th> замість <td>

... </ td>

<tbody> означає колекцію рядків таблиці, які містять дані

<tfoot> означає нижній колонтитул таблиці, але описується до тега <tbody>

*<table>*

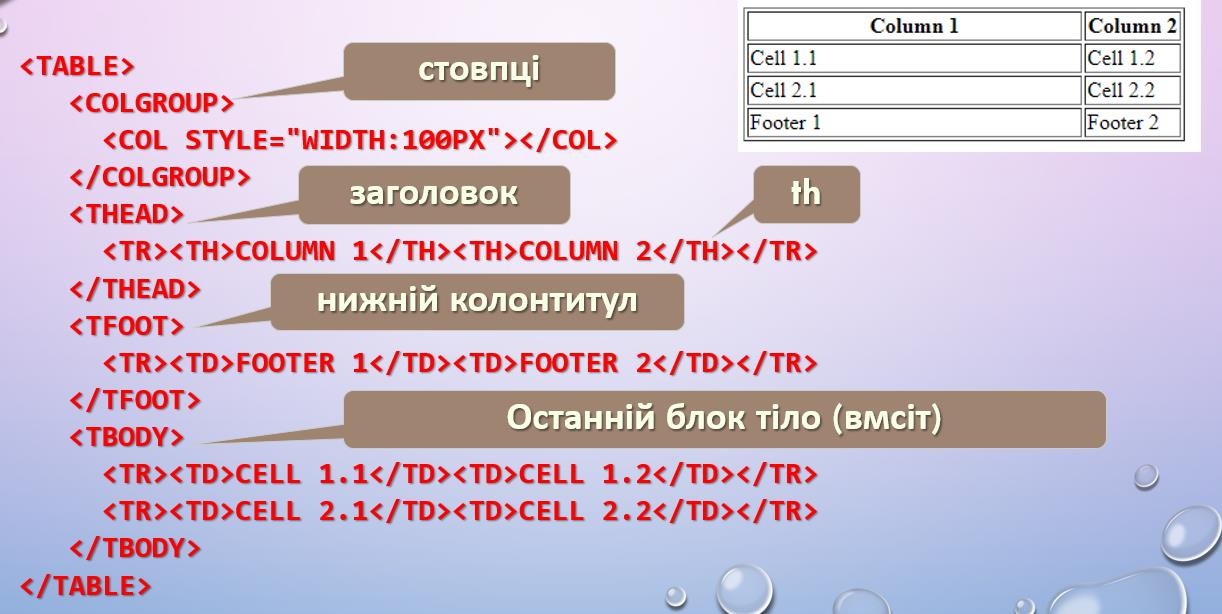
*<thead> <tr> <tH> ... </ tH> </ tr> </ thead>*

*<tfoot> <tr> <td> ... </ td> </ tr> </ tfoot>*

*<tbody> <tr> <td> ... </ td> </ tr> </ tbody>*

*</ table>*

Приклад:



Об'єднання комірок

colspan = "N" - розтягує комірку на N стовпців вліво Приклад:

*<table cellpadding = "15" border = "1">*

*<tr> <td colspan = "2"> 1 </ td> </ tr>*

*<tr> <td> 2 </ td> <td> 3 </ td> </ tr>*

*</ table>*

rowspan = "N" - розтягує комірку на N рядків вниз Приклад:

*<table cellpadding = "15" border = "1">*

*<tr> <td rowspan = "2"> 1 </ td> <td> 2 </ td> </ tr>*

*<tr> <td> 3 </ td> </ tr>*

*</ table>*

Ширина таблиці

Якщо ширина таблиці спочатку не задана, то вона обчислюється виходячи з вмісту комірок.

*<table border = "1">*

*<tr> <td> ширина таблиці не задана! </ td> </ tr>*

*</ table>*

Максимальна ширина таблиці в такому випадку дорівнює ширині вікна. Якщо ж ширина задана атрибутом width, то браузер розставляє переноси слів в тексті комірок таким чином, щоб дотримати заданий розмір.

*<table width = "100" border = "1">*

*<tr> <td> якщо задати атрибут width, текст починає переноситися за словами </ td> </ tr>*

*</ table>*

Додавання заголовка таблиці

Заголовок таблиці можна створити за допомогою відомих вам тегів <h1> -

<h6> Але оскільки ширина таблиці може відрізнятися від ширини вікна оглядача, вирівняти текстовий заголовок щодо таблиці може виявитися досить складно Тому для створення заголовків краще використовувати тег <CAPTION>, який створює заголовок безпосередньо в таблиці.

<table border = 10 width = 100%> <caption> назва таблиці </ caption>

## Плаваючі фрейми

Плаваючий фрейм, або лінійний фрейм, з'являється як окреме, плаваюче вікно для виведення інших документів. Він отримав свою назву з того факту, що може з'являтися вбудованим в нормальний потік елементів сторінки або може зміщуватися вліво або вправо на сторінці з оточуючим його текстом. Фрейм може виводити один документ або може бути місцем, де виводяться кілька з'єднаних документів. Наприклад, кілька посилань на сторінці можуть виводити різні зображення в цьому лінійному фреймі.

Лінійні фрейми створюють за допомогою тега <iframe>, загальна форма якого показана на лістингу.

*<iframe*

*src = "url"*

*name = "framename" frameborder = "1 | 0" scrolling = "auto | yes | no"*

*Виключені:*

*width = "n | n%" height = "n | n%" align = "left | right"*

*align = "top | middle | bottom" vspace = "n"*

*hspace = "n" marginwidth = "n" marginheight = "n"*

*>*

*</ Iframe>*

Атрибут src визначає сторінку для початкового завантаження у фрейм. Атрибут name привласнює кадру ім'я в якості покажчика для посилань. Фрейм не обов'язково з'єднувати з посиланнями. Можна просто вивести всередині фрейму один зовнішній документ, і в цьому випадку треба визначити в атрибуті src фрейма без імені тільки URL.

За замовчуванням навколо фрейма виводяться кордони. Можна відключити кордони за допомогою атрибута frameborder = "0". Якщо контент сторінки, що завантажується у фрейм, більше фрейму, то автоматично виводяться панелі прокрутки.

Можна відключити панелі прокрутки за допомогою атрибута scrolling = "no" або постійно виводити панелі прокрутки за допомогою scrolling = "yes".

Решта атрибути - width, height, align, vspace, hspace, marginwidth, і marginheight - краще ставити за допомогою таблиць стилів. Вони повинні вважатися виключеними атрибутами.

Відзначимо, що закриває тег </ iframe> є обов'язковим, навіть якщо він нічого не замикає. На сторінці Web можна визначити будь-яку кількість плаваючих фреймів.

Наступний код використовується для виведення і активації плаваючого фрейма. Властивості таблиці стилів замінюють більшість атрибутів фрейму. Відзначимо, що в тезі <iframe> атрибут src не заданий. Тому фрейм відкривається без виведення документа, залишаючи фрейм порожнім.

*<iframe name="TheFrame" scrolling="no"*

*style="width:225px; height:200px; float:right; margin-left:15px; border:ridge 5px">*

*</iframe>*

*<div>*

*<a href="Artemis.htm" target="TheFrame">Artemis</a>*

*<a href="Colossus.htm" target="TheFrame">Colossus</a>*

*<a href="Gardens.htm" target="TheFrame">Gardens</a>*

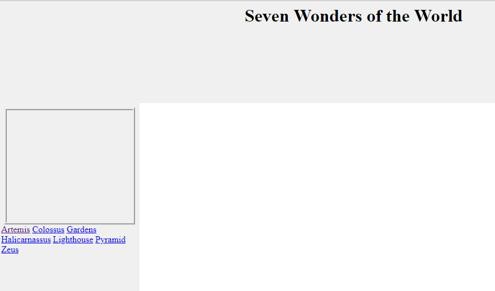
*<a href="Halicarnassus.htm" target="TheFrame">Halicarnassus</a>*

*<a href="Lighthouse.htm" target="TheFrame">Lighthouse</a>*

*<a href="Pyramid.htm" target="TheFrame">Pyramid</a>*

*<a href="Zeus.htm" target="TheFrame">Zeus</a>*

*</div>*



**4.Хід роботи**

**Код:**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Головна</title>

    <meta name="author" content="Tylna Mariia">

</head>

<style>

    body{

        background: #FAEBD7;

    }

</style>

<body>

    <h1><font color="black">Ласкаво просимо</font></h1>

    <font size="4" face="Arial, sans-serif">

        <p><i>Вітаємо вас на нашому сайті про екскурсії.</i> <b>Ми пропонуємо захоплюючі подорожі та незабутні враження!</b></p>

        <p><a href="excursions.html" style="color: rgb(85, 85, 225);">Дізнатися більше про наші екскурсії</a></p>

        <p><a href="ourphotos.html" style="color: rgb(85, 85, 225);">Переглянути наші фото</a></p>

        <p><a href="contacts.html" style="color: rgb(85, 85, 225);">Зв'язатися з нами</a></p>

        <p><a href="#video1" style="color: rgb(85, 85, 225);">Подивіться відео про нас</a></p>

        <h2> <font>Чому обирають нас?</font></h2>

        <p>Над створенням турів працюють досвідчені мандрівники, екскурсоводи, фахівці, які працювали в туристичних комплексах в різних точках світу… Випадкових людей немає. Тільки ті, хто захоплений туризмом і хоче розвивати його тут, в Україні, розповідаючи про її природні та історичні багатства.</p>

        <ol>

            <li>Наші туристи відправляються в продумані тематичні подорожі.</li>

            <li>З нами працюють закохані в професію гіди та екскурсоводи.</li>

            <li>Для кожного туриста ми оформлюємо страховку. Неважливо, розраховано поїздку на день чи на тиждень.</li>

            <li>А для завзятих мандрівників ми розробили програму лояльності: чим частіше подорожуєш, тим більше економиш.</li>

        </ol>

        <!-- Плаваючий фрейм з відео -->

        <div id="video1">

            <iframe width="500" height="300" src="https://www.youtube.com/embed/JA6Bj7Gr3ko" frameborder="0" allowfullscreen></iframe>

        </div>

        <p><a href="excursions.html#excursions-ukraine" style="color: rgb(85, 85, 225);">Дізнатися більше про екскурсії Україною</a></p>

        <p><a href="excursions.html#excursions-table" style="color: rgb(85, 85, 225);">Дізнатися більше про екскурсії закордоном</a></p>

    </font>

</body>

</html>

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Екскурсії</title>

</head>

<style>

    body{

        background: #FAEBD7;

    }

    table {

            width: 100%;

            border-collapse: collapse;

            margin-top: 20px;

        }

        th, td {

            border: 1px solid #dddddd;

            text-align: center;

            padding: 8px;

        }

        th {

            background-color: #d3cb73;

            color: white;

        }

        a {

            color: rgb(205, 203, 126);

        }

</style>

<body>

    <h1><font color="green">Наші екскурсії</font></h1>

    <font size="4" face="Arial, sans-serif">

        <p><i>Дізнайтеся більше про наші захоплюючі екскурсії.</i> <b>Ми пропонуємо різноманітні маршрути та цікаві місця для відвідування.</b></p>

        <p><a href="contacts.html" style="color: green;">Зв'язатися з нами</a></p>

         <!-- Таблиця екскурсій -->

    <table id="excursions-ukraine" cellspacing="10">

        <caption>Екскурсії Україною</caption>

        <tr>

            <th><img src="lviv.jpg" alt="Екскурсія 1" width="250" height="180"></th>

            <th><img src="karpaty.jpg" alt="Екскурсія 2" width="250" height="180"></th>

            <th><img src="odesa.jpg" alt="Екскурсія 3" width="250" height="180"></th>

        </tr>

        <tr>

            <td><a href="contacts.html" style="color: rgb(8, 75, 164);">Забронювати</a></td>

            <td><a href="contacts.html" style="color: rgb(8, 75, 164);">Забронювати</a></td>

            <td><a href="contacts.html" style="color: rgb(8, 75, 164);">Забронювати</a></td>

        </tr>

        <tr>

            <td>Популярний тур по Львову на автобусі з відвіданням відомих місць Львова, розміщених за межами історичного центру міста.</td>

            <td>Дводенний тур-елегія в Карпати, під час якого Ви переосмислите філософію життя, ознайомившись з нашим трагічним минулим у Рогатині, відпочивши різнобічно в Яремче й на Буковелі та здійснивши непросте сходження на найвищу вершину України  - гору Говерлу.</td>

            <td>Познайомитися з Одесою ближче, насолодитися її архітектурою, посмакувати її кухнею, поговорити з одеситами про те, про це, пожартувати, посумувати, зануритися в море або, якщо немає погоди, в атмосферу життєдайного південного міста і закохатися в нього раз і назавжди</td>

        </tr>

    </table>

    <table id="excursions-table" cellspacing="10">

        <caption>Екскурсії закордоном</caption>

        <colgroup>

            <col span="1" style="background-color:#DEB887; color:white; text-align:center;">

            <col span="1" style="background-color:#DEB887; color:white; text-align:center;">

            <col span="2" style="background-color:#DEB887; color:white; text-align:center;">

        </colgroup>

        <thead>

            <tr>

                <th rowspan="2" class="top-align center-align">Дата</th>

                <th rowspan="2" class="top-align center-align">Країна</th>

                <th colspan="2" class="center-align">Деталі</th>

            </tr>

            <tr>

                <th class="center-align">Маршрут</th>

                <th class="center-align">Вартість</th>

            </tr>

        </thead>

        <tbody>

            <tr>

                <td class="center-align">01.05.2024</td>

                <td class="center-align">Італія</td>

                <td class="center-align">Рим - Флоренція - Венеція</td>

                <td class="center-align">1200$</td>

            </tr>

            <tr>

                <td class="center-align">15.06.2024</td>

                <td class="center-align">Франція</td>

                <td class="center-align">Париж - Ліон - Ніцца</td>

                <td class="center-align">1500$</td>

            </tr>

            <tr>

                <td class="center-align">20.07.2024</td>

                <td class="center-align">Іспанія</td>

                <td class="center-align">Мадрид - Барселона - Севілья</td>

                <td class="center-align">1400$</td>

            </tr>

        </tbody>

    </table>

    </font>

</body>

</html>

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Контакти</title>

</head>

<style>

    body{

        background: #FAEBD7;

    }

</style>

<body>

    <h1><font color="purple">Наші контакти</font></h1>

    <dl>

        <font size="4" face="Arial, sans-serif">

            <li><i>Телефон:</i> <a href="tel:+123456789" style="color: blue;">+1 (234) 567-89</a></li>

            <li><i>Електронна пошта:</i> <a href="mailto:info@example.com" style="color: green;">info@example.com</a></li>

            <li><i>Telegram:</i> <a href="https://t.me/yourtelegram" style="color: purple;">@yourtelegram</a></li>

        </font>

    </dl>

    <font size="4" face="Arial, sans-serif">

        <p><a href="index.html" style="color: purple;">Повернутися на головну</a></p>

    </font>

</body>

</html>

<!-- Сторінка фото (foto.html) -->

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Наші фото</title>

</head>

<style>

    body{

        background: #FAEBD7;

    }

</style>

<body>

    <h1><font color="black">Сторінка наших фото</font></h1>

    <font size="4" face="Arial, sans-serif">

        <p><i>Ось кілька неймовірних фотографій з наших подорожей.</i> <b>Перегляньте та насолоджуйтеся красою світу разом з нами!</b></p>

        <!-- Перше зображення -->

        <a href="odesa.jpg">

            <img src="odesa.jpg" alt="Фото 1" width="250" height="180">

        </a>

        <!-- Друге зображення -->

        <a href="lviv.jpg">

            <img src="lviv.jpg" alt="Фото 2" width="250" height="180">

        </a>

        <!-- Третє зображення -->

        <a href="karpaty.jpg">

            <img src="karpaty.jpg" alt="Фото 3" width="250" height="180">

        </a>

        <!-- Четверте зображення -->

        <a href="olvia.jpg">

            <img src="olvia.jpg" alt="Фото 4" width="250" height="180">

        </a>

    </font>

    <font size="4" face="Arial, sans-serif">

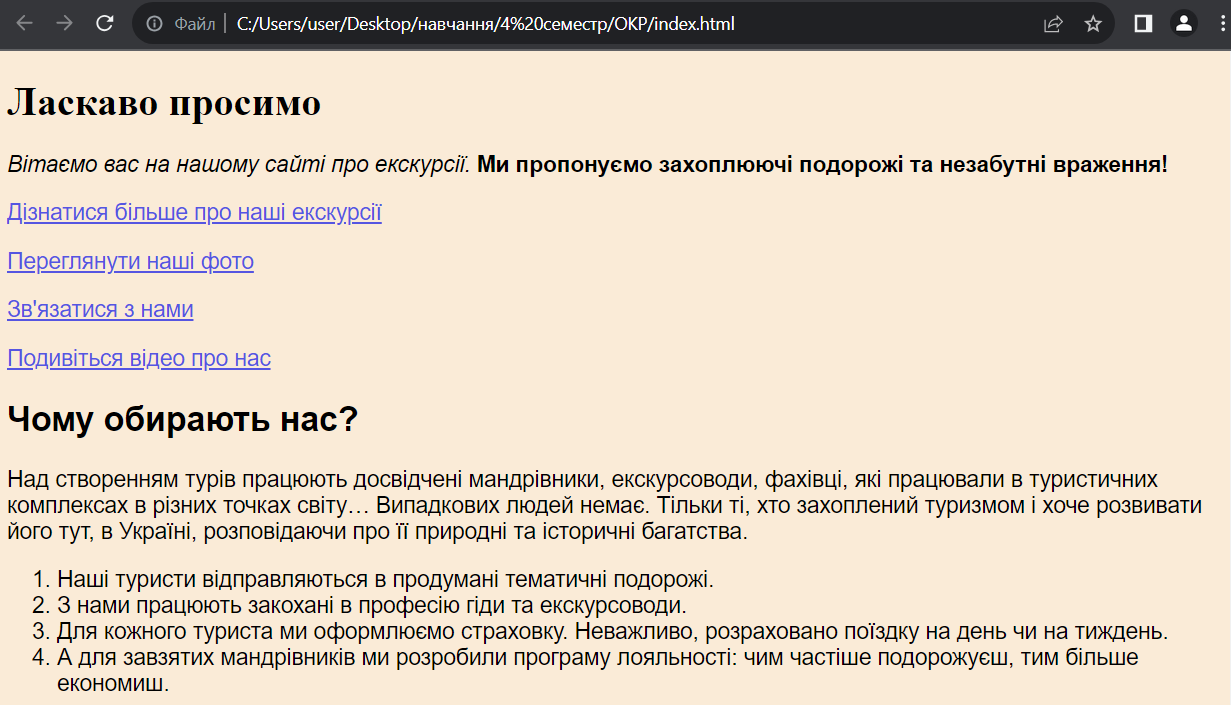
        <p><a href="index.html" style="color: purple;">Повернутися на головну</a></p>

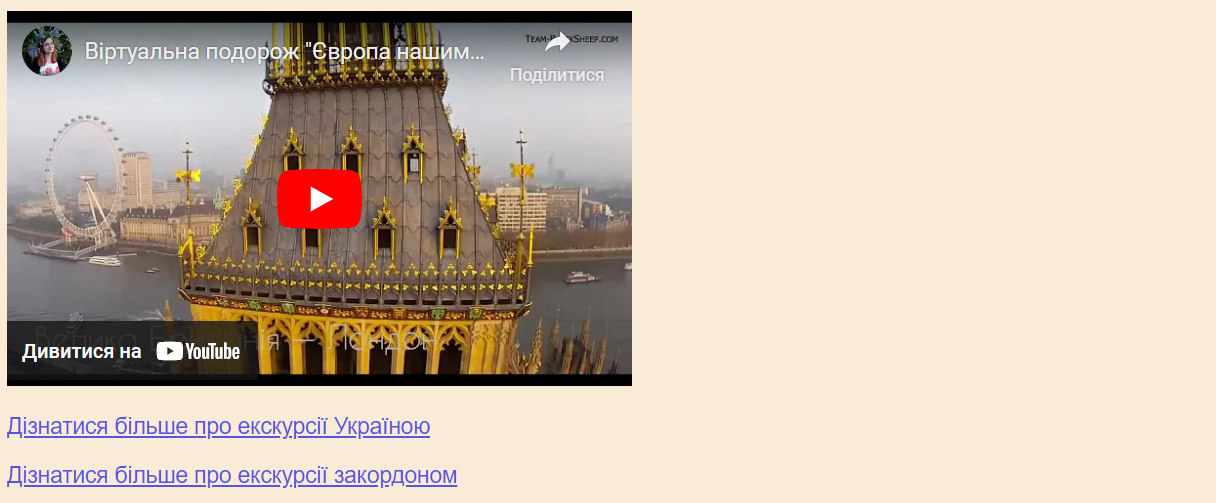
    </font>

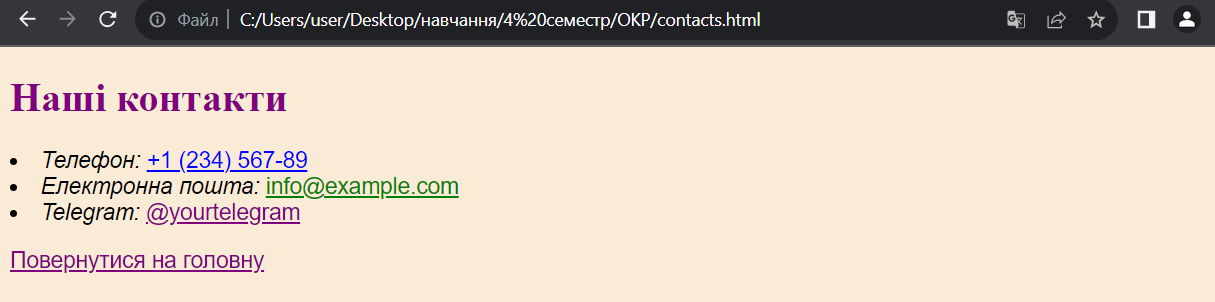
</body>

</html>

**Результат:**

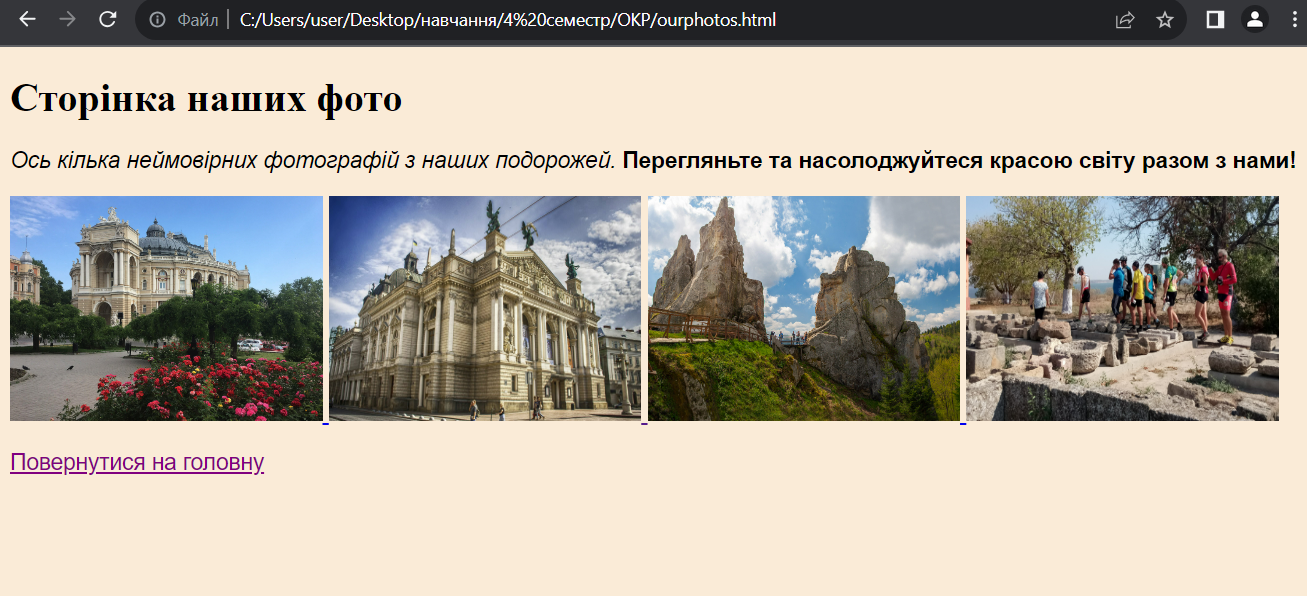












**Висновок:**

В ході виконання лабораторної роботи успішно реалізовала завдання щодо форматування та структуризації веб-сторінок. Використовуючи HTML-теги, було додано абзаци тексту, гіперпосилання з переходами між сторінками та інтернет-ресурсами. Також було використано теги <font> для форматування тексту, змінено розміри та зроблено картинки гіперпосиланнями. Додано списки з різним форматуванням та застосовано графічний фон.

Таблиці були створені для відображення різноманітної інформації, включаючи текст, гіперпосилання та зображення. Застосовано різні стилі форматування, такі як колірні шапки та різні види вирівнювання для складних таблиць. Додано також плаваючий фрейм.

Весь проєкт був успішно завантажений на віддалений репозиторій для зручного обміну та спільної роботи. Всі ці кроки сприяли засвоєнню основ веб-розробки та роботи з різноманітними HTML-тегами.