# **Лабораторна робота №2**

# **Тема «Таблиці в HTML-5. Створення зображень засобами трансформацій в CSS-3»**

**Завдання 1.**

За допомогою мови HTML-5 створити таблицю відповідно за своїм варіантом. У перший рядок таблиці розташувати ПІБ виконавця (окремо в кожну комірку). У другому рядку записати послідовно групу, номер лабораторної роботи та номер варіанту.

В цьому ж файлі додати CSS властивості:

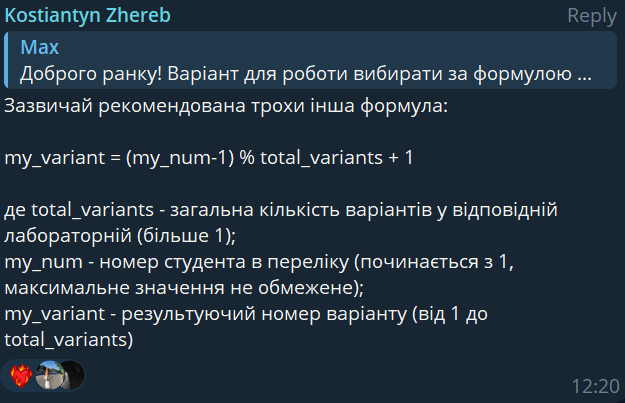
Для 1-го рядка встановити колір #nn8f8f та 2 #f5f5nn (властивість background-color), де nn- номер вашого варіанту.

Для тексту обрати контрастний колір за допомогою сервісу [color.adobe.com](https://color.adobe.com/)

Зробити таким чином, щоб при наведенні курсора на 1-й та 2-й рядки кольори змінювалися на протилежні (використати псевдо-клас hover).

**Завдання 2.**

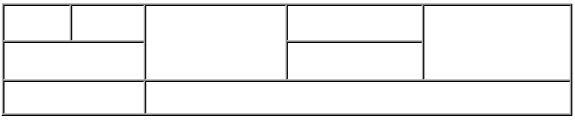
Створити зображення відповідно до свого варіанта: 1. – за допомогою position; 2- за допомогою transform , використовуючи тільки властивості CSS. Властивості CSS зберегти в окремому файлі style.css.

****

**№30=10**

**Варіанти до завдання 1.**

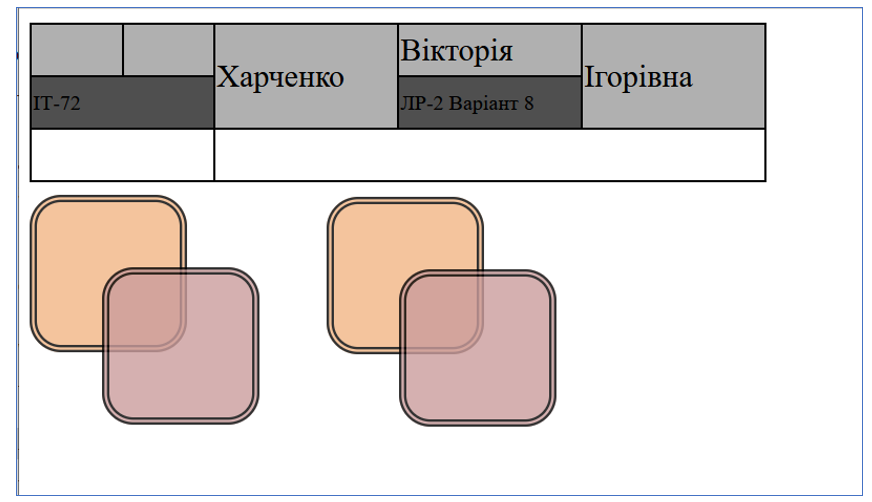
**Варіант 10**



**Варіанти до завдання 2.**

|  |
| --- |
| **Варіант 10:** |
|  |

**Приклад виконання лабораторної роботи:**



**Оформлення звіту та порядок захисту**

**Оформлення звіту та порядок захисту** Лабораторна робота виконується на комп’ютері. Звіт оформлюється на аркушах А4 , в якому стисло відображається зміст, хід роботи та отримані результати. **Обов’язково навести посилання на проект в Git-hub та посилання на створений сайт.**

**Теоретичні відомості**

**Робота з таблицями.**

Сама таблиця має жорстку структуру: головним є тег <table>, усередині якого повинні лежати теги <tr>, які створюють ряди (рядки) таблиці, а всередині них - теги <td>, які створюють осередки.

Як ви бачите, немає тегів, які створювали б стовпці - таблиці створюються по рядах: спочатку перший ряд, потім другий і так далі.

<table border="1">

<!--Це перший ряд таблици:-->

<tr>

<td>Ячейка 1</td>

<td>Ячейка 2</td>

<td>Ячейка 3</td>

</tr>

<!--Це другий ряд таблици:-->

<tr>

<td>Ячейка 4</td>

<td>Ячейка 5</td>

<td>Ячейка 6</td>

</tr>

<!--Це третій ряд таблици:-->

<tr>

<td>Ячейка 7</td>

<td>Ячейка 8</td>

<td>Ячейка 9</td>

</tr>

</table>

**Розширимо Ячейку1 на два стовпці без розвалювання таблиці**

Щоб поправити проблему з розвалювання таблиці з попереднього прикладу, видалимо одну з комірок праворуч від нашої (це Ячейка2 або Ячейка3 - без різниці):

<table>

<tr>

<td colspan="2">Ячейка 1</td>

<td>Ячейка 2</td>

</tr>

<tr>

<td>Ячейка 4</td>

<td>Ячейка 5</td>

<td>Ячейка 6</td>

</tr>

<tr>

<td>Ячейка 7</td>

<td>Ячейка 8</td>

<td>Ячейка 9</td>

</tr>

</table>



Каскадні таблиці стилів CSS (Cascading style sheets) - формальна мова опису зовнішнього вигляду документа, створеного з використанням мови розмітки гіпертексту.

Каскадні таблиці стилів дозволяють розділити опис логічної структури html-документа (виконане за допомогою мови розмітки) і опис зовнішнього вигляду html-документа (виконане за допомогою CSS).

Існує три способи визначення стилів: 1) в окремому файлі, що підключається до html-документів, 2) за допомогою тега <style> безпосередньо в деякому html-документі і 3) за допомогою атрибута style безпосередньо в деякому тезі.

Найбільш високий пріоритет має стиль, визначений у тезі, потім йде визначення стилю за допомогою тега style і найнижчим пріоритетом мають властивості, певні в окремому файлі.

Каскад пріоритетів особливо зручний при розробці великих проектів, наприклад, сайтів, які з значної частини html-документів. У цьому випадку загальне оформлення може бути винесено в окремий файл, в html-документі можуть бути внесені зміни в стиль документа за допомогою тега <style>, атрибут тега style дозволяє змінити оформлення одного тега.

Стилі визначаються парами властивостей і значень, перелік пар полягає в фігурні дужки і пари розділяються крапкою з комою:

{Property\_1: value\_1; property\_2: value\_2; ...; property\_n: value\_n}

**Абсолютне позиціонування**

Спільно з властивістю position використовуються CSS властивості, які керують зміщенням позиціонованого елемента:

• top (зміщення позиціонованого елемента від верхнього краю).

• right (зміщення позиціонованого елемента від правого краю).

• bottom (зміщення позиціонованого елемента від нижнього краю).

• left (зміщення позиціонованого елемента від лівого краю).

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Як значення, які визначають зміщення елемента, допускається використовувати фізичні одиниці (наприклад, пункти ), але частіше використовують візуальні одиниці - пікселі , процентні значення і значення em . Значення можуть бути як позитивні , так і негативні , як і люди, тільки значення.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

При використанні абсолютного позиціонування ( **position : absolute** ) елемент зсувається (позиціонується) щодо заданого краю його предка , при цьому предок повинен мати значення position відмінне від, встановленого за замовчуванням - static , інакше відлік (зміщення) буде вестися щодо, зазначеного краю вікна браузера .

**Відносне позиціонування**

Наступний тип позиціонування, який ми розглянемо це відносне позиціонування . Елементи, для яких задано відносне позиціонування ( [position](https://basicweb.ru/css/css_pr_position.php) : relative ) зміщуються (розміщуються) щодо положення в потоці документа , або іншими словами щодо його поточної позиції . Основна ідея відносного позиціонування полягає в тому, щоб не зрушити кудись елемент, а створити «контейнер» для елемента, який має абсолютне позиціонування . Іншими словами, вкладені елементи будуть зміщуватися не відносно краю вікна браузера, а щодо цього елемента, який буде мати відносне позиціонування і знаходиться в основному потоці документа .