**МІНЕСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**Національний технічний університет**

**«Харківський політехнічний інститут»**

Навчально науковий інститут комп’ютерних наук та інформаційних технологій

Кафедра Інформатики та інтелектуальної власності

**ЗВІТ**

**до лабораторної роботи 6 з дисципліни**

**«Основи WEB-технологій»**

Студент Яковлев І. А.

Викладач Шуба І. В.

м. Харків

Лабораторна робота 6

Завдання 1

Використовуючи властивостi grid, grid-template-areas, grid-template-rows,

grid-template-columns, grid-gap, grid-area Grid-верстки виконати другу

частину четвертого завданні четвертої лабораторної роботи згідно до свого

варіанту, не використовуючи при цьому стандартні способи позиціонування

та змiни розмірів відповідних блоків.

Опис директив:

Грід-розкладка (CSS Grid Layout) є двовимірною системою сіток в CSS. Гриди підійдуть і для верстки основних областей сторінки, і невеликих елементів інтерфейсу користувача. Ця стаття дає загальне уявлення про грид-розкладку та нову термінологію, яка є частиною специфікації CSS Grid Layout Level 1

body{

display: grid;

grid-template-rows: 90vh 10vh;

}

.top{

display: grid;

grid-template-columns: 15vw 70vw 15vw;

}

.center{

display: grid;

grid-template-rows: 10%;

}

.header{

background: rgb(255, 146, 146);

}

.content{

display: grid;

background: rgb(180, 255, 137);

}

.leftNav{

background: rgb(255, 233, 137);

}

.rightNav{

background: rgb(255, 233, 137);

}

.bottom{

background: rgb(137, 198, 255);

}

Завдання 2

Використовуючи властивостi grid, grid-template-areas, grid-template-rows,

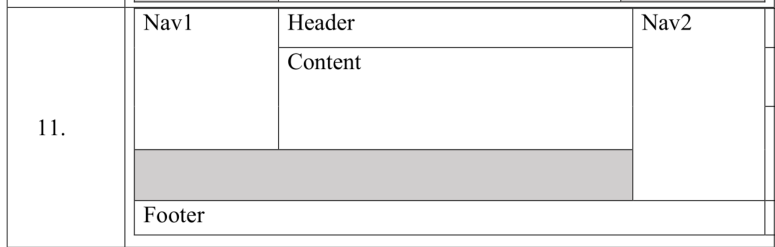
grid-template-columns, grid-gap, grid-area та... Grid-верстки модифікувати

минуле завдання, утворивши порожнi мiсця (сірий колір) в шаблоні сайту

згiдно до свого варiанту, не використовуючи при цьому стандартні способи

позиціонування та зміни розмірів відповідних блоків (сірий колір символізує

порожнє місце макету, тобто відсутність будь-якого контенту).



body{

display: grid;

grid-template-areas:

'ln h rn'

'ln c rn'

'. . rn'

'f f f';

grid-template-columns: 10vw 80vw 10vw;

grid-template-rows: 10vh 70vh 10vh 10vh;

background: rgb(136, 132, 132);

}

header{

grid-area: h;

background: rgb(255, 141, 141);

}

.nav1{

grid-area: ln;

background: rgb(154, 255, 141);

}

.nav2{

grid-area: rn;

background: rgb(141, 255, 166);

}

footer{

grid-area: f;

background: rgb(255, 225, 141);

}

div{

grid-area: c;

background: rgb(141, 168, 255);

}

Завдання 3

Використовуючи властивостi grid, grid-template-rows, grid-template-

columns, grid-gap, grid-row-start, grid-row-end, grid-column-start, grid-

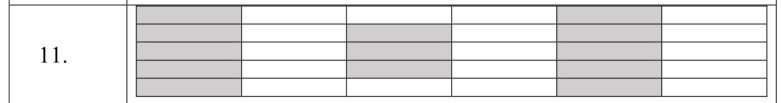
column-end, функції repeat та значення span Grid-верстки розташувати

абстрактні блоки (області сірого кольору) на сітці («гриді») згідно до свого

варіанту, не використовуючи при цьому стандартні способи позиціонування

та змiни розмірів відповідних блоків (лінії сітки потрібні лише для орієнтації

в просторі сторінки, демонструвати їх не треба).



body{

display: grid;

height: 100vh;

grid-template-columns: repeat(6, 1fr);

grid-template-rows: auto;

}

.col{

display: grid;

background: rgb(119, 239, 255);

border: 1px solid red;

}

.in{

border: 1px solid yellow;

}

body .col:nth-child(1), body .col:nth-child(5){

background: #000;

}

body .col:nth-child(3) .in:nth-child(2){

background: #000;

}

body .col:nth-child(3) .in:nth-child(3){

background: #000;

}

body .col:nth-child(3) .in:nth-child(4){

background: #000;

}

Завдання 4

Використовуючи властивості flex, flex-direction, flex-wrap, flex-flow,

justify-content, align-content, align-items, flex-grow, flex-shrink Flex-верстки

виконати другу частину четвертого завданні четвертої лабораторної роботи

згiдно до свого варіанту, не використовуючи при цьому стандартні способи

позиціонування та зміни розмірів відповідних блоків.

Опис директив:

Властивість CSS flex - це скорочена властивість, що визначає здатність гнучкого елемента розтягуватися або стискатися для заповнення доступного вільного простору. Ця властивість встановлює flex-grow, flex-shrink та flex-basis.

body{

display: flex;

height: 100vh;

background: #000;

flex-direction: column;

align-items: stretch;

}

.top{

display: flex;

flex-direction: row;

background: rgb(255, 120, 120);

flex: 9;

}

.ln{

background: rgb(255, 114, 114);

flex: 1.5;

}

.center{

display: flex;

flex-direction: column;

background: rgb(224, 120, 255);

flex: 9;

}

.head{

background: rgb(255, 241, 114);

flex: 1;

}

.content{

background: rgb(114, 163, 255);

flex: 8;

}

.rn{

background: rgb(255, 114, 114);

flex: 1.5;

}

.bottom{

background: rgb(144, 255, 129);

flex: 1;

}

Завдання 5

Використовуючи властивості flex, flex-direction, flex-wrap, flex-flow,

justify-content, align-content, align-items, flex-grow, flex-shrink Flex-верстки

виконати друге завдання цієї роботи згідно до свого варіанту.

body{

display: flex;

height: 100vh;

background: #000;

flex-direction: column;

align-items: stretch;

}

.top{

display: flex;

flex-direction: row;

background: rgb(99, 102, 102);

flex: 9;

}

.left{

display: flex;

flex-direction: column;

flex: 9.5;

}

.topleft{

display: flex;

flex-direction: row;

flex: 8;

}

.botleft{

flex: 1;

}

.ln{

background: rgb(255, 114, 114);

flex: 1.5;

}

.center{

display: flex;

flex-direction: column;

background: rgb(224, 120, 255);

flex: 9;

}

.head{

background: rgb(255, 241, 114);

flex: 1;

}

.content{

background: rgb(114, 163, 255);

flex: 8;

}

.grey{

background: rgb(68, 70, 73);

flex: 8;

}

.rn{

background: rgb(255, 114, 114);

flex: 1.5;

}

.bottom{

background: rgb(144, 255, 129);

flex: 1;

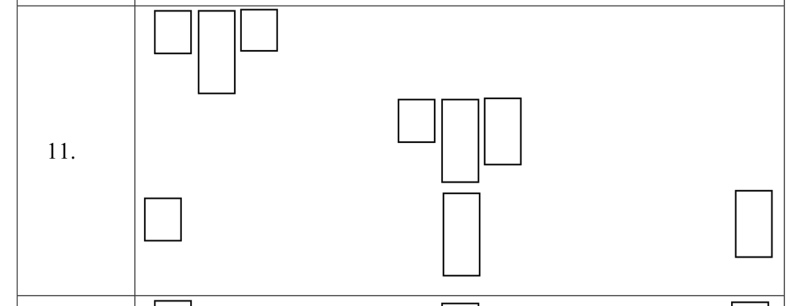
}

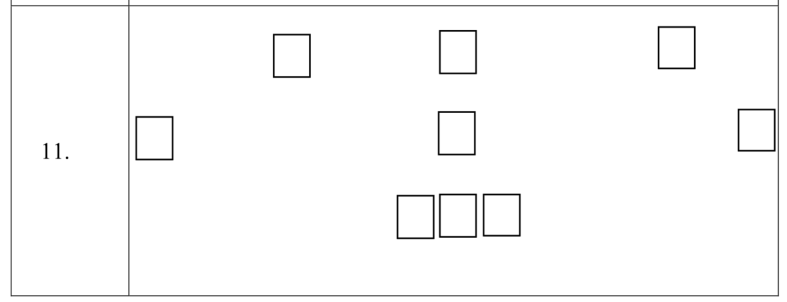
Завдання 6

Використовуючи різні способи вирівнювання блоків та їх рядків Flex-

верстки розташувати абстрактні блоки на сторiнцi згiдно до свого варіанту

згiдно до свого варіанту. Співвідношення та розмiр блокiв створити умовно.





body > div{

display: flex;

flex-direction: column;

gap: 10px;

padding: 10px;

}

.sm{

width: 4vw;

height: 4vh;

background: rgb(255, 105, 230);

}

.mid{

width: 5vw;

height: 7vh;

background: rgb(238, 105, 255);

}

.big{

width: 7vw;

height: 10vh;

background: rgb(243, 105, 255);

}

.top{

display: flex;

flex-direction: row;

column-gap: 1vw;

}

.cen{

display: flex;

justify-content: center;

column-gap: 1vw;

}

.bot{

display: flex;

justify-content: center;

align-items: baseline;

align-content: space-between;

column-gap: 1vw;

}

.bot .big{

margin: auto;

}

.t2{

justify-content: space-evenly;

}

.c2{

justify-content: space-between;

}

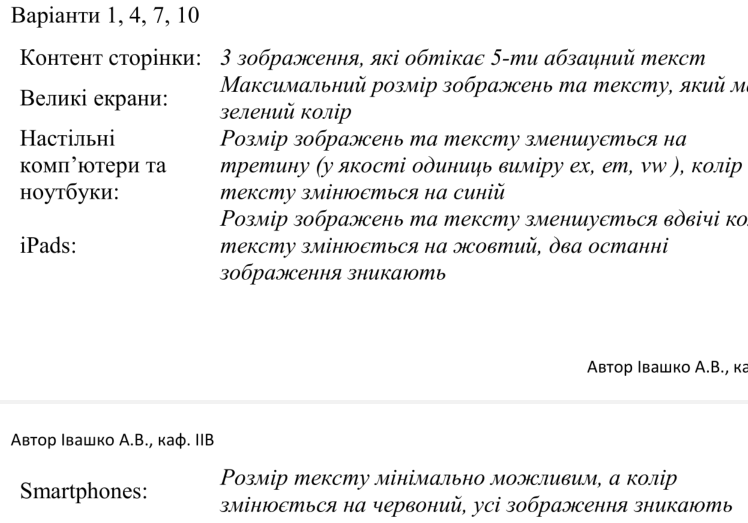
Завдання 7

Використовуючи параметри медіа запитів до відповідних гаджетів створити

адаптивну сторінку згідно до свого варіанту.

Опис директив:

Медіа-запити – це особливі правила застосування CSS, за допомогою яких можна змінювати стиль елементів, відштовхуючись від пристроїв, на яких переглядатимуть веб-сторінку.



@media only screen and (min-width: 320px) and (max-width: 600px){

body{

color: red;

font-size: 10px;

}

img{

display: none;

}

}

@media only screen and (min-width:768px) and (max-width: 1024px) and (orientation: landscape){

body{

color: yellow;

}

img{

width: 200px;

}

p img{

display: none;

}

}

@media only screen and (min-width: 1224px){

body{

color: blue;

}

img{

width: 300px;

}

}

@media only screen and (min-width: 1824px){

body{

color: green;

}

img{

width: 400px;

}

}