ВІДОКРЕМЛЕННИЙ СТРУКТУРНИЙ ПІДРОЗДІЛ

«ОПТИКО-МЕХАНІЧНИЙ ФАХОВИЙ КОЛЕДЖ КИЇВСЬКОГО НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ ІМЕНІ ТАРАСА ШЕВЧЕНКА»

Циклова комісія спеціальності «Інженерія програмного забезпечення»

**З В І Т**

**З НАВЧАЛЬНОЇ ПРАКТИКИ**

спеціальність 121 Інженерія програмного забезпечення

освітньо-кваліфікаційний рівень «фаховий молодший бакалавр»

|  |  |  |
| --- | --- | --- |
| ПІДСУМКОВА ОЦІНКА  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (оцінка цифрою та прописом)  Керівники практики від коледжу:  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ А. С. Нікітенко  (оцінка) (підпис)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Т. І. Лумпова  (оцінка) (підпис)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ М. С. Ходжаєв  (оцінка) (підпис) |  | Виконав: |
| студента 2-го курсу групи ІПЗ-22 |
| Цяк Олега Павловича  залікова книжка № \_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (підпис виконавця)  «\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2024 року |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_ А. В. Степанюк  (оцінка) (підпис) |  |  |
| «\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2024 року |  |  |

м. Київ — 2024

ЗАТВЕРДЖУЮ

Директор ОМФК КНУ

імені Тараса Шевченка  
 \_\_\_\_\_\_\_\_\_\_ Борис ГАПРІНДАШВІЛІ

«17» травня 2024 р.

**ІНДИВІДУАЛЬНЕ ЗАВДАННЯ**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| на період | *навчальної* | | практики | |
|  | (навчальної, технологічної, виробничої) | |  | |
| *Цяка Олега Павловича* | | | | |
| (П.І.Б студента.) | | | | |
| Спеціальність | *121 Інженерія програмного забезпечення* | Група | | *ІПЗ-22* |

|  |  |  |
| --- | --- | --- |
|  | **Питання для вивчення** | **Відмітки про виконання** |
| **1** | **Вступне заняття.** Вивчення питань охорони праці. Ознайомлення з електронною бібліотекою коледжу. Вивчення правил оформлення звітної документації. |  |
| **2** | **Розділ "Інформатика**" |  |
| 2.1 | Засобами MS Word розробити документ складної структури та зберегти у різних форматах |  |
| 2.2 | Розв’язування рівнянь |  |
| 2.3 | Розв’язування оптимізаційних задач |  |
| 2.4 | Основи статистичного аналізу даних. Ряди даних. Кореляційний аналіз даних |  |
| 2.5 | Обчислення основних статистичних характеристик вибірки |  |
| 2.6 | Візуалізація рядів і трендів даних |  |
| 2.7 | Розв‘язок задач за допомогою засобів табличного редактора MS Excel |  |
| 2.8 | Проектування бази даних та створення бази даних *MS ACCESS* |  |
| 2.9 | Основні поняття реляційних баз даних |  |
| 2.10 | Введення, коригування, вилучення інформації з бази даних *MS ACCESS* |  |
| 2.11 | Вибірка, пошук, фільтрація даних в *MS ACCESS* |  |
| 2.12 | Виконання практичних завдань за допомогою засобів табличного редактора MS Access |  |

|  |  |  |
| --- | --- | --- |
|  | **Питання для вивчення** | **Відмітки про виконання** |
| **3** | **Розділ "Програмування С++"** |  |
| 3.1 | Алгоритмізація обчислювальних процесів. Розробка постановки задачі, визначення вхідних та вихідних даних, оформлення блок-схем. |  |
| 3.2 | Типи даних в мові С/С++, внутрішнє представлення даних, операції перетворення типів даних. |  |
| 3.3 | Оператор присвоювання. Операції введення- виведення даних |  |
| 3.4 | Програмування лінійних та розгалужених обчислювальних процесів |  |
| 3.5 | Програмування циклічних обчислювальних процесів |  |
| 3.6 | Складання програм |  |
| 3.7 | Виконання тестів |  |
| 4 | **Розділ "** **Практична робота з С++"** |  |
| 4.1 | Тема 1. Введення та виведення у базових консольних програмах |  |
| 4.2 | Тема 2. Структури даних для управління станом програми |  |
| 4.3 | Тема 3. Робота з файлами: запис та читання |  |
| 4.4 | Тема 4. Формати файлів та їх обробка |  |
| 4.5 | Тема 5. Системи контролю версій (VCS) |  |
| 4.6 | Тема 6. Мова Markdown для опису проекті |  |
| 4.7 | Розроблення ігрової програми |  |
| 5 | Розділ ”Веб-дизайн та HTML” |  |
| 5.1 | Мова розмітки HTML та структура Веб-сторінки |  |
| 5.2 | Розмітка тексту за допомогою HTML. Зображення та посилання |  |
| 5.3 | Каскадні таблиці стилів CSS та їх структура |  |
| 5.4 | Спадковості і каскадування за допомогою CSS |  |
| 5.5 | Розробка та оформлення Постановки задачі |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Питання для вивчення** | | | | **Відмітки про виконання** | | |
| 5.9 | Розробка HTML документу | | | |  | | |
| **6** | Систематизація матеріалів, оформлення звіту з навчальної практики | | | |  | | |
| Голова циклової комісії | |  |  | Савєльєва І.В. | |  |  |
|  | | (підпис) |  | (ініціали, прізвище) | |  | (дата) |
| Керівники практики від коледжу | |  |  |  | |  |  |
|  | |  |  | Нікітенко А.С. | |  | 17.05.2024р. |
|  | | (підпис) |  | (ініціали, прізвище) | |  | (дата) |
|  | |  |  | Лумпова Т.І. | |  | 17.05.2024р. |
|  | | (підпис) |  | (ініціали, прізвище) | |  | (дата) |
|  | |  |  | Нікітенко А.С. | |  | 17.05.2024р. |
|  | | (підпис) |  | (ініціали, прізвище) | |  | (дата) |
|  | |  |  | Лумпова Т.І. | |  | 17.05.2024р. |
|  | | (підпис) |  | (ініціали, прізвище) | |  | (дата) |
|  | |  |  |  | |  |  |
|  | |  |  |  | |  |  |

**ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ**

**ВСТУП**

Автор:Цяк Олег Павлович

Дата:16.06.2024

Мета:

Перелік вивчених тем:

**1 БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ ПРИ РОБОТІ З КОМП’ЮТЕРОМ**

**2 РОЗДІЛ "ІНФОРМАТИКА"**

**2.2 Розв‘язок задач за допомогою засобів табличного редактора MS Excel**

а допомогою засобів табличного редактора MS Excel розв’язав наступні задачі:

1

Катети прямокутного трикутника дорівнюють 8см і 13см. Визначить гіпотенузу цього трикутника та площу. На скільки має змінитися менший катет заданого трикутника, щоб його гіпотенуза дорівнювала 22 см? Чому дорівнює площа такого трикутника?



2

Ціну 10 марок автомобілів, задану в $, перерахувати в грошових одиницях України та Польщі, використовуючи дані з інтернету про ціну автомобілів та курси валют. Визначить, яку суму необхідно сплатити автосалону, якщо він закупає по декілька екземплярів кожної моделі. (Марки автомобілів та їх кількість для автосалону обрати самостійно).



3

Типографія виготовляє зошити трьох видів: по 24 аркуші, по 48 аркушів і по 64 аркуші. Денний дохід від одного зошиту кожного виду 2 грн., 3 грн. і 5 грн. Необхідно визначити, за якої кількості виробів типографія отримає максимальний дохід, з урахуванням того, що:

* загальний обсяг виробництва — 300 зошитів на день;
* підприємству потрібно виготовити 50 зошитів по 24 аркуші для виконання існуючого замовлення;
* 40 зошитів по 48 аркушів — для виконання планового замовлення;
* зошитів по 64 аркуші необхідно виготовляти не більше ніж 40 одиниць на день.

4



Засобами табличного процесора здійсніть статистичний аналіз температурного режиму двох весняних місяців (для введення початкових значень температури скористайтесь зразком таблиці на рисунку 2.18).

За веденими даними потрібно визначити:

* середнє значення температури за місяць;
* максимальну і мінімальну температури;
* значення температури, що найчастіше повторювалось;
* медіану температурних значень;

Використовуючи комбіновану діаграму виконайте візуалізацію значень температур за два місяці.



**3 РОЗДІЛ "ПРОГРАМУВАННЯ С++"**

**3.1 Теоретичні відомості**

C++ — це мова програмування загального призначення, яка підтримує різні парадигми програмування, зокрема процедурне, об'єктно-орієнтоване та узагальнене програмування. Вона була розроблена Б'ярне Страуструпом як розширення мови C. Ось основні концепції C++:

**3.1.1 Загальні поняття. Елементи мови С++ - константи, змінні, операції, перетворення типів**

**Елементи мови:**

* **Константи:**
* **Фіксовані значення, які не змінюються під час виконання програми.**
* **Оголошуються з ключовим словом const.**
* **Наприклад: const int MAX\_SIZE = 100;.**
* **Змінні:**
* **Іменовані області пам'яті, де зберігаються значення.**
* **Оголошуються з вказанням типу.**
* **Наприклад: int age = 25;.**
* **Операції:**
* **Арифметичні: +, -, \*, /, %.**
* **Логічні: &&, ||, !.**
* **Порівняння: ==, !=, <, >, <=, >=.**
* **Присвоєння: =, +=, -=, \*=, /=, %=.**
* **Перетворення типів:**
* **Неявні: автоматичні перетворення типів компілятором (наприклад, з int в float).**
* **Явні: примусове перетворення типів за допомогою операторів або функцій (наприклад, static\_cast<int>(value)).**

**Ці основні елементи мови C++ дозволяють створювати ефективні та структуровані програми**

**3.1.2 Арифметичні операції**

 **Додавання**: +

 **Віднімання**: -

 **Множення**: \*

 **Ділення**: /

 **Ділення за модулем (остача)**: %

**3.1.3 Поняття вхідного та вихідного потоку, найпростіші математичні функції**

* **Вхідний потік (cin)**:
* Потік для отримання введених даних з клавіатури.
* Використовується з оператором >>.
* **Вихідний потік (cout)**:
* Потік для виведення даних на екран консолі.
* Використовується з оператором <<.

**Найпростіші математичні функції**

* **Додавання**: a + b
* **Віднімання**: a - b
* **Множення**: a \* b
* **Ділення**: a / b
* **Модуль числа**: abs(x)
* **Піднесення до степеня**: pow(x, n)
* **Корінь квадратний**: sqrt(x)

**3.1.4 Програмування лінійних та розгалужених обчислювальних процесів**

* **Лінійний процес**:
* Виконується послідовно від початку до кінця без переривань.
* Приклад: обчислення суми двох чисел.

**Програмування розгалужених обчислювальних процесів:**

* **Розгалужений процес**:
* Має різні шляхи виконання залежно від умов.
* Використовує умовні оператори (if, else, switch).
* Приклад: обчислення максимуму з двох чисел залежно від умови.

**3.1.5 Програмування циклічних обчислювальних процесів**

**Циклічний процес**:

* Виконується декілька разів до досягнення заданої умови.
* Використовується циклічний оператор (for, while, do-while).
* Приклад: обчислення суми чисел від 1 до N.

**3.2 Практичне завдання**

Написати код на мові програмування C++ для розрахунку по формулам

**3.2.1 Завдання**

**Завдання**: Обчислити і вивести на екран у табличному вигляді значення функції *f(x)* на заданому інтервалі зміни значень аргументу *х* від *xпоч* до *xкін* з кроком *h.*

Коефіцієнти *a, b, c –* дійсні числа. Значення *a, b, c, xпоч*, *xкін*, *h* вводити з клавіатури. Передбачити перевірку допустимості введених значень.

**3.2.2 Постановка задачі**

Вхідні дані

5,4,8,2,8,4

Вихідні дані



Блок-схема алгоритму роботи програми представлена на рис. ххх

Блок-схема





**3.2.3 Код програми**

#include <iostream> // Підключаємо бібліотеку для вводу-виводу

#include <cmath> // Підключаємо бібліотеку для математичних обчислень

#include <iomanip> // Підключаємо бібліотеку для форматування виведення

#include <windows.h> // Підключаємо бібліотеку для налаштування консолі (колір)

using namespace std; // Використовуємо стандартний простір імен

// Оголошення функції Fun1 для обчислення значень функції на заданому інтервалі

int f(float a, float b, float c, float xPoch, float xKin, float H);

int main()

{

system("color F0"); // Встановлюємо колір фону консолі

float a, b, c, xPoch, xKin, H; // Оголошуємо змінні для зберігання введених даних

cout << "a ="; cin >> a; // Запитуємо і зчитуємо значення a

cout << "b ="; cin >> b; // Запитуємо і зчитуємо значення b

cout << "c ="; cin >> c; // Запитуємо і зчитуємо значення c

cout << "xPoch ="; cin >> xPoch; // Запитуємо і зчитуємо початкове значення x

cout << "xKin ="; cin >> xKin; // Запитуємо і зчитуємо кінцеве значення x

cout << "H ="; cin >> H; // Запитуємо і зчитуємо крок зміни x

// Викликаємо функцію Fun1 з введеними параметрами

int ret = f(a, b, c, xPoch, xKin, H);

return 0; // Повертаємо 0 для позначення успішного завершення програми

}

// Реалізація функції Fun1 для обчислення значень функції на заданому інтервалі

int f(float a ,float b ,float c ,float xPoch ,float xKin ,float H)

{

float x, y; // Оголошуємо змінні для зберігання поточних значень x та y

// Перевірка допустимості введених значень

if (xPoch >= xKin) {

cerr << "Помилка: xPoch повинно бути менше xKin." << endl; // Виводимо повідомлення про помилку

return 1; // Повертаємо 1 для позначення помилки

}

if (H <= 0) {

cerr << "Помилка: крок H повинен бути більшим за 0." << endl; // Виводимо повідомлення про помилку

return 1; // Повертаємо 1 для позначення помилки

}

// Виведення заголовків таблиці

cout << setw(10) << "x" << setw(15) << "f(x)" << endl; // Виводимо заголовки стовпців

cout << "---------------------------" << endl; // Виводимо роздільник

// Цикл для обчислення значень f(x) на заданому інтервалі

for (x = xPoch; x <= xKin; x += H) {

// Обчислення значення функції залежно від умови

if (x < 0 && (b + c) > 5) {

y = (5 \* x - b) / (c - 4 \* x \* (pow(x, 3) + 1)); // Перша умова

}

else if (x > 0 && (b - c) < 0) {

y = (x - a) / (sqrt(x) + c); // Друга умова

}

else {

y = (-2 \* x + c) / (4 \* c \* x + 3 \* a); // Всі інші випадки

}

// Виведення значень x та f(x)

cout << setw(10) << x << setw(15) << y << endl; // Виводимо поточні значення x та f(x)

}

return 0; // Повертаємо 0 для позначення успішного завершення функції

}

**3.2.4 Скріншот виконання програми**

Скріншот



**3.3 Виконання тестів**



**4 РОЗДІЛ " ПРАКТИЧНА РОБОТА З С++"**

**4.1 Тема 1. Введення та виведення у базових консольних програмах**

*У мові програмування C++ введення та виведення даних здійснюються за допомогою стандартної бібліотеки iostream. Основні функції для роботи з введенням та виведенням - це cin для введення даних та cout для виведення даних.*

***#include <iostream> // Підключаємо бібліотеку iostream***

***using namespace std;***

***int main() {***

***int number;***

***cout << "Введіть число: "; // Вивід запиту до користувача***

***cin >> number; // Введення числа з клавіатури***

***cout << "Ви ввели число: " << number << endl; // Вивід введеного числа***

***return 0;***

***}***

**4.2 Тема 2. Структури даних для управління станом програми**

*У мові програмування C++ існує багато структур даних, які використовуються для управління станом програми. Основні з них включають масиви, вектори, списки, множини, словники та черги. Кожна з цих структур має свої особливості та застосування. Розглянемо їх детальніше.*

***Масиви*** *— це прості структури даних, які використовуються для зберігання елементів одного типу. Розмір масиву визначається під час його створення і не може змінюватися.*

***Вектори*** *— це динамічні масиви, які можуть змінювати свій розмір під час виконання програми. Для роботи з векторами використовується бібліотека <vector>.*

***Списки*** *— це двозв'язні списки, які дозволяють ефективно додавати і видаляти елементи з будь-якої позиції. Для роботи зі списками використовується бібліотека <list>.*

***Множини*** — це структури даних, які зберігають унікальні елементи. Для роботи з множинами використовується бібліотека <set>.

***Черги*** — це структури даних, які працюють за принципом FIFO (First In, First Out). Для роботи з чергами використовується бібліотека <queue>.

**4.3 Тема 3. Робота з файлами: запис та читання**

*У мові програмування C++ для роботи з файлами використовується бібліотека <fstream>, яка містить три основні класи: ifstream для читання з файлів, ofstream для запису у файли та fstream для обох операцій.*

**Підключення бібліотеки**

*Для роботи з файлами в C++ необхідно підключити бібліотеку <fstream>. Це робиться за допомогою директиви #include на початку програми:Також часто використовується бібліотека <iostream> для виведення повідомлень на консоль:*

**Запис у файл**

*Для запису даних у файл використовується клас ofstream. Розглянемо приклад запису тексту у файл.*

***Приклад:***

*#include <iostream>*

*#include <fstream>*

*using namespace std;*

*int main() {*

*ofstream outFile("example.txt"); // Відкриваємо файл для запису*

*if (outFile.is\_open()) { // Перевіряємо, чи файл відкрився успішно*

*outFile << "Привіт, світ!" << endl; // Записуємо текст у файл*

*outFile << "Це приклад запису у файл." << endl;*

*outFile.close(); // Закриваємо файл*

*cout << "Дані успішно записані у файл." << endl;*

*} else {*

*cout << "Не вдалося відкрити файл для запису." << endl;*

*}*

*return 0;*

*}*

**4.4 Тема 4. Формати файлів та їх обробка**

*У розробці програмного забезпечення на С++ використовуються різноманітні формати файлів для збереження даних. Обробка цих файлів включає їх зчитування, запис і маніпуляції з їхнім вмістом. Ось основні аспекти форматів файлів та їх обробки:*

*Текстові файли:*

*Опис: Текстові файли зберігають дані у формі тексту, який можна прочитати і відредагувати вручну.*

*Обробка: Для читання і запису текстових файлів використовуються потоки (ifstream для читання, ofstream для запису) або потоки введення-виведення (iostream).*

***Бінарні файли:***

*Опис: Бінарні файли зберігають дані у вигляді послідовності байтів, що є ефективним для зберігання складних структур даних.*

*Обробка: Для читання і запису бінарних файлів використовуються функції read і write, які працюють безпосередньо з байтами файлу.*

***CSV файли:***

*Опис: CSV (Comma-Separated Values) файли зберігають табличні дані у вигляді текстових файлів, де значення розділені комами або іншими символами.*

*Обробка: Для обробки CSV файлів зазвичай використовують сторонні бібліотеки або власну логіку розбиття рядків на поля за допомогою функцій рядкової обробки.*

**4..5 Тема 5. Системи контролю версій (VCS)**

*Системи контролю версій (Version Control Systems, VCS) є невід'ємною частиною сучасного розроблення програмного забезпечення. Вони дозволяють відстежувати зміни в коді, координувати роботу між кількома розробниками та зберігати історію проекту. Ось ключові моменти щодо VCS:*

*Основи VCS:*

*Відстеження змін: VCS зберігає всі зміни, зроблені у файлах проекту, що дозволяє повернутися до попередньої версії у разі необхідності.*

*Історія проекту: Збереження історії змін дозволяє побачити, хто і коли вносив певні зміни.*

*Типи VCS:*

*Локальні VCS: Усі версії файлів зберігаються на локальному комп'ютері. Приклад: RCS (Revision Control System).*

*Централізовані VCS: Використовується єдиний центральний сервер, який містить усі версії файлів. Розробники отримують доступ до файлів через мережу. Приклад: Subversion (SVN).*

*Розподілені VCS: Кожен розробник має повну копію всього репозиторію, включаючи всю історію змін. Це дозволяє працювати автономно. Приклад: Git, Mercurial.*

*Основні операції в VCS:*

*Ініціалізація репозиторію: Створення нового репозиторію для зберігання файлів проекту.*

*Додавання змін (commit): Збереження змін у репозиторії з коментарем.*

*Оновлення (pull): Отримання останніх змін із центрального або віддаленого репозиторію.*

*Злиття (merge): Об'єднання змін з різних гілок розроблення.*

*Вирішення конфліктів: Розв'язання конфліктів, що виникають при злитті змін з різних джерел.*

*Переваги використання VCS:*

*Співпраця: Кілька розробників можуть одночасно працювати над одним проектом без ризику перезапису роботи один одного.*

*Безпека: Історія змін дозволяє відстежувати, хто і коли вносив зміни, а також повертатися до попередніх версій у разі помилки.*

*Гнучкість: Можливість створення гілок для окремих функцій або виправлень, які можна тестувати і зливати з основною гілкою проект*

**4.6 Тема 6. Мова Markdown для опису проекті**

**Markdown** — *це легка мова розмітки, яка дозволяє легко форматувати текст, використовуючи звичайний текстовий редактор. Markdown часто використовується для написання документації, створення README-файлів, написання блогів та статей. Його основна перевага — простота синтаксису, який легко читається як у відформатованому вигляді, так і у сирцевому тексті.*

*Основні можливості Markdown:*

*Заголовки різних рівнів (від <h1> до <h6>)*

*Списки (нумеровані та марковані)*

*Виділення тексту (курсив, жирний шрифт)*

*Посилання та зображення*

*Блоки коду та цитати*

*Таблиці*

**4.7 Розроблення ігрової програми**

**4.7.1 Код програми**

*#include <iostream>*

*using namespace std;*

*char board[3][3] = { {'1', '2', '3'}, {'4', '5', '6'}, {'7', '8', '9'} };*

*char current\_marker;*

*int current\_player;*

*void drawBoard() {*

*system("clear"); // Use "cls" for Windows*

*cout << " " << board[0][0] << " | " << board[0][1] << " | " << board[0][2] << " \n";*

*cout << "---|---|---\n";*

*cout << " " << board[1][0] << " | " << board[1][1] << " | " << board[1][2] << " \n";*

*cout << "---|---|---\n";*

*cout << " " << board[2][0] << " | " << board[2][1] << " | " << board[2][2] << " \n";*

*}*

*bool placeMarker(int slot) {*

*int row = (slot - 1) / 3;*

*int col = (slot - 1) % 3;*

*if (board[row][col] != 'X' && board[row][col] != 'O') {*

*board[row][col] = current\_marker;*

*return true;*

*} else {*

*return false;*

*}*

*}*

*int winner() {*

*// Rows*

*for (int i = 0; i < 3; i++) {*

*if (board[i][0] == board[i][1] && board[i][1] == board[i][2]) {*

*return current\_player;*

*}*

*}*

*// Columns*

*for (int i = 0; i < 3; i++) {*

*if (board[0][i] == board[1][i] && board[1][i] == board[2][i]) {*

*return current\_player;*

*}*

*}*

*// Diagonals*

*if (board[0][0] == board[1][1] && board[1][1] == board[2][2]) {*

*return current\_player;*

*}*

*if (board[0][2] == board[1][1] && board[1][1] == board[2][0]) {*

*return current\_player;*

*}*

*return 0;*

*}*

*void swap\_player\_and\_marker() {*

*if (current\_marker == 'X') {*

*current\_marker = 'O';*

*} else {*

*current\_marker = 'X';*

*}*

*if (current\_player == 1) {*

*current\_player = 2;*

*} else {*

*current\_player = 1;*

*}*

*}*

*void game() {*

*current\_player = 1;*

*current\_marker = 'X';*

*drawBoard();*

*int player\_won;*

*for (int i = 0; i < 9; i++) {*

*cout << "It's player " << current\_player << "'s turn. Enter your slot: ";*

*int slot;*

*cin >> slot;*

*if (slot < 1 || slot > 9) {*

*cout << "Invalid slot! Choose another slot!\n";*

*i--;*

*continue;*

*}*

*if (!placeMarker(slot)) {*

*cout << "Slot occupied! Choose another slot!\n";*

*i--;*

*continue;*

*}*

*drawBoard();*

*player\_won = winner();*

*if (player\_won == 1) {*

*cout << "Player 1 won! Congratulations!\n";*

*break;*

*} else if (player\_won == 2) {*

*cout << "Player 2 won! Congratulations!\n";*

*break;*

*}*

*swap\_player\_and\_marker();*

*}*

*if (player\_won == 0) {*

*cout << "That is a tie game!\n";*

*}*

*}*

*int main() {*

*game();*

*return 0;*

*}*

**4.7.2 Скріншоти виконання програми**



**5 РОЗДІЛ ”ВЕБ-ДИЗАЙН ТА HTML”**

**5.1 Мова розмітки HTML та структура Веб-сторінки**

Основи HTML

HTML (HyperText Markup Language) - це основна мова для створення веб-сторінок та веб-додатків. Вона використовується для структуризації вмісту та визначення елементів на сторінці. HTML складається з набору елементів, які представлені у вигляді тегів.

Структура HTML-документа

Типовий HTML-документ складається з наступних частин:

DOCTYPE декларація: Вказує тип документа і версію HTML.

Елемент <html>: Кореневий елемент, що містить весь вміст HTML-документа.

Елемент <head>: Містить метадані про документ (наприклад, заголовок, стилі, метаінформацію).

Елемент <body>: Містить весь вміст, який відображається на веб-сторінці (тексти, зображення, посилання тощо).

**5.2 Розмітка тексту за допомогою HTML. Зображення та посилання**

HTML надає різноманітні теги для форматування тексту:

* **Заголовки**: <h1> до <h6>

<h1>Головний заголовок</h1>

<h2>Підзаголовок</h2>

* **Параграфи**: <p>

html

Копіювати код

<p>Це параграф тексту.</p>

* **Жирний текст**: <strong>

html

Копіювати код

<p>Це <strong>жирний текст</strong>.</p>

**5.3 Каскадні таблиці стилів CSS та їх структура**

CSS (Cascading Style Sheets) - мова стилів для оформлення HTML-документів.

**Структура CSS**

CSS правило складається з селектора та набору декларацій:

селектор {

властивість: значення;

}

**Селектори**

* **Теги**: p { color: blue; }
* **Класи**: .class-name { font-size: 20px; }
* **ID**: #id-name { background-color: yellow; }
* **Комбіновані**: div p { margin: 10px; }

**Типи CSS**

* **Вбудовані**: <p style="color: red;">Текст</p>
* **Внутрішні**: В межах тегу <style> в HTML:

<style>

p { color: blue; }

</style>

* **Зовнішні**: У файлі .css, підключається через <link>:

<link rel="stylesheet" href="styles.css">

**Каскадність та успадкування**

* **Каскадність**: Стилі переписуються залежно від специфічності та порядку.
* **Успадкування**: Деякі властивості успадковуються від батьківських елементів.

**Приклад:**

<link rel="stylesheet" href="styles.css">

/\* styles.css \*/

p {

color: green;

}

CSS дозволяє створювати красиві та структуровані веб-сторінки за допомогою стилів.

**5.4 Спадковості і каскадування за допомогою CSS**

**Спадковість**

Деякі CSS властивості передаються від батьківських елементів до дочірніх.

body {

font-family: Arial, sans-serif;

}

p {

color: blue;

}

**Каскадування**

CSS правила застосовуються за пріоритетом селекторів і порядком визначення.

p {

color: blue; /\* менш специфічний \*/

}

p.special {

color: red; /\* більш специфічний \*/

}

**Пріоритет селекторів**

* Вбудовані стилі (inline): style="color: green;"
* ID селектори: #id
* Класові селектори: .class
* Селектори тегів: p

**5.5 Практична частина**

Зробити сайт

**5.5.1 Постановки задачі**

Зробити сайт про себе з посиланнями на свої соц мережі

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Oleg Tsiak</title>

<link rel="stylesheet" href="main.css">

</head>

<body>

<div class="container">

<div style="text-align: center;">

<img src="P1011463 (1).JPG" class="profile-image">

<h1>Oleg Tsiak</h1>

<h2>Motion Graphic Designer</h2>

</div>

<section>

<h3>Summary</h3>

<p>Six months of experience in 2D motion design. I work in After Effects and Adobe Premiere Pro. I’m ready to help your company gain greater recognition, and I’m also ready to help your team illustrate videos to train your employees. I have experience working under tight deadlines and rapidly changing priorities. Always open to new complex tasks that require quick learning and the search for one good solution.</p>

</section>

<section>

<h3>Education</h3>

<p>2013 - 2024 Energodarskaya secondary school No. 2</p>

<p>2023 - 2024 AE from FreelStep</p>

</section>

<section>

<h3>Experience</h3>

<p><strong>Freelance( January 2024 - Now): 2D Motion Designer</strong><br>

</section>

<section class="contact-info">

<h3>Contact</h3>

<a href="mailto:tsyak2016@gmail.com" class="button"><img src="email.svg" alt="Email Icon"> Email</a>

<a href="https://facebook.com/Oleg Tsiak" class="button"><img src="facebook.svg" alt="Facebook Icon"> Facebook</a>

<a href="https://instagram.com/nie\_blokajte\_menya" class="button"><img src="instagram.svg" alt="Instagram Icon"> Instagram</a>

<a href="https://github.com/OlegTsiak" class="button"><img src="github.svg" alt="Github Icon"> Github</a>

</section>

</div>

</body>

</html>

**CSS**

body {

font-family: Arial, sans-serif;

margin: 0;

padding: 0;

background-color:#d5d5e6;

color: #333;

display: flex;

flex-direction: column;

align-items: center;

justify-content: center;

}

.container {

background-color: #e0e0ea;

border-radius: 10px;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

width: 90%;

max-width: 800px;

margin: 20px;

padding: 20px;

}

.profile-image {

width: 150px;

height: 150px;

border-radius: 50%;

object-fit: cover;

}

h1 {

margin-top: 0;

color: #205d9a;

}

.contact-info a {

display: inline-block;

margin-right: 10px;

text-decoration: none;

color: #0073e6;

}

.contact-info a:hover {

text-decoration: none;

}

.contact-info {

margin-top: 20px;

}

.contact-info .button {

display: inline-block;

margin-right: 10px;

text-decoration: none;

color: #fff;

background-color: #255585;

padding: 10px 20px;

border-radius: 50px;

border: none;

transition: background-color 0.3s;

display: flex;

align-items: center;

margin-top: 10px;

text-decoration: none;

}

.contact-info .button img {

margin-right: 10px;

width: 20px;

height: 20px;

}

.contact-info .button:hover {

background-color: #005bb5;

}

**ВИСНОВКИ**

**ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ ІНФОРМАЦІЇ**

1. ГОСТ 2.105-95 ЕСКД Общие требования к текстовым документам.

2. ДСТУ 1.5-93 Загальні вимоги до побудови, викладу, оформлення та змісту.

**ДОДАТКИ**