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: |
| Студент 2-го курсу групи ІПЗ-23 |
| Онипчеко Олег Олегович  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) | |  | |
| *Онипченко Олге Олегович* | | | | |
| (П.І.Б студента.) | | | | |
| Спеціальність | *121 Інженерія програмного забезпечення* | Група | | *ІПЗ-23* |

|  | **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) |
|  | |  |  |  | |  |  |
|  | |  |  |  | |  |  |

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

4

Розроб.

Перевір.

Реценз.

В.Я

Н. Контр.

В.Я

Затверд.

Звіт з навчальної практики

Літ.

Аркушів

ІПЗ-23

ОМФК.0121 НП 20

Зміст

[ВСТУП 7](#_Toc167918692)

[1 БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ ПРИ РОБОТІ З КОМП’ЮТЕРОМ 8](#_Toc167918693)

[2 РОЗДІЛ "ІНФОРМАТИКА" 9](#_Toc167918694)

[**2.1** **Розроблення засобами MS Word документа складної структури** 9](#_Toc167918695)

[**2.2 Розв‘язок задач за допомогою засобів табличного редактора MS Excel** 9](#_Toc167918696)

[**2.3 Виконання** **практичних завдань за допомогою засобів табличного редактора MS Access** 9](#_Toc167918697)

[3 РОЗДІЛ "ПРОГРАМУВАННЯ С++" 10](#_Toc167918698)

[**3.1 Теоретичні відомості** 10](#_Toc167918699)

[3.1.1 Загальні поняття. Елементи мови С++ - константи, змінні, операції, перетворення типів 10](#_Toc167918700)

[3.1.2 Арифметичні операції 10](#_Toc167918701)

[3.1.3 Поняття вхідного та вихідного потоку, найпростіші математичні функції 10](#_Toc167918702)

[3.1.4 Програмування лінійних та розгалужених обчислювальних процесів 10](#_Toc167918703)

[3.1.5 Програмування циклічних обчислювальних процесів 10](#_Toc167918704)

[**3.2 Практичне завдання** 11](#_Toc167918705)

[3.2.1 Завдання 11](#_Toc167918706)

[3.2.2 Постановка задачі 11](#_Toc167918707)

[3.2.3 Код програми 11](#_Toc167918708)

[3.2.4 Скріншот виконання програми 11](#_Toc167918709)

[**3.3 Виконання тестів** 11](#_Toc167918710)

[4 РОЗДІЛ " ПРАКТИЧНА РОБОТА З С++" 12](#_Toc167918711)

[**4.1 Тема 1. Введення та виведення у базових консольних програмах** 12](#_Toc167918712)

[**4.2 Тема 2. Структури даних для управління станом програми** 12](#_Toc167918713)

[**4.3 Тема 3. Робота з файлами: запис та читання** 12](#_Toc167918714)

[**4.4 Тема 4. Формати файлів та їх обробка** 12](#_Toc167918715)

[**4..5 Тема 5. Системи контролю версій (VCS)** 13](#_Toc167918716)

[**4.6 Тема 6. Мова Markdown для опису проекті** 13](#_Toc167918717)

[**4.7 Розроблення ігрової програми** 13](#_Toc167918718)

[4.7.1 Код програми 13](#_Toc167918719)

[4.7.2 Скріншоти виконання програми 13](#_Toc167918720)

[5 РОЗДІЛ ”ВЕБ-ДИЗАЙН ТА HTML” 14](#_Toc167918721)

[**5.1 Мова розмітки HTML та структура Веб-сторінки** 14](#_Toc167918722)

[**5.2 Розмітка тексту за допомогою HTML. Зображення та посилання** 14](#_Toc167918723)

[**5.3 Каскадні таблиці стилів CSS та їх структура** 14](#_Toc167918724)

[**5.4 Спадковості і каскадування за допомогою CSS** 14](#_Toc167918725)

[**5.5 Практична частина** 15](#_Toc167918726)

[5.5.1 Постановки задачі 15](#_Toc167918727)

[5.5.2 HTML документ 15](#_Toc167918728)

[ВИСНОВКИ 16](#_Toc167918729)

[ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ ІНФОРМАЦІЇ 17](#_Toc167918730)

[ДОДАТКИ 18](#_Toc167918731)

ОМФК.0121 НП 20

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

# ВСТУП

Автор: Онипченко Олег

Дата:30.05.24

Мета: написати гру на мові С++.

Перелік вивчених тем:

# 1 БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ ПРИ РОБОТІ З КОМП’ЮТЕРОМ

Безпека при роботі з комп’ютером має велике значення з кількох причин. По-перше, недбале ставлення до безпеки може призвести до фізичних травм. Наприклад, погане розташування комп’ютерного столу чи некомфортне крісло можуть спричинити болі у спині або шиї. Крім того, тривала робота за комп’ютером може спричинити очні напруження та інші проблеми зі здоров’ям.

По-друге, безпека даних є іншим важливим аспектом. Крадіжка або втрата цінної інформації може призвести до серйозних проблем, які можуть вплинути на бізнес або особисте життя користувача. Заходи безпеки, такі як використання паролів і шифрування даних, можуть допомогти уникнути таких неприємностей.

**Основні аспекти безпеки при роботі з комп’ютером**

**Віртуальна безпека**

Використання комп’ютера пов’язане з ризиком вірусів, шкідливих програм та кібератак. Щоб забезпечити віртуальну безпеку, потрібно:

**Установити антивірусне програмне забезпечення**

Встановлення надійного антивірусного програмного забезпечення допоможе захистити комп’ютер від шкідливих програм та вірусів. Регулярно оновлюйте антивірусну базу даних для забезпечення найвищого рівня захисту.

**Застосовувати оновлення системи**

Регулярно оновлюйте операційну систему та програмне забезпечення, оскільки це дозволяє усунути вразливості, які можуть бути використані хакерами для вторгнення в систему.

**Фізична безпека**

Фізична безпека також є важливим аспектом при роботі з комп’ютером. Ось декілька порад щодо фізичної безпеки:

**Вірно розташуйте робоче місце**

Переконайтеся, що ваше робоче місце обладнане комфортним столом та стільцем. Забезпечення належної підтримки для спини та шиї може допомогти уникнути болю та напруження.

**Забезпечте достатнє освітлення**

Маючи достатнє освітлення на робочому місці, ви зменшуєте напруження очей і запобігаєте втомі під час роботи за комп’ютером.

**Регулярно робіть перерви**

Не забувайте робити короткі перерви під час тривалої роботи за комп’ютером. Піднімайтесь, розтягуйтеся та робіть прості фізичні вправи для підтримки кровообігу та запобігання м’язовим напругам.

**Висновок**

Безпека життєдіяльності при роботі з комп’ютером є надзвичайно важливою. Захищаючи себе від фізичних травм, вірусів та кібератак, ми забезпечуємо безпечне та продуктивне використання комп’ютера. Пам’ятайте про важливість використання антивірусного програмного забезпечення, оновлення системи, розсудливе поводження з електронною поштою та правильну організацію робочого місця.

# 2 РОЗДІЛ "ІНФОРМАТИКА"

## **2.1 Розроблення засобами MS Word документа складної структури**

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

## **2.3 Виконання** **практичних завдань за допомогою засобів табличного редактора MS Access**

# 3 РОЗДІЛ "ПРОГРАМУВАННЯ С++"

## **3.1 Теоретичні відомості**

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

### 

### 3.1.2 Арифметичні операції

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

### 3.1.4 Програмування лінійних та розгалужених обчислювальних процесів

### 3.1.5 Програмування циклічних обчислювальних процесів

## **3.2 Практичне завдання**

### 3.2.1 Завдання

### 3.2.2 Постановка задачі

Вхідні дані

Вихідні дані

Блок-схема алгоритму роботи програми представлена на рис. ххх

Блок-схема

Рисунок хх - Блок-схема алгоритму роботи програми

### 3.2.3 Код програми

### 3.2.4 Скріншот виконання програми

Скріншот виконання програми представлена на рис. ххх

Скріншот

Рисунок хх - Скріншот виконання програми

## **3.3 Виконання тестів**

# 4 РОЗДІЛ " ПРАКТИЧНА РОБОТА З С++"

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

Введення та виведення у базових консольних програмах на C++ можна здійснити за допомогою стандартних потоків вводу-виводу `std::cin` та `std::cout`.

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

У C++ для управління станом програми ви можете використовувати різні структури даних в залежності від вашої потреби:

1. Структури: Структури дозволяють зберігати кілька пов'язаних даних разом. Вони корисні для представлення складних об'єктів з декількома властивостями.

2. Масиви: Масиви використовуються для зберігання колекції елементів одного типу.

3. Вектори: `std::vector` є динамічним масивом, який можна змінювати за розміром. Це корисно для зберігання колекцій змінної довжини

4. Мапи: `std::map` або `std::unordered\_map` дозволяють зберігати дані у форматі ключ-значення, що дозволяє швидкий доступ до даних за ключем.

5. Стеки та черги: `std::stack` та `std::queue` надають можливість виконання операцій "вставка" і "вилучення" зі структурами даних типу стек та черга відповідно.

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

У C++ для роботи з файлами використовуються потоки вводу-виводу std::ifstream (для читання з файлу) та std::ofstream (для запису у файл).

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

У C++ ви можете обробляти різні формати файлів, такі як текстові файли, бінарні файли та файли у спеціалізованих форматах, наприклад, CSV (Comma Separated Values) або JSON (JavaScript Object Notation). Обробка файлів різних форматів може вимагати використання різних бібліотек або самостійного парсингу.

Ось декілька прикладів обробки різних типів файлів у C++:

1. Текстові файли: Для обробки текстових файлів використовуються звичайні потоки вводу-виводу `std::ifstream` та `std::ofstream`, які я вже показував у попередній відповіді.

2. Бінарні файли: Для обробки бінарних файлів також використовуються потоки вводу-виводу, але читання та запис може виконуватися за допомогою функцій `read` та `write`, які працюють з байтовими потоками.

3. CSV файли: Для обробки CSV файлів можна використовувати бібліотеку, наприклад, `boost::tokenizer` або `std::getline` для розбиття рядка на поля.

4. JSON файли: Для обробки JSON файлів можна використовувати бібліотеку, таку як `nlohmann/json`, яка забезпечує зручний інтерфейс для парсингу та створення JSON об'єктів.

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

Системи контролю версій (VCS) важливі для ефективного управління розробкою програмного забезпечення, включаючи проекти на C++.

Ось декілька популярних VCS та як їх використовувати у контексті C++:

1. Git: Git є однією з найпопулярніших систем контролю версій. Він дозволяє вам відстежувати зміни у коді, створювати гілки для роботи над новими функціями та об'єднувати їх, а також співпрацювати з іншими розробниками.

2. Subversion (SVN): SVN - це централізована система контролю версій, яка працює за принципом "центральний сервер", на відміну від розподіленого підходу Git.

3. Mercurial (Hg): Це ще одна розподілена система контролю версій, подібна до Git, але з більш простим інтерфейсом користувача.

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

Markdown - це простий інструмент для створення форматованого тексту, який часто використовується для написання документації, опису проектів та README файлів у репозиторіях. Він може бути корисним для опису проектів на C++, де ви можете включати опис коду, інструкції зі збирання, використання та іншу корисну інформацію.

## **4.7 Розроблення ігрової програми**

Розроблюю гру «2048»

### 4.7.1 Код програми

#include <iostream>

#include <vector>

#include <algorithm>

#include <cstdlib>

#include <ctime>

#include <fstream>

#include <iomanip> // For std::setw

using namespace std

// Function to initialize the game grid

vector<vector<int>> initializeGrid() {

return {{0, 0, 0},

{0, 0, 0},

{0, 0, 0}};

}

// Function to add a new 2 or 4 tile to the grid

bool addNewTile(vector<vector<int>>& grid) {

vector<pair<int, int>> emptyCells;

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

for (int j = 0; j < 3; ++j) {

if (grid[i][j] == 0) {

emptyCells.push\_back({i, j});

}

}

}

if (!emptyCells.empty()) {

int index = rand() % emptyCells.size();

int value = (rand() % 2 + 1) \* 2; // Either 2 or 4

grid[emptyCells[index].first][emptyCells[index].second] = value;

return true;

}

return false;

}

// Function to print the game grid

void printGrid(const vector<vector<int>>& grid) {

cout << "-------------------" << endl;

for (const auto& row : grid) {

cout << "|";

for (int val : row) {

if (val == 0) {

cout << setw(4) << " " << " |";

} else {

cout << setw(4) << val << " |";

}

}

cout << endl;

cout << "-------------------" << endl;

}

}

// Function to check if the game is over

bool isGameOver(const vector<vector<int>>& grid) {

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

for (int j = 0; j < 3; ++j) {

if (grid[i][j] == 0) {

return false;

}

if (j < 2 && grid[i][j] == grid[i][j+1]) {

return false;

}

if (i < 2 && grid[i][j] == grid[i+1][j]) {

return false;

}

}

}

return true;

}

// Function to swipe left

bool swipeLeft(vector<vector<int>>& grid) {

bool moved = false;

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

// Move all tiles to the left

vector<int> newRow;

for (int j = 0; j < 3; ++j) {

if (grid[i][j] != 0) {

newRow.push\_back(grid[i][j]);

}

}

while (newRow.size() < 3) {

newRow.push\_back(0);

}

// Merge adjacent tiles of the same value

for (int j = 0; j < 2; ++j) {

if (newRow[j] == newRow[j+1] && newRow[j] != 0) {

newRow[j] \*= 2;

newRow[j+1] = 0;

moved = true;

}

}

// Move merged tiles to the left

vector<int> finalRow;

for (int j = 0; j < 3; ++j) {

if (newRow[j] != 0) {

finalRow.push\_back(newRow[j]);

}

}

while (finalRow.size() < 3) {

finalRow.push\_back(0);

}

if (grid[i] != finalRow) {

moved = true;

grid[i] = finalRow;

}

}

return moved;

}

// Function to swipe right

bool swipeRight(vector<vector<int>>& grid) {

// Reverse each row, then swipe left, then reverse again

bool moved = false;

for (auto& row : grid) {

reverse(row.begin(), row.end());

}

moved = swipeLeft(grid);

for (auto& row : grid) {

reverse(row.begin(), row.end());

}

return moved;

}

// Function to swipe up

bool swipeUp(vector<vector<int>>& grid) {

// Transpose the grid, then swipe left, then transpose again

vector<vector<int>> transposedGrid = grid;

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

for (int j = 0; j < 3; ++j) {

transposedGrid[i][j] = grid[j][i];

}

}

bool moved = swipeLeft(transposedGrid);

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

for (int j = 0; j < 3; ++j) {

grid[i][j] = transposedGrid[j][i];

}

}

return moved;

}

// Function to swipe down

bool swipeDown(vector<vector<int>>& grid) {

// Transpose the grid, reverse each row, swipe left, reverse again, then transpose again

vector<vector<int>> transposedGrid = grid;

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

for (int j = 0; j < 3; ++j) {

transposedGrid[i][j] = grid[j][i];

}

}

for (auto& row : transposedGrid) {

reverse(row.begin(), row.end());

}

bool moved = swipeLeft(transposedGrid);

for (auto& row : transposedGrid) {

reverse(row.begin(), row.end());

}

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

for (int j = 0; j < 3; ++j) {

grid[i][j] = transposedGrid[j][i];

}

}

return moved;

}

// Function to save game statistics to a file

void saveGameStats(const vector<vector<int>>& grid, bool reached2048) {

ofstream file("game\_stats.txt", ios::app);

if (!file.is\_open()) {

cout << "Error: Unable to open file for saving game stats." << endl;

return;

}

file << "Game Grid:" << endl;

for (const auto& row : grid) {

for (int val : row) {

file << val << " ";

}

file << endl;

}

file << "Reached 2048: " << (reached2048 ? "Yes" : "No") << endl;

file << "--------------------------" << endl;

file.close();

}

// Main function to start the game

void playGame() {

srand(time(0));

vector<vector<int>> grid = initializeGrid();

addNewTile(grid);

addNewTile(grid);

bool reached2048 = false;

while (true) {

system("cls||clear");

printGrid(grid);

if (reached2048 || isGameOver(grid)) {

cout << "Game Over!" << endl;

saveGameStats(grid, reached2048); // Save game stats

break;

}

cout << "Enter swipe direction (up/down/left/right): ";

string direction;

cin >> direction;

bool moved = false;

if (direction == "left") {

moved = swipeLeft(grid);

} else if (direction == "right") {

moved = swipeRight(grid);

} else if (direction == "up") {

moved = swipeUp(grid);

} else if (direction == "down") {

moved = swipeDown(grid);

} else {

cout << "Invalid direction! Please enter one of: up/down/left/right" << endl;

}

if (moved) {

if (addNewTile(grid)) {

// Check if 2048 is reached

for (const auto& row : grid) {

for (int val : row) {

if (val == 2048) {

reached2048 = true;

cout << "Congratulations! You reached 2048. Do you want to continue? (Y/N): ";

char choice;

cin >> choice;

if (choice == 'N' || choice == 'n') {

cout << "Game Over!" << endl;

saveGameStats(grid, reached2048); // Save game stats

return;

}

}

}

}

} else {

cout << "Game Over! No more moves possible." << endl;

saveGameStats(grid, reached2048); // Save game stats

break;

}

}

}

cout << "Press Enter to exit..." << endl;

cin.ignore(); // Ignore the newline left in the input buffer

cin.get(); // Wait for the user to press Enter

}

// Main function

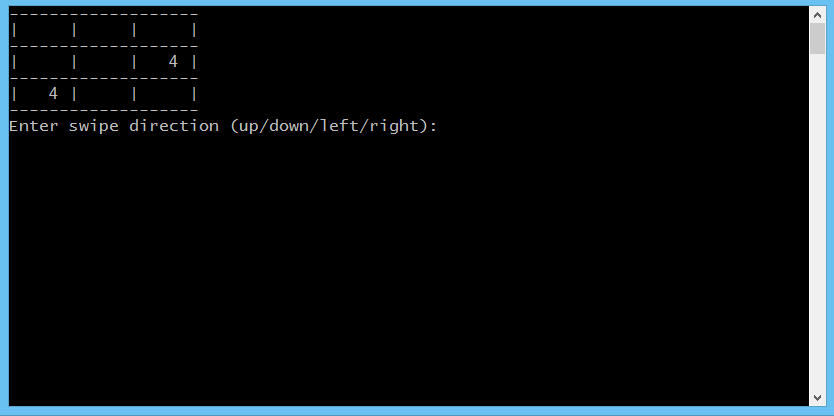
int main() {

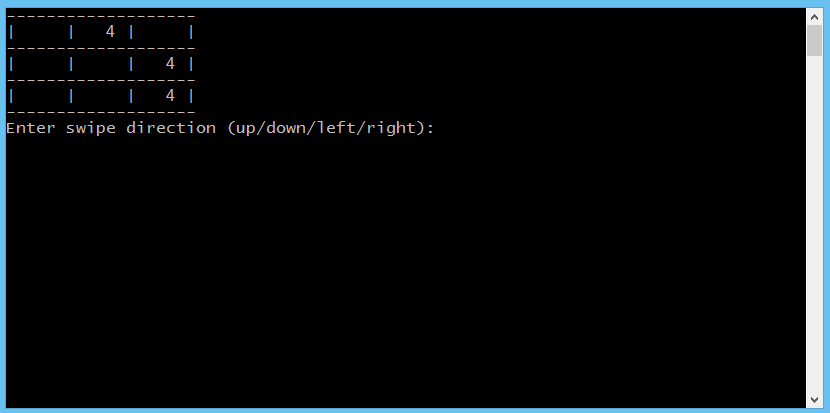
playGame();

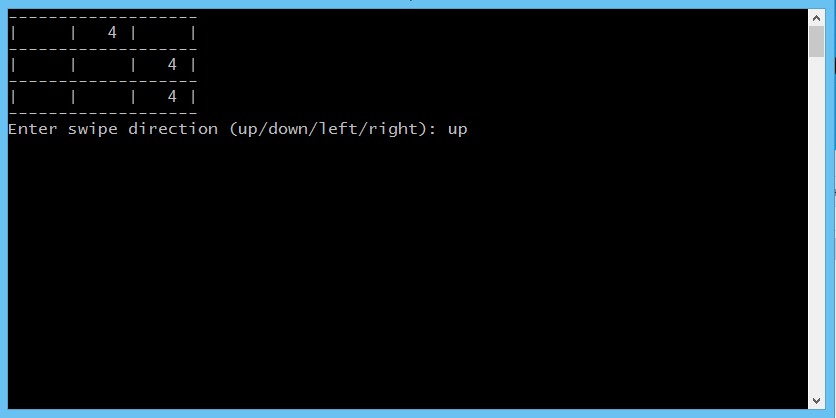
return 0;

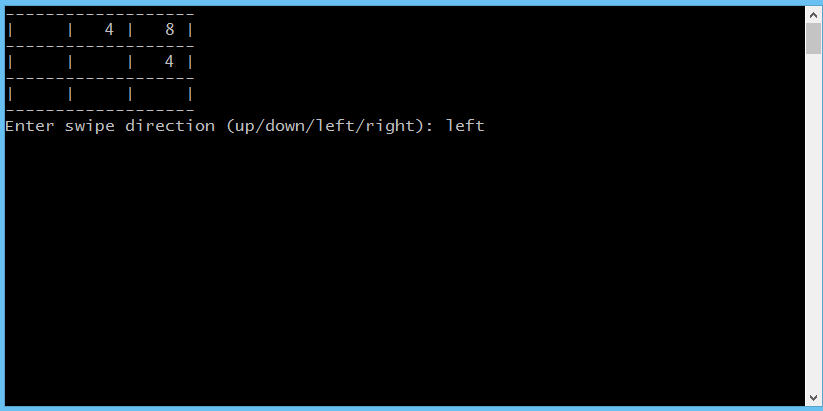
}

### 4.7.2 Скріншоти виконання програми











# 5 РОЗДІЛ ”ВЕБ-ДИЗАЙН ТА HTML”

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

**Тег** — це записана в кутових дужках < > позначка мови HTML, яку опрацьовує браузер.

<html> Початок документа

<head> Початок заголовка — службової частини коду

<title> Назва, що виводитиметься на вкладці браузера

</title> Кінець назви

</head> Кінець заголовка

<body> Початок «тіла» документа, яке виводить браузер

</body> Кінець «тіла» документа

</html> Кінець документа

Теги бувають **парними**і **непарними**. Парний тег складається з відкривального і закривального тегів. **Закривальний**тег починається символом «**/**».Для створення маркованого списку використовують тег <ul>…</ul>.

Елементи списку розмічають тегами <li>…</li> .

Для створення гіперпосилань на вебсторінки сайту та зовнішні ресурси мережі інтернет використовують тег **<a>...</a>**.

Теги розмітки таблиці:

Тег Призначення

<table>…</table> Обмежують таблицю

<tr>…</tr> Обмежують рядок таблиці

<td>…</td> Обмежують клітинку таблиці

<th>…</th> Використовують замість тегу <td>…</td> для виокремлення заголовків

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

Текст в HTML вводиться в будь-якому місці між тегами **<BODY></BODY>.**

Тег розриву абзацу **<P></P**>, та розриву рядка **<BR>.** . Для тегу розриву абзацу використання закриваючого тегу **</P>** не є необхідним.

**Теги виділення тексту**

|  |  |  |
| --- | --- | --- |
| **Тег** | **Стиль** | **Приклад** |
| **<B>** | Напівжирний шрифт | **Напівжирний** |
| **<strong>** | Напівжирний шрифт | **Напівжирний** |
| **<I>** | Курсив | *Курсив* |
| **<em>** | Курсив | *Курсив* |
| **<U>** | Підкреслення | Підкреслений текст |
| **<SUB>** | Підрядковий текст | Текст підрядковий текст |
| **<SUP>** | Надрядковий текст | Текст надрядковий текст |
| **<STRIKE>** | Перекреслення | Перекреслений |

У мові HTML для виділення заголовків передбачено ціле сімейство тегів: від **<h1>** до **<h6>.**Тег **<h1>** позначає найважливіший заголовок, а тег **<h6>** позначає підзаголовок самого нижнього рівня. Часто використовуваними тегами заголовків є: **<h1>, <h2> і <h3>.**

**Урок 2. Додавання посилань и додавання тексту.**

Для створення посилання необхідно повідомити браузеру, який елемент є посиланням, а також вказати адресу документа, на який слід зробити посилання. Обидва дії виконуються за допомогою тега <a>.

трибут href визначає URL, тобто адресу документа, на який слід перейти, а вміст контейнера <a> є посиланням. Текст, розташований між тегами <a> і </a>.

Щоб вставити малюнок на вебсторінку, необхідно скористатися тегом <img src = "my.jpg">, де img — ім'я тегу, призначеного для вставляння зображення, src = "my.jpg" — атрибут, в якому вказується ім'я файлу з малюнком.

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

CSS можна додати до елементів HTML трьома способами:

* Inline (вбудований або рядковий) - використовуючи атрибут style в HTML елементах
* Internal (внутрішній) - використовуючи елемент <style> в разділі <head>
* External (зовнішній) - за допомогою зовнішнього файлу CSS

Найбільш розповсюджений спосіб додати CSS - це зберігати стилі в окремих файлах CSS.

Вбудований CSS використовується для застосування унікального стиля до одного HTML элемента.

Вбудований CSS використовує атрибут style HTML елемента.

Внутрішній CSS використовується для визначення стилю окремої HTML сторінки.

Внутрішній CSS вказується в розділі <head> HTML сторінки за допомогою елемента <style>.

**За допомогою зовнішньої таблиці стилів ви можете змінити зовнішній вигляд всього веб-сайту, змінивши лише один файл!**

## **5.4 Спадковості і каскадування за допомогою CSS**

**Спадковість**

Стилі, присвоєні деякого елементу, успадковуються всіма нащадками (вкладеними елементами), якщо вони не перевизначені (перекриті) явно.

**Спадковість властивості**

До спадкоємною властивостей відносяться в першу чергу властивості, що визначають параметри відображення тексту:

font-size, font-family, font-style, font-weight, color, text-align, text-transform, text-indent, line-height, letter-spacing, word-spacing, white-space, direction і т. д.

Також до спадкоємною властивостей відносяться list-style, cursor, visibility, border-collapse і деякі інші.

### 5.5 Практична частина

**Код:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>My Resume</title>

</head>

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

<body>

<div class="container">

<img src="C:\Users\Олег\Desktop\Практика\my\_photo (2).jpg" alt="Photo" class="round-img">

<div class="name"><strong>Oleg Onypchenko</strong></div>

<div class="os"><strong>Windows/macOS</strong></div>

<h2> Summary </h2>

<hr>

<p1>My name is Oleg. I am a person who always strives for self-development and achieving new heights. <br> Thanks to my desire for knowledge and constant desire to learn, I quickly absorb new information and effectively apply it in practice.

I am passionate about programming and web development, and during college I successfully learned the basics of web programming. <br> My skills in this area allow me to create quality web applications and websites.</p1>

<h2>Educatio</h2>

<hr>

<p2>Finished 9 classes at KZO SZSh No. 17. <br> Then he studied at the Optical-Mechanical College with a degree in Software Engineering.</p2>

<h2>Experience</h2>

<hr>

<p3>Completed a programming internship as part of the college curriculum.<br>

- Learned basic programmingconcepts, including programming languages, data structures, and algorithms.<br>

- Practically applied the acquired knowledge to create simple programs and web applications.<br>

- Participated in projects to develop websites and web applications.<br>

- Studied the basics of web programming, including <strong>HTML, CSS.</p3>

<br>

<div class="btn-container">

<a href="https://www.instagram.com/\_olleeeeg\_?igsh=ZzBxdnF3eWt3bWtx" target="\_blank" class="btn"><img src="instagram\_icon.png" alt="Instagram Icon">Instagram</a>

<a href="https://t.me/Ufyrj" target="\_blank" class="btn"><img src="telegram\_icon.png" alt="Telegram Icon">Telegram</a>

<a href="mailto:jzjzjzjzjzjzjz351@gmail.com" target="\_blank" class="btn" ><img src="gmail\_icon.png" alt="Gmail Icon">Email</a>

</div>

</div>

</body>

</html>

**Стилі:**

<style>

.oleg\_id{

text-align: center;

color: black;

}

.name {

text-align: center;

font-size: 25px;

color: blue;

margin-bottom: 10px;

}

.round-img {

border-radius: 50%;

width: 160px;

height: 160px;

margin: 0 auto;

display: block;

}

.os{

text-align: center;

font-size: 15px;

}

.btn-container{

padding: 20px;

text-align: center;

}

.btn {

position: relative;

display: block;

padding: 10px 200px;

margin: 10px auto;

color: black;

background-color: white;

border: 2px solid black;

border-radius: 5px;

text-decoration: none;

transition: background-color 0.3s, color 0.3s;

width: calc(100% - 40px); ;

max-width: 200px; /

overflow: hidden;

}

.btn:hover {

background-color: black;

color: white;

}

.btn img {

position: absolute;

left: 10px;

top: 50%;

transform: translateY(-50%);

width: 20px;

height: 20px;

transition: filter 0.3s;

}

.btn:hover img {

filter: invert(100%);

}

.container {

text-align: left;

background-color: lightgray;

padding: 20px;

border-radius: 10px;

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

width: 100%;

max-width: 800px;

}

body {

text-align: left;

display: flex;

justify-content: center;

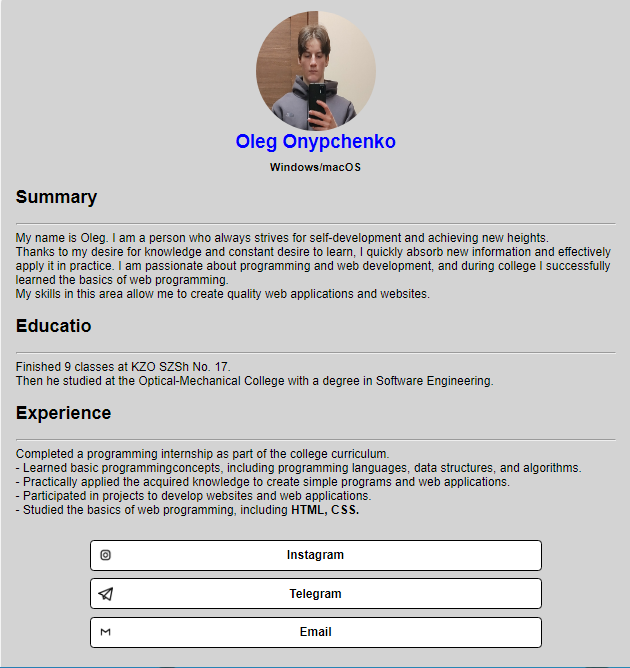
align-items: center;

min-height: 100vh;

font-family: Arial, sans-serif;

}

</style>



### 5.5.1 Постановки задачі

### 5.5.2 HTML документ

# 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