МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ВІДОКРЕМЛЕНИЙ СТРУКТУРНИЙ ПІДРОЗДІЛ

«ПОЛТАВСЬКИЙ ПОЛІТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ

НАЦІОНАЛЬНОГО ТЕХНІЧНОГО УНІВЕРСИТЕТУ

«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Циклова комісія дисциплін програмної інженерії

**КУРСОВА РОБОТА**

Розробка гри на Unreal Engine 5

на тему \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

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

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

Виконав: здобувач освіти 2 курсу, групи \_\_\_\_\_\_\_  
 спеціальності 121   
 Інженерія програмного забезпечення

Попович I.I.

25

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (прізвище та ініціали)

Керівник

Олійник В.В.

\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
 (підпис) (прізвище та ініціали)

Полтава – 2024

**ЗМІСТ**

[ВСТУП 3](#_Toc169265641)

[1.1. Основні вимоги до продукту 6](#_Toc169265642)

[1.2. Вимоги до інтерфейсу 9](#_Toc169265643)

[2. ПЛАНУВАННЯ СИСТЕМИ 11](#_Toc169265644)

[2.1. Алгоритм роботи програми. 11](#_Toc169265645)

[2.2. Тестування 15](#_Toc169265646)

[2.3. Інструкція з використання системи 16](#_Toc169265647)

[ВИСНОВКИ 17](#_Toc169265648)

[СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ 19](#_Toc169265649)

[ДОДАТОК А. ПРОТОТИП IНТЕРФЕЙСУ 20](#_Toc169265650)

[ДОДАТОК Б. ВИХIДНI КОДИ 22](#_Toc169265651)

[ДОДАТОК В. РЕЗУЛЬТАТИ ТЕСТУВАННЯ 23](#_Toc169265652)

[ДОДАТОК Г. ЗНIМКИ ЕКРАНУ 25](#_Toc169265653)

# ВСТУП

Ця робота присвячена дослідженню та практичному застосуванню сучасних технологій 3D-моделювання, текстурування та програмування, зокрема в контексті розробки ігор. Вона є результатом самостійного вивчення та практичного досвіду, отриманого під час роботи з різноманітним програмним забезпеченням забезпеченням.

Мета роботи полягає в удосконаленні знань та навичок у сфері 3D-моделювання, текстурування та програмування, зокрема в мові C++ та ігровому движку Unreal Engine 5, а також у практичному застосуванні отриманих знань для створення власних 3D-моделей, текстур та ігрових механік.

Предмет роботи – це процес створення 3D-моделей та текстур, а також розробка ігрових механік, використовуючи мову C++ та ігровий движок Unreal Engine 5.

Об'єкт роботи – це 3D-моделі, текстури та ігровий код, створені в рамках цього дослідження, які реалізують певний ігровий функціонал.

Актуальність роботи обумовлена зростаючою популярністю ігрової індустрії та розвитком технологій віртуальної реальності. Навчання 3D-моделюванню, текстуруванню та програмуванню для ігор є важливим кроком для освоєння нових професій та отримання конкурентних переваг на ринку праці.

Завдання роботи.

Удосконалити знання та навички в програмуванні на C++, зокрема в частині роботи з класами, структурами даних та іншими елементами, що використовуються в Unreal Engine.

Ознайомитися з основами роботи в Unreal Engine 5 та навчитися створювати прості ігрові механіки та функціонал.

Навчитися створювати 3D-моделі за допомогою програмного забезпечення Blender.

Оволодіти навичками текстурування за допомогою Substance 3D Painter та Substance 3D Designer.

Інтегрувати створені 3D-моделі та текстури в Unreal Engine 5 та налаштувати їх візуальні ефекти.

Розробити власний ігровий функціонал за допомогою C++ та Unreal Engine 5.

Для виконання поставлених завдань були використані наступні програмні засоби:

Програмне забезпечення:

* Unreal Engine 5. Unreal Engine 5: ігровий двигун, який буде використовуватися для розробки гри;
* Visual Studio. Середовище розробки для написання коду на C++, та вiдладки проєкту;
* Blender. Програмне забезпечення для створення 3D-моделей;
* Substance Painter. Програмне забезпечення для текстурування всіх об'єктів;
* Rizom UV. Програмне забезпечення для розгортки 3D моделі та подальшого пакування отриманого результату;
* Photoshop. Програмне забезпечення для правки карт та планування приміщень, тощо;
* Marmoset ToolBag. Програмне забезпечення для запікання карт (нормалей, ambient occlusion, та інших).

**1. ПОСТАНОВКА ЗАВДАННЯ**

Створити гру на Unreal Engine 5 з базовими функціями, що включають:

* пересування. Персонаж повинен мати можливість рухатися по локації, перестрибувати перешкоди та взаємодіяти з об'єктами;
* взаємодія з об'єктами. персонаж повинен мати можливість взаємодіяти з різними об'єктами на локації, наприклад, підбирати предмети, активувати механізми тощо;
* створена локація. гра повинна мати власну локацію, розроблену в Unreal Engine 5, з різними елементами ландшафту, об'єктами та перешкодами.

Додаткові функції:

* пересування через перешкоди. Персонаж повинен мати можливість долати перешкоди, використовуючи різні механіки, наприклад, стрибки, лазіння, та iншi;
* штучний інтелект. Штучний інтелект, який буде перешкоджати гравцеві пройти рівень;
* візуальні ефекти. Візуальні ефекти, що покращують її візуальну складову, наприклад, ефекти вибухів, диму, вогню тощо;
* окуляри. Гра повинна мати окуляри, які надають гравцеві додаткові можливості, наприклад прибирати завали;
* збереження. Гра повинна мати функцію збереження прогресу гравця;
* iгрове меню. Гра повинна мати ігрове меню, що дозволяє гравцеві переглядати налаштування, вибирати рівні, переглядати статистику тощо;
* налаштування гри. Гра повинна мати можливість налаштування графіки, звуку та інших параметрів;
* синематики. Синематики, що допомагають розповісти історію гри або представити нові елементи;
* iнтерфейс. Iнтерфейс який буде виводити кiлькiсть витривалостi, яка залишилася у гравця та предмети, які є у гравця в ігровому інвентарі.

Середовище та мова розробки:

* Unreal Engine 5: ігровий двигун, який буде використовуватися для розробки гри;
* Visual Studio. Середовище розробки для написання коду на C++, та вiдладки проєкту;
* C++ – мова програмування, яка буде використовуватися для написання коду гри.

## Основні вимоги до продукту

Розробити гру в програмі Unreal Engine 5 з використанням мови програмування C++.

Побудова гри:

Гра побудована за модульним принципом, де майже кожна функцiя є окремим модулем, що відповідає за певну функціональність. Модулі взаємодіють між собою, що дозволяє легко змінювати та розширювати функціональність гри.

Технології та інструменти:

* Unreal Engine 5. Unreal Engine 5: ігровий двигун, який буде використовуватися для розробки гри..
* Visual Studio. Середовище розробки для написання коду на C++, та вiдладки проєкту.
* Blender. Програмне забезпечення для створення 3D-моделей.
* Substance Painter. Програмне забезпечення для текстурування всіх об'єктів.
* Rizom UV. Програмне забезпечення для розгортки 3D моделі та подальшого пакування отриманого результату
* Photoshop. Програмне забезпечення для правки карт та планування приміщень, тощо.
* Marmoset ToolBag. Програмне забезпечення для запікання карт (нормалей, ambient occlusion, та інших)
* Gameplay Framework. Фреймворк Unreal Engine, що надає базові функції для створення ігрового процесу.
* C++. Мова програмування, що використовується для написання коду гри.

Залежності та встановлення.

Гра зiбрана в один архiв. Для правильної роботи гри, треба:

* встановити та разпакувати архiв з GitHub репозиторiю;
* встановити компонент DirectX12.

ФУНКЦІОНАЛЬНІ ВИМОГИ:

* пересування. Персонаж може рухатися по локації, перестрибувати перешкоди та взаємодіяти з об'єктами;
* взаємодія з об'єктами. Персонаж може взаємодіяти з різними об'єктами на локації, наприклад, підбирати предмети, активувати механізми тощо;
* Штучний інтелект. Штучний інтелект, який перешкоджає гравцеві пройти рівень;
* Окуляри. Окуляри, які надають гравцеві додаткові можливості;
* Збереження. Збереження прогресу гравця;
* Інтерфейс. Iгрове меню, iнвентар, виведення витривалостi;
* Налаштування гри. Налаштування графіки, звуку та інших параметрів.

НЕФУНКЦІОНАЛЬНІ ВИМОГИ:

* cинематики. Синематики, що допомагають розповісти історію гри або представити нові елементи;
* візуальні ефекти. Візуальні ефекти, що покращують її візуальну складову.

ПЕРЕЛІК РЕАЛІЗОВАНИХ ЗА ЧАС ПРАКТИКИ ФУНКЦІОНАЛЬНИХ ВИМОГ

* Пересування.
* Взаємодія з об'єктами.
* Створена локація.
* Iнтерфейс.

ПЕРЕЛІК НЕРЕАЛІЗОВАНОЇ ФУНКЦІОНАЛЬНОСТІ

* Пересування через перешкоди;
* штучний інтелект;
* візуальні ефекти;
* iгровий предмет “окуляри”, якi будуть змiнювати вид локацiї;
* збереження;
* iгрове меню;
* налаштування гри;
* синематики.

Функціональність була не реалізована через невелику кількість часу (вивчення програм для роботи з 3D моделями; вивчення початкового рівня роботи з Unreal Engine; початкового рівня для роботи з (ООП) та Blueprints; реалізації 3D моделей, текстур; та проблем з знаходженням інформацією роботи з Unreal Engine).

## Вимоги до інтерфейсу

Iнвентар.

Інтерфейс інвентаря складається з головної задньої частини "кордонів", по якій налаштовують свій розмір кнопки, і наявні предмети.

Предмети розміщуються з відстанню один від одного, один за одним, і мають міні зображення предмета.

Автоматичний розподіл по області, з автоматичним масштабуванням, зручний варіант для розробки та гравців.

Прототип в додатку А, зображення 1.

Витривалiсть.

Інтерфейс витривалості складається з меж і смужки витривалості.

При використанні витривалості смужка звужується до середини. При відновленні розтягується до меж.

Розтягування та стиснення смужки робить інтерфейс зрозумілим для будь-якої людини.

Прототип в додатку А, рисунок 2, 3, та 4.

Iгрове меню.

У центрі екрану розташований логотип гри з написом "RE MEMORY INC.", де слово "RE" виділене фіолетовим кольором, а решта логотипу сірого кольору.

Праворуч від логотипу знаходяться чотири чорні кнопки з білими написами:

1. Continue (Продовжити);
2. Load Game (Завантажити гру);
3. Settings (Налаштування);
4. Quit (Вийти).

Кожна кнопка має прямокутну форму і розташована вертикально одна під одною.

Прототип в додатку А, рисунок 5.

Використаний GUI-фреймворк для інтерфейсу - Unreal Motion Graphics (UMG).

Використаний сервісу для планування GUI - draw.io.

# 2. ПЛАНУВАННЯ СИСТЕМИ

## 2.1. Алгоритм роботи програми.

Перемiщення вперед та назад (MoveForwardBackwfrd).

Якщо натиснута клавiша W, персонаж переміщається на +1 значення осі X.

Якщо натиснута клавiша S, персонаж переміщається на -1 значення по осi X.

Перемiщення влiво та вправо (MoveRightLeft).

Якщо натиснута клавiша D, персонаж переміщається на +1 значення осі Z.

Якщо натиснута клавiша A, персонаж переміщається на +1 значення осі Z.

Стрибок (Jump).

Якщо натиснута клавiша Space, викликаємо функцію Jump(), в якій встановлюємо для успадкованої змінної "bPressedJump" від класу Character, значення true (стрибок). Пiсля відтискання клавiши, викликаємо функцію StopJump(), в якій встановлюємо для успадкованої змінної "bPressedJump" від класу Character, значення false (зупинка стрибка).

Присідання (Crouch).

Якщо натиснута клавiша Left Ctrl, викликаємо успадковану функцію від класу Character "Crouch", зi значенням false (присісти). Після відтискання клавiши, викликаємо успадковану функцію від класу Character "UnCrouch", зi значенням false (встати).

Бiг (Sprint).

Якщо натиснуто клавiшу Shift, змінюємо значення для змінної “bIsSprint” на значення true; для змінної MaxWalkSpeed встановлюємо значення 800 (збільшуємо швидкість ходьби на 400); в будованнiй функцiї Tick, викликаємо функцiю DecreaseStamina() (віднімання заданого значення від витривалості).

Якщо клавішу не відпущено, а витривалiсть дорiвнює, або меньше 0 - викликаємо функцiю StopSprint() (примусова зупинка бiгу); для змінної MaxWalkSpeed встановлюємо значення 400 (зменшуємо швидкість ходьби); встановлюємо для значення витривалостi, значення 0 (для запобігання нескінченному віднімання від значення витривалості, якщо витривалiсть меньше 0); викликаємо функцiю IncreaseStamina() (відновлення витривалості, раз у Tick).

Якщо клавiшу вiдпущено, для змінної MaxWalkSpeed встановлюємо значення 400 (зменшуємо швидкість ходьби); викликаємо функцiю IncreaseStamina() (відновлення витривалості, раз у Tick).

Iнвентар (Inventory).

Якщо натиснуто клавiшу I, та iнвентар не виведено на екран - виводимо WBP\_Inventory (iнтерфейс iнвентарю) за допомогою функцiї AddToViewport() на екран, та відображаємо курсор миші за допомогою функції SetCursorVisibility(true).

Якщо натиснуто клавiшу I, та iнвентар виведений на екран - прибираємо WBP\_Inventory (iнтерфейс iнвентарю) за допомогою функції RemoveFromParent(), та прибираємо курсор миші за допомогою функції SetCursorVisibility(false).

Взаємодія(Interact).

Якщо натиснуто клавiшу E, створюємо лiнiю за допомогою функцiї LineTraceSingle.

Якщо лiнiя торкається об'єкта, який успадкований від класу Pickups - підібрати в інвентар.

Якщо торкається об'єкта, який має ComponentTag “Door”, перевiряємо, чи зачиненi дверi на ключ. Якщо дверi не зачиненi, викликаємо функцiю ToggleDoor() (вiдкриття, або зачинненя дверей).

Якщо дверi зачиненi на ключ – перевiряємо, чи є в iнветарi ключ. Якщо ключ в iнвентарi є, викликаємо функцiю ToggleDoor() (вiдкриття, або зачинненя дверей). Якщо немає, нiчого не робимо.

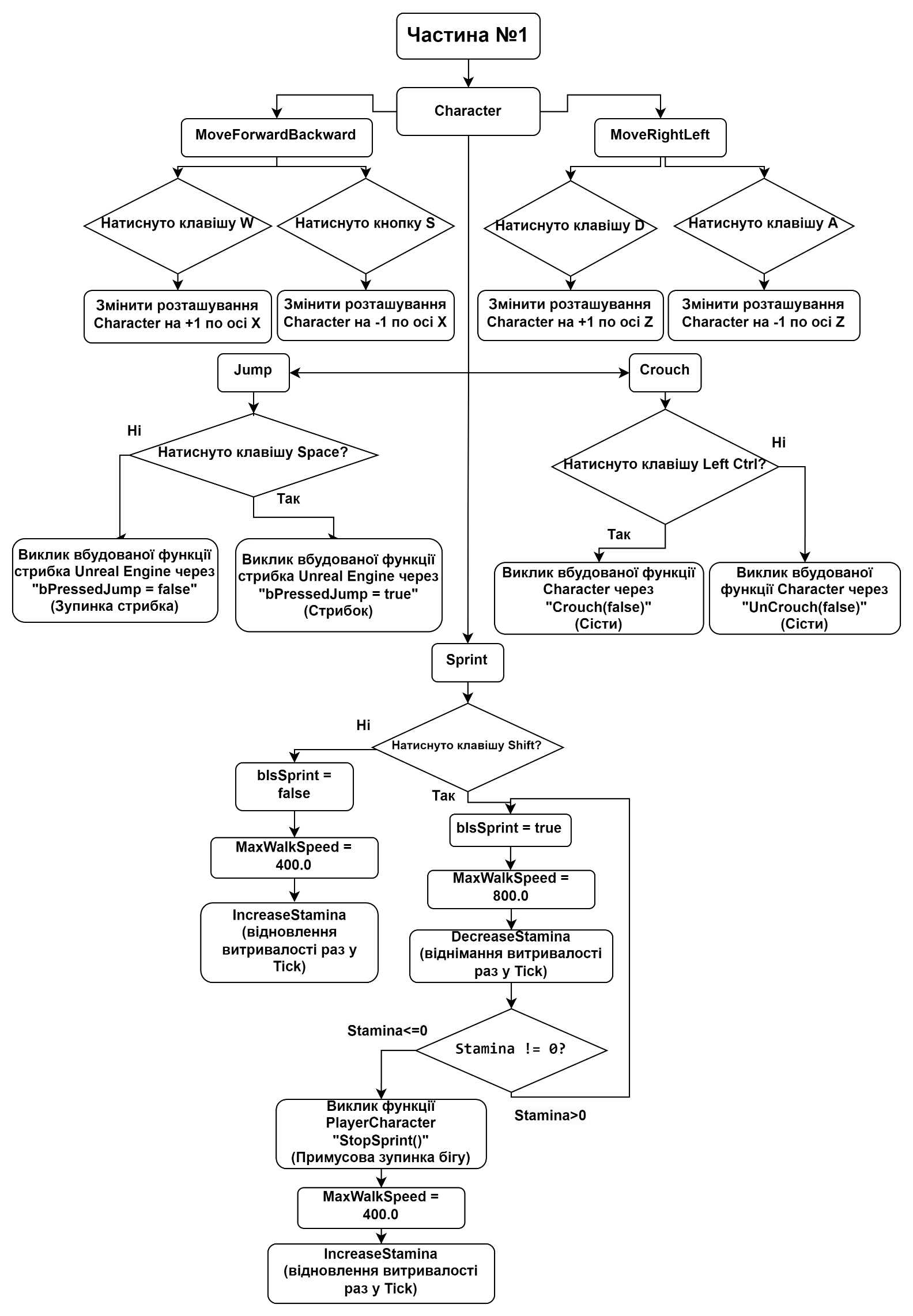


Рисунок 1. Алгоритм роботи програми, частина №1.

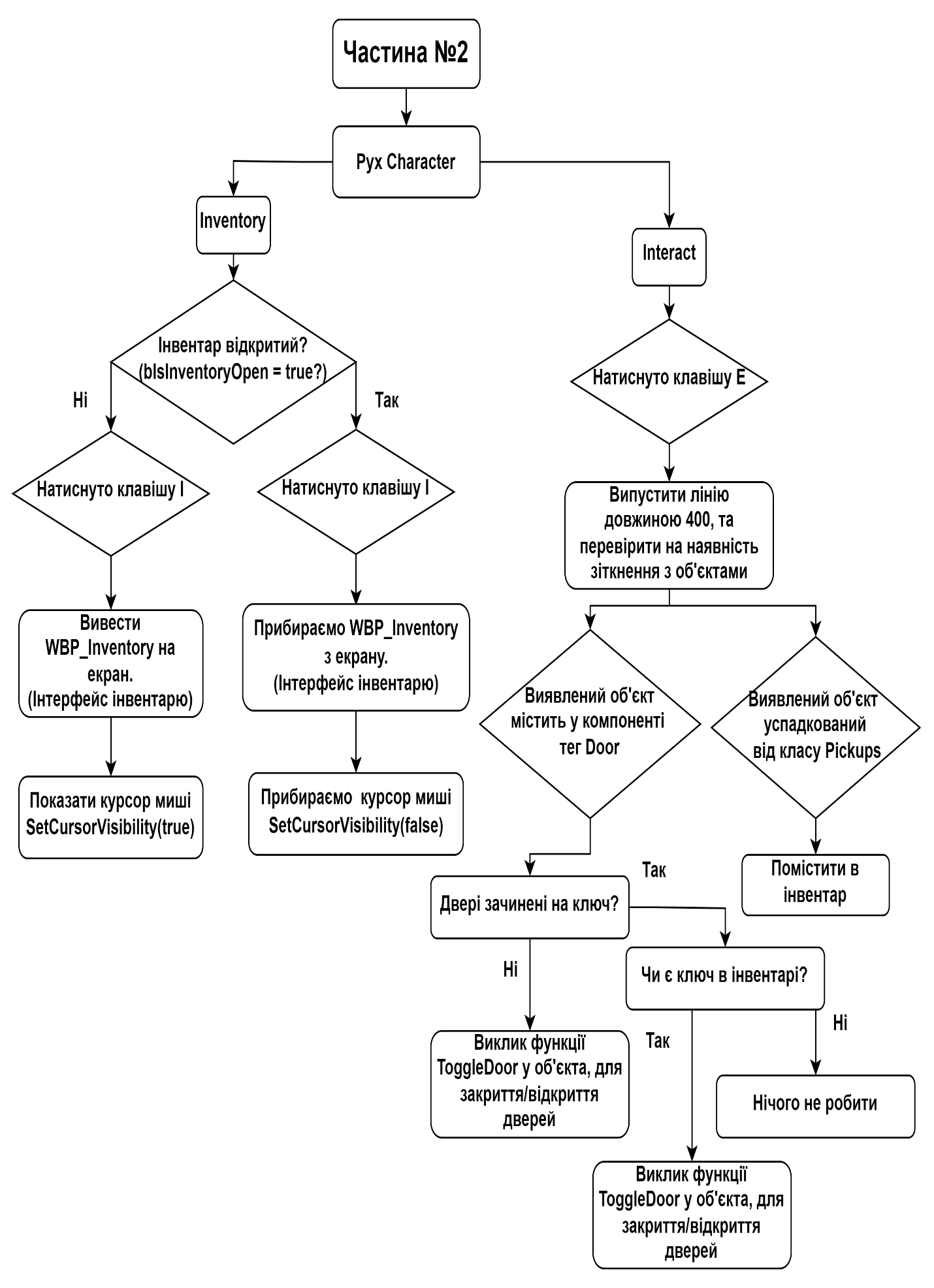


Рисунок 1. Алгоритм роботи програми, частина №2.

## 2.2. Тестування

1. Функціональне тестування (Functional Testing) - це процес перевірки роботи всіх функцій гри, щоб переконатися, що вони виконують свої завдання правильно. Це один із найважливіших видів тестування, оскільки він спрямований на перевірку основної функціональності гри.

2. Тестування юзабіліті (Usability Testing) - це процес оцінки зручності та інтуїтивності інтерфейсу гри, а також загального досвіду користувача. Це важливо для того, щоб гравці могли легко розуміти і використовувати всі функції гри.

Сценарій тестування:

1. запуск гри;
2. управління персонажем (біг, стрибки, перевірка стаміни);
3. використання інвентарю, відкриття інвентарю;
4. взаємодія з оточенням, збирання нових предметів і додавання в інвентар;
5. перевірка оновлення стаміни під час виконання дій;
6. перевірка реакції інтерфейсу на дії гравця;
7. перевірка зміни рівню стаміни при бігу.

Результати тестування в додатку В.

## 2.3. Інструкція з використання системи

Залежності та встановлення.

Гра зiбрана в один архiв.

Для правильної роботи гри, треба:

1. встановити та разпакувати архiв;
2. встановити компонент DirectX12;
3. запустити файл RM.exe.

Для запуску вiдладки проєкту треба:

1. завантажите весь проєкт з репозиторiю на GitHub;
2. разпакувати;
3. завантажити Unreal Engine 5.3;
4. натиснути правою кнопкою мишi по файлу “RM.uproject”, та натиснути “Generate Visual Studio project files (додаток Г. Рисунок 1);
5. вiдкриваємо файл RM.sln;
6. нажимаємо кнопку “Локальний вiдладчик Windows” (додаток Г. Рисунок 2).

# ВИСНОВКИ

Основною метою курсової роботи було практичне оволодіння навичками розробки ігор на Unreal Engine 5, поглиблення знань у C++ та об'єктно-орієнтованому програмуванні (ООП), створення першого великого проекту, а також вивчення нових навичок у сфері 3D моделювання, текстурування та інших суміжних областях.

Завдання проекту.

Завдання полягало у створенні простої гри, яка демонструє знання основних принципів ігрової розробки:

* cтворення навколишнього середовища;
* розробка логіки гри;
* робота з редакторами 3D моделювання та 3D моделями;
* користувацький інтерфейс (UI).

Навчальні аспекти:

* оволодіння інтерфейсом та інструментами Unreal Engine 5;
* використання Blueprint та C++ скриптів для розробки ігрової логіки;
* 3D моделювання;
* основи моделювання 3D об'єктів;
* iмпортування та експортування моделей;
* текстурування та планування;
* проектування ігрових механік;
* розробка логіки взаємодії між об'єктами у грі;
* створення та налаштування елементів UI;
* вдосконалення навичок;
* логічне мислення та алгоритмізація ігрової механіки;
* творче вирішення проблем та усунення помилок;
* розвиток проекту.

Прототип гри, створений у рамках курсової роботи, планується розвивати до повноцінного продукту та викласти в інтернет-магазин Steam або Epic Games. Цей проект також може слугувати демонстрацією навичок студента у швидкому навчанні, незважаючи на короткі терміни, та можливості розвивати навички в інших сферах, таких як 3D моделювання, текстурування та інші.

Курсова робота продемонструвала здатність студента швидко опановувати нові технології та інструменти, розробляти комплексні проекти, а також вирішувати технічні та творчі задачі. Прототип гри може слугувати основою для подальшого розвитку та вдосконалення, що підвищує професійний рівень розробника.

# СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Шилдт Г. Довідник програміста C/C++. Харків: Ранок, 2018. 432 с.
2. Герберт Ш. C++: базовий курс. Київ: Науковий світ, 2022. 624 с.
3. Вступ до програмування мовою С++. Організація обчислень: навчальний посібник / Бєлов Ю. А. та ін. . Київ: Київський університет, 2012. 175 с.
4. Документацiя Unreal Engine: [Веб-сайт]. URL: https://dev.epicgames.com/documentation/ru-ru/unreal-engine/unreal-engine-5-3-documentation (дата звернення: 01.03.2024).
5. МЕТОДИЧНI ВКАЗIВКИ ДО ВИКОНАННЯ КУРСОВИХ РОБIТ ДЛЯ ЗДОБУВАЧIВ ОСВIТИ СПЕЦIАЛЬНОСТI 121 «IНЖЕНЕРIЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ»: МЕТОДИЧНI ВКАЗIВКИ / Вiдокремлений структурний пiдроздiл «Полтавський полiтехнiчний фаховий коледж Нацiонального технiчного унiверситету «Харкiвський полiтехнiчний iнститут»; [уклав.: В.В.Олiйник]. – Полтава: ВСП ППФК НТУ “ХПI”, 2022. – 22 с.

# ДОДАТОК А. ПРОТОТИП IНТЕРФЕЙСУ

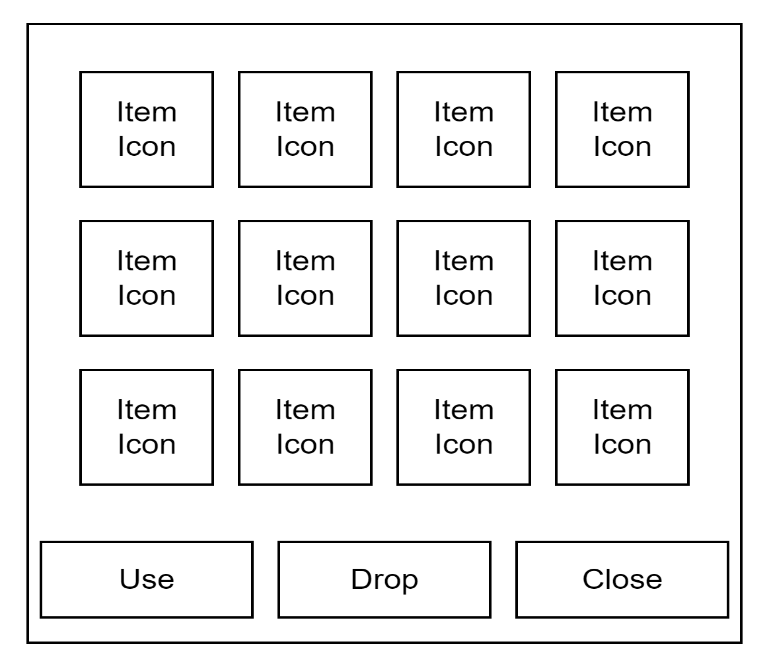


Рисунок 1. Прототип інтерфейсу інвентарю.



Рисунок 2. Прототип iнтерфейсу витривалостi (використання витривалостi).

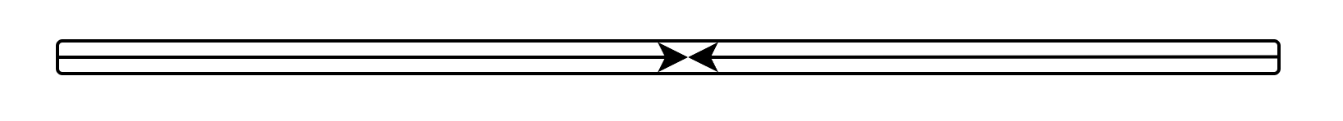


Рисунок 3. Прототип iнтерфейсу витривалостi (витривалiсть використана).



Рисунок 4. Прототип iнтерфейсу витривалостi (витривалiсть повна).

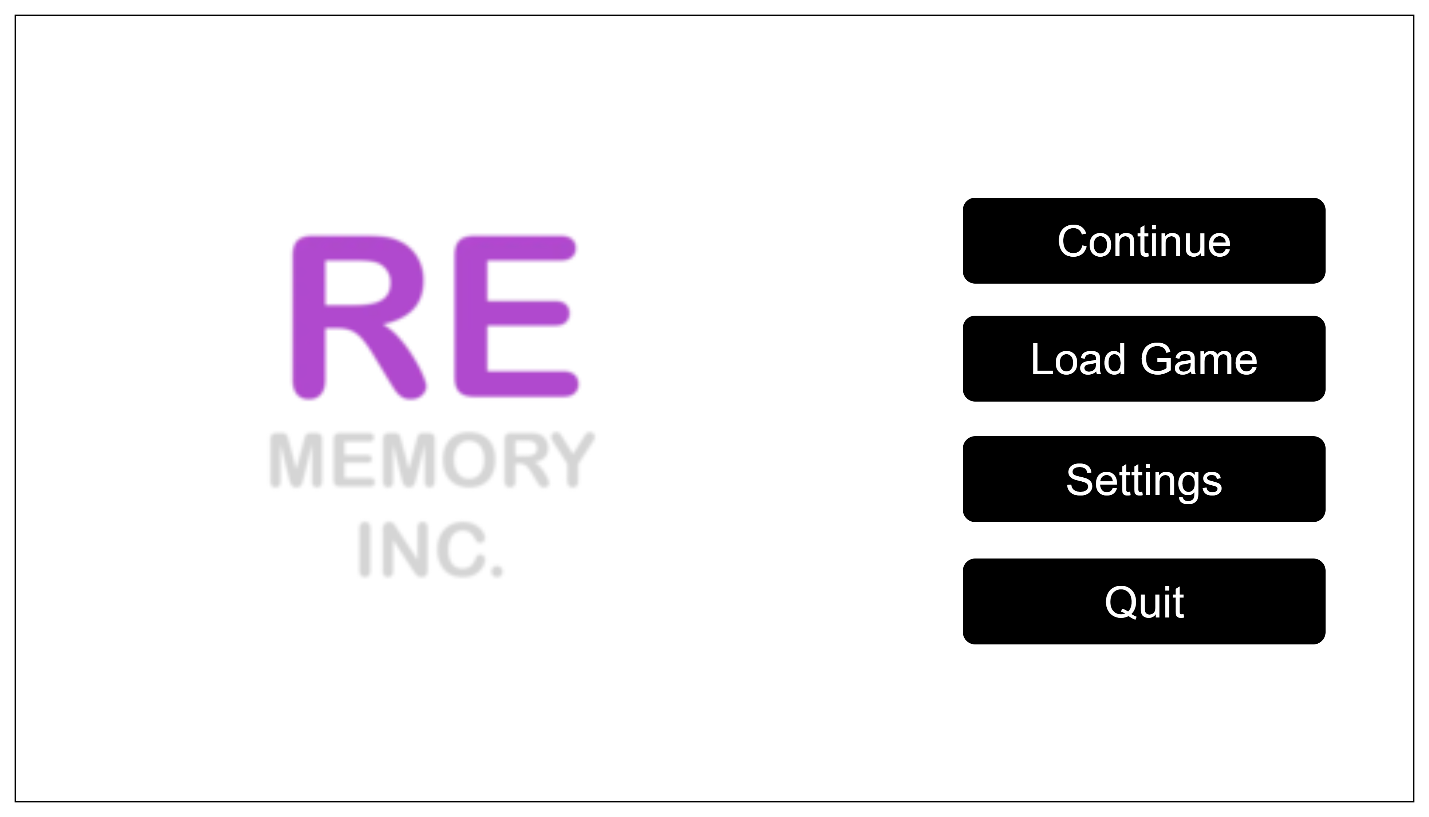


Рисунок 5. Прототип iнтерфейсу головного меню.

# ДОДАТОК Б. ВИХIДНI КОДИ



QR-Code на GitHub репозиторiй https://github.com/Glerik23/HorrorGame

# ДОДАТОК В. РЕЗУЛЬТАТИ ТЕСТУВАННЯ

Результати тестування:

У ході ручної перевірки були виявлені проблеми з виходом витривалості в мінус, при роботі з не цілими значеннями (float). Було виправлено помилку, додаванням перевірки на витривалість менше нуля, примусової зупинки бігу, та встановлення значення для витривалості 0.

* Біг та стрибки працюють коректно, стаміну зменшується при її використанні.
* Візуальний індикатор стаміни (смуга) легко помітний, але трохи дрібний.
* Використання інвентарю:
* Відкриття інструменту відбувається швидко, інтерфейс зрозумілий.
* Перевірка реакції інтерфейсу на дії гравця:
* Індикатор стаміни чітко показує зміни, але потребує покращення візуального відображення.
* Повідомлення про вичерпання стаміни з'являються, але гравці зазначили, що вони надто дрібні та непомітні.
* Нові предмети відразу відображаються в інвентарі, що добре.

Запропоновані покращення:

* збільшити розмір візуального індикатора стаміни: Зробити смугу стаміни помітнішою і додати яскраві візуальні ефекти при її вичерпанні;
* додати підказки в інвентар (показати короткий опис предметів при наведенні курсору);
* ввести кнопки для автоматичного сортування предметів Це полегшить організацію інвентарю;
* поліпшити сповіщення про вичерпання стаміни: Зробити їх більш помітними, можливо, з використанням звукових сигналів та анімацій;

Висновок.

Тестування юзабіліті інтерфейсу зі стаміном та інвентарем виявило декілька областей для покращення. Впровадження запропонованих змін допоможе зробити інтерфейс більш інтуїтивним та зручним для гравців, що підвищить загальний рівень задоволеності грою.

# ДОДАТОК Г. ЗНIМКИ ЕКРАНУ

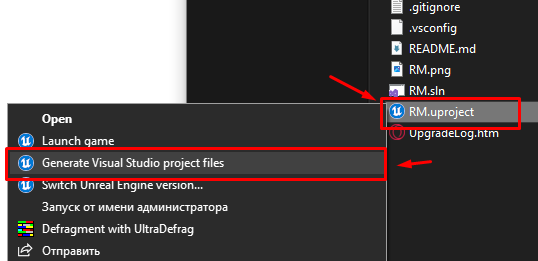


Рисунок 1. Створення файлів для вiдладки.

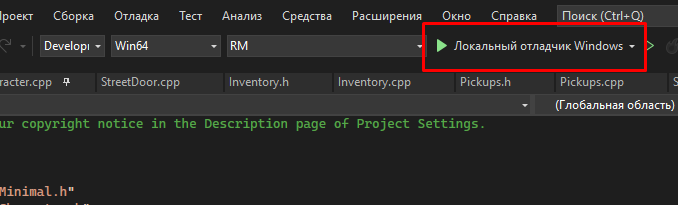


Рисунок 2. Збiрка та вiдладка проєкту.