Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

Інститут електроенергетики
(інститут)

Факультет інформаційних технологій
(факультет)

Кафедра Програмного забезпечення комп'ютерних систем
(повна назва)

ПОЯСНЮВАЛЬНА ЗАПИСКА
кваліфікаційної роботи ступеня

бакалавра
(назва освітньо-кваліфікаційного рівня)

студента *Кудрявцева Дмитра Сергійовича*
(ПІБ)

академічної групи *122-19-1*
(шифр)

спеціальності *122 Комп'ютерні науки*
(код і назва спеціальності)

освітньої програми *Комп'ютерні науки*
(назва освітньої програми)

на тему:
Розробка мобільної гри на мові програмування Python з використанням бібліотеки Pygame

Керівники	**Прізвище, ініціали**	**Оцінка за шкалою**		**Підпис**
		рейтинговою	**інституційною**	
кваліфікаційної роботи	*проф. Мороз Б.І*			
розділів:				
спеціальний	*проф. Мороз Б.І*			
економічний	*проф. Вагонова О.Г*			
Рецензент				
Нормоконтролер	*доц. Гуліна І.Г*			

Дніпро
2023

Міністерство освіти і науки України
НТУ «Дніпровська політехніка»

ЗАТВЕРДЖЕНО:
завідувач кафедри
програмного забезпечення комп'ютерних систем
(повна назва)

_____ І.М. Удовик
(підпис) (прізвище, ініціали)

« » _____ 2023 року

ЗАВДАННЯ
на кваліфікаційну роботу

бакалавра
(назва освітньо-кваліфікаційного рівня)

студента *122-19-1* *Кудрявцев Д.С*
(група) (прізвище та ініціали)

тема кваліфікаційної роботи *Розробка мобільної гри на мові програмування Python з використанням бібліотеки Pygame*

затверджена наказом ректора НТУ «ДП» від №

Розділ	Зміст виконання	Термін виконання
Спеціальний	***На основі матеріалів виробничої практики та інших науково-технічних джерел провести аналіз стану рішення проблеми та постановку задачі.*** ***Обґрунтувати вибір та здійснити реалізацію методів вирішення проблеми***	***13.05.2023 р.***
Економічний	***Провести розрахунок трудомісткості розробки програмного забезпечення, витрат на створення ПО й тривалості його розробки***	***27.05.2023 р.***

Завдання видав _____ *Проф. Мороз Б.І*
(підпис) (посада, прізвище, ініціали)

Завдання прийняв до виконання _____ *Кудрявцев Д.С*
(підпис) (прізвище, ініціали)

Дата видачі завдання: *05.01.2023 р.*

Термін подання кваліфікаційної роботи до ЕК: *22.06.2023 р.*

РЕФЕРАТ

Пояснювальна записка: 64 с., 21 рис., 2 табл., 20 джерел.

Об'єкт розробки: мобільна гра

Мета кваліфікаційної роботи: надати користувачу незабутнє відчуття від гри та поширити його майстерність та реакцію

У вступі розглядається аналіз та сучасний стан проблеми, конкретизується мета кваліфікаційної роботи та галузь її застосування, наведено обґрунтування актуальності теми та уточнюється постановка завдання.

У першому розділі проведено аналіз предметної області, визначено актуальність завдання та призначення розробки, розроблена постановка завдання, задані вимоги до програмної реалізації, технологій та програмних засобів.

У другому розділі виконано аналіз існуючих рішень, обрано вибір платформи для розробки, виконано проектування і розробка програми, наведено опис алгоритму і структури функціонування підсистеми, визначені вхідні і вихідні дані, наведені характеристики складу параметрів технічних засобів, описаний виклик та завантаження застосунку, описана робота програми.

В економічному розділі визначено трудомісткість розробленої інформаційної системи, проведений підрахунок вартості роботи по створенню застосунку та розраховано час на його створення.

Практичне значення полягає в наданні користувачу цієї програми за для розваги у вільний час

Актуальність даної роботи полягає в тому, що надає багато функцій для роботи з графікою, звуком та введенням користувача, що дозволяє створювати різноманітні ігрові проєкти. Багато людей і команд продовжують використовувати Pygame для розробки ігор, особливо для простих 2D-проєктів.

Список ключових слів: Призначення розробки, економічний розділ, практичне значення, ведення користувача, платформа розробки

ABSTRACT

Explanatory note: 64 pp., 21 fig., 2 tables., 20 sources.

Object of development: mobile game.

The purpose of the qualification work: to provide the user an unforgettable experience of the game and spread their skills and reactions.

The introduction considers the analysis and current state of the problem, specifies the purpose of the qualification work and the field of its application, provides a justification for the relevance of the topic and clarifies the problem.

In the first section the analysis of the subject area is carried out, the urgency of the task and purpose of development are defined, the statement of the task is developed, requirements to software realization, technologies and software are set.

The second section analyzes the existing solutions, selects the platform for development, performs design and development of the program, describes the algorithm and structure of the subsystem, determines the input and output data, provides characteristics of the parameters of technical means, describes the call and application load, describes the program.

The economic section determines the complexity of the developed information system, calculates the cost of work to create an application and calculates the time to create it.

The practical significance lies in providing users with entertainment during their free time.

The relevance of this work lies in its ability to provide numerous features for working with graphics, sound, and user input, allowing for the creation of diverse game projects. Many individuals and teams continue to use Pygame for game development, especially for simple 2D projects.

Keywords: Purpose of development, economic section, practical significance, user input, development platform.

ЗМІСТ

СПИСОК УМОВНИХ ПОЗНАЧЕНЬ

RNG - генератор випадкових чисел

ЕОМ - електронно-обчислювальна машина

ІС - інформаційна система

ПЗ - програмне забезпечення

ІТ - інформаційні технології

JSON - формат обміну даними

ВСТУП

Історія комп'ютерних ігор починається з 1940-х років, [1] коли були створені перші експериментальні "ігри на екрані" на основі електронних пристроїв. Протягом наступних десятиліть ігри поступово розвивалися і ставали все більш складними та реалістичними.

Однією з перших відомих комп'ютерних ігор був "Spacewar!"[2], створений у 1962 році на основі PDP-1, одного з перших мінікомп'ютерів. З того часу ігри продовжили свій розвиток, пройшли шлях від простих 2D аркад до складних 3D віртуальних світів.

З появою персональних комп'ютерів у 1980-х роках та популярності домашніх ігрових консолей, комп'ютерні ігри стали доступнішими широкій аудиторії. Такі ігри, як "Tetris", "Super Mario Bros", "The Legend of Zelda" і "Final Fantasy", стали культовими та мають значний вплив на відеоігрову індустрію.

З розвитком графічних технологій, швидкого Інтернету та мобільних пристроїв, комп'ютерні ігри стали більш реалістичними, іммерсивними та соціальними. Виникли онлайн-ігри, мультиплеєрні ігри та ігри в реальному часі, які дозволяють гравцям спілкуватися та співпрацювати віртуально.

Щодо впливу комп'ютерних ігор на людину, дослідження в цій області продовжуються і мають різні результати. Деякі дослідження показують, що комп'ютерні ігри можуть сприяти розвитку когнітивних навичок, таких як увага, концентрація, швидкість реакції та проблемне мислення.

Однак, важливо враховувати, що деякі види комп'ютерних ігор можуть мати потенційно негативний вплив на людину, особливо якщо

використовуються неконтрольовано або велику кількість часу витрачається на гру. Перевитрати часу на гру можуть призвести до занедбання реальних життєвих обов'язків, соціальної ізоляції, проблем з фізичним здоров'ям та психічним станом.

Також було проведено дослідження щодо можливого впливу насильственого вмісту деяких відеоігор на агресивну поведінку. Деякі дослідження показали кореляцію між великою кількістю годин, витрачених на гру з насильственим вмістом, та збільшеним рівнем агресії у деяких осіб. Однак, цей вплив є складним і індивідуальним, і не можна стверджувати, що відеоігри призводять безпосередньо до насильства. [4]

Важливо знати, що вплив комп'ютерних ігор на людину є комплексним і залежить від багатьох факторів, включаючи індивідуальні особливості, контекст використання та тип ігор. Щоб отримати максимальну користь від ігор та мінімізувати можливі негативні наслідки, важливо розуміти свої власні межі, контролювати час гри та розуміти вплив вмісту ігор на свої емоції та поведінку.

Наприкінці, важливо брати до уваги основні принципи здорового способу життя, включаючи регулярну фізичну активність, належний сон, здорове харчування та збалансовану соціальну взаємодію, оскільки ці елементи також мають велике значення для загального благополуччя.та іншими аспектами життя. Важливо мати свідоме ставлення до гри, встановлювати обмеження щодо часу гри, враховувати вікові рекомендації та контролювати вміст ігор, особливо для дітей та підлітків.

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1. Загальні відомості з предметної галузі

Python - це високорівнева, інтерпретована мова програмування загального призначення. Вона була створена Гвідо ван Россумом і вперше була випущена у 1991 році. Python славиться своєю простотою і зручністю в використанні, що робить його популярним в середовищі програмістів, починаючи від початківців до професіоналів.[3]

Основні риси Python:

Синтаксис: Python має чистий і лаконічний синтаксис, що сприяє зрозумілості коду. Використання пробілів для відступів (індентування) має велике значення в Python і допомагає підтримувати зрозумілість.

Інтерпретованість: Python - інтерпретована мова програмування, що означає, що код виконується рядок за рядком. Це забезпечує швидку зміну інструкцій та легкість відладки.

Об'єктно-орієнтоване програмування: Python підтримує об'єктно-орієнтований підхід, що дозволяє створювати класи, об'єкти, методи і спадкування. Це сприяє модульності, повторному використанню коду і структуруванню програм.

Багаті бібліотеки: Python має велику кількість стандартних бібліотек, які включають різні функції і модулі для різних потреб, таких як робота з рядками, мережами, базами даних, графікою і багато іншого. Також існує велика кількість сторонніх бібліотек, розроблених спільнотою, що розширюють можливості мови.

Переносимість: Python може працювати на різних платформах, таких як Windows, macOS, Linux і інші.

1.2. Призначення розробки та область застосування

Розробка ігор на мові програмування Python може мати різноманітні призначення та області застосування. Ось кілька прикладів:

Навчальні цілі: Графічне програмування на Python може бути використане для створення навчальних ігор, які допомагають усвідомити концепції програмування, алгоритмів та логічного мислення. Такі ігри можуть бути корисними для початківців, які вчаться програмуванню, та допомогти їм зрозуміти основи кодування.

Розваги та релаксація: Python може використовуватися для створення простих графічних ігор, які призначені для розваги та релаксації. Це можуть бути аркади, головоломки, карточні або настільні ігри, які допомагають користувачам провести час за забавою.

Симуляції та моделювання: Python є популярною мовою для створення симуляцій та моделювання реальних процесів. Ігри на Python можуть бути використані для створення віртуальних середовищ, де користувачі можуть взаємодіяти з моделями фізичних, соціальних або економічних процесів.

Навчання та тренування: Python-ігри можуть використовуватися для навчання та тренування у різних областях. Наприклад, це можуть бути симулятори польотів для пілотів, тренажери для навчання мови або ігри, які допомагають удосконалювати навички розв'язування проблем або управління проектами.

Машинне навчання та штучний інтелект: Python є однією з найпопулярніших мов програмування для розробки алгоритмів машинного навчання та штучного інтелекту. Гри на Python можуть бути використані для створення середовищ та завдань, які допомагають навчити та натренувати моделі машинного навчання або алгоритми штучного інтелекту. Наприклад, це можуть бути ігри, де модель агента навчається пристосовуватись до змінюваних умов, виробляти оптимальні стратегії або взаємодіяти з іншими агентами.

Візуалізація даних: Python зазвичай використовується для аналізу даних та візуалізації результатів. Графічні ігри можуть бути використані для візуалізації складних даних або статистичних моделей, що дозволяє користувачам краще розуміти та взаємодіяти зі зібраними даними.

Загалом, розробка ігор на Python має широкий спектр застосувань, починаючи від навчальних цілей та розваг, до симуляцій, навчання та досліджень у різних областях. Python надає зручний та гнучкий інструментарій для реалізації ігрових проектів з різними цілями та функціональністю.

1.3. Підстава для розробки

Відповідно до освітньої програми, згідно навчального плану та графіків навчального процесу, в кінці навчання студент виконує кваліфікаційну роботу. Тема роботи узгоджується з керівником проекту, випускаючою кафедрою, та затверджується наказом ректора.

Отже, підставами для розробки (виконання кваліфікаційної роботи) є:

- освітня програма спеціальності 122 «Комп’ютерні науки»;
- навчальний план та графік навчального процесу;
- наказ ректора Національного технічного університету «Дніпровська політехніка» № _____ від __.__.2023 р;

- завдання на кваліфікаційну роботу на тему: «*Розробка гри на мові програмування Python з використанням бібліотеки Pygame*».

1.4. Постановка завдання

Назва проекту: Розробка гри на мові програмування Python з використанням бібліотеки Pygame.

Мета проекту: Наша мета - розробити просту гру за допомогою мови програмування Python і бібліотеки Pygame. Гра буде включати гравця, який керує об'єктом на екрані та намагається досягти мети, уникаючи перешкод.

Опис проекту: Встановлення Pygame: Перш за все, необхідно встановити бібліотеку Pygame. Це можна зробити за допомогою менеджера пакетів pip, виконавши команду `pip install pygame` у терміналі.

Ініціалізація гри: Створимо новий файл Python та імпортуємо бібліотеку Pygame. Задамо основні параметри гри, такі як розмір вікна, колір фону та швидкість оновлення.

Створення об'єктів: Створимо об'єкти, які будуть присутні на екрані гри, такі як гравець, перешкоди та бонуси. Кожен об'єкт буде мати свої властивості, такі як розмір, швидкість та позиція.

Управління гравцем: Налаштуємо управління гравцем за допомогою клавіш на клавіатурі або інших джерел вводу. Визначимо рух гравця та реакцію на дії гравця.

Логіка гри: Розробимо логіку гри, таку як визначення зіткнень між об'єктами, рахунок очків та стан гри (початок, пауза, кінець). Реалізуємо механіку взаємодії між гравцем, перешкодами та бонусами.

Рендеринг графіки (продовження): Застосуємо функції Pygame для відображення графічних елементів на екрані. Створимо цикл, який оновлює та відображає стан гри на кожній ітерації. Налаштуємо відображення графічних об'єктів, їх позицію та анімацію.

Обробка взаємодії: Реалізуємо обробку взаємодії між гравцем та іншими об'єктами. Наприклад, при зіткненні з перешкодою гра може завершитись, а при збиранні бонусу гравець може отримати додаткові очки.

Звуковий супровід: Додамо звуковий супровід до гри, включаючи звукові ефекти при взаємодії з об'єктами, фонову музику та звукові сигнали.

Меню та інтерфейс користувача: Розробимо меню гри, в якому користувач зможе почати гру, переглянути правила або вийти з гри. Застосуємо елементи інтерфейсу користувача для відображення інформації, такої як рахунок очків, стан гравця та повідомлення.

Тестування та налагодження: Після розробки гри проведемо тестування для виявлення помилок та недоліків. Виправимо виявлені проблеми та зробимо необхідні виправлення.

Оптимізація та удосконалення: Підвищимо продуктивність гри та удосконалимо її ефективність. Можна оптимізувати код, покращити алгоритми, додати нові функції або рівні гри, щоб зробити її більш захоплюючою.

Документація (продовження): пояснює основні функціональні можливості гри, правила гри, використовувані клавіші управління та інші важливі аспекти. Документація допоможе іншим розробникам або користувачам зрозуміти, як користуватись та розширювати гру.

Пакування та розповсюдження: Збережемо гру в окрему виконувану програму або пакет, щоб її можна було легко встановити та виконати на інших комп'ютерах без необхідності встановлення Python та Pygame. Підготуємо інструкції з встановлення та використання гри для користувачів.

Дострокова оцінка та зворотній зв'язок: Передбачимо можливість здати гру на оцінку іншим розробникам або геймерам для отримання відгуків та пропозицій щодо поліпшення. Це можна зробити, наприклад, публікуючи гру на форумах чи відкритих ресурсах для розробників.

Підтримка та розвиток: Після випуску гри забезпечимо підтримку для користувачів, відповідаючи на їх запити, виправляючи помилки та видаючи оновлення з новими функціями або виправленнями. Поступово розвиватимемо гру, додаючи нові рівні, функції або можливості, щоб збільшити інтерес користувачів та підтримувати активність гравців.

Це загальна постановка розробки гри на мові програмування Python з використанням бібліотеки Pygame. Звичайно, деталі та кроки можуть

різнитися залежно від конкретних вимог та складності гри, яку ми плануємо розробити.

1.5. Вимоги до програми або програмного виробу

1.5.1. Вимоги до функціональних характеристик

Вимоги до функціональних характеристик гри на мові програмування Python з використанням бібліотеки Pygame можуть включати наступні пункти:

Головне меню: Розробка головного меню гри, де користувач може почати нову гру, переглянути правила, налаштування або вийти з гри.

Рух гравця: Реалізація руху гравця по екрану за допомогою клавіш на клавіатурі або іншого введення.

Графічний інтерфейс: Створення візуальних елементів, таких як фонове зображення, спрайти для гравця, перешкод. Візуалізація різних станів гри, таких як початок, пауза, кінець гри.

Генерація перешкод: Реалізація механізму генерації перешкод на екрані, які гравець повинен уникати. Перешкоди можуть бути статичними або рухомими, з різною складністю та розмірами.

Рахунок очків: Ведення рахунку очків гравця в залежності від досягнень, таких як пройдені рівні, зібрані бонуси або виживаний час.

Звуковий супровід: Додавання звукових ефектів, фонової музики та звукових сигналів, які відтворюються відповідно до дій гравця та подій в грі.

Завершення гри: Реалізація умови для завершення гри, наприклад, якщо гравець зіткнеться з перешкодою або досягне кінцевої мети.

1.5.2 Вимоги до інформаційної безпеки

Вимоги до інформаційної безпеки гри на мові програмування Python з використанням бібліотеки Pygame можуть включати наступні аспекти:

Захист від вразливостей програмного забезпечення: Переконатися, що код написаний безпечно та захищений від типових вразливостей, таких як введення через клавіатуру, SQL-ін'єкції, переповнення буфера тощо. Використовувати надійні методи та бібліотеки для обробки даних та взаємодії з користувачем.

Шифрування: Застосовувати шифрування для захисту конфіденційної інформації, такої як паролі, особисті дані користувачів тощо. Використовуйте надійні алгоритми шифрування та зберігайте ключі безпечно.

Аутентифікація та авторизація: Реалізувати механізми аутентифікації користувачів, щоб перевірити їхню ідентичність перед наданням доступу до обмеженого функціоналу гри. Також використовувати механізми авторизації, щоб обмежити доступ до ресурсів та функцій залежно від ролі користувача.

Захист від шахрайства та використання читів: Розробити заходи для запобігання шахрайству та використанню читів в грі. Це може включати виявлення та блокування недозволених програм або алгоритмів, контроль цілісності даних та анти-чити системи.

Захист від зловмисницьких атак: Приділіть увагу захисту від зловмисницьких атак, таких як внедрення шкідливого коду, вимагання викупу, використання слабкостей мережевого стеку тощо.

1.5.3 Вимоги до складу та параметрів технічних засобів

Вимоги до складу та параметрів технічних засобів для гри можуть включати наступні аспекти:

Операційна система: Гра повинна підтримувати певну операційну систему, наприклад, Windows, macOS, Linux або іншу. Вимоги до версії операційної системи повинні бути чітко визначені.

Процесор: Вказати мінімальні та рекомендовані характеристики процесора, такі як тип (наприклад, Intel Core i5 або AMD Ryzen 5), тактова частота і кількість ядер. Це допоможе забезпечити плавну роботу гри на різних системах.

Оперативна пам'ять: Вказати мінімальний та рекомендований обсяг оперативної пам'яті, наприклад, 4 ГБ або 8 ГБ. Більша кількість оперативної пам'яті дозволяє запускати гру з вищою продуктивністю і забезпечує більш плавний геймплей.

Відеокарта: Вказати мінімальні та рекомендовані характеристики відеокарти, наприклад, модель (наприклад, NVIDIA GeForce GTX 1060 або AMD Radeon RX 580), обсяг відеопам'яті і підтримку графічних API (наприклад, DirectX 11 або OpenGL 4.5).

Місце на диску: Вказати необхідний обсяг вільного місця на жорсткому диску для встановлення гри. Це може бути визначено в гігабайтах (ГБ) або терабайтах (ТБ), залежно від розміру гри і її компонентів.

1.5.4 Вимоги до інформаційної та програмної сумісності

Вимоги до інформаційної та програмної сумісності гри включають:

Процесор: Мінімум 1 ГГц одно- або багатоядерний процесор.

Оперативна пам'ять: Мінімум 1 ГБ оперативної пам'яті.

Відеокарта: Підтримка OpenGL ES 2.0 або вище.

Місце на диску: Зазвичай вимагається приблизно 30-50 МБ вільного місця на диску для інсталяції гри. Однак, остаточний розмір гри може залежати від конкретної реалізації.

Операційна система: Android версії 4.1 (Jelly Bean) або вище.

Дисплей: Рекомендована роздільна здатність екрану 480x800 пікселів або вище.

Акселерометр: Деякі версії гри можуть використовувати акселерометр для керування, тому пристрій має мати акселерометр.

Рекомендовані вимоги: Додатково вказати рекомендовані характеристики апаратного забезпечення, які забезпечать кращу продуктивність та якість геймплею. Це може бути вища кількість процесорних ядер, більший обсяг оперативної пам'яті, потужніша відеокарта тощо. [4]

РОЗДІЛ 2
ПРОЕКТУВАННЯ ТА РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ

2.1. Функціональне призначення системи

Функціональне призначення включає такі елементи:

Система здатна відтворювати гру на мобільному пристрої.

Система забезпечує відображення графічних елементів гри на екрані пристрою.

Система дозволяє користувачеві взаємодіяти з грою за допомогою дотиків та кнопок на мобільному пристрої.

Система реалізує правила та алгоритми гри, включаючи обробку рухів гравців, взаємодію об'єктів, рівні складності та інші аспекти геймплею.

Система забезпечує відтворення звукових ефектів та музики в грі.

Відображення меню гри, включаючи кнопку старту та кнопку таблиці рекордів.

Реалізація паузи та відповідного відображення на екрані.

2.2. Опис застосованих математичних методів

У підрозділі «Опис застосованих математичних методів» повинні бути наведені всі використані математичні методи, приклади та умови їх застосування в розробленій системі. Якщо таких методів не використовується, то ця інформація повинна бути вказана в тексті цього підрозділу.

2.3. Опис використаних технологій та мов програмування

Код написаний мовою програмування Python. Python - це інтерпретована мова програмування загального призначення, яка просунута, проста для вивчення та має велику активну спільноту розробників. Python добре

підходить для розробки ігор завдяки своїй лаконічній синтаксису та великому набору бібліотек. [5]

Pygame - це бібліотека, яка базується на бібліотеці PySDL2 і надає можливості для розробки ігор та графічних програм з використанням Python. Pygame має різноманітні функції, які допомагають у керуванні графікою, аудіо, музикою та обробці подій. У цьому коді Pygame використовується для створення графічного ігрового інтерфейсу, обробки подій клавіатури та миші, малювання графіки, відтворення звуків та музики. [6]

JSON (JavaScript Object Notation) - це легкий формат обміну даними, який широко використовується для збереження та передачі структурованих даних. У цьому коді бібліотека JSON використовується для збереження даних таблиці рекордів у файлі "high_scores.json". Вона дозволяє зчитувати та записувати дані у форматі JSON, що робить його зручним для збереження та обміну даними у програмах. [7]

Для графіки гри використовуються зображення, які зберігаються у відповідних файлах. Ці зображення використовуються для представлення об'єктів гри, таких як НЛО та астероїди. Зображення завантажуються за допомогою Pygame та відображаються на екрані гри.

Звукові файли: Для фонової музики використовуються звукові файли. Pygame надає можливості відтворення звуків та музики у форматах, які підтримуються бібліотекою, таких як WAV або MP3. У цьому коді використовується звуковий файл для відтворення фонової музики під час гри. [8]

В цьому коді використовуються мова програмування Python, бібліотека Pygame для розробки ігор, бібліотека JSON для збереження даних таблиці

рекордів, зображення для графіки гри та звукові файли для фонової музики. Використання цих технологій дозволяє розробникам створювати ігри з графічним інтерфейсом та звуковими ефектами у Python.

2.4. Опис структури системи та алгоритмів її функціонування

Імпорт бібліотек та необхідних модулів:

1. pygame: для реалізації ігрового інтерфейсу.
2. random: для генерації випадкових значень.
3. json: для роботи з файлами формату JSON.
4. sys: для виходу з програми. [9]

Ініціалізація Pygame та створення вікна гри.

Завантаження зображень та звукових файлів, ініціалізація глобальних змінних.

Оголошення функцій:

1. draw_score(): відображення поточного рахунку гри на екрані.
2. generate_asteroid(): генерація нового астероїда.
3. check_collision(): перевірка на зіткнення з астероїдами або краєм екрану.
4. load_high_scores(): завантаження таблиці рекордів з файлу.
5. save_high_scores(): збереження таблиці рекордів у файл.
6. draw_high_scores(): відображення таблиці рекордів на екрані.
7. draw_menu(): відображення головного меню гри.
8. Ініціалізація змінних, необхідних для гри.

Запуск головного циклу гри:

1. Обробка подій (натискання клавіш, клацання миші, тощо).

2. Оновлення екрану.

3. Обробка руху НЛО та астероїдів.

4. Перевірка на зіткнення та збільшення рахунку.

5. Перевірка умов завершення гри (зіткнення або виходу за межі екрану).

6. Збереження рекордів, якщо рахунок є одним із найкращих.

7. Збільшення швидкості астероїдів залежно від рахунку.

8. Перевірка на паузу та відображення відповідного тексту.

Це загальний опис структури та алгоритмів функціонування гри "Lost in Space". Код здійснює обробку подій, оновлює стан гри та відображає його на екрані. Крім того, він здатний завантажувати та зберігати рекорди гравців у файлі формату JSON та відображати таблицю рекордів на екрані.

2.5. Обгрунтування та організація вхідних та вихідних даних програми

Дана програма є простою грою, реалізованою з використанням бібліотеки Pygame. Вона має наступну організацію вхідних та вихідних даних.

Вхідні дані:

1. Зображення, які завантажуються з файлів: "Images/Космос.png", "Images/нло.png" і "Images/астероиды.png".

2. Звуковий файл музики, який завантажується з файлу "Sound/music.mp3".

Вихідні дані:

1. Вікно гри, розміром 500x750 пікселів.
2. Зображення, яке відображається на фоні екрана.
3. Зображення НЛО та астероїдів, які відображаються на екрані гри.
4. Текст, який відображає поточний рахунок гравця.
5. Текст, який відображає час, проведений гравцем у грі.
6. Текст, який відображає заголовок головного меню гри.
7. Текст, який відображає повідомлення для початку гри або перезапуску.
8. Текст, який відображає найкращі рахунки гравців у таблиці рекордів.
9. Текст, який відображає інструкцію для введення імені гравця.
10. Текст, який відображає введене ім'я гравця.
11. Текст, який відображає кнопку "Back" у таблиці рекордів.

Загальний виконавчий код містить цикл, який оновлює екран, обробляє різні події та відображає відповідні зображення та тексти на екрані, залежно від стану гри та вхідних подій.

Програма також має функції для завантаження та збереження таблиці рекордів гравців у файл "high_scores.json". Функції load_high_scores() та save_high_scores() відповідають за цю операцію.

2.6. Опис роботи розробленої системи

2.6.1. Використані технічні засоби

Даний код написаний за допомогою середовища розробки PyCharm (Рис 2.1), яке є популярним інтегрованим середовищем розробки для мови програмування Python. [10] PyCharm надає розширену підтримку Python, включаючи автодоповнення коду, перевірку синтаксису, візуальне відлагодження та багато іншого. [11]

Рис.2.1. Середовище розробки PyCharm

PyCharm надає зручний та інтуїтивно зрозумілий інтерфейс, що сприяє комфортній роботі з кодом. Він має багато корисних функцій, таких як автодоповнення, підказки, перевірка синтаксису та багато іншого, що полегшує процес розробки. [12]

Також для розробки гри використовуються такі пристрої:

Комп'ютер. Для виконання коду Python і запуску гри використовується комп'ютер. Комп'ютер може бути заснований на будь-якому сучасному апаратному забезпеченні, що підтримує встановлене середовище Python та бібліотеку Pygame.

Монітор. Візуальний вивід гри та інтерфейс користувача здійснюється за допомогою монітора. Монітор відображає графіку, зображення фону, об'єкти гри та текстові повідомлення.

Телефон. Взаємодія користувача з грою здійснюється за допомогою клавіатури. Клавіатура дозволяє користувачу керувати рухом НЛО, починати гру, ввести своє ім'я для таблиці рекордів та виконувати додаткові дії, такі як пауза в грі.

Акустична система. Система має звукову складову, яка відтворює фонову музику під час гри. Для цього використовується акустична система, що підключена до телефона

Файли. Система використовує зовнішні файли для збереження даних. Зокрема, використовується файл "high_scores.json" для збереження таблиці рекордів.

Графічні зображення. Система використовує растрові зображення для відображення графіки гри та інтерфейсу користувача. В коді згадуються зображення фону, НЛО та астероїдів.

Звукові файли. Для відтворення фонової музики у грі використовується звуковий файл "music.mp3".

Зазначений код використовує вищезгадані технічні засоби та пристрої для створення та відтворення гри з використанням бібліотеки Pygame у середовищі Python.

2.6.2. Використані програмні засоби

Функціонування розробленої системи забезпечується за допомогою операційної системи та програмних засобів. Операційна система є основною системною програмою, яка керує роботою комп'ютера і надає інтерфейс для взаємодії між апаратним і програмним забезпеченням. У даному випадку, система використовує операційну систему, таку як Windows (Рис. 2.2) та за для отримання apk-файлу дистрибутив Linux - Ubuntu. (Рис. 2.3)

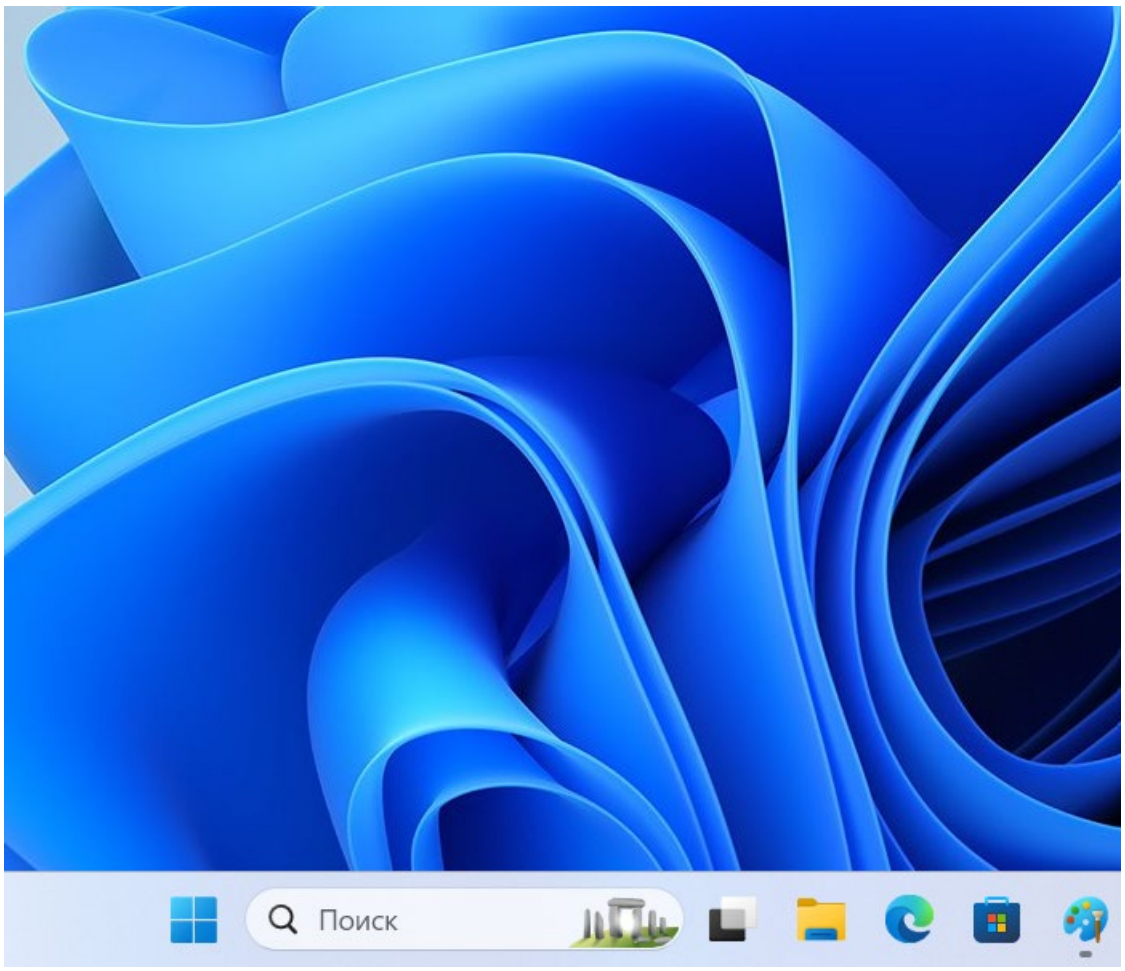

Рис. 2.2. Використовування ОС Windows 11

Рис 2.3. Дистрибутив Linux Ubuntu

Windows 11 - це остання версія операційної системи Windows, розроблена компанією Microsoft. Вона є наступником популярної Windows

10 і пропонує ряд нових функцій, вдосконалень і змін в інтерфейсі користувача. [13]

Має оновлений дизайн, включаючи нове центральне меню Пуск, перероблену панель завдань і змінений вигляд вікон. Вона надає більші можливості для персоналізації, зокрема, можливість розміщувати віджети на робочому столі та використовувати нові теми.

Також пропонує покращені можливості для гри, зокрема, підтримку технології DirectStorage для швидшого завантаження ігор, покращену підтримку контролерів Xbox та нові функції, спрямовані на поліпшення продуктивності та використання мультизадачності.

Окрім того, Windows 11 впроваджує покращену підтримку для роботи зі зовнішніми пристроями, такими як сенсорні екрани, планшети та 2-в-1 пристрої, і має покращену підтримку пензлів і стилусів. [14]

Загалом, Windows 11 є новою версією операційної системи Windows, яка пропонує оновлений інтерфейс, нові функції та покращену продуктивність для користувачів комп'ютерів та інших пристроїв.

Ubuntu - це операційна система, яка належить до сімейства Linux. Вона є однією з найпопулярніших дистрибутивів Linux і відома своєю простотою в використанні та доступністю для широкого кола користувачів. [15]

Ubuntu розробляється та підтримується спільнотою розробників, і вона має відкритий вихідний код. Основні цінності Ubuntu включають безкоштовність, відкритість, безпеку та спрощений інтерфейс користувача.

Операційна система Ubuntu надає повноцінне середовище для виконання різних завдань, включаючи веб-перегляд, редагування документів, розробку програмного забезпечення, налагодження системи та багато іншого. Вона має широкий вибір програм та пакетів, доступних для встановлення з використанням системи керування пакетами apt. [16]

Ubuntu також відома своєю регулярною випуском нових версій з оновленнями та вдосконаленнями. Крім основного випуску, що призначений для настільних комп'ютерів та ноутбуків, існують інші варіанти Ubuntu, такі як Ubuntu Server для серверних систем, Ubuntu Core для вбудованих пристроїв та Ubuntu Touch для мобільних пристроїв. [17]

Узагальнюючи, Ubuntu - це операційна система заснована на ядрі Linux, яка пропонує безкоштовну, відкриту та легку у використанні платформу для роботи з комп'ютером та за для якої ми можемо зробити apk-файл нашого проекту [18]

2.6.3. Виклик та завантаження програми

Цей код використовує бібліотеку Pygame для створення простої гри "Lost in Space". Гра має на меті уникати зіткнень з астероїдами, переміщуючи НЛО вгору та вниз за допомогою пробілу. Гра завершується, якщо НЛО зіткнеться з астероїдом або виходить за межі екрану.

Для запуску цієї програми потрібно мати встановлену бібліотеку Pygame та Python. Для запуску програми необхідно мати всі зображення та звуки, які використовуються у коді, розташовані у відповідних директоріях (Images та Sound).

Розмір програми залежить від розмірів зображень, які використовуються, та кількості астероїдів, які генеруються. Також, розмір програми може збільшуватись залежно від кількості рекордів, які зберігаються у файлі "high_scores.json".

Умови завантаження та використання програми повинні відповідати ліцензійним умовам, які вказані у початку коду. Крім того, вам можуть знадобитись додаткові дозволи або права для використання зображень та звуків, які використовуються у грі.

Оперативна пам'ять, необхідна для виконання програми, залежить від розміру зображень та обсягу даних, які зберігаються у пам'яті для відстеження позицій об'єктів у грі. Зазвичай, така проста гра не вимагає великої кількості оперативної пам'яті для виконання.

Перед використанням цього коду, впевніться, що ви виконуєте всі необхідні вимоги та умови для використання бібліотеки Pygame, а також щодо використання зображень та звуків, які використовуються у грі.

Також за для отримання apk-файлу, щоб нашу гру можна встановити на телефон використаємо у нашому терміналі інструмент Buildozer.

Buildozer - це інструмент, призначений для автоматизації процесу збірки, пакування і розгортання мобільних додатків. Він широко використовується у розробці мобільних додатків на платформах Android і iOS. [19]

Buildozer надає зручний спосіб сконфігурувати параметри збірки додатку, включаючи залежності, налаштування компіляції, включення різних ресурсів та багато іншого. Він автоматично вирішує багато проблем,

пов'язаних зі збіркою мобільних додатків, і дозволяє розробникам швидше і простіше розгортати свої додатки на різних платформах.

Buildozer використовується в комбінації з іншими інструментами розробки, такими як Python, Kivy, Plyer та інші, і допомагає розробникам зосередитися на самому процесі розробки додатку, забезпечуючи автоматизацію процесу збірки та розгортання.

Щоб викликати програму за допомогою buildozer та створити APK-файл, нам знадобиться налаштувати свій проект та виконати декілька кроків.

Переконаймося, що у нас встановлені Python та Buildozer. Встановлення Buildozer можна здійснити за допомогою команди pip install buildozer. (Рис 2.4)

Рис. 2.4. Встановлення Buildozer у терміналі

Перейдемо до кореневої папки нашого проекту, де знаходиться файл buildozer.spec. (Рис 2.5)

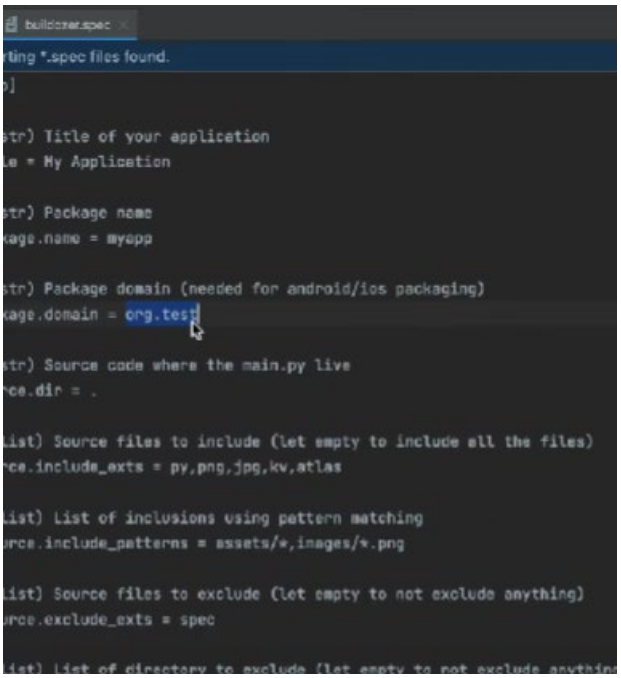

Рис. 2.5. Корнева папка buildozer.spec

Відредагуємо файл buildozer.spec, щоб відповідати налаштуванням вашого проекту. Особливо зверніть увагу на параметри, такі як title, package.name, package.domain, source.dir, source.include_exts та інші. Встановімо requirements для включення необхідних пакетів Pygame та інших залежностей вашого проекту.

Відкриємо командний рядок або термінал та перейдемо до кореневої папки нашого проекту. (Рис 2.6)

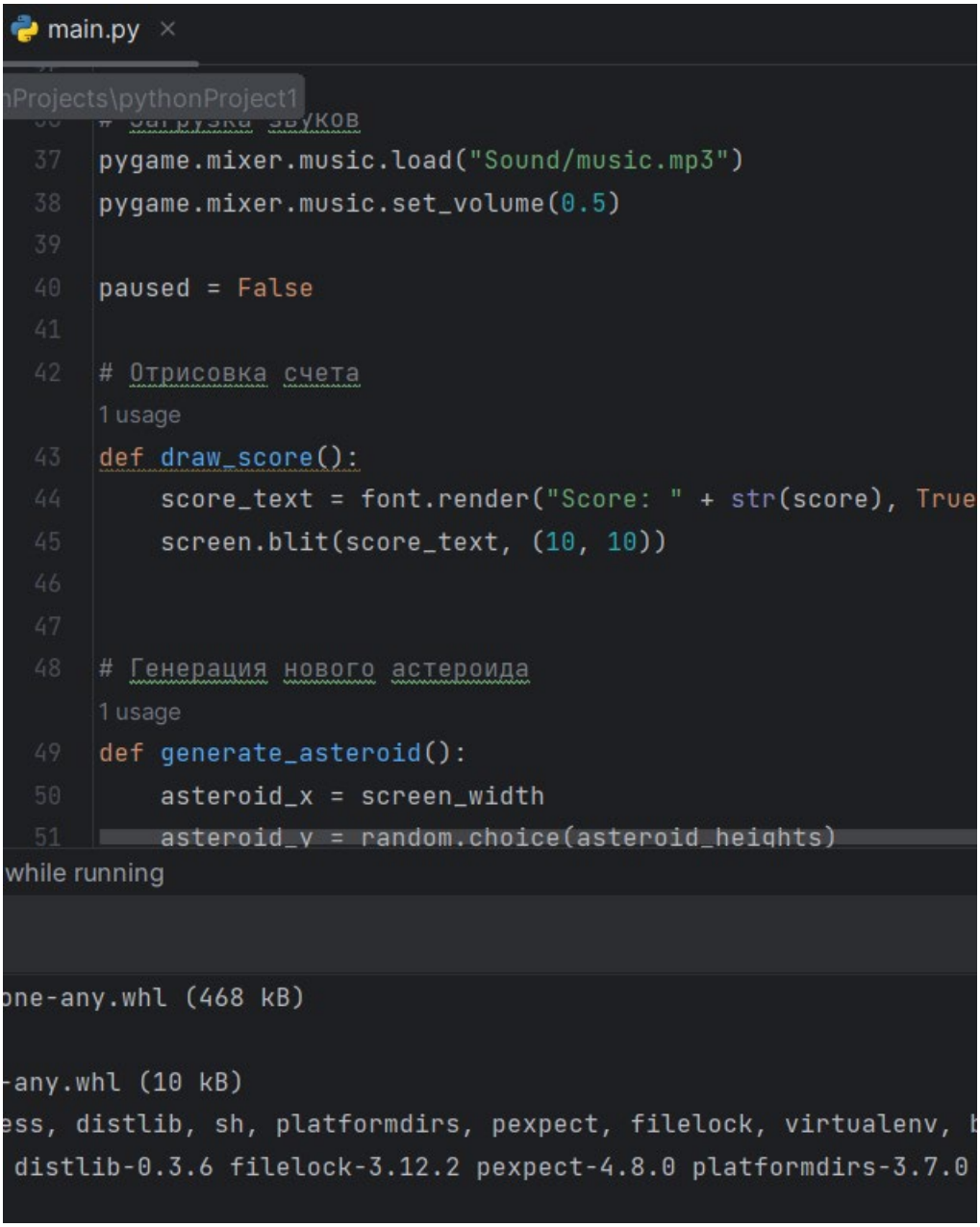

Рис 2.6. Корнева папка нашого проекту та відкритий термінал

Виконаємо наступну команду (Рис. 2.7), щоб збудувати APK-файл.

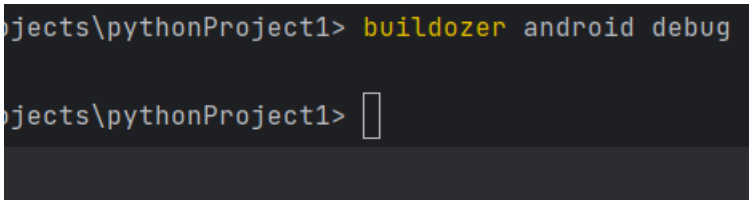

Рис. 2.7. Запуск команди buildozer

Почекаємо, поки buildozer завершить процес збирання. (Рис.2.8) Це може зайняти кілька хвилин, залежно від розміру проекту та налаштувань.

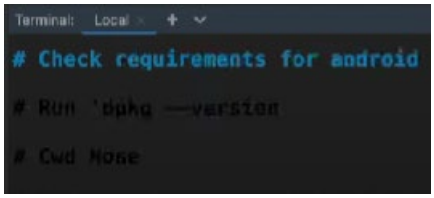

Рис. 2.8. - процесс збирання у APK-файл

Після завершення ми знайдемо згенерований APK-файл у папці bin в кореневій папці нашого проекту. (Рис. 2.9)

Рис 2.9. згенерований apk-файл

Скопіюємо APK-файл (Рис 2.10) на свій пристрій Android та встановимо його. (Рис 2.11)

Рис 2.10. apk-файл

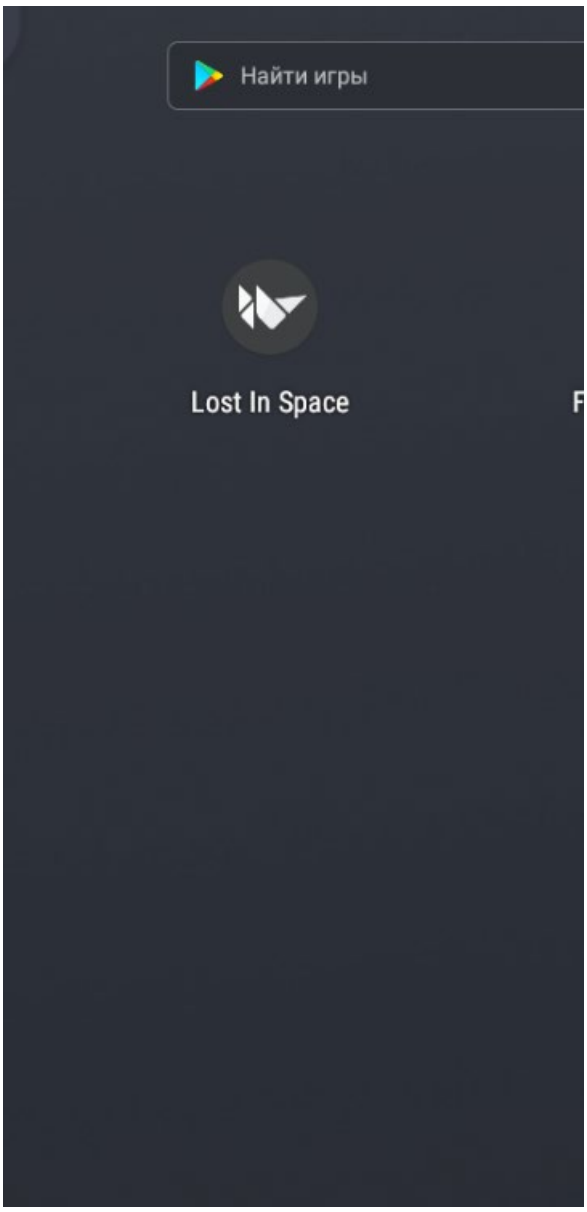

Рис. 2.12. Встановлена гра на android

2.6.4. Опис інтерфейсу користувача

Вікно гри: Вікно має розміри 500 пікселів у ширину та 750 пікселів у висоту. Воно використовується для відображення графічних елементів гри. (Рис. 2.13)

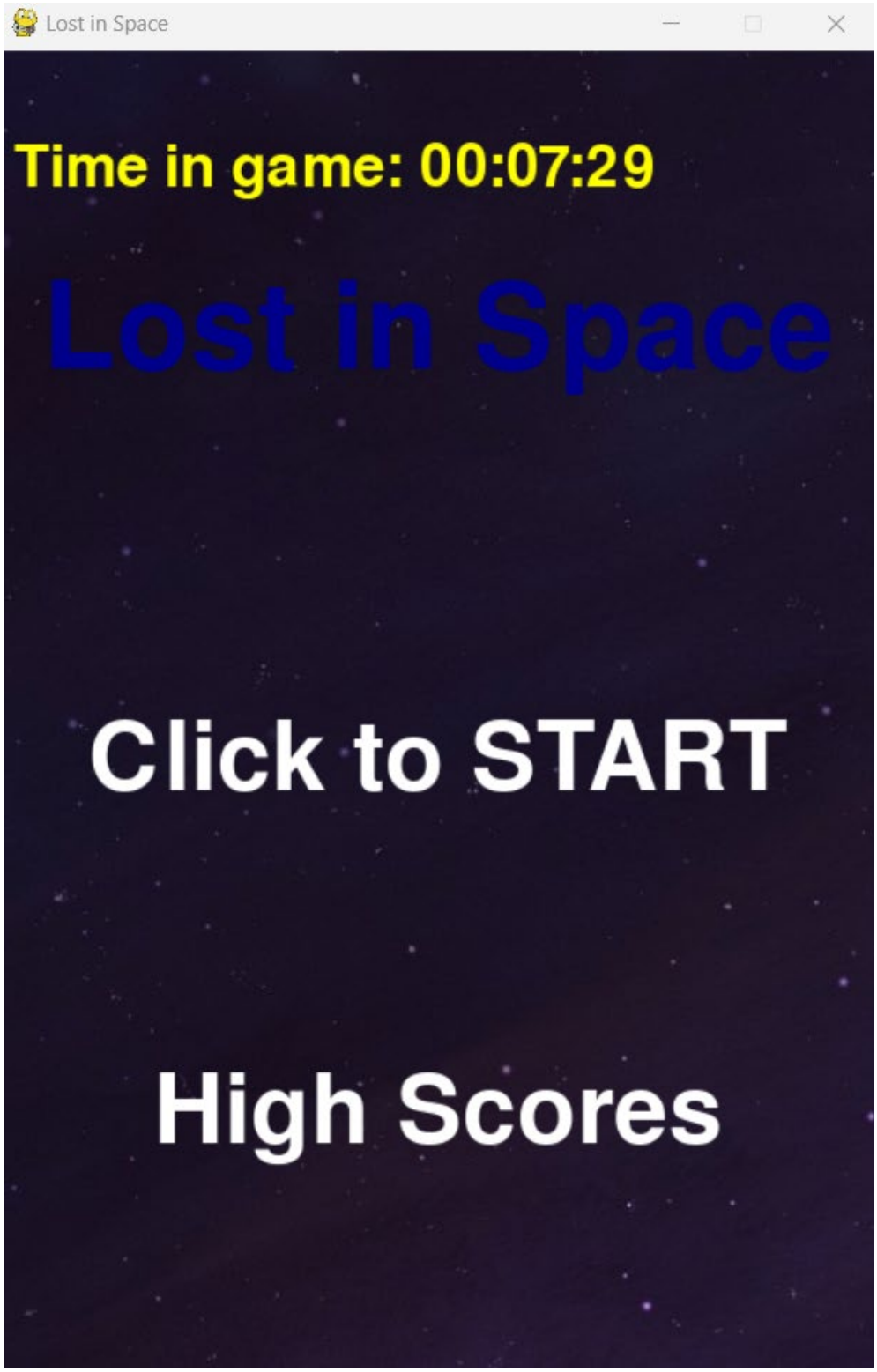

Рис 2.13. Вікно гри

Зображення фону: Завантажується зображення космосу, яке використовується як фонове зображення гри. (Рис. 2.14)

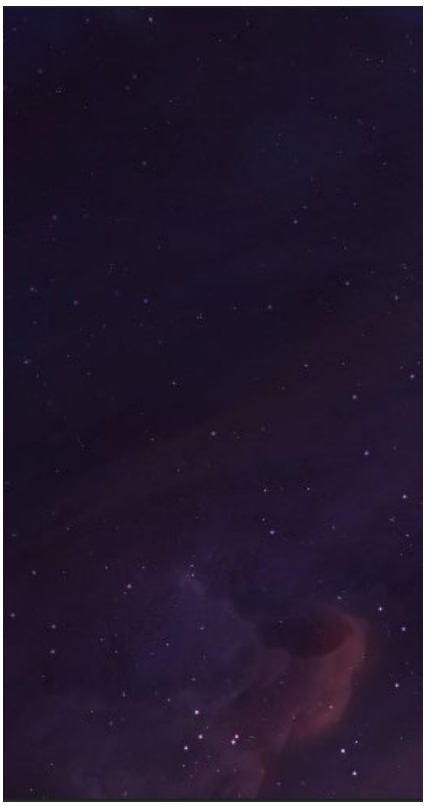

Рис 2.14. Фон гри

Зображення НЛО: Завантажується зображення НЛО, яке представляє головний графічний об'єкт гравця. (Рис. 2.15)

Рис 2.15. НЛО

Зображення астероїдів: Завантажується зображення астероїдів, яке використовується для створення астероїдів у грі. (Рис. 2.16)

Рис. 2.16. Астероїд

Текстові повідомлення: Використовуються тексти, щоб відображати різну інформацію для гравця. Відображення рахунку гравця, часу гри. (Рис. 2.17)

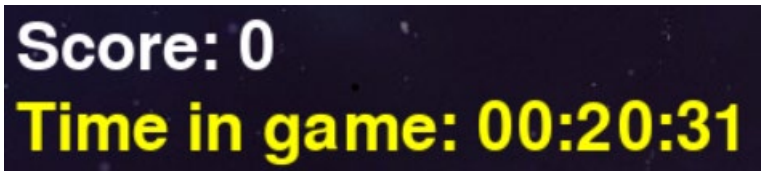

Рис. 2.17. Повідомлення під час гри

Меню: Гра має початкове меню з назвою гри, кнопкою "START" та кнопкою "High Scores". Гравець може почати гру, натиснувши кнопку "START", або переглянути таблицю рекордів, натиснувши кнопку "High Scores". (Рис. 2.18)

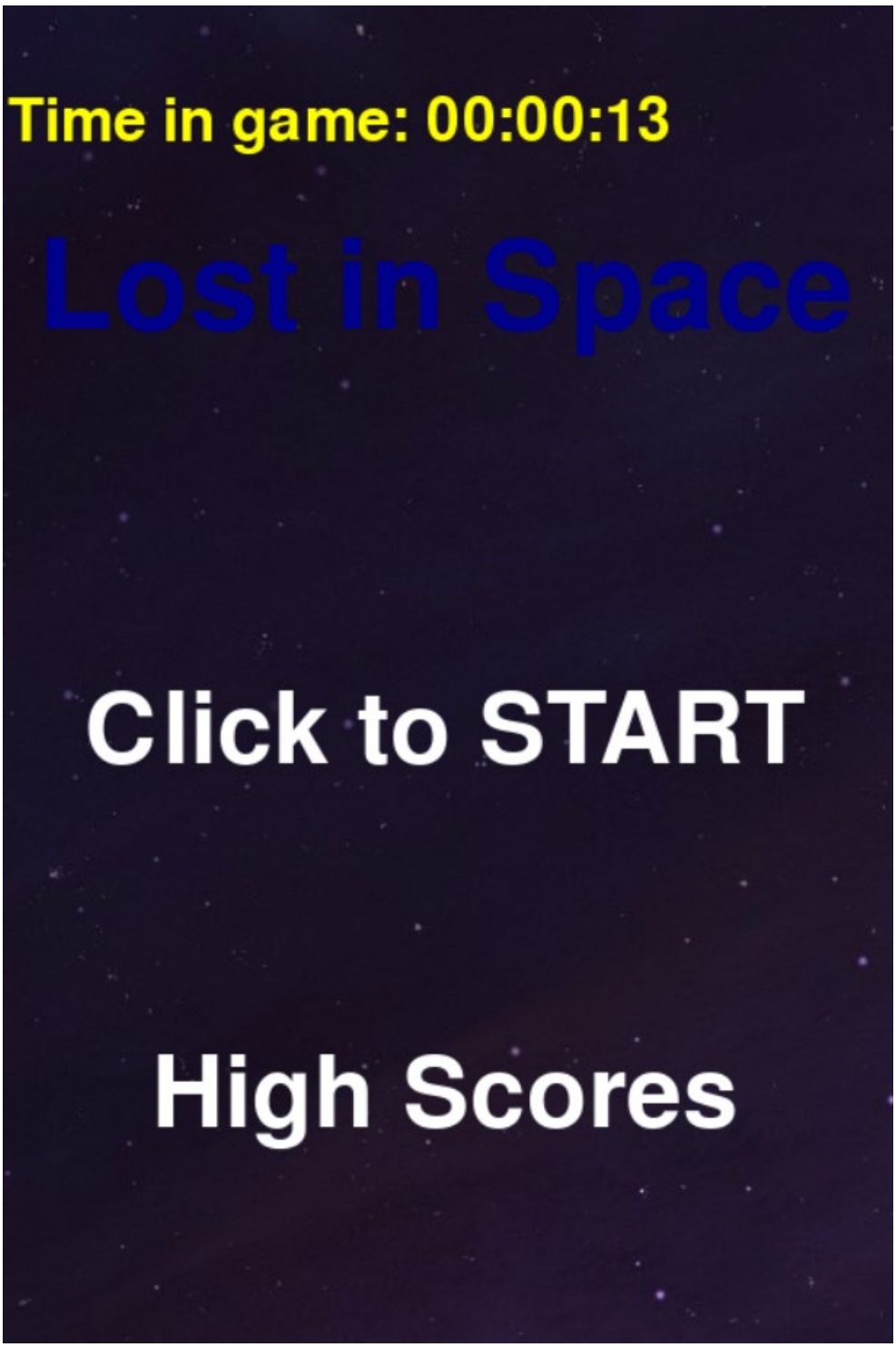

Рис 2.18. - Меню гри

Таблиця рекордів: Гра має таблицю рекордів (Рис. 2.19), яка відображає найкращі результати гравців. Гравець може переглянути цю таблицю, натиснувши кнопку "High Scores" у меню.

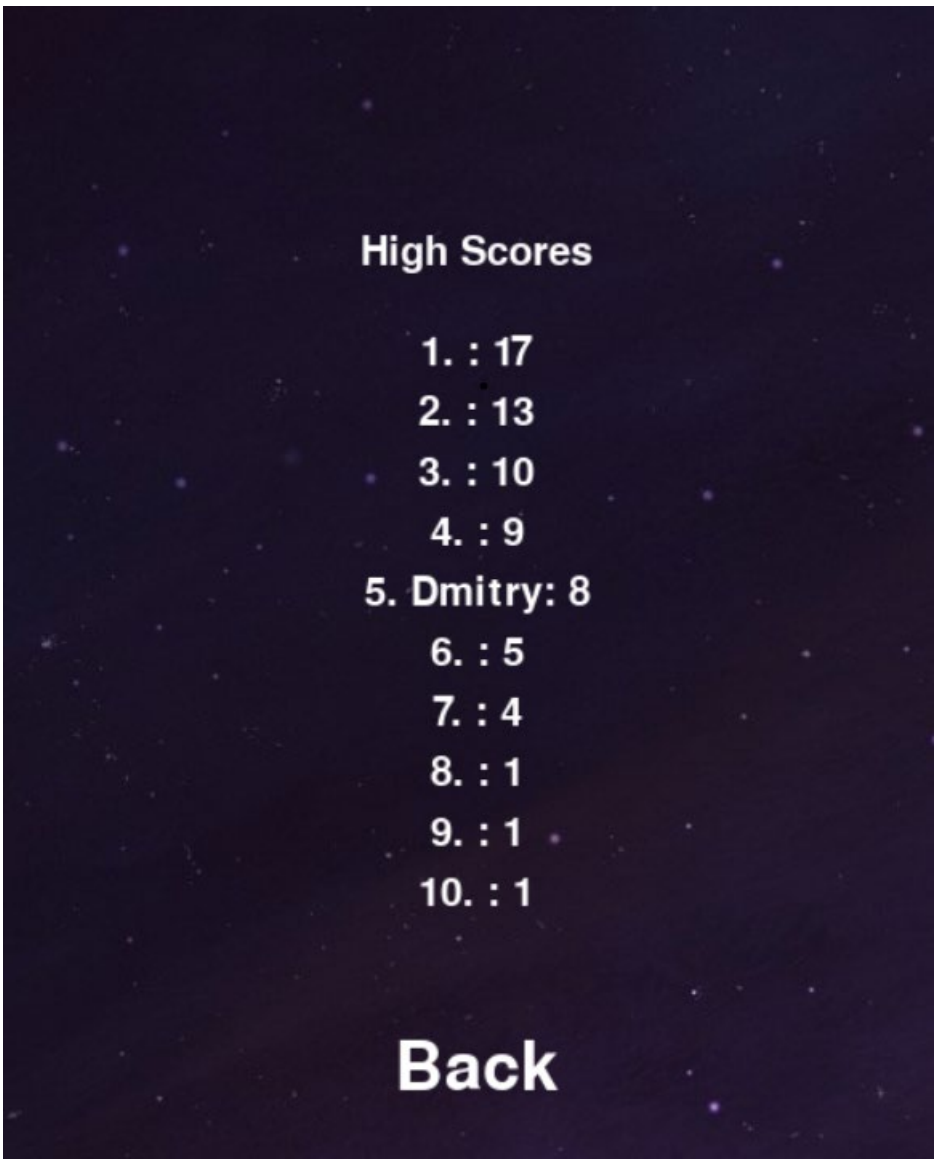

Рис 2.19. Таблиця рекордів

Введення імені гравця: У таблиці рекордів гравець може ввести своє ім'я для збереження результату. (Рис 2.20)

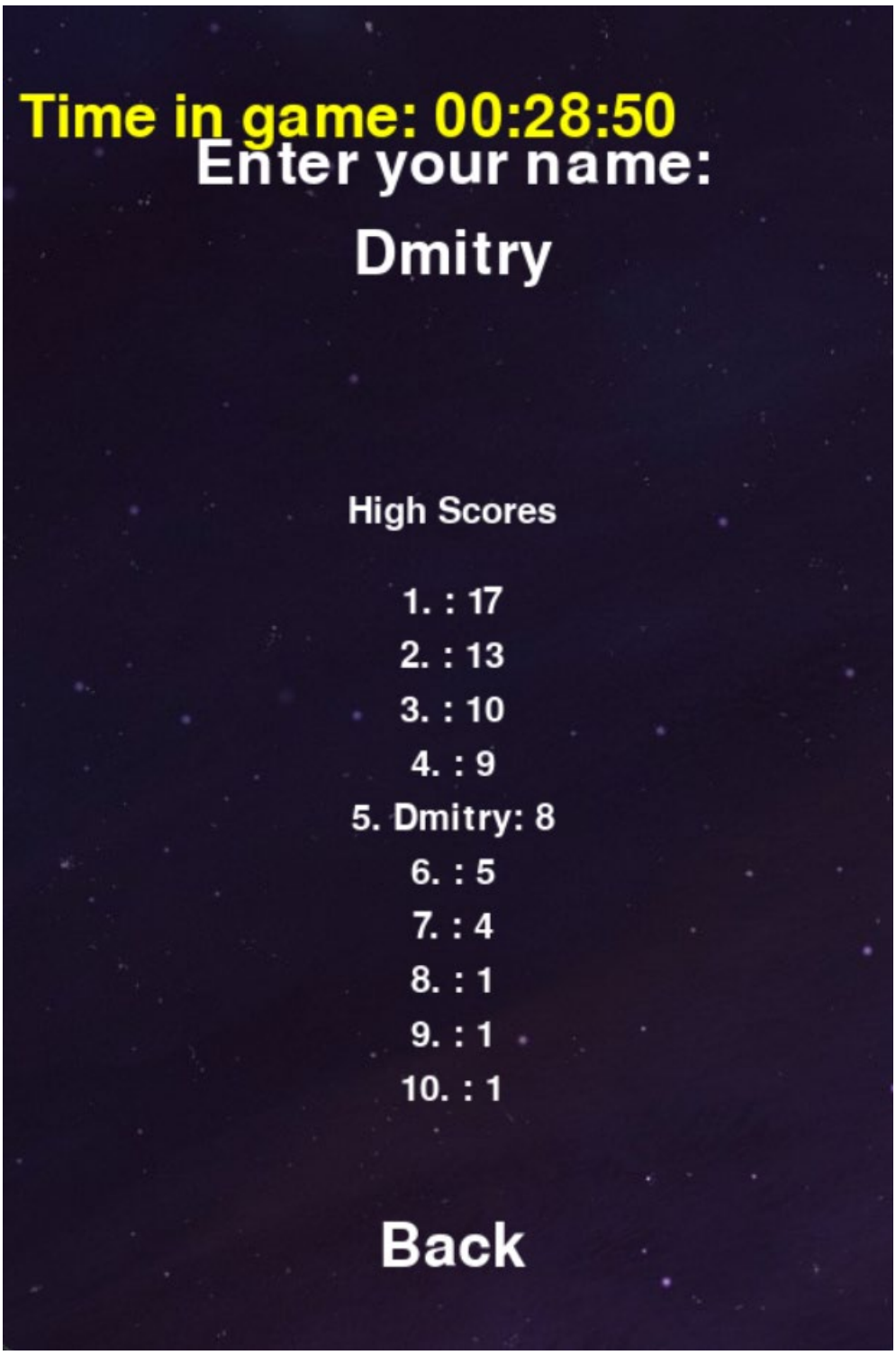

Рис. 2.20 Ім’я гравця у таблиці рекордів

Пауза: Гравець може поставити гру на паузу. (Рис 2.21)

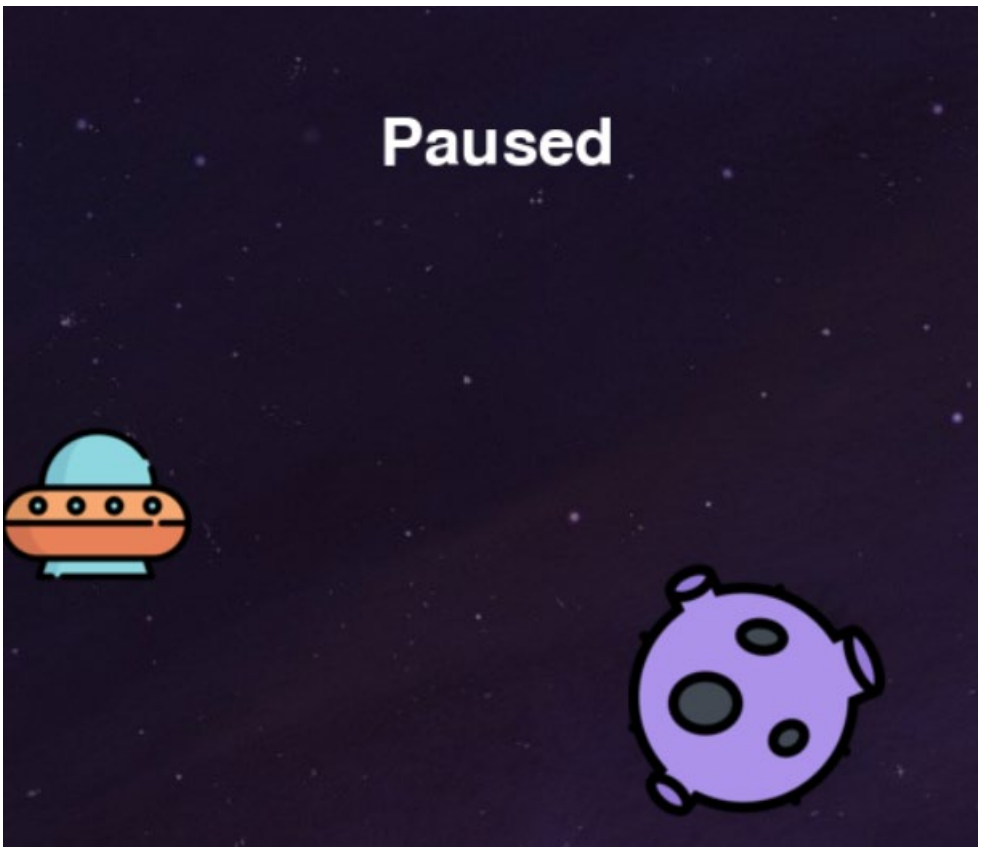

Рис 2.21. Пауза у грі

РОЗДІЛ 3
ЕКОНОМІКА

3.1. Визначення трудомісткості та вартості розробки програмного продукту

Початкові дані:

1. передбачуване число операторів програми – 300;
2. коефіцієнт складності програми – 1,25;
3. коефіцієнт корекції програми в ході її розробки – 0,2;
4. годинна заробітна плата програміста– 102 грн/год; [20]
5. коефіцієнт збільшення витрат праці внаслідок недостатнього опису задачі – 1,25;
6. коефіцієнт кваліфікації програміста, обумовлений від стажу роботи з даної спеціальності – 1,1;
7. вартість машино-години ЕОМ – 10.30 грн/год.

За годину ігровий ноутбук споживає 0,18 кВт/год, а вартість електроенергії склала 1,68 кВт/год. Тобто 0,18*1,68=0.30 грн. Амортизація комп'ютерного обладнання складає 10 грн.

Нормування праці в процесі створення ПЗ істотно ускладнено в силу творчого характеру праці програміста. Тому трудомісткість розробки ПЗ може бути розрахована на основі системи моделей з різною точністю оцінки.

Трудомісткість розробки ПЗ можна розрахувати за формулою:

$$t = t_o + t_u + t_a + t_n + t_{отл} + t_д \text{, людино-годин,} \quad (3.1)$$

де t_o- витрати праці на підготовку й опис поставленої задачі (приймається 50 людино-годин);

t_u - витрати праці на дослідження алгоритму рішення задачі;

t_a- витрати праці на розробку блок-схеми алгоритму;

t_n-витрати праці на програмування по готовій блок-схемі;

$t_{отл}$-витрати праці на налагодження програми на ЕОМ;

$t_д$ - витрати праці на підготовку документації.

Складові витрати праці визначаються через умовне число операторів у програмногму забезпеченні, яке розробляється.

Умовне число операторів:

$$Q = q \cdot C \cdot (1 + p), \quad (3.2)$$

де q - передбачуване число операторів (300);

C - коефіцієнт складності програми (1,25);

p - коефіцієнт корекції програми в ході її розробки (0,2).

$$Q = 300 * 1{,}25 * (1 + 0{,}2) = 450$$

Витрати праці на вивчення опису задачі t_u визначається з урахуванням уточнення опису і кваліфікації програміста:

$$t_u = \frac{Q \times B}{(75 \ldots 85) \times K} \text{ людино} - \text{годин}, \quad (3.3)$$

де B – коефіцієнт збільшення витрат праці внаслідок недостатнього опису задачі;

k – коефіцієнт кваліфікації програміста, обумовлений від стажу роботи з даної спеціальності.

$$t_u = \frac{450 \times 1{,}25}{75 \times 1{,}1} = 6{,}82 \text{ людино} - \text{годин}$$

Витрати праці на розробку алгоритму рішення задачі:

$$t_\alpha = \frac{Q}{(20 \ldots 25) \times K'}, \quad (3.4)$$

$$t_{\alpha} = \frac{450}{20 \times 1{,}1} = 24{,}75 \text{ людино} - \text{годин}$$

Витрати на складання програми по готовій блок-схемі:

$$t_n = \frac{Q}{(20 \ldots 25) \times k'}, \quad (3.5)$$

$$t_n = \frac{450}{21 \times 1{,}1} = 23{,}57 \text{ людино} - \text{годин}$$

Витрати праці на налагодження програми на ЕОМ:

- за умови автономного налагодження одного завдання:

$$t_{\text{отл}} = \frac{Q}{(4 \ldots 5) \times k}, \quad (3.6)$$

$$t_{\text{отл}} = \frac{450}{4 \times 1{,}2} = 135 \text{ людино} - \text{годин}$$

- за умови комплексного налагодження завдання:

$$t_{\text{отл}}^{k} = 1{,}5 * t_{\text{отл}}, \quad (3.7)$$

$$t_{\text{отл}}^{k} = 1{,}5 * 135 = \textit{202,5}$$

Витрати праці на підготовку документації:

$$t_{\partial} = t_{\partial\rho} + t_{\partial o}, \quad (3.8)$$

де $t_{\partial\rho}$ - трудомісткість підготовки матеріалів і рукопису.

$$t_{\partial\rho} = \frac{Q}{(15...20)\times k}, \text{ (3.9)}$$

$$t_{\partial\rho} = \frac{450}{15 \times 1{,}1} = 33 \text{ людино} - \text{годин}$$

$t_{\partial o}$ - трудомісткість редагування, печатки й оформлення документації.

$$t_{\partial o} = 0{,}75 * t_{\partial\rho}, \text{ (3.10)}$$

$$t_{\partial o} = 0{,}75 * 33 = 24{,}75 \text{ людино} - \text{годин}$$

$$t_{\partial} = 33 + 24{,}75 = 57{,}75 \text{ людино} - \text{годин}$$

$$t = 50 + 6{,}82 + 24{,}75 + 23{,}57 + 135 + 57{,}75 = 297{,}89 \text{ людино} - \text{годин}$$

3.2. Витрати на створення програмного забезпечення

Витрати на створення ПЗ *Кпо* включають витрати на заробітну плату виконавця програми *Зз/п* і витрат машинного часу, необхідного на налагодження програми на ЕОМ.

$$К_{ПО} = З_{ЗП} + З_{МВ}, \text{грн.}, \text{ (3.11)}$$

Заробітна плата виконавців визначається за формулою:

$$З_{ЗП} = t * С_{ПР}, \text{ грн.}, (3.12)$$

де: t - загальна трудомісткість, людино-годин;

$С_{пр}$ - середня годинна заробітна плата програміста, грн/година

$$З_{ЗП} = 297{,}89 * 102 = 30384.78 \text{ грн}$$

Вартість машинного часу, необхідного для налагодження програми на ЕОМ:

$$З_{МВ} = t_{отл} * С_{мч}, \text{ грн.}, (3.13)$$

де $t_{отл}$ - трудомісткість налагодження програми на ЕОМ, год.

$С_{мч}$ - вартість машино-години ЕОМ, грн/год.

$$З_{МВ} = 135 * 10.30 = 1390.5 \text{ грн}$$

$$K_{ПО} = 30384.78 + 1390.5 = 31775.28 \text{ грн.}$$

Визначені в такий спосіб витрати на створення програмного забезпечення є частиною одноразових капітальних витрат на створення АСУП.

Очікуваний період створення ПЗ:

$$T = \frac{t}{В_к * F_p}, \text{ міс.}, (3.14)$$

де $В_к$ - число виконавців;

F_p - місячний фонд робочого часу (при 40 годинному робочому тижні F_p = 176 годин).

$$T = \frac{297{,}89}{1 * 176} \approx 1{,}7$$

Висновок: Для розробки цього проекту було затрачено 297,89 годин, а вартість створення програмного забезпечення оцінюється в суму 31775,28 грн. Це вказує на очікувану тривалість розробки, яка складає 1,7 місяців.

ВИСНОВКИ

Ця гра є простою аркадною грою "Lost in Space". У грі гравець керує космічним кораблем (НЛО) і повинен уникати зіткнень з астероїдами, які з'являються на екрані. Гравець отримує очки за кожен пройдений астероїд і

може побити свій власний рекорд. Швидкість астероїдів збільшується залежно від кількості очок гравця. Гра має можливість паузи, а також відображення таблиці рекордів з іменами гравців.

Основний цикл гри включає обробку подій (натискання клавіш, натискання кнопок миші), оновлення позицій НЛО та астероїдів, перевірку зіткнень та відображення різних елементів гри, таких як фон, НЛО, астероїди, рахунок тощо. Також використовується модуль Pygame для реалізації графіки та звуків.

Гра має певні можливості для розширення, наприклад, можна додати більше різновидів астероїдів, використати різні типи контролю (наприклад, керування за допомогою сенсорного екрану або геймпада), вдосконалити механіку гри та включити більше функціональних можливостей, таких як power-ups або різні рівні складності.

Загалом, цей код може служити як основа для створення аркадних ігор, і його можна розширити та модифікувати залежно від ваших потреб і ідей.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Історія комп'ютерних ігор: http://ukped.com/rozrobky-urokiv/informatyka/218-istoriia-rozvytku-kompiuternoi-tekhniky.html
2. Перша відома комп’ютерна гра:

http://tsn.ua/special-projects/computer_graphics/
3. Про Python:
https://acode.com.ua/
4. Дослідження щодо негативного впливу відеоігор
https://ukr.media/science/385960/
5. Путівник мовою програмування Python
https://pythonguide.rozh2sch.org.ua/
6. Бібліотека Pygame
https://www.pygame.org/news
7. JSON (JavaScript Object Notation)
https://apix-drive.com/ua/blog/useful/scho-take-json
8. Звукові файли
https://naurok.com.ua/
9. sys — Системні параметри та функції
https://docs.python.org/uk/3/library/sys.html
10. PyCharm
http://technologies-school.blogspot.com/2018/03/pycharm-python.html
11. PyCharm. Його розширена підтримка Python
https://dou.ua/lenta/articles/tools-for-python-developers/
12. Функціональність PyCharm
https://blog.ithillel.ua/articles/creating-virtual-environments-and-installing-libraries-for-python-3-in-the-pycharm-ide
13. Windows 11
https://compx.ua/reliz-windows-11/
14. Керування Windows 11
https://itc.ua/ua/novini/novi-funktsiyi-windows-11-u-2023-rotsi-planshetnyj-rezhym-povnoekranni-vidzhety-pokrashhenyj-poshuk-ta-bagato-inshogo/
15. Ubuntu - опис ОС
https://hyperhost.ua/info/uk/ubuntu-opis-os-aktualni-versii-plyusi-minusi
16. Властивості і характеристики інтерфейсу Ubuntu
https://naurok.com.ua/kompleks-urokiv-sistemne-programne-zabezpechennya-57469.html
17. Різноманітні варіанти Ubuntu
https://ubunlog.com/uk/
18. APK-файл
https://tmginfo.net/2020/05/shho-take-apk-fayli-i-navishho-voni-potribni/
19. Buildozer
https://repository.kpi.kharkov.ua/server/api/core/bitstreams/a4786ba9-e3ae-431c-bd93-a1be1a0eb64b/content
20. Годинна заробітна плата програміста
https://www.work.ua/jobs-python+developer/?sort=salary

ДОДАТОК А

КОД ПРОГРАМИ

```
import pygame
import random
import json
import sys

# Инициализация Pygame
pygame.init()

# Инициализация таймера
clock = pygame.time.Clock()
start_time = pygame.time.get_ticks()

# Определение размеров экрана
screen_width = 500
screen_height = 750

# Создание экрана
screen = pygame.display.set_mode((screen_width, screen_height))
pygame.display.set_caption("Lost in Space")

# Загрузка изображений
background_img = pygame.image.load("Images/Космос.png").convert()
ufo_img = pygame.image.load("Images/нло.png").convert_alpha()
asteroid_img = pygame.image.load("Images/астероиды.png").convert_alpha()

# Определение глобальных переменных
gravity = 0.35
ufo_movement = 0
ufo_position = [50, screen_height // 2]
asteroid_width = 1
```

```
asteroid_heights = [100, 200, 300, 400, 500, 600]
asteroid_speed = 8
score = 0
font = pygame.font.Font(None, 48)

# Загрузка звуков
pygame.mixer.music.load("Sound/music.mp3")
pygame.mixer.music.set_volume(0.5)

paused = False

# Отрисовка счета
def draw_score():
    score_text = font.render("Score: " + str(score), True, (255, 255, 255))
    screen.blit(score_text, (10, 10))



# Генерация нового астероида
def generate_asteroid():
    asteroid_x = screen_width
    asteroid_y = random.choice(asteroid_heights)
    asteroid_rect = asteroid_img.get_rect(midtop=(asteroid_x, asteroid_y))
    return asteroid_rect



# Проверка столкновения с астероидами
def check_collision():
    for asteroid in asteroids:
        if ufo_rect.colliderect(asteroid):
            return True
```

```
    if ufo_rect.top <= 0 or ufo_rect.bottom >= screen_height:
        return True
    return False


# Загрузка таблицы рекордов из файла
def load_high_scores():
    try:
        with open("high_scores.json", "r") as file:
            high_scores = json.load(file)
            return high_scores
    except FileNotFoundError:
        return []


# Сохранение таблицы рекордов в файл
def save_high_scores(high_scores):
    with open("high_scores.json", "w") as file:
        json.dump(high_scores, file)


# Отрисовка таблицы рекордов
def draw_high_scores(high_scores):
    scores_font = pygame.font.Font(None, 30)
    title_text = scores_font.render("High Scores", True, (255, 255, 255))
    screen.blit(title_text, (screen_width // 2 - title_text.get_width() // 2,
screen_height // 2 - 100))

    for i, (score, player_name) in enumerate(high_scores):
```

```
        score_text = scores_font.render(f"{i + 1}. {player_name}: {score}", True, (255, 255, 255))
        screen.blit(score_text, (screen_width // 2 - score_text.get_width() // 2, screen_height // 2 - 50 + i * 30))


# Отрисовка меню
def draw_menu():
    title_font = pygame.font.Font(None, 100)
    title_text = title_font.render("Lost in Space", True, (0, 0, 139))
    screen.blit(title_text, (screen_width // 2 - title_text.get_width() // 2, screen_height // 2 - 250))

    start_font = pygame.font.Font(None, 80)
    start_text = start_font.render("Click to START", True, (255, 255, 255))
    start_text_rect = start_text.get_rect(topleft=(screen_width // 2 - start_text.get_width() // 2,
                                                   screen_height // 2 + 0))
    screen.blit(start_text, start_text_rect)

    high_scores_text = start_font.render("High Scores", True, (255, 255, 255))
    high_scores_rect = high_scores_text.get_rect(topleft=(screen_width // 2 - high_scores_text.get_width() // 2,
                                                          screen_height // 2 + 200))
    screen.blit(high_scores_text, high_scores_rect)

    return start_text_rect, high_scores_rect


# Игровой цикл
```

```
running = True
clock = pygame.time.Clock()

game_active = False
high_scores_active = False
high_scores = load_high_scores()

asteroids = []
SPAWN_ASTEROID = pygame.USEREVENT
pygame.time.set_timer(SPAWN_ASTEROID, 1000)

# Воспроизведение фоновой музыки
pygame.mixer.music.play(-1)

# Получение прямоугольников текста "START" и "High Scores"
name = ""
start_text_rect, high_scores_rect = draw_menu()

while running:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            running = False
            pygame.quit()
            sys.exit()

        # Обработка событий нажатия клавиш
        elif event.type == pygame.KEYDOWN:
            if event.key == pygame.K_SPACE:
                # Если игра активна и не на паузе, нажатие пробела изменяет
движение НЛО
```

```
            if game_active and not paused:
                ufo_movement = 0
                ufo_movement -= 7
        elif event.key == pygame.K_p:
            paused = not paused
        elif event.key == pygame.K_BACKSPACE:
            if high_scores_active and len(name) > 0:
                name = name[:-1]
        elif event.unicode.isalnum() and len(name) < 10:
            name += event.unicode

            # Обработка событий нажатия кнопок мыши
    elif event.type == pygame.MOUSEBUTTONDOWN:
        if start_text_rect.collidepoint(event.pos):
            if game_active:
                ufo_movement = 0
                ufo_movement -= 7
            else:
                game_active = True
                ufo_movement = 0
                ufo_position = [50, screen_height // 2]
                asteroids.clear()
                score = 0
        elif high_scores_rect.collidepoint(event.pos):
            high_scores_active = True
        elif high_scores_active and back_rect.collidepoint(event.pos):
            high_scores_active = False
            save_high_scores(high_scores)  # Сохранение таблицы рекордов при
нажатии кнопки "Back"
```

```
        elif event.type == SPAWN_ASTEROID and game_active:
            asteroids.append(generate_asteroid())

    screen.blit(background_img, (0, 0))

    # Расчет времени пользователя в игре
    current_time = pygame.time.get_ticks() - start_time
    seconds = current_time // 1000

    # Отрисовка времени
    hours = seconds // 3600
    minutes = (seconds % 3600) // 60
    seconds = seconds % 60
    time_text = font.render(f"Time in game:
{hours:02d}:{minutes:02d}:{seconds:02d}", True, (255, 255, 0))
    screen.blit(time_text, (10, 50))

    if game_active:
        # Обновление позиции НЛО
        ufo_movement += gravity
        ufo_rect = ufo_img.get_rect(center=ufo_position)
        ufo_position[1] += ufo_movement
        screen.blit(ufo_img, ufo_rect)

        # Обновление позиций астероидов
        asteroids = [asteroid for asteroid in asteroids if asteroid.right > -10]
        passed_asteroids = []
        for asteroid in asteroids:
            if asteroid.right < ufo_rect.left and asteroid not in passed_asteroids:
                score += 1
```

```
            passed_asteroids.append(asteroid)
        asteroid.centerx -= asteroid_speed
        screen.blit(asteroid_img, asteroid)
    # Отрисовка счета
    draw_score()

    # Обновление счета и проверка столкновения
    if check_collision():
        game_active = False
        if score > 0:
            high_scores.append((score, name))
            high_scores.sort(key=lambda x: x[0], reverse=True)
            high_scores = high_scores[:10]
            save_high_scores(high_scores)

    # Увеличение скорости астероидов
    if score >= 1 and score <= 2:
        asteroid_speed = 8 * 1.01
    if score >= 3 and score <= 4:
        asteroid_speed = 8 * 1.02
    if score >= 5 and score <= 6:
        asteroid_speed = 8 * 1.03
    if score >= 7 and score <= 8:
        asteroid_speed = 8 * 1.04
    if score >= 9 and score <= 10:
        asteroid_speed = 8 * 1.05
    if score >= 11 and score <= 12:
        asteroid_speed = 8 * 1.06
    if score >= 13 and score <= 14:
        asteroid_speed = 8 * 1.07
```

```
if score >= 15 and score <= 16:
    asteroid_speed = 8 * 1.08
if score >= 17 and score <= 18:
    asteroid_speed = 8 * 1.09
if score >= 19 and score <= 20:
    asteroid_speed = 8 * 1.1
if score >= 21 and score <= 22:
    asteroid_speed = 8 * 1.15
if score >= 22 and score <= 24:
    asteroid_speed = 8 * 1.2
if score >= 25 and score <= 30:
    asteroid_speed = 8 * 1.25
if score >= 31 and score <= 35:
    asteroid_speed = 8 * 1.3
if score >= 36 and score <= 40:
    asteroid_speed = 8 * 1.35
if score >= 41 and score <= 45:
    asteroid_speed = 8 * 1.4
if score >= 46 and score <= 50:
    asteroid_speed = 8 * 1.45
if score >= 51 and score <= 55:
    asteroid_speed = 8 * 1.5
if score >= 56 and score <= 60:
    asteroid_speed = 8 * 1.2
if score >= 61 and score <= 65:
    asteroid_speed = 8 * 1.25
if score >= 66 and score <= 70:
    asteroid_speed = 8 * 1.3
if score >= 71 and score <= 80:
    asteroid_speed = 8 * 1.4
```

```
        if score >= 81 and score <= 100:
            asteroid_speed = 8 * 1.5
        if score >= 101 and score <= 1000:
            asteroid_speed = 8 * 2

    elif high_scores_active:
        # Отрисовка таблицы рекордов
        draw_high_scores(high_scores)
        back_font = pygame.font.Font(None, 50)
        back_text = back_font.render("Back", True, (255, 255, 255))
        back_rect = back_text.get_rect(topleft=(screen_width // 2 -
back_text.get_width() // 2,
                                        screen_height // 2 + 300))
        # Отрисовка кнопки "Back"
        screen.blit(back_text, back_rect)

        name_font = pygame.font.Font(None, 50)
        name_text = name_font.render("Enter your name:", True, (255, 255, 255))
        name_rect = name_text.get_rect(topleft=(screen_width // 2 -
name_text.get_width() // 2,
                                         screen_height // 2 - 300))
        # Отрисовка текста "Enter your name"
        screen.blit(name_text, name_rect)

        input_font = pygame.font.Font(None, 50)
        input_text = input_font.render(name, True, (255, 255, 255))
        input_rect = input_text.get_rect(topleft=(screen_width // 2 -
input_text.get_width() // 2,
                                          screen_height // 2 - 250))
        # Отрисовка введенного имени
```

```
            screen.blit(input_text, input_rect)

        else:
            # Отрисовка меню (текст "START" и "High Scores")
            start_text_rect, high_scores_rect = draw_menu()
        if paused:
            # Отрисовка текста "Paused"
            pause_text = font.render("Paused", True, (255, 255, 255))
            screen.blit(pause_text, (screen_width // 2 - pause_text.get_width() // 2,
screen_height // 2))

        # Обновление экрана
        pygame.display.update()
        clock.tick(60)
```

ДОДАТОК Б

ВІДГУК
керівника економічного розділу
на кваліфікаційну роботу бакалавра
на тему:

ДОДАТОК В

Перелік файлів на диску

Ім’я файлу	**Опис**
Пояснювальні документи	

ПЗ_Кудрявцев.doc	Пояснювальна записка до кваліфікаційної роботи. Документ Word.
ПЗ_Кудрявцев.pdf	Пояснювальна записка до кваліфікаційної роботи в форматі PDF
Програма	
Program.rar	Архів. Містить коди програми і откомпільовану програму
Презентація	
Презентація_Кудрявцев.ppt	Презентація кваліфікаційної роботи