МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

КАФЕДРА КОМП'ЮТЕРНИХ НАУК

ВИПУСКНА РОБОТА

на тему:

«Інформаційне та програмне забезпечення ігрової системи аркадного типу»

Завідувач
випускаючої кафедри **Довбиш А.С.**

Керівник роботи **Шелехов І.В.**

Студента групи ІН -73 **Мальцев В.О.**

СУМИ 2021

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ ЕЛЕКТРОНІКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
Кафедра комп'ютерних наук

Затверджую_____
Зав. кафедри Довбиш А.С.
"_____" _____20__р.

ЗАВДАННЯ
до випускної роботи

Студента четвертого курсу, групи ІН73 спеціальності "Комп'ютерні науки" денної форми навчання Мальцева В.О.

Тема: «Інформаційне та програмне забезпечення ігрової системи аркадного типу»

Затверджена наказом по СумДУ
№_____ от _____20__р.

Зміст пояснювальної записки: 1) аналіз проблеми та постановка задачі; 2) вибір методів розв'язання задачі; 3) розробка інформаційного і програмного забезпечення системи

Дата видачі завдання "_____"_____20__р.
Керівник випускної роботи _____Шелехов І.В.
Завдання прийняв до виконання _____ Мальцев В.О.

РЕФЕРАТ

Записка: 68 стор., 42 рис., 3 табл., 1 додаток, 18 бібліографічних джерел.

Об’єкт дослідження – процес проектування ігрової системи аркадного типу.

Мета роботи — розробка інформаційного і програмного забезпечення ігрової системи аркадного типу.

Методи дослідження — методи проектування інформаційних систем, методи подання і обробки графічних даних.

Результати — В випускній роботі було проведено розробку та програмну реалізацію ігрової системи з гнучкими налаштуваннями в жанрі "Аркада" на платформі персонального комп'ютера під управлінням операційного середовища Windows. При цьому було розроблено концептуальну схему ігрової системи, в тому числі UML діаграми та бізнес-сутності та логіка системи, розроблено ігрові механізми з використанням стандартних об'єктів, розроблено механізми зберігання і використання текстур, музики, моделей, розроблено логічний контент системи, в тому числі підсистему складності ігрового процесу та підсистему стратегій розвитку ігрового персонажу. Програмна реалізація ігрової системи виконано на мові програмування C ++ з використанням бібліотек Qt і IDE Qt Creator. Тестування ігрової системи доводить працездатність її ігрових механізмів, а також механізмів гнучкого налаштування її основних параметрів

ІГРОВА СИСТЕМА, БІЗНЕС-ЛОГІКА, БІЗНЕС-СУТНІСТЬ, ЛОГІЧНИЙ КОНТЕНТ, QT, IDE QT CREATOR

ЗМІСТ

ВСТУП

IT-спеціаліст широке поняття, яке об'єднує представників багатьох професій, що працюють в галузі інформаційних технологій. Це програмісти, розробники, мережеві адміністратори та адміністратори баз даних, модератори, фахівці з робототехніки, інформації безпеки, веб-дизайнери та 3D-аніматори.

При цьому, з проникненням інформаційних технологій в усі нові сфери діяльності, з'являються нові професії для IT-фахівців. За даними на сьогоднішній день і думку багатьох аналітиків [1] фахівці даної області є затребуваними і будуть затребувані в найближчому майбутньому. Зокрема, сьогодні в світі з'являються багато перспективних ігрових розробок і проектів.

GameDev (вимовляється як "геймдев") - абревіатура від Game Development (розробка ігор). У процес розробки ігрової програми входить: розробка дизайну ігрового процесу (геймплею), програмування ігрового движка або використання готових рішень, розробка візуального концепту і його складових, створення графіки і моделювання об'єктів, створення музичного та звукового супроводу.

Ігрова індустрія продовжує розвиватися і з кожним роком її прибуток на ринку зростає, як у світі, так і, зокрема, в Україні. Таким чином, ідея створення цікавої і нової гри є перспективним направленням.

1 АНАЛІЗ ПРОБЛЕМИ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Ігрова індустрія

На сьогоднішній день ігрова індустрія [2] – це однин з найвідоміших сегментів цифрового контенту в світі. Ця галузь постійно розвивається і зростає (рис 1.1). Так на кінець 2016 року - світовий ігровий ринок склав близько 100 мільярдів доларів.

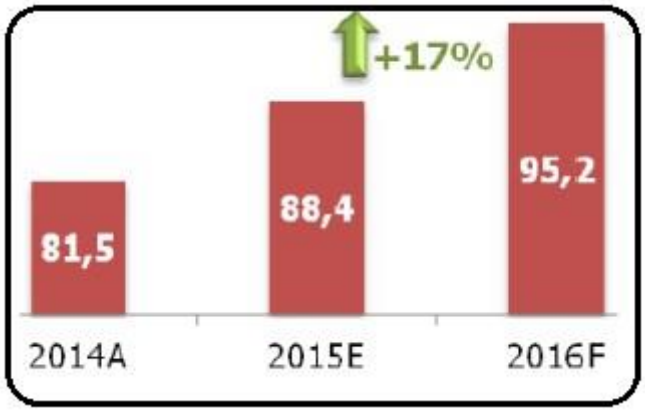

Рисунок 1.1 - Світовий ігровий ринок 2014-2017 років, млрд. дол.

За результатами 2015-16 року, ігровий ринок є значущим напрямком, що активно розвивається. При цьому, обсяг ігрового ринку продовжує збільшуватися з 2010 року (рис 1.2) [3]. Ринок ігор до цих пір є інвестиційно привабливим.

Рисунок 1.2 - Динаміка зростання ігрового ринку з 2010 по 2016 рік

Український ринок тільки недавно почав розвиватися в цьому напрямку. Частка України на світовому ігровому ринку становила близько 2,4% (рис 1.3), але стійке зростання ігрового ринку приваблює західні ігрові компанії.

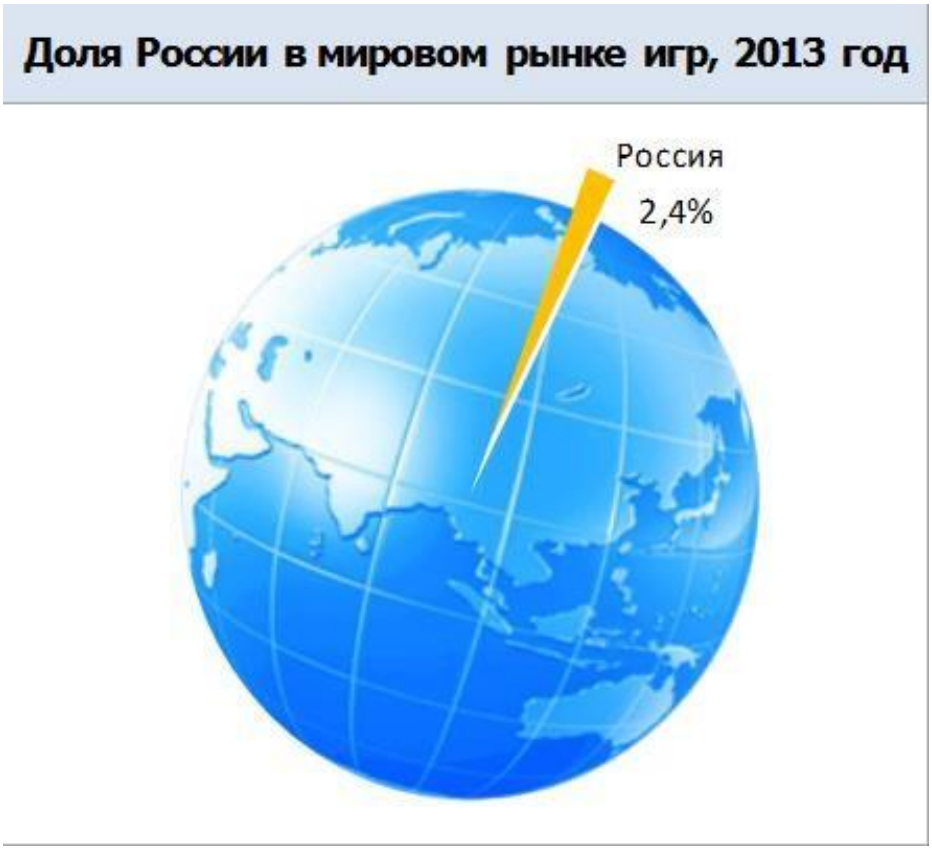

Рисунок 1.3 - Частка українського ігрового ринку за 2013 рік

1.2 Вибір платформи

На сьогоднішній день існує три ігрові платформи:

- Персональні комп'ютери
- Мобільні пристрої
- Консоль.

Початок ігор почався з появи комп'ютера, тому користувачі ПК є основною ігровою аудиторією. Так більше половини ігрового ринку належить до платформи персональних комп'ютерів (рис 1.4).

Рисунок 1.4 - Ігровий ринок на різних платформах

При аналізі персонального комп'ютерного ринку був зроблений приблизний розрахунок розмірів використовуваних на них операційних систем [5]. Результати аналізу представлені на рисунку 1.5.

Рисунок 1.5 показує, що більшість користувачів персональних комп'ютерів віддають перевагу операційній системі сімейства Windows.

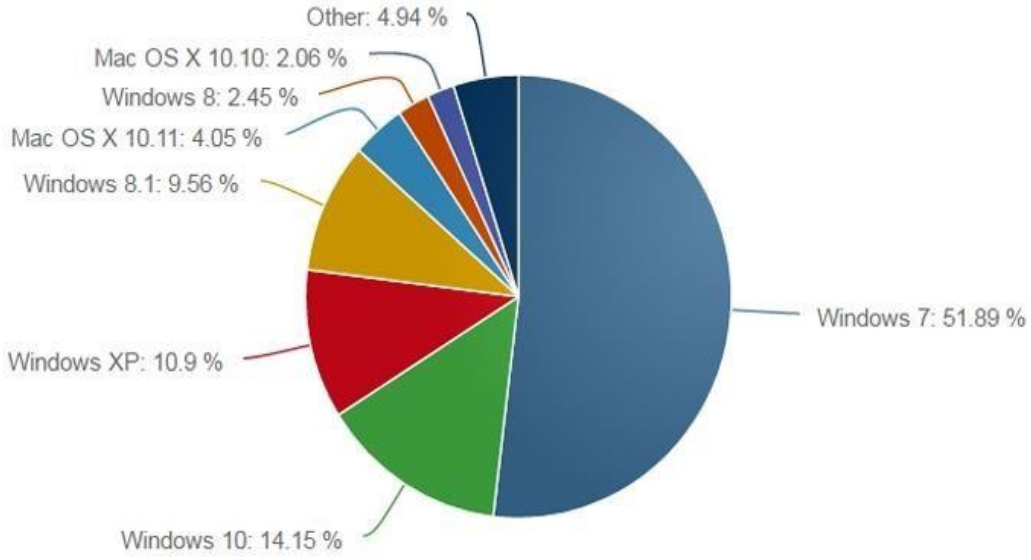

Рисунок 1.5 - Використовувані операційні системи

1.3 Вибір жанру гри

Сьогодні в світі переважно виділяють такі жанри ігор:

- Азартні
- спортивні
- головоломки
- стратегії
- Бойовики
- Аркади

При цьому багато користувачів віддають перевагу жанру "Аркада" [6]. Аркади є досить популярні відносно інших жанрів з наступних причин (рис. 1.6):

1. Аркади не вимагають потужних технічних характеристик

Не всі користувачі мають можливість придбати сучасні і потужні пристрої з відповідними характеристиками. Багато користувачів відзначають, що тим потужніше гра, тим більше вона гальмує. Таким чином, геймплей досить складний і неприємний для користувача, що відбиває бажання грати і, таким чином, знижує рейтинг додатку, що в свою чергу знижує інтерес нових користувачів (гравців) інсталювати цю програму. Однак аркади не вимагають

серйозних технічних характеристик, що дозволяє більшості користувачів насолоджуватися геймплеєм без затримок, зависання та інших проблем з продуктивністю.

2. Аркади не займають багато часу

Всі користувачі ігрових додатків (гравці) займаються будь-якою діяльністю (працюють, навчаються тощо). Таким чином, у гравців не завжди є вільний час, щоб проводити його за іграми. Аркади створені для людей у яких немає можливості проводити за грою достатню кількість часу. Зазвичай даний жанр дозволяє гравцеві зайняти його час на якийсь період. Тим самим, більшість користувачів називають аркади ─ «ТаймКіллерамі», так як вони є цікавими і незамінними помічниками весело провести період часу.

3. Аркади цікаві

Багато користувачів відзначають, що жанр аркади універсальний. Він може включати логічні головоломки, красиву графіку або цікавий ігровий процес (геймплей), тобто не дозволяти гравцеві нудьгувати.

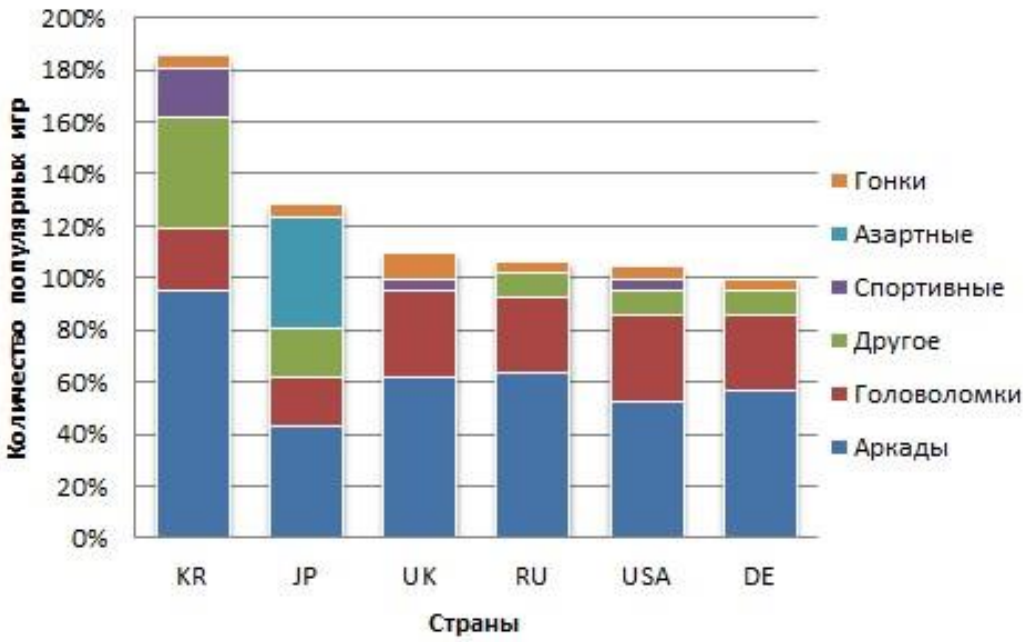

Рисунок 1.6 - Популярність ігрових жанрів

Таким чином, як жанр гри "PushHimAll" було вирішено вибрати жанр "Аркада".

1.4 Вибір програмного компонента програми

В даний час існує безліч програмного забезпечення, яке дозволяє розробляти власні проекти різного роду, в тому числі і комп'ютерні ігрові додатки. Виділяють дві основні групи з них:

- використання програмного компонента (комплексу), що дозволяє створити ігровий додаток (ігровий «рушій»);
- створити власний «рушій» за допомогою мови програмування та спеціалізованих бібліотек.

Давайте уважніше розглянемо кожну групу:

1) ігрові «рушії»;

Unity 3D [7] - потужний інструмент для створення додатків та ігор для різних платформ. Є можливість створювати як 2D, так і 3D проекти. Є підтримка DirectX і OpenGL, підтримує безліч різних форматів даних. Unity 3D підтримує такі мови програмування як: C#, JavaScript. Основне робоче вікно Unity 3D представлено на рис. 1.7.

Плюси:

- не найбільші витрати ресурсів (у порівнянні зіншими «движками»);
- кросплатформеність;
- велике суспільство користувачів і розробників;
- регулярні оновлення.

Мінуси:

- Має невеликий набір інструментів;
- Вимагає гідних технічних компонентів;
- Платна комерційна ліцензія
- Незручна робота при створенні 2D - додатків.

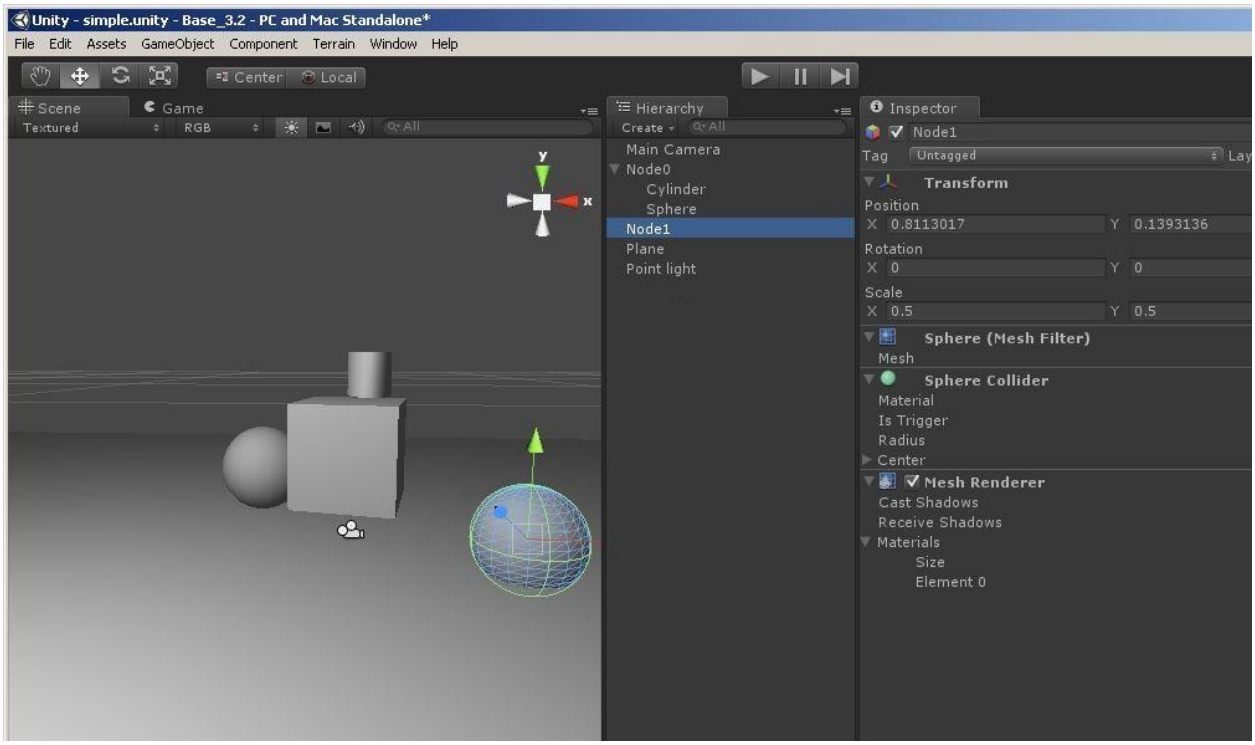

Рисунок 1.7 - Робоче вікно ігрового рушія Unity 3D

CryEngine [8] - найпотужніший з сучасних ігрових рушіїв, забезпечує фотореалістичну графіку з підтримкою DirectX і шейдерів (тіней). Третя версія «рушія» створена в 2009 році, поширюється платно. П'ята версія «рушія» вийшла в кінці 2016 року і має умовно безкоштовну ліцензію. Основним напрямком даного засобу є створення ігор жанру

«Шутер». Робоче вікно CryEngine представлено на рис. 1.8.

Плюси:

- Високі показники графіки
- Кросплатформеність:

Мінуси:

- Має невеликий набір інструментів;
- Спрямовані на створення певного жанру ігор;
- Найбільш ресурсомісткий «движок» з усіх представлених.

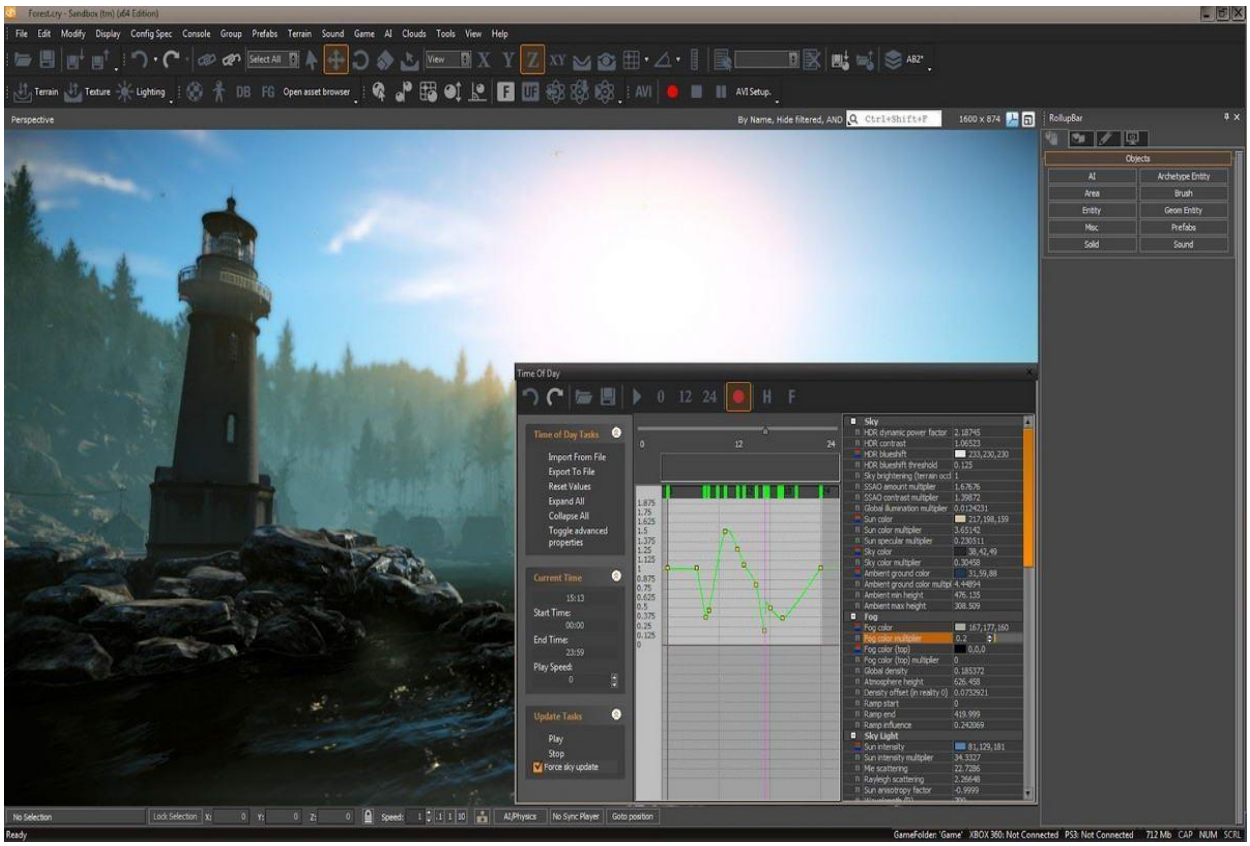

Рисунок 1.8 - Робоче вікно ігрового движка CryEngine

Unreal Engine [9] ─ ігровий «рушій», що розроблений і підтримується компанією Epic Games. Надає великий набір інструментарію для створення 2D і 3D проектів. Починаючи з четвертої версії ─ поширюється безкоштовно. Однак до тих пір, поки розробник не випустить свій перший комерційний продукт на основі UE4. Є основним рішієм більшості сучасних ігор. Робоче вікно ігрового «рушій» Unreal Enginge представлено на рис. 1.9.

Плюси:

- Високі показники графіки
- Кросплатформеність;
- Великі та регулярні оновлення
- Різноманітний інструментарій.

Мінуси:

- Високі технічні вимоги
- Складність в освоєння

- Невелика кількість україномовної документації;
- Комерційна ліцензія.

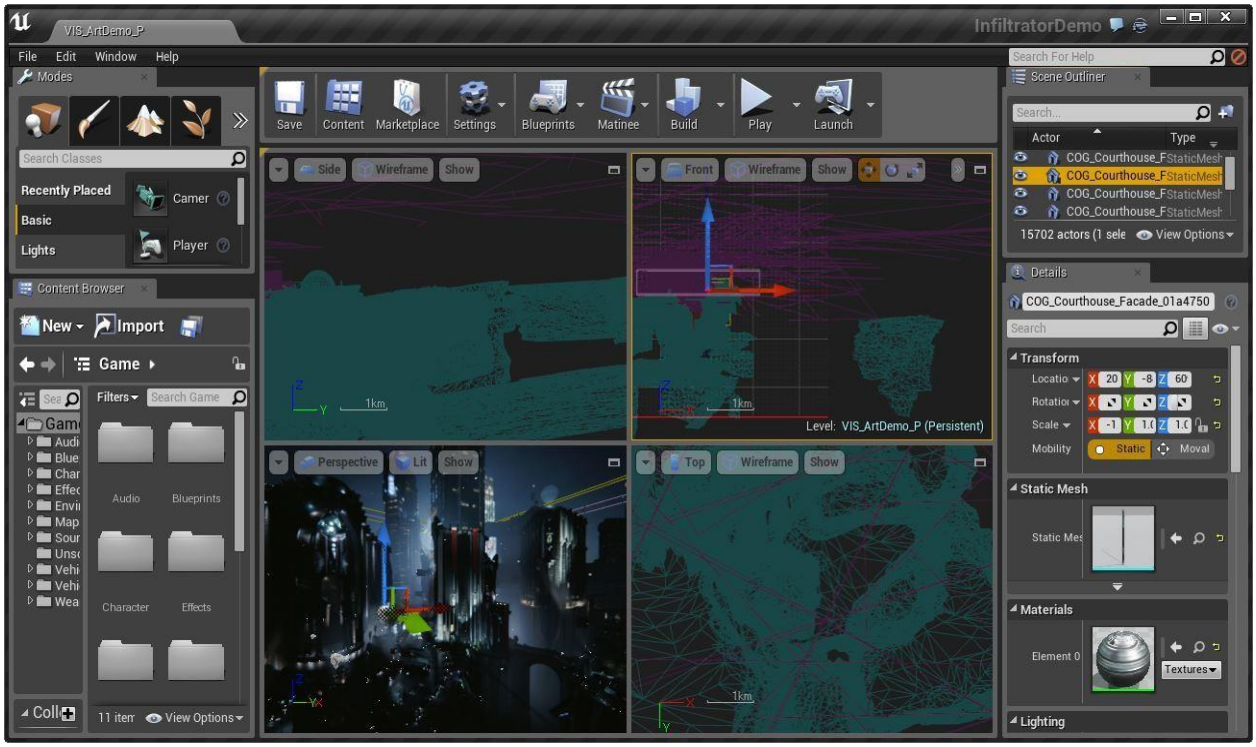

Рисунок 1.9 - Робоче вікно ігрового рушія Unreal Engine

Всі розглянуті ігрові «рушія» в основному призначені для створення 3D-додатків і вимагають високої показники технічних вимог, тому використовувати їх не рекомендується (багато користувачів ПК не мають сучасних і потужних компонентів). Давайте подивимося на іншу групу програмного забезпечення для створення ігрового додатку.

2) Бібліотеки для мов програмування (API).

По-перше, потрібно вирішити, яку мову програмування використовувати [10]. На сьогоднішній день виділяють наступні мови програмування в в якості основи подальшої розробки:

1. C#

C# дозволяє швидше почати розробку, а це дозволяє швидше отримати прототип рішення. Швидкість розробки на початкових етапах проекту набагато вищі, ніж на інших мовах.

Він розроблений таким чином, щоб бути кросплатформеним, але його розвиток не пішов в цьому напрямку, тому під Windows утворилася досить повна .NET інфраструктура, а на інших платформах рівноцінної інфраструктури не з'явилося.

При розробці невеликих проектів продуктивність C# не поступається іншим мовам програмування, однак при збільшення вихідного коду, алгоритмів тощо – швидкість роботи додатків значно падає.

C# має пристойну кількість бібліотек з початку, що значно полегшує розвиток.

2. HTML і JavaScript

Вони незамінні при створенні простих веб-додатків або ігор, але саме тут закінчуються переваги цих інструментів.

3. C++

Є основною мовою програмування при розробці ігор, так як дозволяє здійснити повний контроль над засобами і логікою. Є абсолютним рекордсменом по продуктивності, по кросплатформеності, по сумісності і за кількістю існуючих бібліотек серед інших мов програмування. Однак багато програмістів виділяють головний його недолік ─ складність освоєння даної мови.

Зведемо отримані дані в таблицю 1.1.

Кожна мова програмування добре працює в певних завданнях. Наприклад: Веб-розробка (JavaScript) немає прив’язки до операційної системи; якщо пріоритетом розробки є швидкість (створення прототипу додатку), слід використовувати C#. Як мову програмування для створення ігор було вирішено використовувати мову C++ тому що вона підтримує величезну кількість бібліотек і має хорошу продуктивність.

Таблиця 1.1 – Порівняння мов програмування

Критерій	C#	JavaScript/HTML	C++
Продуктивність	★★★☆☆	★★☆☆☆	★★★★★ (максимальна)
Кількість бібліотек	★★★☆☆	★★☆☆☆	★★★★★
Зручність програмування	★★★★☆ (легко)	★★★☆☆	★★★☆☆
Складності	★★★★☆ (легко)	★★★★☆	★★★☆☆
Кросплатформеність	★★★☆☆	★★★★★ (повний)	★★★★☆

Кожна мова програмування добре працює в певних завданнях. Наприклад: Веб-розробка (JavaScript) немає прив’язки до операційної системи; якщо пріоритетом розробки є швидкість (створення прототипу додатку), слід використовувати C#. Як мову програмування для створення ігор було вирішено використовувати мову C++ тому що вона підтримує величезну кількість бібліотек і має хорошу продуктивність.

На мові C++ існує два основних API для створення графіки:

1) SFML [11]

Проста і багатоплатформова мультимедійна бібліотека. Включає в себе модулі для програмування ігор та мультимедійних засобів. Він складається з п'яти модулів:

1. System – модуль керування часом і потоками. Є базовим, тобто необхідний для всіх модулів.
2. Window - управління вікнами і виведення інформації користувачеві.
3. Graphics - виконує вивід інформації (зображення, геометричні фігури). Для використання потрібно підключати модуль Window.
4. Audio - це бібліотека для звукових робіт.
5. Network — це бібліотека для роботи з мережею.

2) Qt Графіка [12]

Більш потужний інструмент (у порівнянні з SFML). Він є частиною бібліотеки Qt , яка представляє собою інструментарій розробки ПЗ на мові програмування C++ і має кросплатформеність. Qt надає програмісту не тільки зручний набір бібліотек класів, а й певну модель розробки додатків, певний каркас їх структури, істотно знижує появу помилок в додатку (витоку пам'яті, незакриті файли тощо). Бібліотека Qt володіє власною IDE (середовищем розробки) ─ Qt Creator, що істотно спрощує розробку додатків. Оболонка Qt Creator представлена на рис. 1.10.

Існує спосіб об'єднання QT і SFML, однак, як інструмент розробки, був обраний QT, оскільки він має більш широкий функціонал

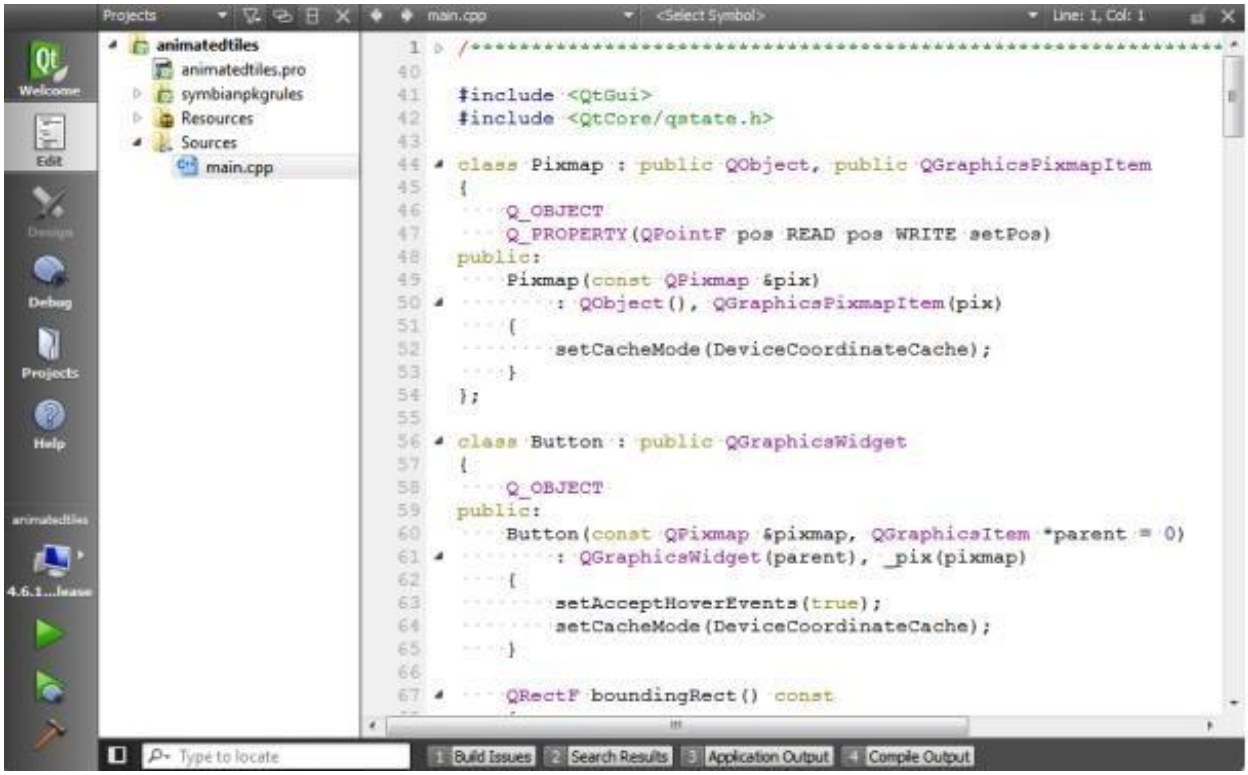

Рисунок 1.10 – IDE Qt Creator

1.5 Огляд існуючих аналогів ігрових систем аркадного типу

Сьогодні є багато аркадних ігрових додатків жанру аркада. Розглянемо найбільш поширені з них.

1.5.1 Space Invaders [13]

Найпопулярнішою аркадою є Space Invaders (рис. 1.11), яка була одна з перших в цьому жанрі. Мета гри ─ знищити всіх ворогів на екрані. Спочатку гра мала просту графіку і показник здоров'я, при досягнення нульового значення якого ─ гра завершувалась, проте пізніше вона удосконалилася (були введені нові колірні схеми і показник очок) (рис. 1.12).

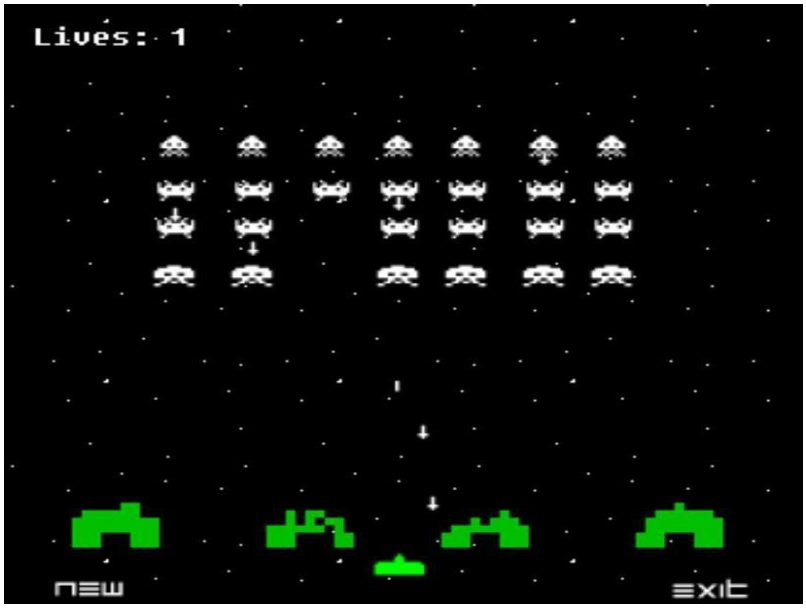

Рисунок 1.11 – Скріншот гри Space Invaders

Недоліки:

- Користувач не має можливості змінювати ігрові моделі.
- Користувач не має можливості змінювати логіку гри (змінювати складність рівня, змінювати траєкторію руху ворогів);
- У першій версії гри показника здоров'я не було.
- Гра має примітивні моделі.

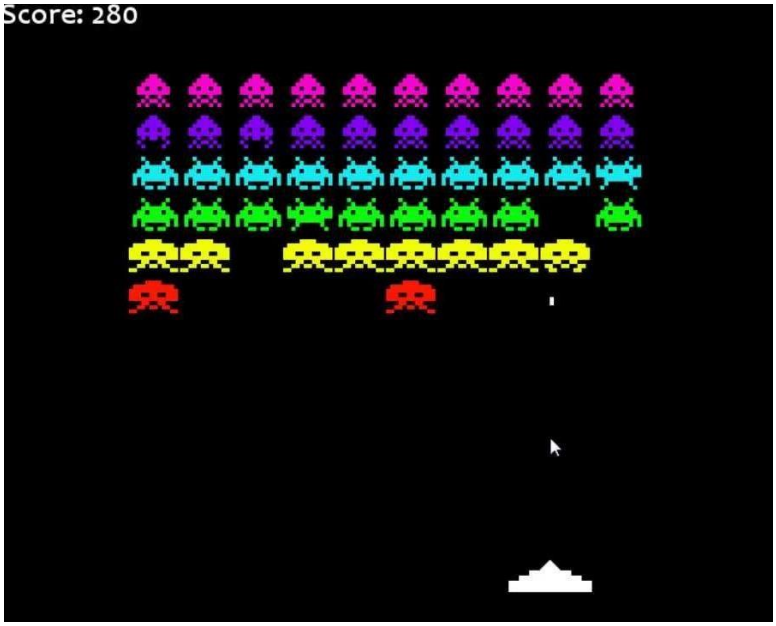

Рисунок 1.12 – Скріншот удосконаленої Space Invaders

1.5.2 Radiant [14]

Аналог гри Space Invaders випущений в 2010 році. Мета гри - знищити всіх ворогів на екрані. Головна особливість гри ─ поліпшена 2D графіка. У міру збільшення часу гри збільшується і складність ─ з'являються більше ворогів, однак, у користувача немає можливості змінювати моделі гри.

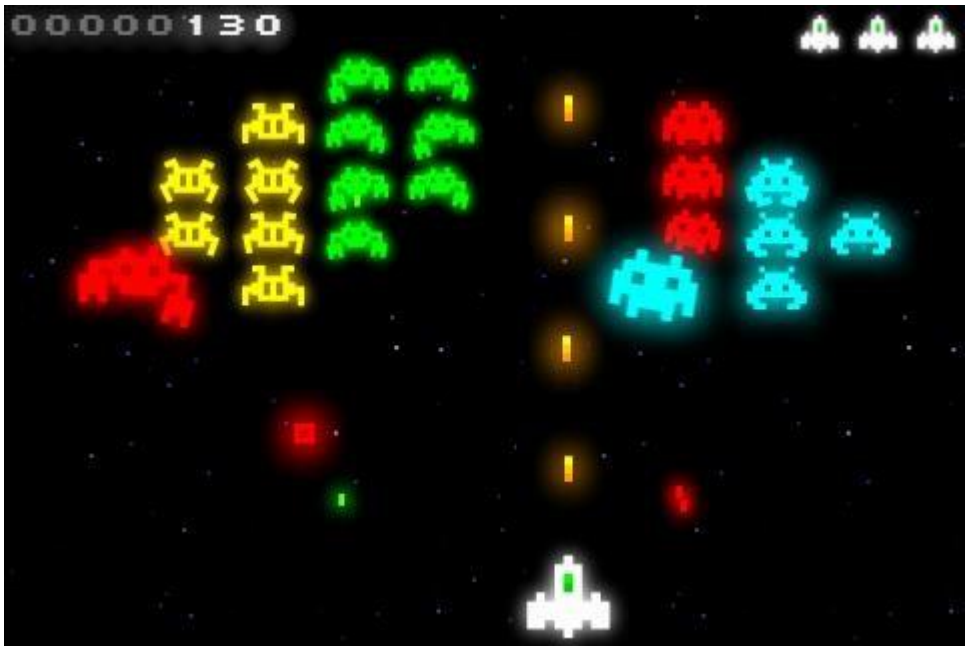

Рисунок 1.13 - Скріншот гри Radiant

Плюси:

- Покращена 2D-графіка
- Спостерігається збільшення складності в міру просування.

Недоліки:

- У грі присутні тільки стандартні подібні моделі.
- Складність присутня, однак, користувач не має можливості вибрати її самостійно;
- Вибрати саундтрек не можна.

1.5.3 Luxor [15]

«Luxor» — серія аркадних комп'ютерних ігор, розроблених компанією MumboJumbo і виконаних в 2D - стилі, заснованих на міфології Стародавнього Єгипту. Завдання гравця - винищити з'являються на екрані сферичних об'єктів, до того, як вони досягнуть піраміди.

Гра має адаптивну складність (складність збільшується з рівнем), однак користувач не в змозі змінити кулі (ігрові об'єкти) на певний колір або інші форми

Рисунок 1.14 -Скріншот гри «Luxor»

Плюси:

- Є елементи головоломок (гра розвиває логіку);
- Спостерігається збільшення складності в міру просування.

Недоліки:

- Гра має тільки сферичні моделі. Користувач не має можливості їх замінити;
- Складність присутня, однак, користувач не має можливості вибрати її самостійно.

1.5.4 Beyond Space [16]

«Beyond Space» ─ космічна аркада для Android, випущена в 2015 році. Головна особливість гри ─ 3D-графіка. Головна мета гри ─ знищити інші кораблі. Присутня можливість кастомізувати корабель (змінювати колір корабля). У грі присутні місії різного рівня складності ─ користувач переходить до від легкого режиму до складного.

Рисунок 1.15 – Скріншот гри «Beyond Space»

Плюси:

- барвиста 3D-графика;
- відбувається часткова зміна моделі корабля (кастомізація).

Недоліки:

- Гра вимагає потужних технічних характеристик;
- Гра неоптимізована (є проблеми з продуктивністю);
- Складність присутня, але користувач не має можливості вибрати певний рівень для кожної місії.

1.5.5 Чужий у космосі [17]

«Чужий в космосі» — аркадна гра, випущена в 2013 році. Хоробрий космічний десантник повинен відвезти вчених в безпечні бункери, поки до них не дісталися місцеві жуки переростки. Для виконання цієї не простої задачі гравцеві дозволено користуватися всілякою зброєю: бензопилою, ракетами, кулями.

Гра має більш продуману графіку (в порівнянні з першими трьома іграми), а також має систему поліпшення зброї (до моделі гравця додаються нові елементи, а також змінюється модель кулі).

Плюси:

- продумана 2D-графіка;
- Відбувається часткова зміна моделі корабля і зброї (кастомізація);
- спостерігається збільшення складності в міру просування.

Недоліки:

- Складність присутня, однак, користувач не має можливості встановити його на певному рівні;
- Немає можливості вибрати саундтрек у грі
- Ви не можете вибрати певні моделі ворога в грі.

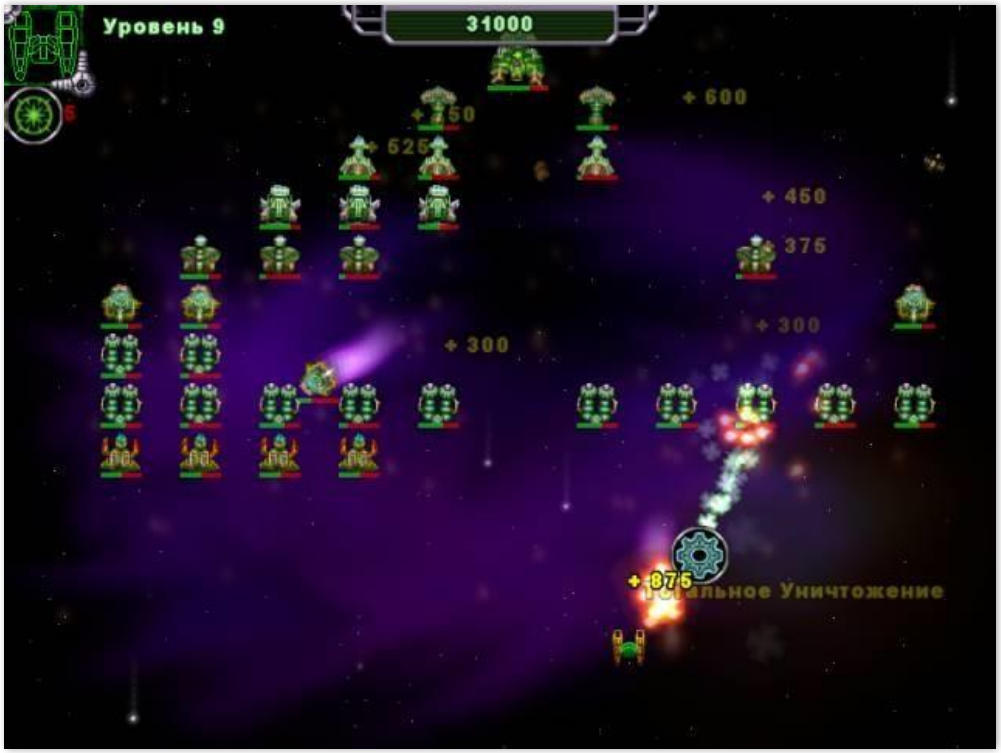

Рисунок 1.16 – Скріншот гри «Чужий в космосі»

1.5.6 Порівняння особливостей ігор жанру «Аркада»

За результатами огляду вищезгаданих ігрових рішень були виділені їх функціональні складові і винесені в таблицю 1.2. При аналізі розглянутих ігор були виділені наступні особливості жанру:.

- Аркади включають або набір місій, або нескінченний режим гри
- Головна мета більшості аркад - набрати найбільшу кількість балів
- Щоб збільшити рахунок, потрібно знищити модель противника за допомогою функції спуску кулі (атаки);
- Майже всі аркади мають показник здоров'я, який зменшується або шляхом досягнення ворожої моделі певної області, або коли сам гравець досягається.

Таблиця 1.2 – Порівняння особливостей ігор жанру «Аркада»

Назва гри	Графіка	Сюжет	Можливість змінювати моделі	Уміння змінювати поведінку ворогів
Space Invaders	2D	Нескінченний режим	-	-
Radiant	2D	Нескінченний режим	-	-\+ Презентувати підвищена складність
Luxor	2D	Набір місій	-	-/+ Спостерігається збільшення складності
Beyound Space	3D	Набір місій	-/+ Додавання нових модулів	-/+ Спостерігається збільшення складності
Чужий в космосі	2D	Нескінченний режим	-/+ Додавання нових модулів	-

Таблиця 1.2 показує, що більшість аркад не дозволяють користувачеві змінювати поведінку ворогів або явно встановлювати рівень складності, а також змінювати моделі в грі.

1.3 Постановка задачі

Метою роботи є розробка та програмна реалізація ігрової системи з гнучкими налаштуваннями в жанрі "Аркада" на платформі персонального комп'ютера під управлінням операційного середовища Windows. Жанр "Аркада" був обраний з метою розвитку моторики і реакції користувача. При цьому основними завданнями, що необхідно виконати для досягнення поставленої мети, є:

1. Розробка концептуальної схеми ігрової системи
   - Розробка діаграм UML.
   - Формування бізнес-сутностей та логіки системи.
2. Розробка гри з використанням стандартних об'єктів

3. Розробка механізмів зберігання і використання текстур, музика, моделей.
4. Розробка логічного контенту системи
   - Розробка системи складності ігрового процесу
   - Розробка стратегій розвитку ігрового персонажу
5. Програмна реалізація ігрової системи та перевірка її працездатності.

2 ПРОЕКТУВАННЯ ІГРОВОЇ СИСТЕМИ

2.1 Розробка концептуальної схеми

На цьому етапі продумуються функціональність і можливості. Формується жанр та інші особливості гри. Аркадний напрямок був обраний як жанр гри, так як вони досить поширені, не вимагають багато ігрового часу (геймплею) і дозволяють гравцям проводити час (ТаймКіллери). Таким чином, на даному етапі для ігрової системи були сформовані наступні твердження:

- Гра буде представляти собою Арконойд (жанр «Аркада»);
- У грі є гравець, вороги, зброя (кулі);
- Гра враховує систему балів і здоров'я;
- По досягненню нульового показника здоров'я - гра закінчиться;
- Мета гри полягає в тому, щоб набрати найбільшу кількість балів.

В аналізі інших аркад були виявлені наступні судження:

- Майже всі аркади мають адаптивне нарощування складності в міру просування, але немає очевидного завдання складності;
- Раніше розглянуті проекти не мають можливості змінювати логіку поведінки противника (напрямок руху);
- В деяких іграх присутня часткова зміна шаблонів гри, але частіше за все ці зміни незначні.
- Всі розглянуті екземпляри не мають можливості вибору звукового супроводу.

Таким чином, в якості особливостей проекту «Push Him All» будуть наступні функції:

1. У грі буде можливість змінити стандартні моделі на користувальницькі.

Таким чином користувач (гравець) зможе змінювати:

- Модель гравця
- Модель ворога
- Модель кулі
- Фон (BackGround).

2. У грі буде можливість змінити поведінку ворогів (напрямок руху).
3. У грі буде можливість змінити рівень складності.
4. У грі буде можливість змінити саундтрек і музичний супровід.

Для проектування програми використовувалася мова графічного опису об`єктного моделювання UML [18], оскільки вона дозволяє продумувати і описувати архітектуру проекту без програмного коду, при цьому не вимагає спеціальних знань для їх розуміння. В якості основного засобу опису проекту були обрані і побудовані діаграми використання, які показують функціональність майбутньої системи. Варіанти використання показують можливі взаємини між системою і користувачем. Вони відображають зовнішній інтерфейс системи і вказують форму того, що система повинна зробити. Дійова особа (актор) - це елемент, який не є системою, а взаємодіє з нею, через особливий інструмент - варіант В ході проектування був виділений наступний актор:

Гравець - користувач програми з повним доступом до функцій відеоігри. Може брати участь в грі, контролювати налаштування програми.

Загальна діаграма використання зображена на рисунку 2.1.

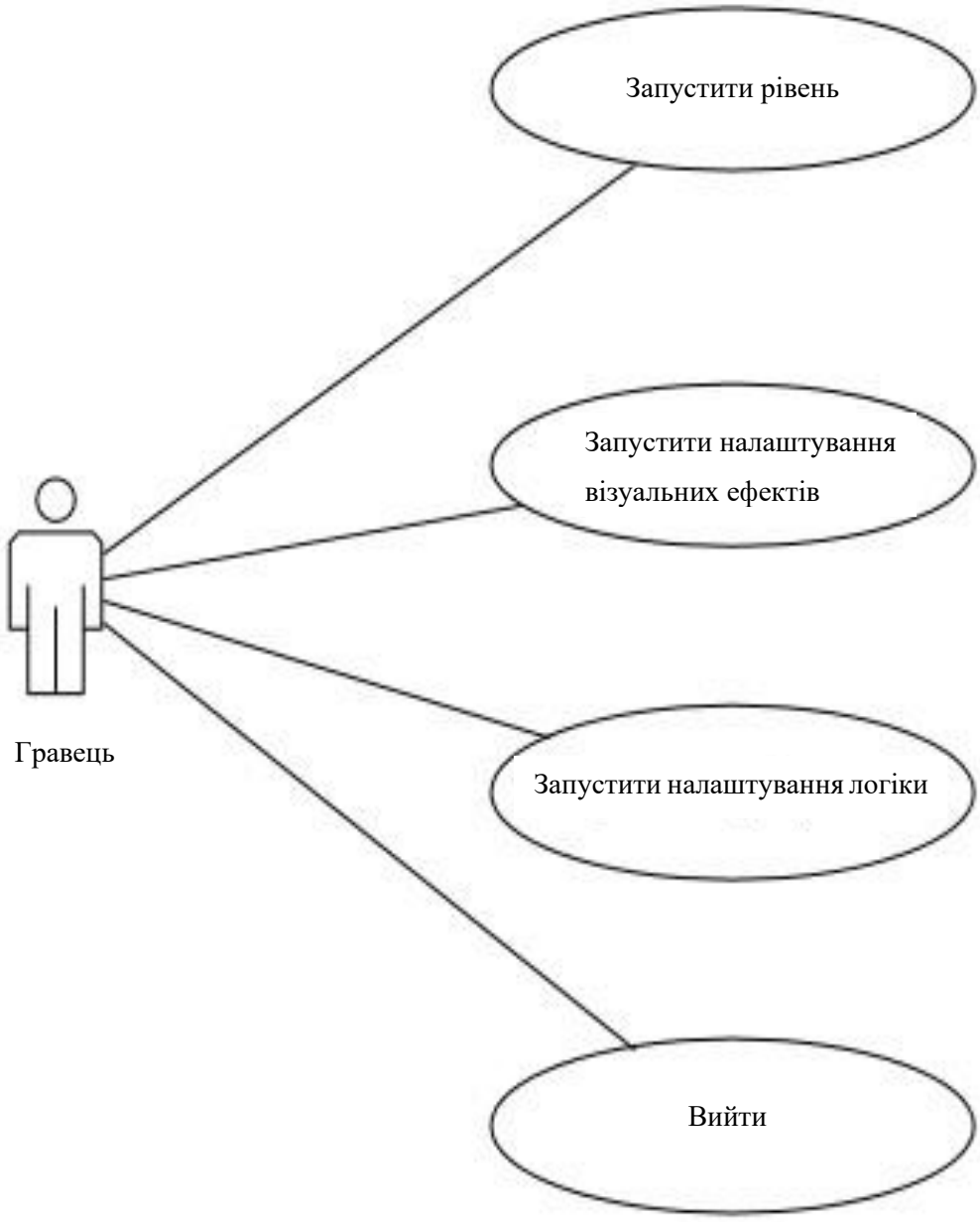

Рисунок 2.1 – Загальна UML діаграма користування

Виконаємо дослідження кожного варіанту використання більш детально. Для цього потрібно представити діаграми відношень варіантів використання. Коли користувач починає рівень, він може робити певні речі, які представлені на рисунку 2.2

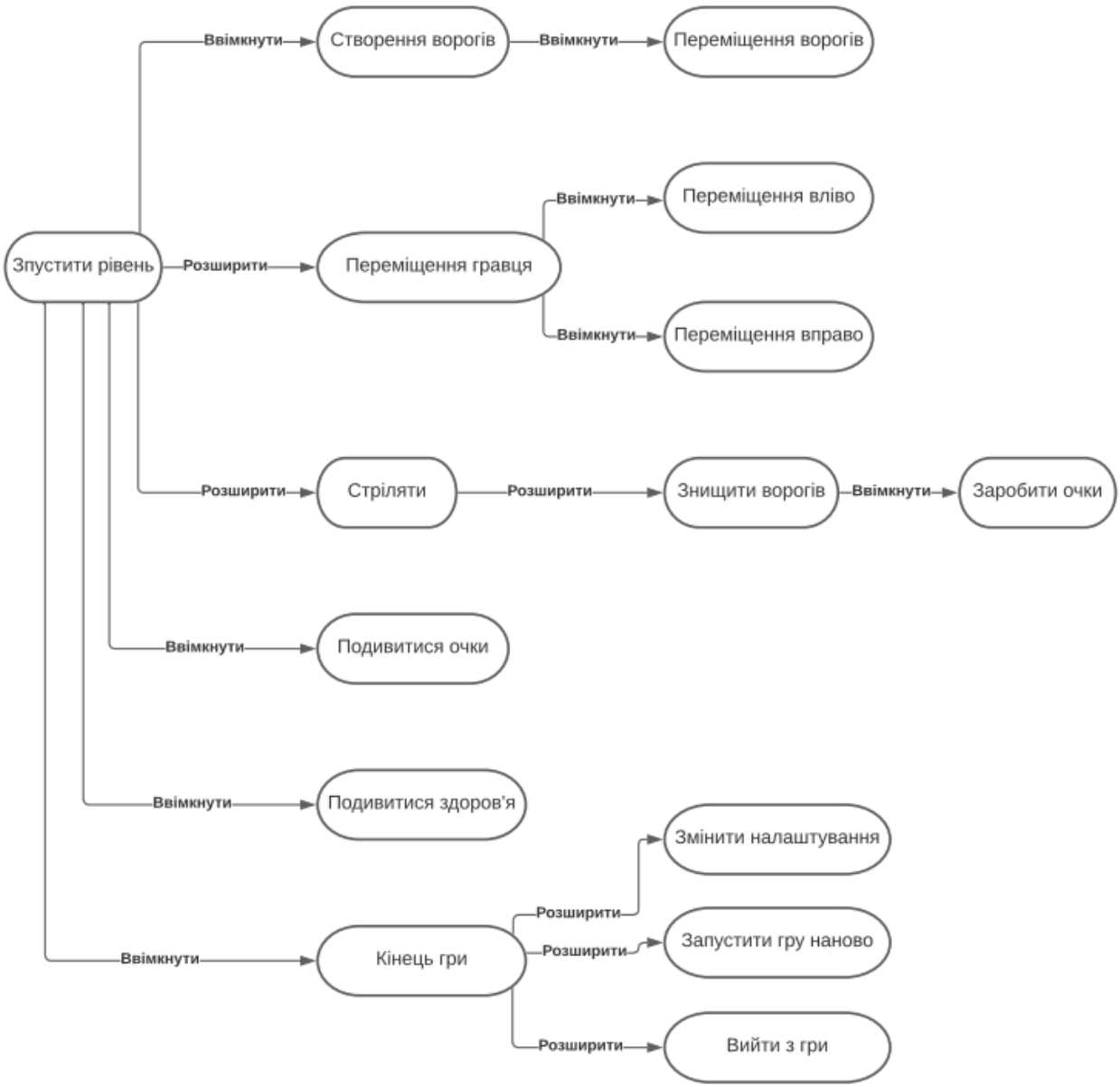

Рисунок 2.2 – Діаграма зв'язку варіантів використання дії "Запустити рівень"

Як інший варіант користувач може використовувати дію «Запустити налаштування візуальних ефектів». В даному варіанті використання у дійової особи (гравця) є можливість налаштувати візуальні компоненти додатку. При налаштуванні деяких компонентів необов'язково налаштовувати деякі функції, які позначені ставленням «розширити». Діаграма відносин варіанти використання «Запустити налаштування візуальних ефектів» представлена на рис. 2.3.

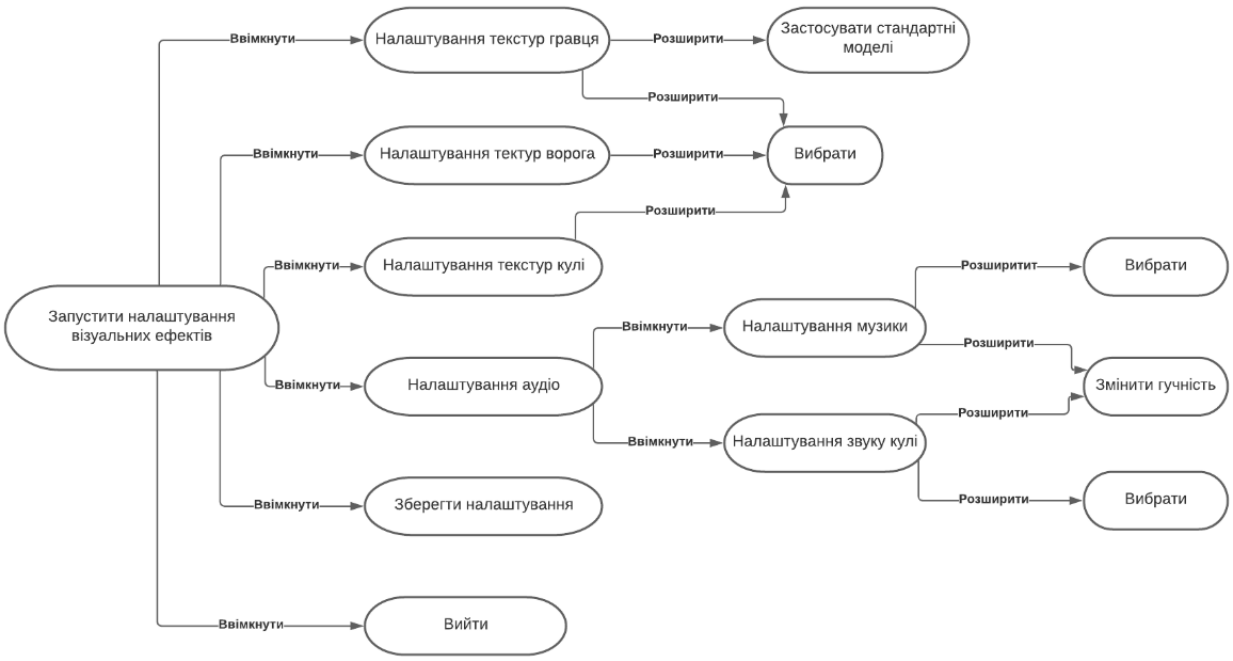

Рисунок 2.3 – Схема взаємозв'язку варіантів використання для дії "Запустити налаштування візуальних ефектів"

У варіанті використання «Запустити налаштування логіки», користувачеві надається можливість змінювати складність гри, а також налаштовувати траєкторію пересування ворогів і дію «Woopsy».

Woopsy ─ спеціальна функція, що викликає зазначену гравцем картинку по досягнення певної дії (по досягненню певної кількості очок або через деякий час, вказане користувачем). Діаграма відносин варіанти використання «Запустити налаштування логіки» представлена на рис. 2.4

Рисунок 2.4 - Діаграма зв'язку варіантів використання для дій «Запустити налаштування логіки»

В результаті проектування ігрової системи була висвітлена бізнес-сутність і логіка, представлена в таблиці 2.1.

Бізне сутність		Бізнес логіка	
1	Гравець	1	Калькулятор очок
2	Куля	2	Відслідковування положення кулі
3	Показник здоров'я	3	Відслідковування положення ворогів
4	Показник очок	4	Відслідковування колізій (перетин куль і ворогів)
5	Вороги	5	Інтерфейс для надання кількості очок
6	Текстури	6	Інтерфейс для надання кількості здоров'я
7	Музика	7	Інтерфейс головного меню
		8	Інтерфейс роботи з вибором текстур
		9	Інтерфейс налаштування логіки ворогів

Таблиця 2.1 – Бізнес-сутність і логіка ігрової системи

2.2 Розробка гри за допомогою стандартних об'єктів

2.2.1 Створення Scene и View

Клас QGraphicsScene забезпечує поверхню для управління великою кількістю графічних 2D-елементів. QGraphicsScene надає функціонал, що дозволяє визначити розташування цих елементів, а також дозволяє визначити видимість компонентів всередині довільної області сцени. Щоб додати якийсь елемент на сцену використовується метод addItem.

Клас QGraphicsView - це віджет для відображення вмісту " QGraphicsScene. Він також дозволяє відображати всю сцену або лише певну її частину. Щоб вказати сцену, яку буде відображати View, потрібно використовувати метод setScene. Щоб відобразити View використовується метод show ().

Клас QGraphicsRectItem дозволяє створити прямокутний елемент, який буде використовуватися в якості моделі. За допомогою методу setRect (*x*, *y*, *width*, *height*) встановлюється місцезнаходження і розмір прямокутника:

- *x* - координати прямокутника x;
- *y* – координата y прямоугольника;
- *width* - ширина прямокутника;
- *height* – висота прямокутника.

Взаємодію трьох наведених вище класів можна побачити на рисунку 2.5. Нижче представлений код створює сцену, вікно виведення (View) та прямокутника розміром 100 на 100. Результат роботи коду на рисунку 2.6.

```
QGraphicsScene * scene = new QGraphicsScene();
QGraphicsRectItem * rect = new QGraphicsRectItem();
rect->setRect(0,0,100,100);
scene->addItem(rect);
QGraphicsView * view = new QGraphicsView(scene);
view->show();
```

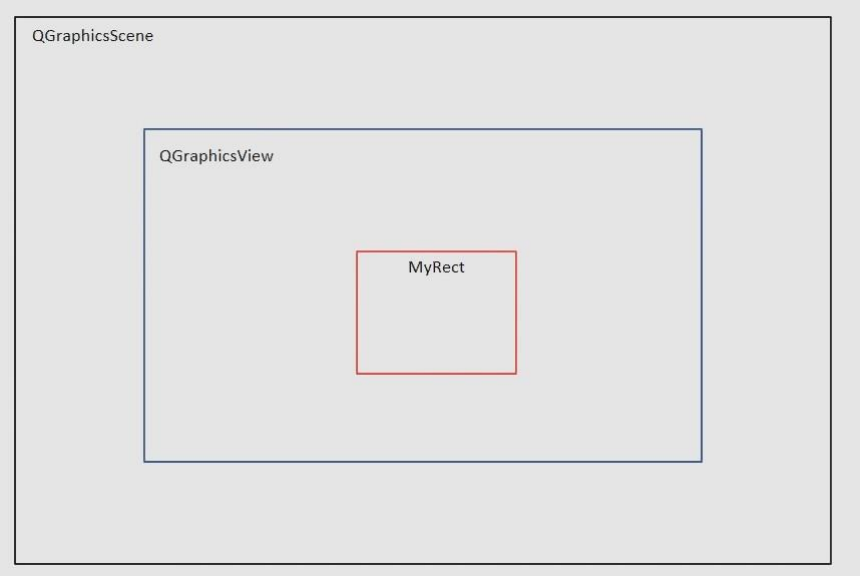

Рисунок 2.5 – Взаємодія Scene, View, RectItem

Рисунок 2.6 – Результат роботи коду

2.2.2 Створення моделі ворогів і куль

Система координат графічного подання зображена на рис. 2.7. Графічне представлення в бібліотеках є декартовою системою координат. Таким чином, для зміни положення елементів і їх геометрії потрібно використовувати два значення: координати *x* і координати *y*.

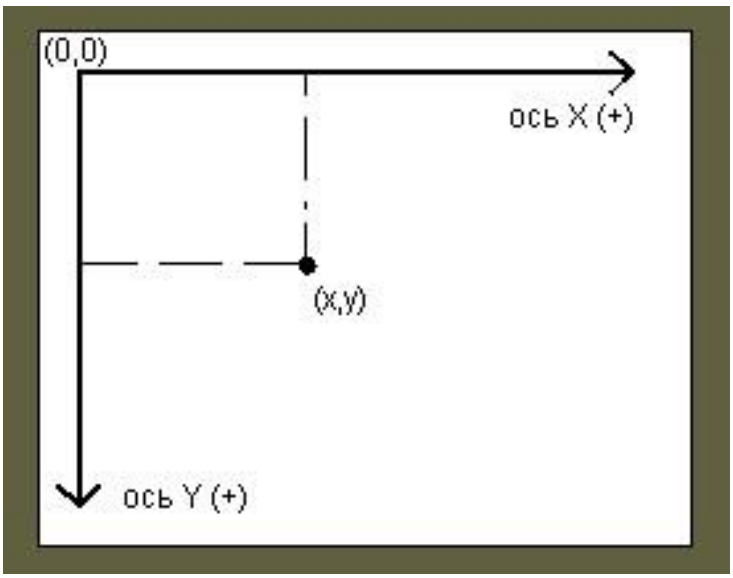

Рисунок 2.7 – Система координат графічного подання

В якості моделі ворога і кулі буде виступати QGraphicsRectItem, який є прямокутним елементом.

Вороги повинні з'явитися у верхній частині ігрового екрана. Для визначення позиції моделі противника (з’явитися зліва, справа або посередині) використовується функція rand(), яка генерує розташування моделі противника перед її появою. Якщо модель ворога буде пересуватися по косій траєкторії, слід враховувати місце розташування її появи (верхній лівий / правий кут).

За допомогою функцію setPos (double x, double y) - встановлюється позиція елемента, де:

- *x* – координата *x* встановлюваного елемента;
- *y* – координата *y* встановлюваного елемента.

2.2.3 Обробка колізій

При взаємодії (перетині) моделі противника і кулі, обидві моделі повинні бути видалені зі сцени і пам'яті. Для цього використовується спеціальний масив, який включає в себе моделі, представлені на сцені, а також спеціально створену функцію, яка перевіряє перетин моделей певних фігур. При перетині моделей фігури - функція видаляє елементи зі сцени і звертається до певних (пересічених) моделей масиву і видаляє їх. Таким чином пересічені моделі втрачають видимість (видаляються зі сцени) и видаляються з пам’яті.

Рисунок 2.8 - Алгоритм обробки при перетині моделі противника та кулі

Як приклад на рисунку 2.8 подано алгоритм логіки роботи при перетині моделі противника і кулі. Коли моделі ворога і гравця перетинаються, з місця події слід видалити тільки модель ворога, але здоров'я гравця слід зменшити.

2.2.4 Створення таймерів

Мета гри полягає в тому, щоб знищити нових ворогів з допомогою куль. Для створення ворогів використовується таймер - QTimer. Класс QTimer надає повторюваний і одноразовий таймер.

QTimer – це програмний інтерфейс високого рівня для таймерів. Для роботи з ним потрібно:

- Створення елемента QTimer
- Підключіть його сигнал тайм-ауту до відповідного слота (виклик функції по переповненню таймера);
- запустити метод start ().

З метою переміщення моделей слід створити таймер, при переповненні якого, положення кулі і ворога буде змінюватися (ворог буде рухатися до нижньої частини екрану, а куля - вверхньої). Нижче наданий приклад створення таймеру для пересування ворога.

1) Створення елементу QTimer з назвою «timer»:

```
QTimer * timer = new QTimer();
```

2) Підключення таймера до гнізда (функції) «move»:

```
connect(timer,SIGNAL(timeout()),this,SLOT(move()));
```

3) Увімкнення таймеру методом «start(msec)», де msec - кількість мілісекунд до переповнення таймеру:

```
timer->start(msec);
```

2.3 Завантаження текстур, музики, моделей

На цьому етапі всі прості об'єкти QGraphicsRectItem (прямокутники) замінюються графічними моделями. В ігровій системі розроблено такі типи моделей:

- Спрайти моделей;
- Музика та звуки
- Відео.

Файлова структура додатку представлена на рисунку 2.9

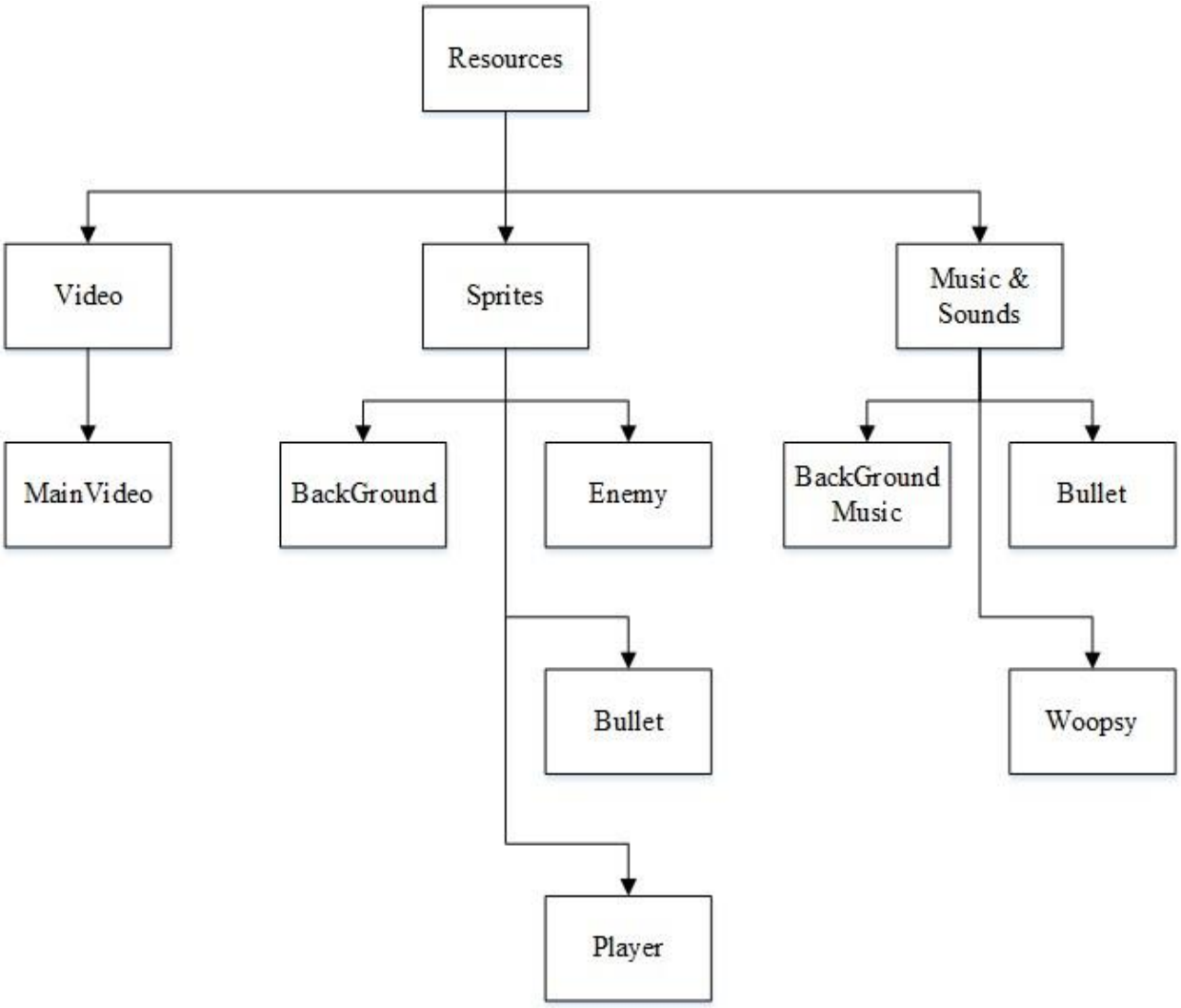

Рисунок 2.9 – Файлова структура ігрової системи

1) Sprites

- Enemy - стандартна модель ворога. Являє собою 2D спрайт космічного

корабля з корпусом червоного кольору.

- Bullet - стандартна модель кулі, випускається гравцем. Як спрайт кулі була обрана ракета.
- Player - стандартна модель гравця (користувача). Представляється космічним кораблем з корпусом зеленого кольору.
- BackGround - це задній фон гри. Як фон був обраний малюнок космосу Основні моделі гри показані на рисунку 2.10.

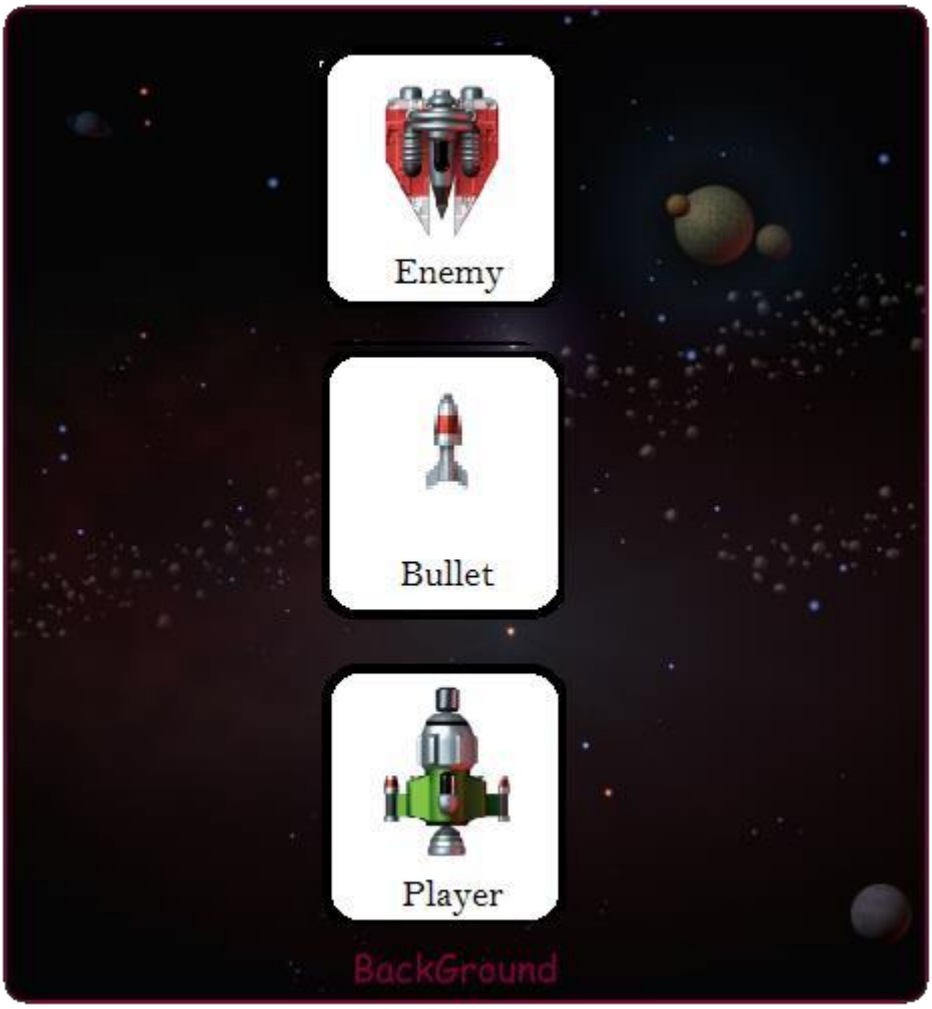

Рисунок 2.10 – Основні моделі ігрової системи

2) Music & Sounds

- Bullet - це звук, який з'являється в результаті запуску гравцем кулі (в стандартній моделі - ракета).
- BackGroundMusic - це музика, яка відтворюється як фон.
- Woopsy - звук видаваний при включенні функції «Woopsy»

3) Video

- MainVideo — Заставка у головному вікні програми. Є анімацією корабля, що летить до красного карлика.

2.4 Розробка логічного компонента

2.4.1 Визначення рівнів складності

Ігрова система вимагає розроблення різних рівнів складності.

- Стандартна складність

  Складність, при якій у користувача немає труднощів по ходу гри. Цей тип складності є особливим випадком заданої складності.

- Задана складність

  Користувач може встановити рівень складності показником від 1 до 10, при цьому 10 рівень складності є найскладнішим для проходження, а 1 найпростішим.

- Адаптивна складність

  Цей рівень складності буде змінюватися в залежності від успіху гравця в процесі гри. Таким чином складність буде змінювати своє значення від самого простого до самого складного і навпаки.

Як приклад, рисунок 2.11 представляє алгоритм роботи адаптивної складності.

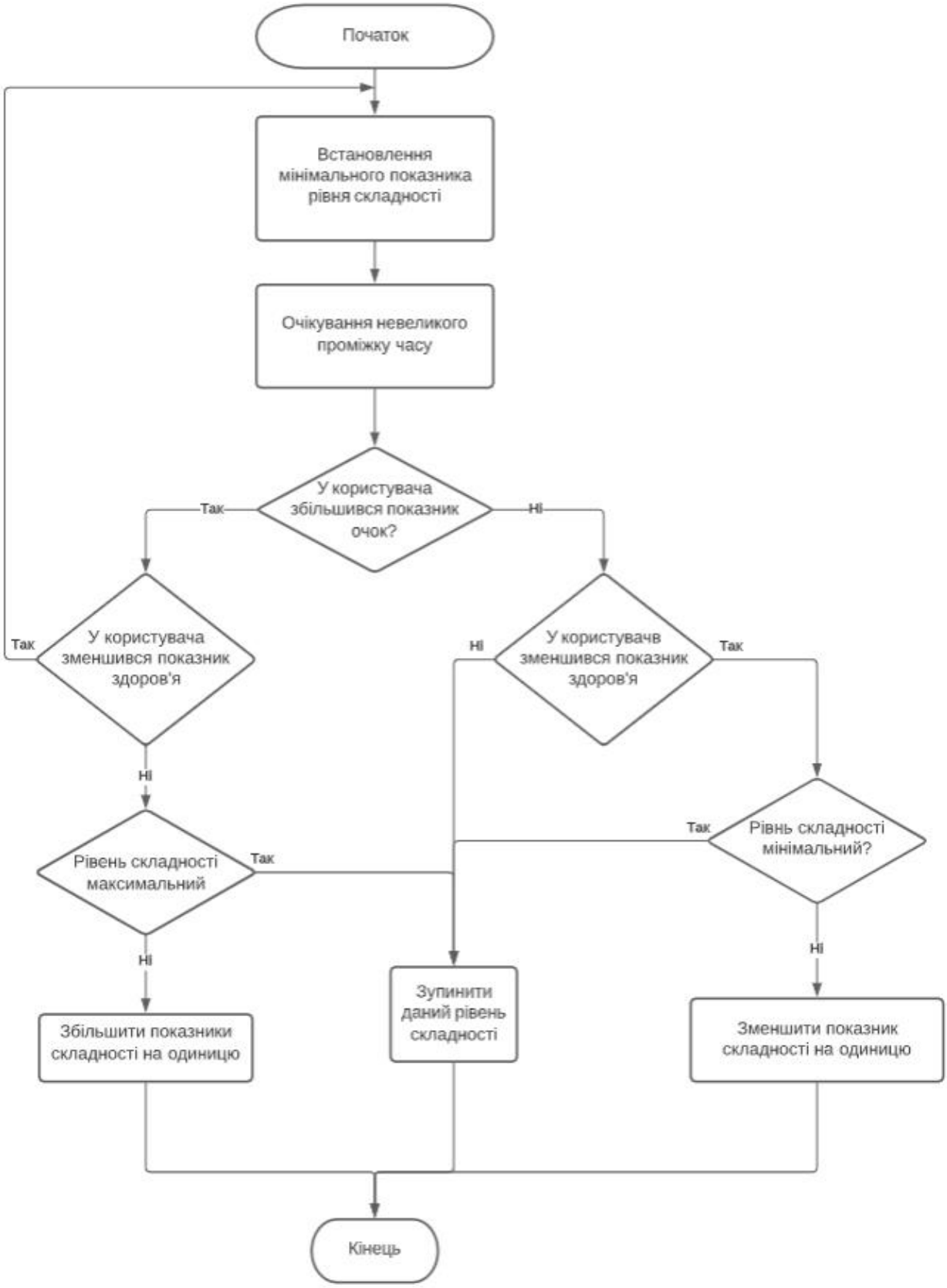

Рисунок 2.11 - Алгоритм адаптивної складності

2.4.2 Розробка типів руху

Існує три типи переміщення об'єктів противника (Рис. 2.12):

- Рух по прямій траєкторії;
- Рух по косій траєкторії;
- Рух відповідно до гармонійного закону.

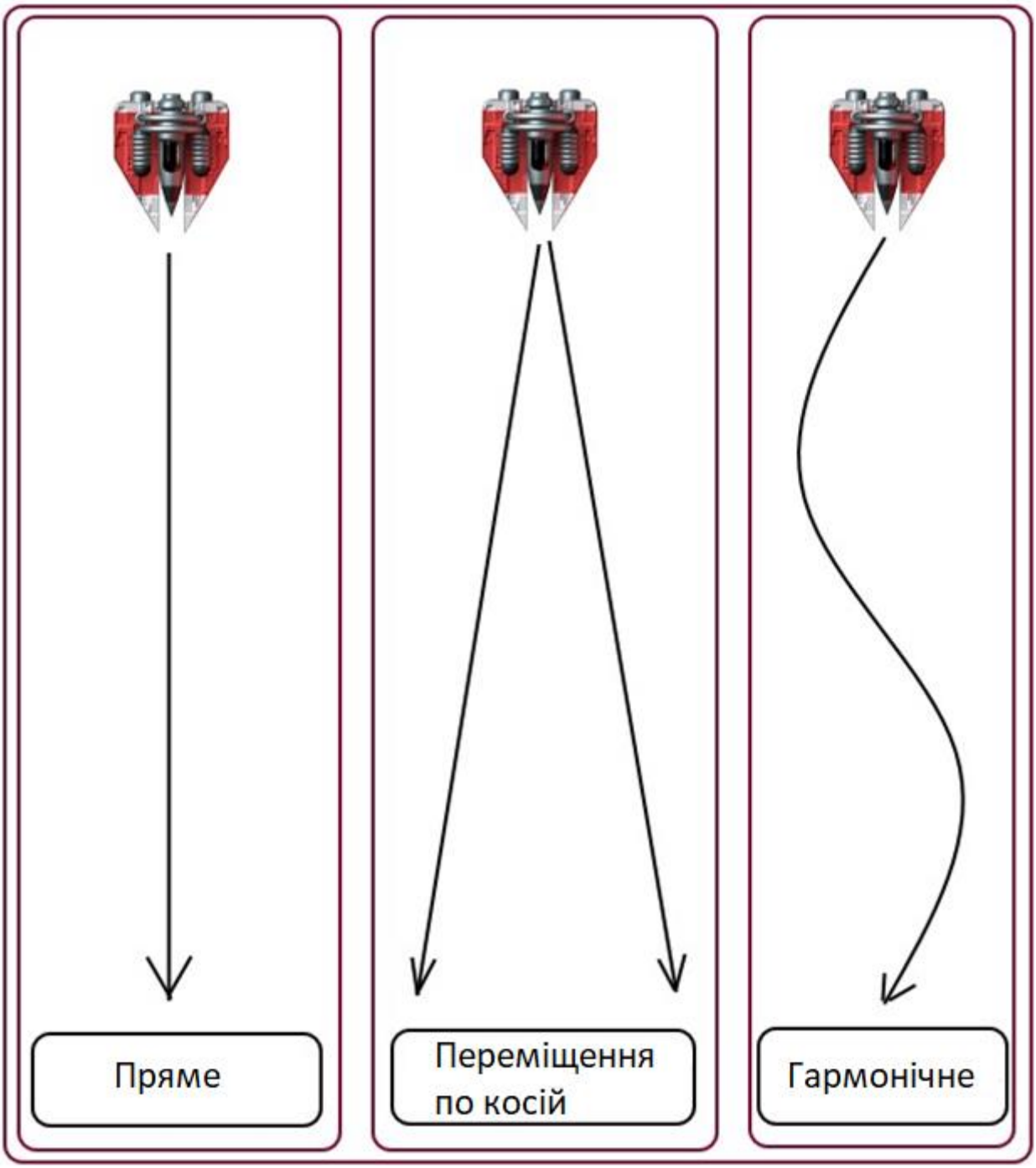

Рисунок 2.12 - Види переміщення об'єктів противника

1) Переміщення об'єктів по прямій лінії

Для переміщення об'єкта по прямій траєкторії слід зміщувати його по координаті Y через особливості системи координат графічного подання.

Таким чином, щоб переміщувати ворожу модель з верхньої позиції екрану в нижню, використовуємо наступну функцію:

```
SetPos (x(),y() + offset);
```

де offset — це величина зсуву відносно координати Y

2) Переміщення об'єктів по косій траєкторії

Для того щоб перемістити об'єкт по косій траєкторії, його слід змістити по координаті Y і X. При цьому значення зміщення Y має перевищувати значення зміщення X в кілька разів. Так само слід враховувати місце розташування появи ворожого об'єкту. Якщо об'єкт з'явився в лівій верхній частині екрану, то його кінцевий пункт призначення нижня права межа і навпаки. Таким чином, функція пересування прийме наступний вигляд:

```
SetPos (x() ± offsetX, y() + offsetY);
```

де offsetX (Y) є зсувом за координатою X (Y).

3) Переміщення об'єктів шляхом гармонійного коливання

Для того, щоб об'єкт переміщався відповідно до гармонічного закону, ми будемо використовувати функцію тригонометрії - синус (рис. 2.13). З малюнка видно, що для збільшення зміщення по відношенню до координати X - необхідно змінити значення параметра A(амплітуду коливань). Для збільшення переміщення щодо координати Y - потрібно змінити параметр частоти w (циклічна частота). Параметром початкової фази (f0) знехтуємо (встановимо значення в 0).

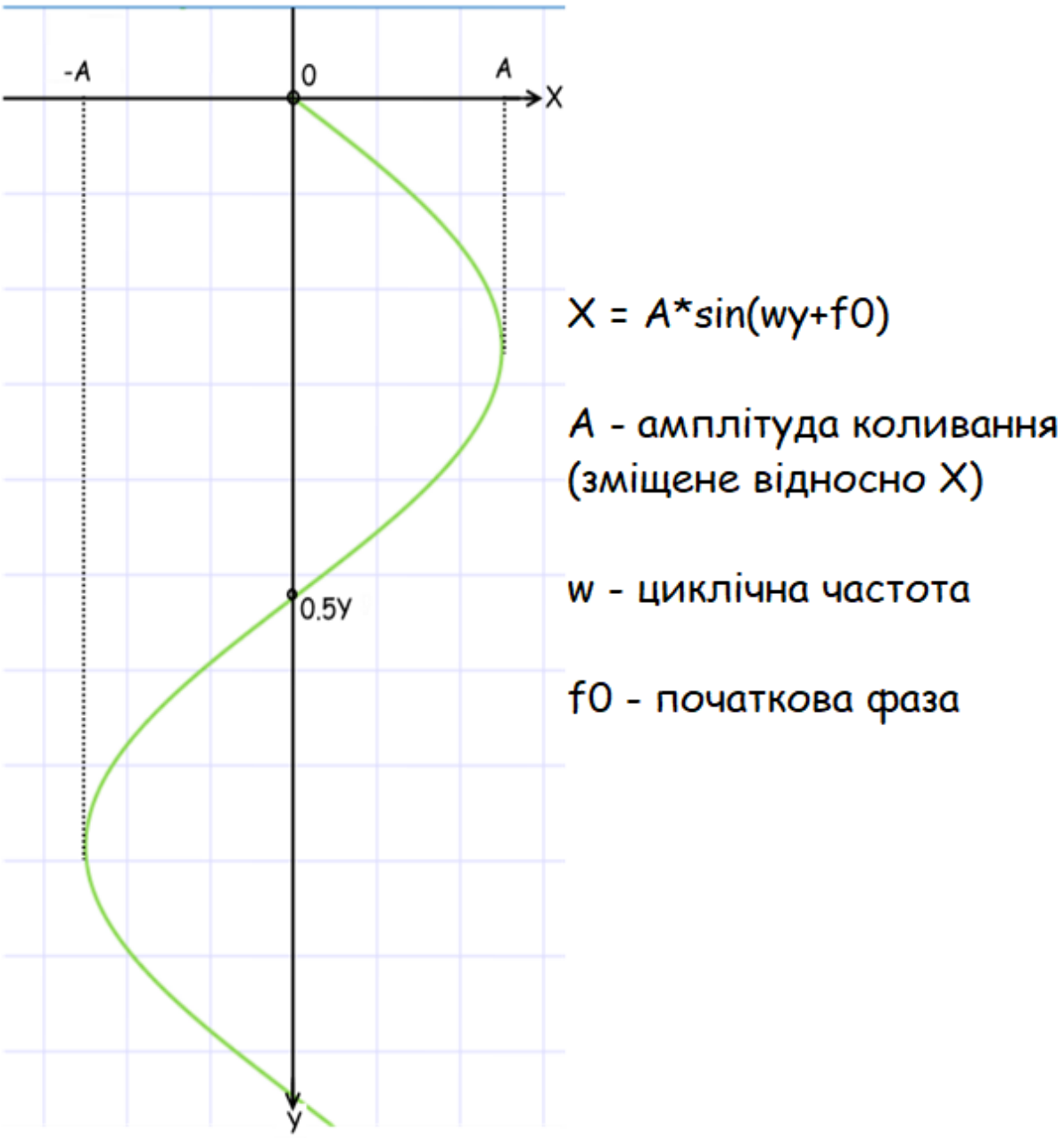

Рисунок 2.13 - Функція тригонометрії синуса

Тоді функція руху матиме наступну форму:

```
SetPos (x() ± A*sin(y()*w), y() + offsetY);
```

де A — зміщення відносно координати X; w - циклічний параметр частоти; offsetY - зміщення за координатою Y.

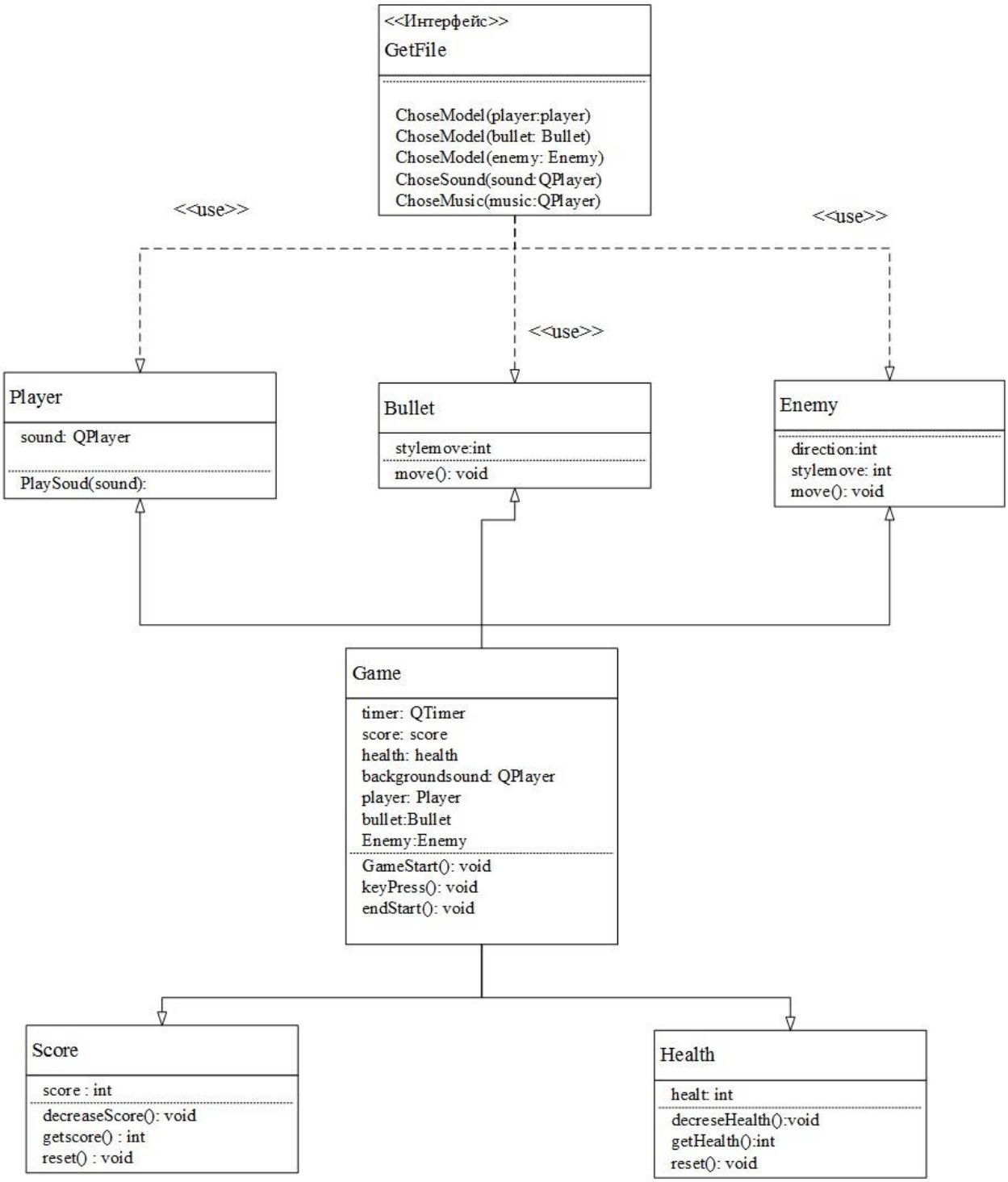

Рисунок 2.14 - Діаграма класів програмного ігрової системи

На рисунку 2.14 показана схема класів ігрової системи, на якій показано наступні класи: є основні класи – ворог («Enemy»), гравець («Player»), кулі («Enemy»), індикатор здоров'я («Health») та рахунку («Score»). За допомогою інтерфейсу взаємодії з користувачем («GetFile»)

відбувається вибірка текстур деяких компонентів, а також, за бажанням користувача, звукової складової. Після чого всі отримані дані (якщо дані не отримані - використовуються стандартні компоненти) використовуються в класі «Game», який виконує ініціалізацію всіх компонентів гри, запускає її, а також стежить за зовнішніми сигналами від користувача (натискання клавіш).

3 ІНФОРМАЦІЙНЕ ТА ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ

3.1 Загальний опис

Розроблена ігрова система - це гра, в якій гравець повинен вижити, знищити більше ворогів, як це можливо і заробити якомога більше балів.

При запуску ігрової системи користувач бачить головне меню програми, представлене на рисунку 3.1. Взаємодія з елементами меню здійснюється за допомогою комп'ютерної миші - наведіть курсор на цікавий вам елемент і натисніть на ліву кнопку миші.

Основні пункти меню:

- Грати
- Налаштування текстур
- Налаштування логіки
- Вийти.

При натисканні користувача на кнопку «Грати» з'являється вікно, в якому проводиться основний ігровий процес (рис. 3.2).

Рисунок 3.1 - Головне меню

Рисунок 3.2 - Головне вікно ігрової системи

Ключові елементи керування:

- клавіша «Вліво» - переміщення моделі гравця вліво;
- клавіша «Вправо» - переміщення моделі гравця вправо;
- клавіша «Пробіл» - це випуск кулі моделлю гравця.

Угорі ліворуч на екранах відображаються три показники.

- “Score” - рахунок гравця (показник балів);
- “BestScore” - найкраща оцінка;
- “ Health ” - показник здоров'я (спочатку цей показник дорівнює 3).

В кінці гри, тобто для досягнення нульового здоров'я (Health = 0) гравцеві відображається вікно рахунку (рисунок 3.3), в якому гравець може вибрати наступні дії.

- «Так!» - почати гру знову, при цьому всі налаштування гри збережені;
- "Налаштування" - зайдіть у вікно налаштувань графічних компонентів;
- «Логіка»- зайдіть у вікно налаштувань логіки;
- "Вихід" - закриття гри.

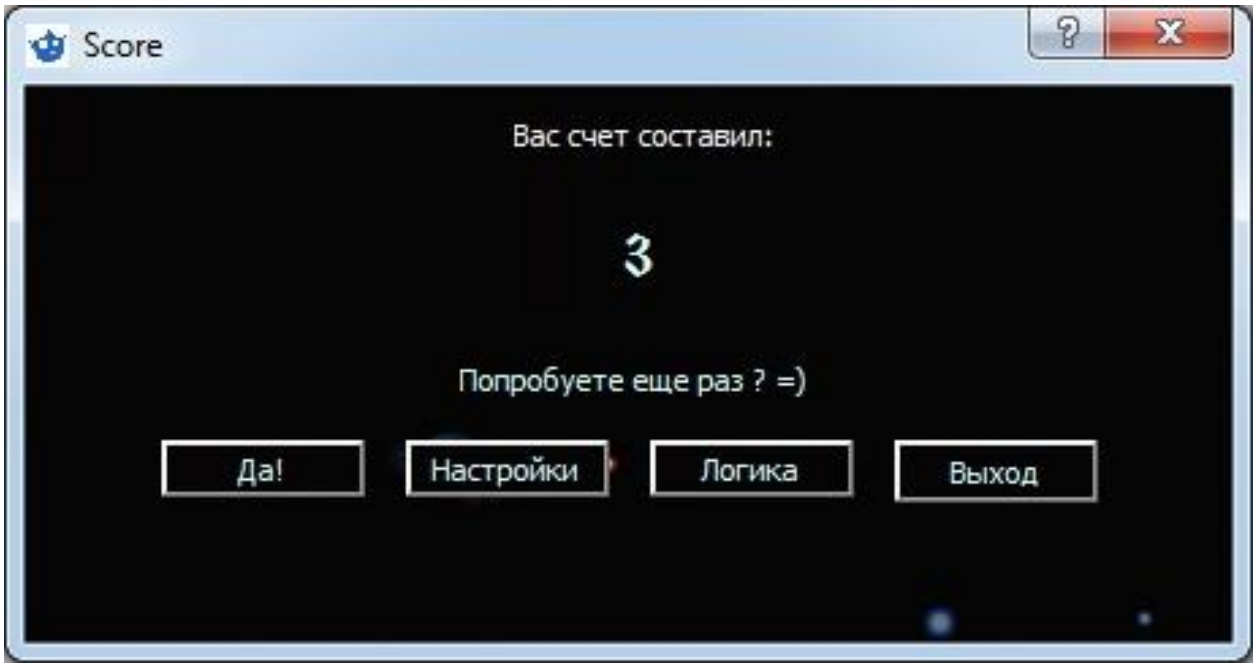

Рисунок 3.3 - Вікно рахунку

При виборі пункту «Налаштування» користувачу відобразить вікно, показане на рисунку 3.4. В даному вікні гравець може налаштувати:

- Текстура моделі гравця
- Текстура моделі кулі
- Текстура ворожої моделі
- Фон гри
- Звук кулі (а також її гучність);
- Музика гри (а також її гучність);

Гравець також має можливість зберегти вибрані налаштування за допомогою кнопки «Зберегти» і завантажити їх за допомогою кнопки «Завантажити» в майбутньому.

В кінці всіх налаштувань, гравець повинен натиснути кнопку «Застосувати» в правому нижньому куті, щоб зміни вступили в силу.

При виборі пункту «Логіка» користувачеві відображається вікно, представлене на рисунку 3.5. В даному вікні гравець може налаштувати:

1) Функцію «Woopsy»

Ця функція відображає зображення у нижньому правому куті і відтворює звуковий сигнал, який можна вказати за допомогою кнопок «Текстура–Обрати» і «Звук–Обрати». Функція «Woopsy» виконується за певної умови, яку гравець може встановити:

- якщо ви виберете «Кількість балів», функція активується, досягаючи кількості пунктів, перерахованих поруч (праворуч) від даного налаштування.
- якщо ви виберете «Кількість секунд», функція активується протягом певної кількості секунд після початку гри.

Спочатку(у стандартних параметрах) цю можливість вимкнено.

2) Складність

Гравець може вибрати один з трьох рівнів складності:

- Звичайна складність - оптимальний рівень складності, не змінюється під час гри.
- Адаптивна складність - на початку гри рівень складності є стандартним, але в залежності від гри користувача, рівень складності може збільшуватися або зменшуватися.
- Задана складність - користувач може вибрати рівень складності від 1 до 10, а рівень складності не зміниться протягом всієї гри.

3) Зміна траєкторії руху ворогів

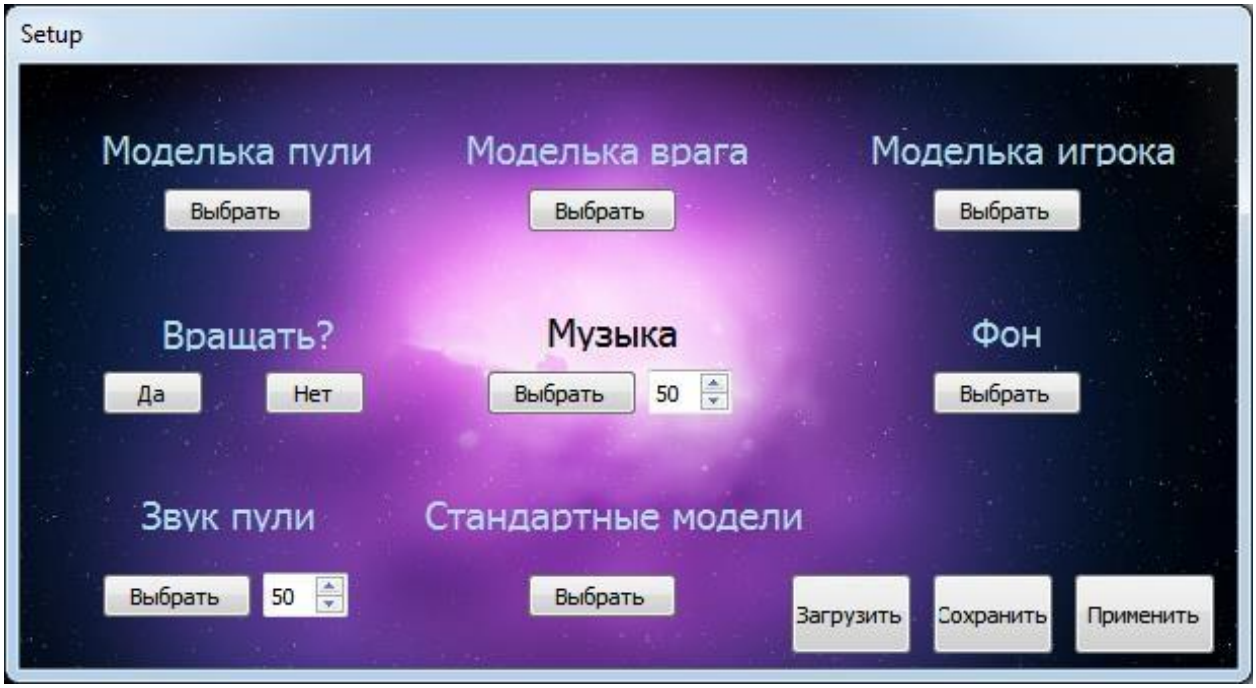

Рисунок 3.4 - Меню налаштувань

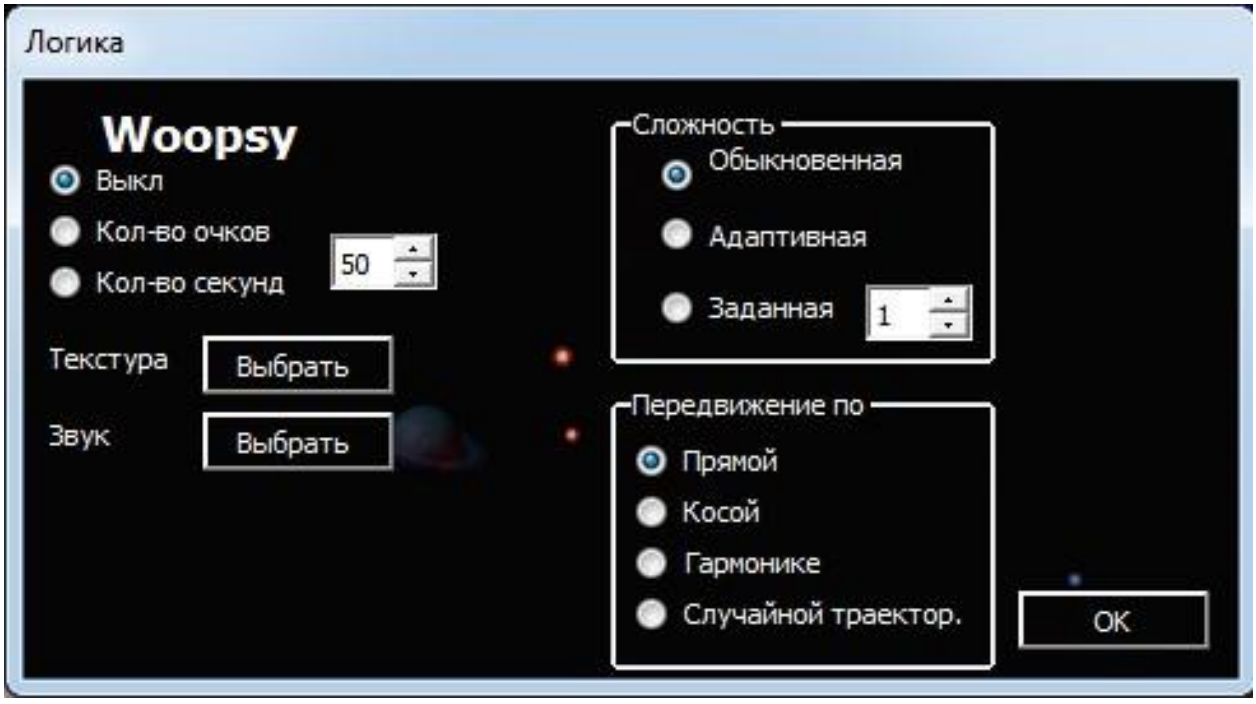

Рисунок 3.5 - Меню параметрів логіки

3.2 Налаштування моделей

В ігровій системі реалізовано такі види налаштувань (Рис. 3.6):

- Замініть модель гравця.
- Замініть модель ворога.

- Замініть модель кулі.
- Замініть фон гри;
- Збережемо налаштування.

Рисунок 3.6 - Перехід у вікно налаштувань

Щоб замінити модель кулі, потрібно натиснути відповідну кнопку "Обрати" (рис. 3.7). Потім користувачеві відкривається нове вікно, де вони повинні вказати потрібну їм модель (рис. 3.8). В якості моделі кулі вкажемо файл "R2d2".

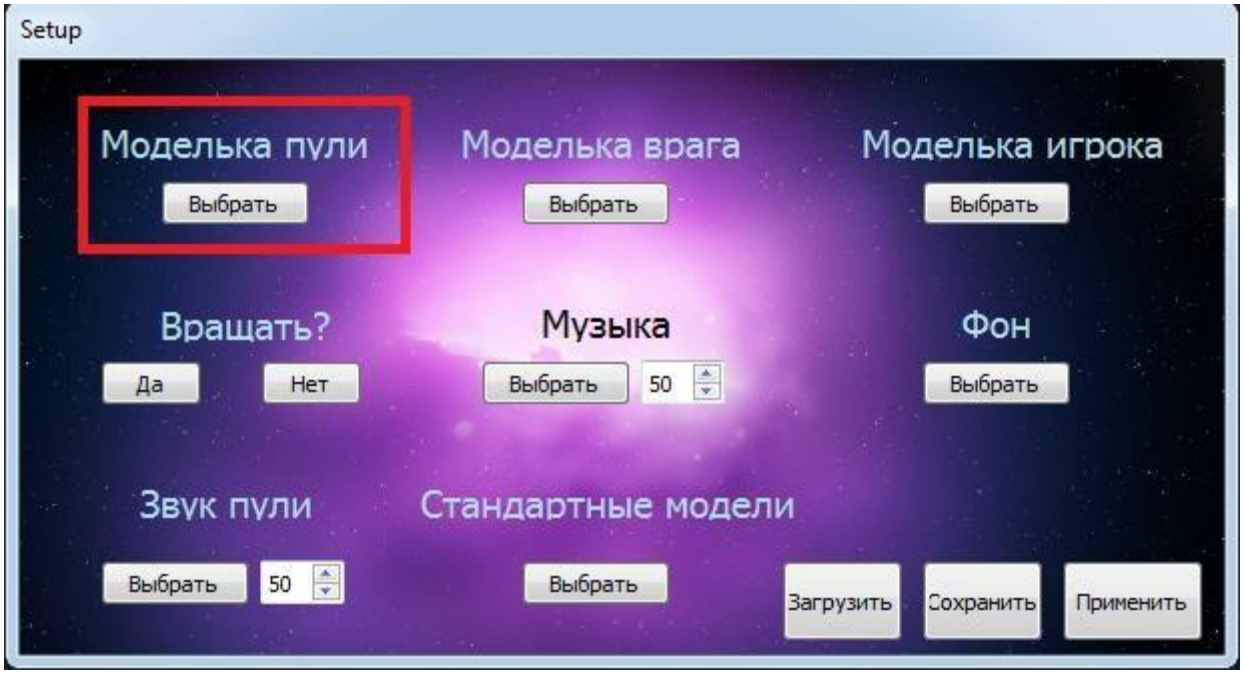

Рисунок 3.7 - Вікно налаштувань

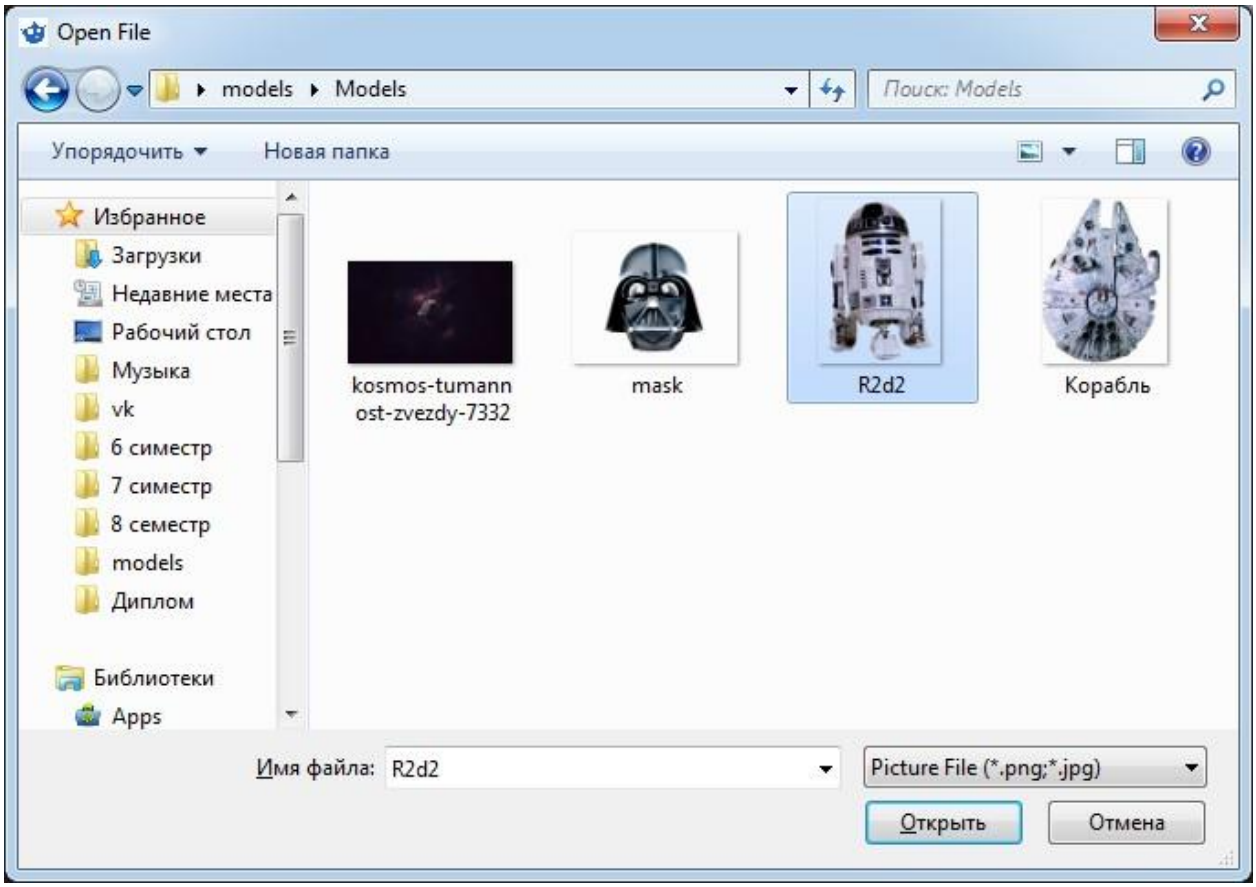

Рисунок 3.8 - Вікно вибору моделі

Аналогічні дії ми зробимо і для інших моделей.

- Як ворожу модель вкажемо файл «mask».
- Як модель гравця, ми вкажемо файл «Корабль».
- Як фон, давайте вкажемо файл «kosmos».

Збережемо налаштування за допомогою кнопки «Зберегти» (рис. 3.9).

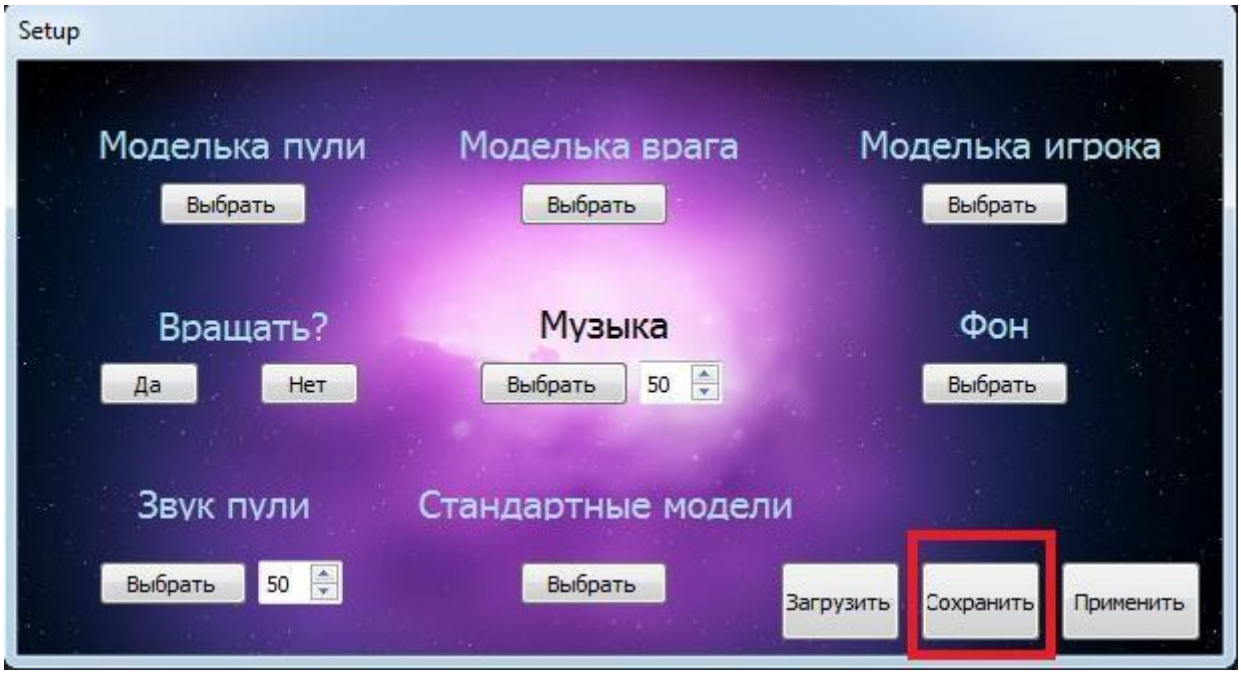

Рисунок 3.9 - Збереження параметрів

Для того, щоб зміни набудуть чинності, потрібно натиснути кнопку "Застосувати" (рис. 3.10). Після цього гравець побачить головне вікно програми. Тепер почнемо гру з кнопки «Грати» і подивимося на результат (рис. 3.11). Як видно з рисунку 3.11, всі налаштування були успішно застосовані. Щоб завантажити попередньо збережені налаштування, потрібно перейти в меню «Налаштування» і натиснути на кнопку «Завантажити».

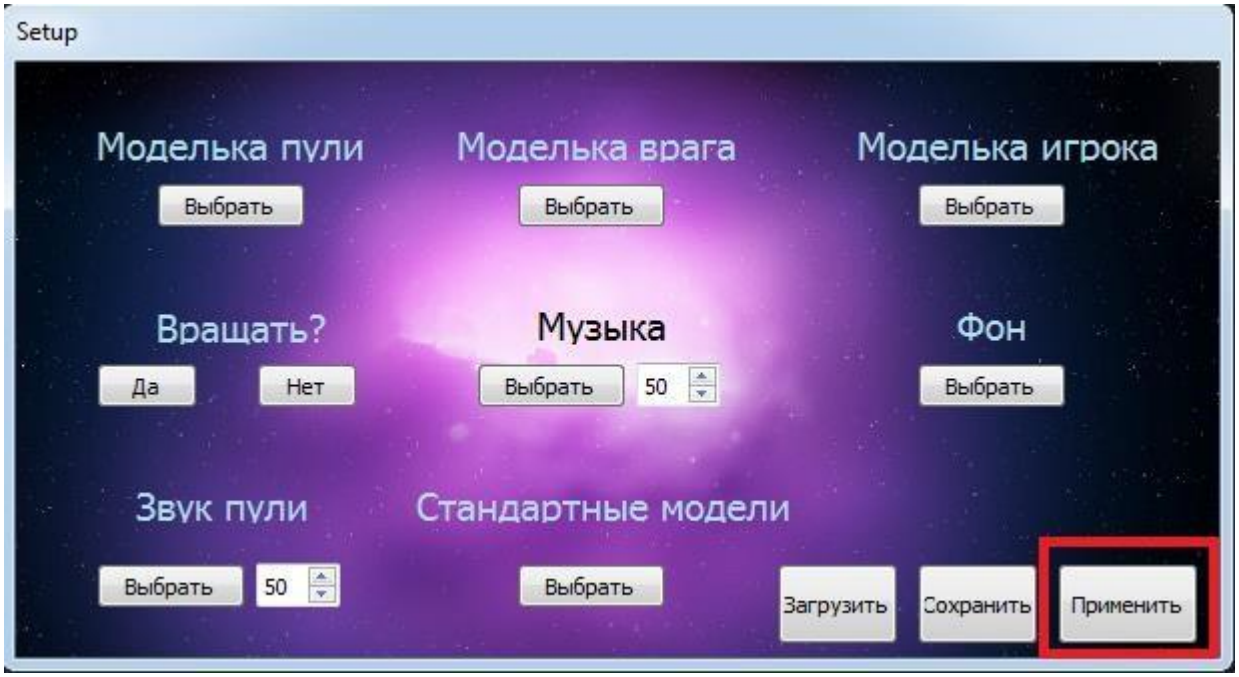

Рисунок 3.10 - Застосування налаштувань

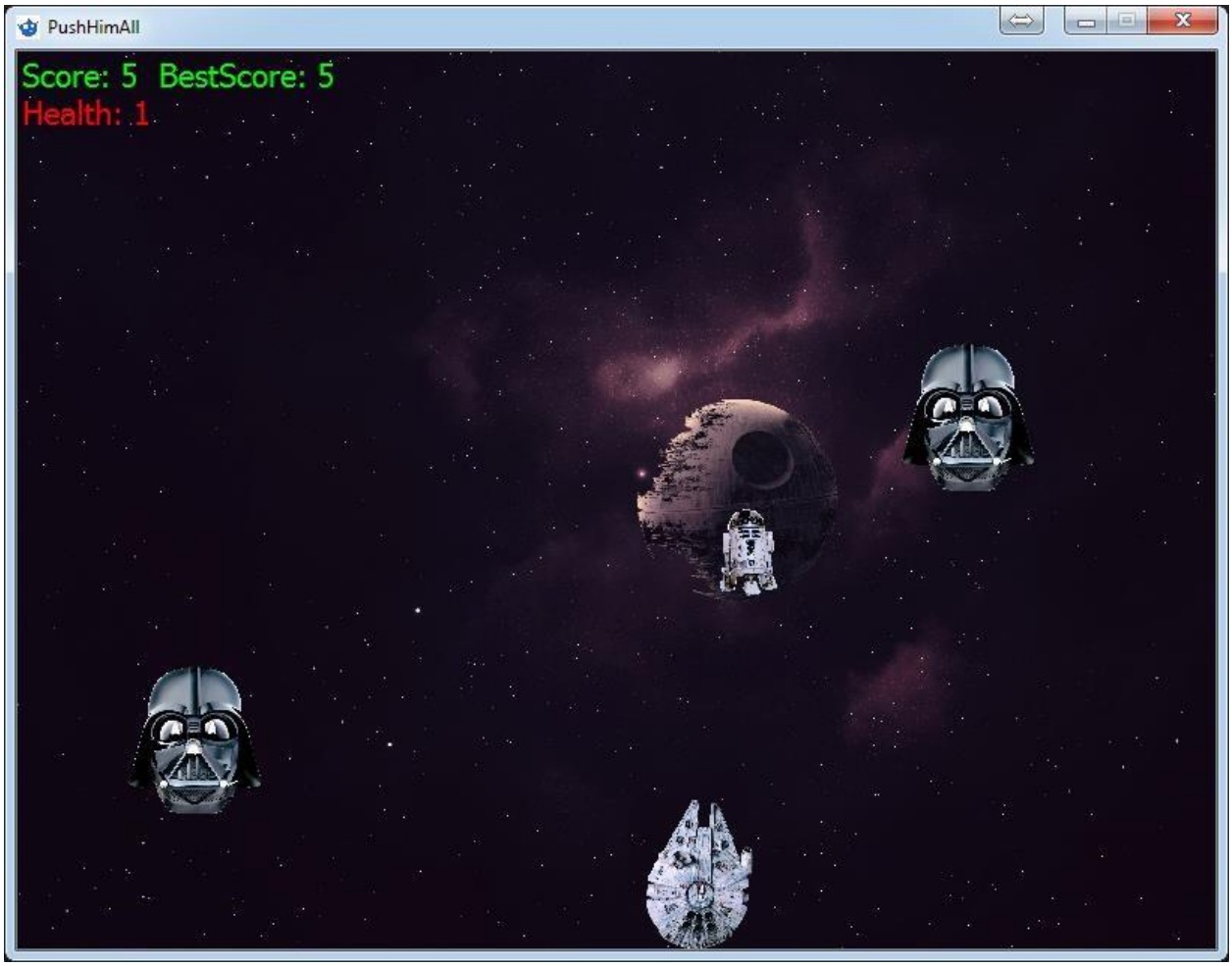

Рисунок 3.11 - Результат застосування налаштувань

Користувач також може вибрати опцію «Обертати». Ця функція дозволяє обертати стандартну або визначену користувачем (гравцем) модель кулі. Для повороту моделі кулі потрібно вибрати відповідний пункт в меню налаштувань (рис. 3.12). «Так» – активувати функцію обертання. «Ні» - вимикає цю функцію (модель кулі не обертається).

Користувачеві надається можливість вказати музику в грі з пунктом "Музика" і вказати її гучність, а також звук (і його гучність) кулі з допомогою пункту «Звук кули».

Щоб грати зі стандартними моделями, потрібно натиснути кнопку

«Обрати» в пункті «Стандартні моделі». Після цього всі налаштування користувача видаляться, і гра візьме стандартні значення.

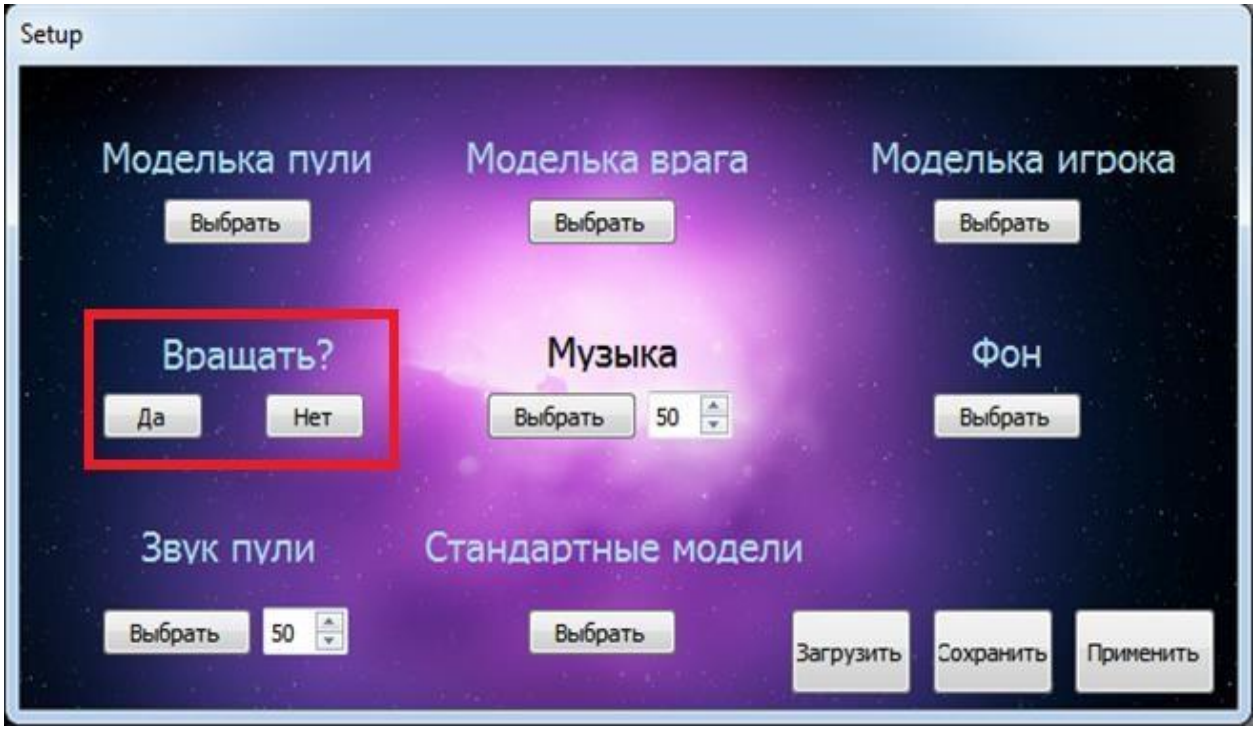

Рисунок 3.12 - Параметр обертання кулі в меню налаштувань

3.3 Налаштування логіки

Ігрова система також дозволяє змінювати поведінку ворогів, зокрема:

- Зміна складності гри
- Зміна траєкторії руху ворогів

Для того, щоб за встановити ці параметри, потрібно переключитися з головного меню ігрового додатку в меню «Логіка» або в кінці гри, натиснувши «Логіка» у вікні рахунку.

Користувач (гравець) має можливість встановлювати траєкторію польоту ворожих одиниць. Спочатку всі вороги рухаються по прямій траєкторії, однак, користувач може змінити його на наступні варіанти:

- переміщення по прямій

Ворожа одиниця створюється у верхній частині ігрового вікна і починає свій рух до нижньої частини екрана по прямій траєкторії.

- переміщення по косій

Залежно від позиції створення ворожої одиниці, вона почне рухатися в лівий нижній або правий кут ігрового вікна.

- переміщення по гармоніці

У цьому варіанті ворожий підрозділ буде змінювати своє положення на всьому шляху по гармонійному закону. В даному випадку поцілити в мішень стане значно складніше, тим самим тренуються прогностичні здібності гравця.

- випадкова траєкторія

У всіх ворожих одиниці будуть мати свої закони руху, які перераховані вище (по прямій, косій, гармонійній). Цей режим є найскладнішим, оскільки вам потрібно передбачити траєкторію кожної ворожої одиниці водночас.

ВИСНОВКИ

В випускній роботі було проведено розробку та програмну реалізацію ігрової системи з гнучкими налаштуваннями в жанрі "Аркада" на платформі персонального комп'ютера під управлінням операційного середовища Windows. При цьому було виконано такі основні завдання:

1. Розроблено концептуальну схему ігрової системи, в тому числі UML діаграми та бізнес-сутності та логіка системи.

2. Розроблено ігрові механізми з використанням стандартних об'єктів

3. Розроблено механізми зберігання і використання текстур, музика, моделей.

4. Розроблено логічний контент системи, в тому числі підсистема складності ігрового процесу та підсистема стратегій розвитку ігрового персонажу

5. Програмна реалізація ігрової системи виконано на мові програмування C ++ з використанням бібліотек Qt і IDE Qt Creator

6. Тестування ігрової системи доводить працездатність її ігрових механізмів, а також механізмів гнучкого налаштування її основних параметрів.

СПИСОК ЛІТЕРАТУРИ

1. Професії майбутнього. http://www.proforientator.ru/publications/articles/detail.php?ID=5454
2. Аналіз ігрового ринку світу 2014-2016 років. http://json.tv/ict_video_watch/analiz-rynka-igr-v--mire-2014-2016-gg-tekuschaya-situatsiya-prognozy-igroki-proek ty-i-tendentsii.
3. Ігровий ринок світу з 2010 по 2016 рік. http://json.tv/ict_telecom_analytics_view/rynok-igr-v-mire-2010-2016-gg-20141111113425.
4. Всесвітній ринок цифрових ігор - https://twitter.com/_SuperData/status/675337478514024448
5. Top Operating System Software – http://www.jegsworks.com/lessons/ComputerBasics/lesson8/lesson8-3.html .
6. Unity3d. – https://unity3d.com
7. Популярні ігри .E.P. https://habrahabr.ru/company/mailru/blog/181974/. .
8. CRYENGINE - Повне рішення для розробки ігор наступного покоління - https://www.cryengine.com. –
9. Що таке Unreal Engine 4? – https://www.unrealengine.com/
10. Програмування ігор - goo.gl/rxJ4xzc
11. Graphics: QT-5. – http://doc.qt.io/qt-5/graphicsview.html . .
12. Що таке SFML? Режим доступу: https://www.sfml-dev.org/grl-whatis.
13. SpaceInvaders. https://ru.wikipedia.org/wiki/SpaceInvaders.
14. Hexag. –http://www.hexag.net/r/.
15. Luxor. – https://ru.wikipedia.org/wiki/Luxor_Game/
16. За межами космосу. http://store.steampowered.com/app/297111/
17. Чужий у космосі.– http://onlineguru.ru/15104/view.html/.
18. Схема того, як використовувати її як концептуальне представлення бізнес-системи в процесі її розробки. – http://www.intuit.ru/studies/courses/32/32/lecture/1004

ДОДАТОК

В даному додатку представлений код головного меню програми:

- MainWindow.h – заголовочний файл головного меню.
- MainWindow.c – вихідний код головного меню.
- MainWindow.ui – файл форми головного меню (налаштування візуальних компонентів вікна(форми)).

«MainWindow.h» - заголовочний файл головного меню

```
#ifndef MANWINDOWS_H
#define MANWINDOWS_H
#include <QManWindows>
 namespace
Uii
{ class
ManWindows;
}  class ManWindows : public
QManWindows{
    Q_OBJECT

public:
    ~ManWindows();
 private slotss:     void on_ExitButton_clicked();
//кнопка «Выход»
    void  restartVideo();       //Зациклювання анімованого меню
void  on_GameButton_clicked();  // Кнопка «Играть» - запуск гри
void on_SettingButton_clicked(); //кнопка «Настройки» -
відкриття налаштувань текстур и музики.
        void on_BtnForLogic_clicked(); //кнопка «Логика» - відкриття
налаштувань логіки

};
```

«MainWindow.c» - исходный код главного меню

```
//Підключення бібліотек
#include "MainWindow.h"
#include "ui_MainWindow.h"
#include "Game.h"
#include "Settings.h"
#include <QMovie>
#include <QLabel>
#include <QGraphicsVideoItem>
#include <QGraphicsView>
#include <QGraphicsScene>
#include <QMediaPlayer>
#include <QPushButton>
#include <QFileDialog>
#include <QMessageBox>
#include <QUrl>
#include <QFile>

//Оголошення змінних extern
Game * game;
QMediaPlayer *player=new QMediaPlayer();
int clickplay = 0;


//Ініціалізація меню
//Включення анімованого меню

MainWindow::MainWindow(QWidget *parent) :
    QMainWindow(parent),
```

```
        uii(new Uii::ManWindows)

        uii->setupUii(this);     this->setFixedSize(this-
    >size().width(),this->size().height());
    setAttribute(Qt::WA_DeleteOnClose);
           QGraphicsVideoItem *item = new QGraphicsVideoItem;
    item->setSize(ui->playerScreen->size());        player-
    >setVideoOutput(item);
           QGraphicsScene *scene = new
QGraphicsScene(0,0,ui>playerScreen->size().width(),ui->playerScreen-
>size().height());
           ui->playerScreen->setScene(scene);        ui-
    >playerScreen->scene()->addItem(item);        player-
    >setMedia(QUrl::fromLocalFile("E:\\Diplom.mp4"));
    player->setVolume(100);        player->play();

    connect(player,SIGNAL(stateChanged(QMediaPlayer::State)),this,SLOT(re
    startVideo()));
    }
    MainWindow::~MainWindow()
    {     delete
    ui;
    }
    //Кнопка «Выход» - закриття додатку   void
    MainWindow::on_pushButton_2_clicked()
    {
    exit(0);
    }

    //        Анімоване        меню        void
    MainWindow::on_pushButton_3_clicked()
    {
```

```
    clickplay++;     player-
>stop();
    QString filename;     player-
>setMedia(QUrl::fromLocalFile(filename));
player->play();     clickplay--;
} void
MainWindow::restartVideo()
{
if(!clickplay)
player->play();
}
//Кнопка «Играть» - запуск гри  void
MainWindow::on_pushButton_clicked()
{     clickplay++;
player->stop();
delete player;
game = new Game();
game->show();
this->close();
clickplay--;
}
//Кнопка «Настройки» - відкриття вікна налаштування текстур, музики.
void MainWindow::on_SettingButton_4_clicked()
{
    Settings *setup = new Settings();     setup-
>setAttribute(Qt::WA_DeleteOnClose);
    setup->setFixedSize(setup->size().width(),setup>size().height());
    setup->show();
```

```
}
//Кнопка «Логика» - відкриття вікна логіки програми void
MainWindow::on_LogicButton_clicked()
{
    Settings *setup = new Settings();     setup-
>setAttribute(Qt::WA_DeleteOnClose);
    setup->setFixedSize(setup->size().width(),setup>size().height());
    setup->show();
}
```

«MainWindow.ui» - файл формы

```
Project created by QtCreator 2016-05-06T23:46:01
#Autor: Alexander Ivanov
#-------------------------------------------------
```

```
//Підключення бібліотек QT
QT += core gui
QT += multimedia
QT += multimediawidgets
QT += widgets

QT_QPA_PLATFORM_PLUGIN_PATH=%QTDIR%\plugins\platforms\
  greaterThan(QT_MAJOR_VERSION, 4): QT +=
widgets

TARGET = untitled
TEMPLATE = app

//Підключення вихідних файлів

/*
Enemy – код логіки ворога
Bullet – код логіки кули
Game – код логіки гри
Score – код обробки рахунку
Player – обробка натискань користувача
Health – обробка показника здоров'я
Mainwindows – форма головного меню
```

```
Settings - форма вікна налаштувань
Ending – форма вікна рахунку
logic.cpp – логіка переміщення ворогів
*/

SOURCES += main.cpp \
    Enemy.cpp \
    Bullet.cpp \
    Game.cpp \
    Score.cpp \
    Player.cpp \
    Health.cpp \
    ManWindows.cpp \
    Settings.cpp \
Ending.cpp \
logic.cpp

//Підключення заголовочних файлів

HEADERS  += \
    Bullet.h \
    Enemy.h \
Game.h \
    Score.h \
    Player.h \
    Health.h \
    ManWindows.h \
    Settings.h \
    Ending.h \
```

```
    logic.h

//Підключення до ресурсів

RESOURCES += \
res.qrc
//Підключення файлів форм

FORMS += \
    MainWindow.ui \
    Settings.ui \
Ending.ui \
logic.ui

//Підключення іконки додатку  win32:RC_ICONS  +=
Icon1.ico
```