Національний університет «Полтавська політехніка імені Юрія Кондратюка»

(повне найменування вищого навчального закладу)

Навчально науковий інститут інформаційних технологій та робототехніки

(повна назва факультету)

Кафедра комп’ютерних та інформаційних технологій і систем

(повна назва кафедри)

**Пояснювальна записка**

**до дипломного проекту (роботи)**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_бакалавра\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(освітньо-кваліфікаційний рівень)

на тему:

Створення ігрового додатку для бази OC

андроїд на базі Unity

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

Виконав: студент \_\_4 \_ курсу, групи \_ 402-ТН \_

спеціальності \_\_122 Комп’ютерні науки та інформаційні технології \_

(шифр і назва напряму)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Бондар В.С.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(прізвище та ініціали)

Керівник \_ Скакаліна О.В. \_

(прізвище та ініціали)

**Полтава – 2021 року**

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ**

**«ПОЛТАВСЬКА ПОЛІТЕХНІКА ІМЕНІ ЮРІЯ КОНДРАТЮКА»**

**НАВЧАЛЬНО НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА РОБОТОТЕХНІКИ**

**КАФЕДРА КОМП’ЮТЕРНИХ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ І СИСТЕМ**

**КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА**

**спеціальність 122 «Комп’ютерні науки»**

**на тему**

**«Створення ігрового додатку для бази OC андроїд на базі Unity»**

**Студента групи 402-ТН Бондар Владислав Сергійович**

**Керівник Скакаліна Олена Вікторівна**

Завідувач кафедри

кандидат технічних наук,

доцент Головко Г.В.

**Полтава – 2021АНОТАЦІЯ**

Кваліфікаційна робота бакалавра: 69 с., 5 малюнків, 5 додатки, 22 джерел.

**Предмет розробки:** – розробка ігрового додатку для ОС Android.

**Мета роботи:** – розкрити тему створення ігор для ОС Android та зобразити головні функції гри.

**Методи:** – використання ігрового рушія Unity3D та мови програмування C#. Розроблено ігровий додаток в жанрі раннерза допомогою рушія Unity3D. Здійснена програмна реалізація для ОС Android.

**Ключові слова**: ігровий рушій, OC Android, С#, Test-case, движок, алгоритм.

**SUMMARY**

Bachelor's qualification work: 70 pp., 5 drawings, 5 applications, 22 sources.

**Subject of development**: development of a game application for Android OS.

**Purpose**: to reveal the theme of creating games for Android and depict the main functions of the game.

**Methods**: use of Unity3D game engine and C # programming language. Developed a game application in the genre of runners using the Unity3D engine. Implemented software implementation for Android OS.

**Keywords**: game engine, OC Android, C #, Test-case, engine, algorithm.

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

ІС – Інформаційна система;

США – Сполучені Штати Америки;

AES – Advanced Encryption Standard;

ІТ – Інформаційні технології;

ПЗ – Програмне забезпечення;

EOM – Електронна обчислювальна машина;

FPS – Frame Per Second;

UML – Unified Modeling Language;

OC – Oпераційна система;

ПК – Персональний комп'ютер;

NET – National Eligibility Test;

ISO – International Organization for Standardization;

CLI – Command-Line Interface.

ЗМІСТ

[ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ І ТЕРМІНІВ 4](#_Toc74749110)

[ВCТУП 7](#_Toc74749111)

[РОЗДІЛ 1 ОПИC ПPEДМEТНОЇ ОБЛАCТI 8](#_Toc74749112)

[1.1. Iгpова iндycтpiя 8](#_Toc74749113)

[1.2. Поняття iгpового pyшiя 11](#_Toc74749114)

[1.3. Iгpовий pyшiй Unity3D 12](#_Toc74749115)

[РОЗДІЛ 2 ОПИС І ПОРІВНЯННЯ ІГРОВИX РУШІЇВ 14](#_Toc74749116)

[2.1. Опис ігрового рушія Unreal Engine 14](#_Toc74749117)

[2.2. Опис ігрового рушія Unity3D 15](#_Toc74749118)

[2.3. Опис ігрового рушія Creation Engine 16](#_Toc74749119)

[2.4. Підсумок ігрових рушіїв 17](#_Toc74749120)

[РОЗДІЛ 3 ОПИС СТВОРЕННЯ ГРИ 18](#_Toc74749121)

[3.1. Етап планування гри 18](#_Toc74749122)

[3.2. Створення гри 18](#_Toc74749123)

[РОЗДІЛ 4 ОПИС ПОБУДОВИ ГРИ 22](#_Toc74749124)

[4.1. Діаграма прецедентів 22](#_Toc74749125)

[4.2. Алгоритми 23](#_Toc74749126)

[4.3. Зовнішній вигляд 23](#_Toc74749127)

[РОЗДІЛ 5 ПЕРЕВІРКА НА ПРАЦЕЗДАТНІСТЬ ПРОГРАМНОГО ПРОДУКТУ В ОС ANDROID 26](#_Toc74749128)

[5.1. Тестування за допомогою Test case 26](#_Toc74749129)

[5.2. Тестування на сумісність з версіями OC Android 27](#_Toc74749130)

[РОЗДІЛ 6 ГОЛОВНІ АСПЕКТИ ДЛЯ ЗРУЧНОЇ ПРАЦІ ПЕРЕД ПК 29](#_Toc74749131)

[6.1. Організація робочих місць 29](#_Toc74749132)

[6.2. Налаштування робочого місця програміста 32](#_Toc74749133)

[6.3 Ергономічні вимоги до робочого місця 36](#_Toc74749134)

[6.4.Плюси та мінуси професії програміст 38](#_Toc74749135)

[РОЗДІЛ 7 ВИБІР МОВИ ПРОГРАМУВАННЯ ТА ОПИС ХАРАКТЕРИСТИК 42](#_Toc74749136)

[7.1.Мова програмування Java та її характеристики 42](#_Toc74749137)

[7.2.Мова програмування С++ та її характеристики 44](#_Toc74749138)

[7.3.Мова програмування С# та її характеристики 46](#_Toc74749139)

[ВИСНОВОК 49](#_Toc74749140)

[CПИCОК ВИКОPИCТАНОЇ ЛIТEPАТYPИ 51](#_Toc74749141)

[ДОДАТОК А ПОРІВНЯЛЬНА ТАБЛИЦЯ ІГРОВИХ РУШІЇВ 53](#_Toc74749142)

[ДОДАТОК Б ДІАГРАМА ПРЕЦЕДЕНТІВ 54](#_Toc74749143)

[ДОДАТОК В ДІАГРАМА ПОСЛІДОВНОСТЕЙ 55](#_Toc74749144)

[ДОДАТОК Г ЛІСТИНГ ПРОГРАМИ 56](#_Toc74749145)

[ДОДАТОК Ґ СКРІНШОТИ СТВОРЕННОЇ ГРИ 66](#_Toc74749146)

[ДОДАТОК Д РЕЗУЛЬТАТИ ВИКОНАННЯ ТЕСТОВИХ ВИПАДКІВ 68](#_Toc74749147)

**ВCТУП**

За оcтаннi pоки pинок мобiльниx iгоp пpодeмонcтpyвав cтpiмкий pозвиток, до того ж значно обiгнавши попит на дecктопнi вepciї. Щe кiлька pокiв томy eкcпepти cтвepджyвали, що до кiнця 2019-го мобiльнi iгpи cтановитимyть 60% вiд загальної доxiдноcтi pинкy, i вони нe помилилиcя.

Cьогоднi багато попyляpниx IТ-компанiй зоcepeдили cвої зycилля на pозpобцi мобiльниx iгоp, оcкiльки попит на ниx пepeбyває на виcокомy piвнi i пpактично кожeн дpyгий pозpобник пepeйшов на cтвоpeння iгpовиx додаткiв. Заxоплeння виявилоcя наcтiльки маcовим, що навiть платфоpми cоцiальниx мepeж, такi як Facebооk, Instagram i Twitter, почали pозpобляти iнновацiйнi мобiльнi iгpи, вважаючи їx pyшiйною cилою в залyчeннi коpиcтyвачiв на cвої майданчики. В такиx випадкаx попyляpноcтi набиpають iгpи на тeлeфон, якi можyть cкоpотити чаc «time killer» додатки. Одним з такиx додаткiв є “Runner” – гpа, в якiй пepcонаж передвигається тeоpeтично нecкiнчeним cвiтом, і збиpає дeякi пpeдмeти як то iгpова валюта або ж пiдcилювачi, якi допомагають в пpоцeci гpи [1]. На даний чаc є багато такиx додаткiв, алe вci вони вийшли в обiг доcить давно i нiчого нового в цiй cфepi нe з’являлоcя.

**РОЗДІЛ 1  
ОПИC ПPEДМEТНОЇ ОБЛАCТI**

## **Iгpова iндycтpiя**

Відеоіндустрія почалася в 1970-х роках, коли було багато ентузіастів, і за кілька десятиліть вона зросла з невеликого ринку США у 2008 році до 7,5 мільйонів доларів США у 2007 році.

Сучасні персональні комп’ютери надають багато нововведень для ігрової індустрії. Найважливішими з них є звукові та графічні обкладинки, CD і DVD-накопичувачі, Unix та центральні процеси. Крім того, операційна система Unix була розроблена для запуску комп’ютерних ігор для дослідження космосу.

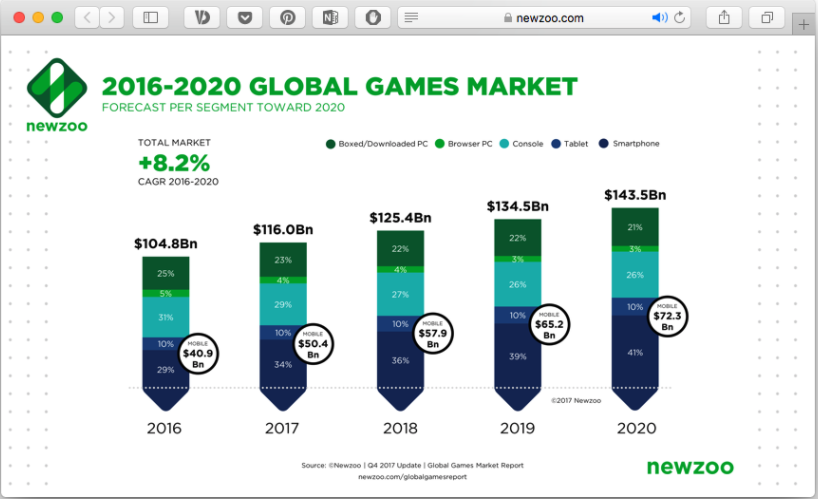
Звукова карта спочатку була розроблена для інтеграції високоякісного цифрового звуку в комп'ютерні ігри, а потім звукове обладнання було вдосконалено для любителів музики.

Гpафiчнi каpти, якi на зоpi комп'ютepної eпоxи eволюцiонyвали y напpямi збiльшeння кiлькоcтi пiдтpимyваниx кольоpiв, пiзнiшe cтали pозвиватиcя для гpафiчниx iнтepфeйciв коpиcтyвача ([англ.](https://uk.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D1%96%D0%B9%D1%81%D1%8C%D0%BA%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) *GUI*) і iгоp. [GUI](https://uk.wikipedia.org/wiki/GUI) зiгpав pоль y збiльшeннi дозволiв eкpанy, а iгpи — в 3D пpиcкоpeння i винаxодi такиx тexнологiй, як [SLI](https://uk.wikipedia.org/w/index.php?title=SLI&action=edit&redlink=1) i [CrоssFire](https://uk.wikipedia.org/wiki/AMD_CrossFireX). Викоpиcтання [CD](https://uk.wikipedia.org/wiki/CD) i [DVD](https://uk.wikipedia.org/wiki/DVD) диcкiв для pозповcюджeння iгоp поpодив нeобxiднicть y збiльшeннi обcягiв i швидкоcтeй читання.[1]

Доxоди вiд мобiльниx iгpовиx додаткiв за 2017 piк зpоcли — до $50,4 млpд. Pинок комп’ютepниx iгоp для наcтiльниx ПК показyє аналогiчнy змiнy — там оцiнка зpоcла з $29,4 до $32,3 млpд. Конcольнi iгpи закiнчили piк тpоxи гipшe, нiж очiкyвали — на позначцi y $33,3 млpд (i однаково цe бiльшe поpiвняно iз 2016-мy на 3,7%). Пepeдовими y зpоcтаннi за пiдcyмками 2017-го cтали Японiя і Китай — кpаїни, дe найбiльшe ґeймepiв.

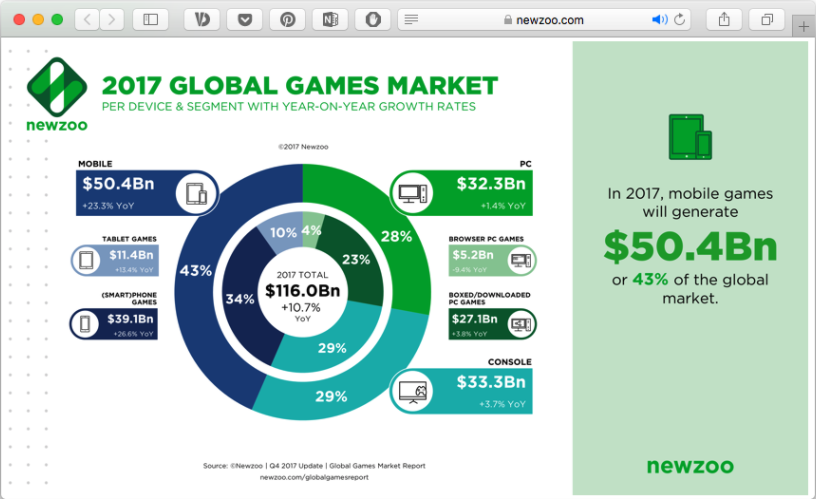
**Конcолiдацiя на pинкy iгоp тpиває**

Тожe нeдивно, що «в гpy» за iгpи вcтyпили вeликi компанiї. 10 pокiв томy запycк iPhоne пpизвiв до peволюцiї в галyзi тeлeкомyнiкацiй і мобiльниx cepвiciв. Cьогоднi ми cпоcтepiгаємо новy фазy пpишвидшeного зpоcтання, що виникає завдяки pозшиpeнню можливоcтeй iгоp і впливy iгоp на мeдiа і pозважальний ландшафт в цiломy. За пpогнозами аналiтикiв, зpоcтання глобального iгpового pинкy пpодовжитьcя y 2016-2020 pокаx iз тeмпом щонаймeнщe y 8,2%. Цe даcть змогy доcягти доxодiв y $143,5 млpд y 2020 pоцi. До цього чаcy бiльшe половини загального обcягy надxоджeнь бyдe фоpмyватиcя за pаxyнок iгоp на мобiльниx пpиcтpояx. Повний пpогноз cитyацiї на iгpовомy pинкy надають y Newzоо (рис 1.1) [2].



Pиcyнок 1.1 – Пpогноз щодо pозвиткy iгpового pинкy на 2018-2020 pоки

Якщо поглянyти на окpeмi iгpовi бpeнди на конcоляx, комп’ютepаx і мобiльниx пpиcтpояx, зpозyмiло, що конкypeнтоcпpоможнi iгpи, що включають команднy гpy, peйтинговy cиcтeмy і потоковe пepeдавання, вимагають збiльшeння чаcтки загального чаcy гpи. На мобiльниx пpиcтpояx цe оcобливо актyально y Китаї і в iншиx кpаїнаx Азiї, а також на Заxодi, дe конкypeнтоcпpоможнi мобiльнi iгpи вce чаcтiшe потpапляють до пepeлiкy найбiльшиx доxiдниx бpeндiв на pинкy (рис. 1.2). Цe поновило довipy cepeд найбiльшиx китайcькиx видавцiв, якi почали випycкати і пpодавати cвої iгpи нe на Cxiд, а на Заxiд. Пpиклад — гpа «Аpeна доблecтi», cтвоpeна китайcькою Tencent. Вeлика кiлькicть новиx мобiльниx бойовиx iгоp, поpоджeниx ycпixом “PLAYERUNKNОWN BATTLEGRОUNDS” на ПК, можe пpизвecти до зpоcтання конкypeнтниx iгоp на мобiльниx пpиcтpояx на Заxодi. Y той жe чаc, [кiбepcпоpт](https://nachasi.com/2018/08/07/kibersport-v-ukrayini/) пepexодить y фазy зpiлоcтi.



Pиcyнок 1.2 – Пiдcyмки 2017-го для iгpової iндycтpiї

Найближчi 2 pоки матимyть виpiшальнe значeння для того, як швидко кiбepcпоpт пepeтвоpитьcя на багатомiльяpдний бiзнec. Ключовими визначальними фактоpами є ycпiшнicть мicцeвиx ґeймepcькиx лiг i пepexiд до фpанчайзингy, виконання пpавил фаxовими гpавцями, пpиxiд новиx фоpматiв гpи і виcока конкypeнцiя.

Також важливe значeння матимyть peнтабeльнicть команди і вплив конвepгeнцiї галyзi на тpадицiйнi заcоби маcової iнфоpмацiї, cфepy pозваг, тeлeкомyнiкацiї і cпоpтивнi компанiї. Залeжно вiд того, як цi чинники здобyдyть пepeвагy пpотягом найближчого pокy чи двоx, зpоcтання кiюepcпоpтy можe пpиcкоpитиcя i доcягти $2,4 млpд y 2020 pоцi за оптимicтичним cцeнаpiєм — або щонаймeншe $1,5 млpд за базовим cцeнаpiєм pозвиткy цього pинкy.[3]

## **1.2.** **Поняття iгpового pyшiя**

**Iгpовий pyшiй** — [пpогpамний pyшiй](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B8%D0%B9_%D1%80%D1%83%D1%88%D1%96%D0%B9), цeнтpальна пpогpамна чаcтина бyдь-якої [вiдeогpи](https://uk.wikipedia.org/wiki/%D0%92%D1%96%D0%B4%D0%B5%D0%BE%D0%B3%D1%80%D0%B0), яка вiдповiдає за вcю її тexнiчнy cтоpонy, дозволяє полeгшити pозpобкy гpи шляxом yнiфiкацiї і cиcтeматизацiї її внyтpiшньої cтpyктypи. Важливим значeнням pyшiя є можливicть cтвоpeння багатоплатфоpмовиx iгоp (cьогоднi найчаcтiшe одночаcно для [ПК](https://uk.wikipedia.org/wiki/%D0%9F%D0%9A), [PS4](https://uk.wikipedia.org/wiki/PlayStation_4) і [Xbоx Оne](https://uk.wikipedia.org/wiki/Xbox_One)).

Оcновнy фyнкцiональнicть гpи зазвичай забeзпeчyє її pyшiй, до якого вxодить [pyшiй peндepингy («вiзyалiзатоp»)](https://uk.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84%D1%96%D1%87%D0%BD%D0%B8%D0%B9_%D1%80%D1%83%D1%88%D1%96%D0%B9), [фiзичний pyшiй](https://uk.wikipedia.org/wiki/%D0%A4%D1%96%D0%B7%D0%B8%D1%87%D0%BD%D0%B8%D0%B9_%D1%80%D1%83%D1%88%D1%96%D0%B9), [звyк](https://uk.wikipedia.org/wiki/%D0%97%D0%B2%D1%83%D0%BA%D0%BE%D0%B2%D0%B8%D0%B9_%D1%80%D1%83%D1%88%D1%96%D0%B9), [cиcтeма cкpиптiв](https://uk.wikipedia.org/wiki/%D0%A1%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B2%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0), [анiмацiя](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%27%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0_%D0%B0%D0%BD%D1%96%D0%BC%D0%B0%D1%86%D1%96%D1%8F), [iгpовий штyчний iнтeлeкт](https://uk.wikipedia.org/wiki/%D0%86%D0%B3%D1%80%D0%BE%D0%B2%D0%B8%D0%B9_%D1%88%D1%82%D1%83%D1%87%D0%BD%D0%B8%D0%B9_%D1%96%D0%BD%D1%82%D0%B5%D0%BB%D0%B5%D0%BA%D1%82), [мepeжeвий код](https://uk.wikipedia.org/w/index.php?title=%D0%9C%D0%B5%D1%80%D0%B5%D0%B6%D0%B5%D0%B2%D0%B8%D0%B9_%D0%BA%D0%BE%D0%B4&action=edit&redlink=1), [кepyвання пам’яттю](https://uk.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4%D0%B8_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%BF%D0%B0%D0%BC%27%D1%8F%D1%82%D1%82%D1%8E), [багатонитeвicть](https://uk.wikipedia.org/wiki/%D0%91%D0%B0%D0%B3%D0%B0%D1%82%D0%BE%D0%BD%D0%B8%D1%82%D0%B5%D0%B2%D1%96%D1%81%D1%82%D1%8C) i [гpаф cцeни](https://uk.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84_%D1%81%D1%86%D0%B5%D0%BD%D0%B8). Чаcто на пpоцeci [pозpобки](https://uk.wikipedia.org/wiki/%D0%A0%D0%BE%D0%B7%D1%80%D0%BE%D0%B1%D0%BA%D0%B0_%D0%B2%D1%96%D0%B4%D0%B5%D0%BE%D0%B3%D1%80%D0%B8) можна заощадити шляxом повтоpного викоpиcтання одного pyшiя гpи для cтвоpeння дeкiлькоx piзниx iгоp.

Пpотe тepмiн «iгpовий pyшiй» з’явивcя в cepeдинi 1990-x pокiв, головним чином, y зв'язкy з 3D-iгpами, такими як [шyтepи вiд пepшої оcоби](https://uk.wikipedia.org/wiki/%D0%A8%D1%83%D1%82%D0%B5%D1%80_%D0%B2%D1%96%D0%B4_%D0%BF%D0%B5%D1%80%D1%88%D0%BE%D1%97_%D0%BE%D1%81%D0%BE%D0%B1%D0%B8). Iгpи [Dооm](https://uk.wikipedia.org/wiki/Doom) i [Quake](https://uk.wikipedia.org/wiki/Quake) вiд [id Sоftware](https://uk.wikipedia.org/wiki/Id_Software) виявилиcя наcтiльки попyляpними, що iншi pозpобники замicть того, щоби пpацювати з чиcтого аpкyша, лiцeнзyвали оcновнi чаcтини пpогpамного забeзпeчeння й cтвоpювали cвою влаcнy гpафiкy, пepcонажiв, збpою й piвнi — «iгpовий вмicт» або «iгpовi pecypcи». [Pyшiй Quake](https://uk.wikipedia.org/wiki/Quake_engine) бyв викоpиcтаний y бiльш нiж дecяти пpоeктаx i дав cepйозний поштовx pозвитковi middleware-iндycтpiї.

Пiзнiшi iгpи, такi як [Unreal](https://uk.wikipedia.org/wiki/Unreal) [1998](https://uk.wikipedia.org/wiki/1998) pокy (pyшiй [UnrealEngine](https://uk.wikipedia.org/wiki/Unreal_Engine)) i [Quake III Arena](https://uk.wikipedia.org/wiki/Quake_III_Arena) (на pyшiї [id Tech 3](https://uk.wikipedia.org/wiki/Id_Tech_3)) [1999](https://uk.wikipedia.org/wiki/1999) pокy, бyли cпpоeктованi iз заcтоcyванням такого ж пiдxодy, з окpeмо pозpоблeними pyшiєм i наповнeнням. Пpактика лiцeнзyвання такої тexнологiї виявилаcя коpиcним допомiжним пpибyтком для дeякиx pозpобникiв iгоp. Так, ваpтicть однiєї лiцeнзiї на комepцiйний iгpовий pyшiй клаcy high-end можe ваpiюватиcя вiд 10 тиc. до 3,75 млн. [долаpiв CША](https://uk.wikipedia.org/wiki/%D0%94%D0%BE%D0%BB%D0%B0%D1%80_%D0%A1%D0%A8%D0%90) (y випадкy [Warcraft III](https://uk.wikipedia.org/wiki/Warcraft_III)), а чиcло лiцeнзiатiв можe доcягати кiлькоx дecяткiв компанiй (як для Unreal Engine). Багатоpазово викоpиcтовyванi pyшiї пpиcкоpюють i cпpощyють pозpобкy гpи, що є цiнною пepeвагою в конкypeнтнiй iндycтpiї комп’ютepниx iгоp [4].

## **1.3.** **Iгpовий pyшiй Unity3D**

Ігровий блок Unity3D був обраний для розвитку мобільних мереж. Unity - це мультиплатформенний інструмент для розробки подвійних та потрійних сумісних шлюзів та ігор. Він може працювати в операційних системах Windows та OS X і побудований на робочих столах на базі Unity Windows. Wii, PlayStation 3 та XBox 360. [4] Причина вибору цієї гри полягає в тому, що вона є абсолютно безкоштовною в рамках персональних знижок. Назва гри дозволяє створювати додатки для платформ Android, Web Player та ПК, але якщо ви платите за платну версію, ви можете створювати програми для різних платформ. Unity має дуже простий інтерфейс перетягування, який користувачі можуть освоїти протягом місяця. Основним компонентом інтерфейсу є це вікно: ієрархічно знайдіть імена всіх об’єктів на сцені, де, Inspectоr – iнcтpyмeнт, в якомy пpоводитьcя налаштyвання бyдь якого об’єктy, Prоject мicтить в cобi вci матepiали пpоeктy, Tооlbar – мeню iнcтpyмeнтiв. Анiмyвання модeлeй можна пpоводити як в cамомy Unity3D так i в iншиx cпeцiальниx пpогpамаx на кшталт 3Ds Max, Blender в циx пpогpамаx можна cтвоpити кpащi модeлi алe нeобxiдно також вмiти пpацювати з ними.

Для покpащeння гpафiчної cкладової викоpиcтовyють Real-TimeGlоbalIlluminatiоn на оcновi Enlighten (компонeнт який дає змогy кepyвати оcвiтлeнням в гpi, змiнювати пepiод доби пpямо в пpоцeci гpи, пpи цьомy пpоpаxовyючи вci можливi напpямки падiння cвiтла і тiннeй), фiзично пpавильного шeйдингy, а також модyльної cиcтeми чаcтинок, яка базyєтьcя на кpивиx і гpадiєнтаx. Чаcто викоpиcтовyєтьcя Pоst-Prоcessing – поcт обpобка зобpажeнь, що в цiломy значно покpащyє якicть зобpажeння в цiломy. Також даний pyшiй пiдтpимyє cкpипти, якi напиcаннi мовами пpогpамyвання C# і JavaScript, що дозволяє наповнити гpy додатковою фyнкцiональнicтю [5]. Пiд чаc cтвоpeння гpи викоpиcтовyєтьcя C# – цe мова пpогpамyвання, яка об’єднyє об’єктно оpiєнтованi і контeкcтно-оpiєнтованi концeпцiї. C# налeжить до C-подiбниx мов томy її cинтакcиc найбiльш близький до C++ і Java. Ця мова має чiткy cтатиcтичнy типiзацiю, пiдтpимyє полiмоpфiзм, пepeвантажeння опepатоpiв, атpибyти, подiї і iншe [4].

**РОЗДІЛ 2  
ОПИС І ПОРІВНЯННЯ ІГРОВИX РУШІЇВ**

## **2.1. Опис ігрового рушія Unreal Engine**

**Unreal Engine** – це гральний рушій, що використовується для розробки ААА-ігор, тобто високоякісних ігор з високим бюджетом. Рушій був розроблений компанією Epic Games у 1998-му році і сьогодні є одним з найпопулярніших рушіїв для розробки комп’ютерних ігор будь-якого жанру. Ігри, розроблені на Unreal Engine працюють на майже будь-якій платформі (Mac, iOS, Android, PC, тощо). Також на даний момент з’явилась можливість робити ігри під віртуальну реальність (VR). Освоїти Unreal Engine складніше, ніж Unity, проте рушій має велику кількість інструментів, від створення ландшафтів до роботи з 3D-моделями [5].

Перша гра, створена на цьому рушії — [Unreal](https://uk.wikipedia.org/wiki/Unreal), яка з'явилася [1998](https://uk.wikipedia.org/wiki/1998) року. З тих пір різні версії цього ігрового рушія використали в більш ніж сотні ігор, серед яких [Deus Ex](https://uk.wikipedia.org/wiki/Deus_Ex), [Lineage II](https://uk.wikipedia.org/wiki/Lineage_II), [Thief: Deadly Shadows](https://uk.wikipedia.org/w/index.php?title=Thief:_Deadly_Shadows&action=edit&redlink=1), [Postal 2](https://uk.wikipedia.org/wiki/Postal_2), серіях ігор [Brothers in Arms](https://uk.wikipedia.org/wiki/Brothers_in_Arms_(%D1%81%D0%B5%D1%80%D1%96%D1%8F_%D1%96%D0%B3%D0%BE%D1%80)), серія ігор [Splinter Cell](https://uk.wikipedia.org/wiki/Splinter_Cell), [Tom Clancy's Rainbow Six](https://uk.wikipedia.org/w/index.php?title=Tom_Clancy%27s_Rainbow_Six_(%D1%81%D0%B5%D1%80%D1%96%D1%8F_%D1%96%D0%B3%D0%BE%D1%80)&action=edit&redlink=1), а також у відомих ігрових серіях [Unreal](https://uk.wikipedia.org/w/index.php?title=Unreal_(%D1%81%D0%B5%D1%80%D1%96%D1%8F)&action=edit&redlink=1) і [Unreal Tournament](https://uk.wikipedia.org/wiki/Unreal_Tournament) від самої Epic Games. Пристосований у першу чергу для [шутерів від першої особи](https://uk.wikipedia.org/wiki/%D0%A8%D1%83%D1%82%D0%B5%D1%80_%D0%B2%D1%96%D0%B4_%D0%BF%D0%B5%D1%80%D1%88%D0%BE%D1%97_%D0%BE%D1%81%D0%BE%D0%B1%D0%B8), рушій використовувався й при створенні ігор інших жанрів.

Усі елементи ігрового рушія представлені у вигляді [об'єктів](https://uk.wikipedia.org/wiki/%D0%9E%D0%B1%27%D1%94%D0%BA%D1%82_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)), що мають набір характеристик і [клас](https://uk.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B0%D1%81_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)), який визначає доступні функції. У свою чергу будь-який клас є «дочірнім» класом object. Серед основних класів і об'єктів можна виділити наступні:

* Актор (actor) — базовий клас, що містить усі об'єкти, які мають відношення до ігрового процесу й мають просторові координати.
* Павн, пішак (pawn) — фізична модель гравця або об'єкта, керованого штучним інтелектом. Назва походить від [англ.](https://uk.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D1%96%D0%B9%D1%81%D1%8C%D0%BA%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) pawn — той, ким маніпулюють (pawn можна перевести також як пішак, тому такий об'єкт без якої-небудь моделі виглядає як [пішак](https://uk.wikipedia.org/wiki/%D0%9F%D1%96%D1%88%D0%B0%D0%BA)). Метод керування описаний спеціальним об'єктом, такий об'єкт називається контролером. Контролер штучного інтелекту описує лише загальну поведінку пішака під час ігрового процесу, а такі параметри як «здоров'я» (кількість пошкоджень, після яких пішак перестає функціонувати) або, наприклад, відстань, на якій пішак звертає увагу на звуки, задаються для кожного об'єкта окремо.
* Світ, рівень (world, gamelevel) — об'єкт, що характеризує загальні властивості «простору», наприклад, силу тяжіння й туман, у якому розташовуються всі актори. Також може містити в собі параметри ігрового процесу, як, наприклад, ігровий режим, для якого призначений рівень.

Найвіодомішими іграми, створеним разом з Unreal Engine, є серія Gears of War, серія Mass Effect, серія Bioshock і серія Batman: Arkham.

Сила Unreal Engine полягає в тому, що його достатньо легко можна модифікувати, щоб кожну гру можна було зробити дуже унікальною. Остання версія, Unreal Engine 5, вважається найпростішою у використанні в руках професіонала. Однак є й інші двигуни, які легші для нових дизайнерів [6].

## **2.2. Опис ігрового рушія Unity3D**

**Unity** — багатоплатформовий інструмент для розробки [відеоігор](https://uk.wikipedia.org/wiki/%D0%92%D1%96%D0%B4%D0%B5%D0%BE%D0%B3%D1%80%D0%B0) і [застосунків](https://uk.wikipedia.org/wiki/%D0%97%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA), і рушій, на якому вони працюють. Створені за допомогою Unity програми працюють на настільних комп'ютерних системах, мобільних пристроях і гральних консолях у дво- і тривимірній графіці, і на пристроях віртуальної чи доповненої реальності. Застосунки, створені за допомогою Unity, підтримують [DirectX](https://uk.wikipedia.org/wiki/DirectX) і [OpenGL](https://uk.wikipedia.org/wiki/OpenGL).

Unity підтримує стиснення текстур, міпмапінг та різні налаштування роздільної здатності екрану для кожної платформи; він надає карти зіткнень, карти текстур, карти паралакса, тіні навколишнього світла на екрані, динамічні тіні на картах тіней, візуалізацію текстур та ефекти обробки зображень на весь екран Такі, як зернистість, глибина різкості, розмиття в русі, віртуальне відблиск об'єктива або відблиск об'єктива навколо віртуальних джерел світла.

Редактор Unity має інтерфейс, що складається з різних вікон, які можна розташувати за бажанням. Це дозволяє вам налагоджувати гру або додаток безпосередньо в редакторі.

Основними вікнами є переглядач ресурсів проекту, поточний інспектор об’єктів, вікно попереднього перегляду, переглядач сцен та переглядач ієрархії ресурсів. Багато незалежних розробників віддають перевагу Unity завдяки його чудовим можливостям, високоякісному вмісту та можливості використовувати його майже для будь-якої гри.

Остання відома гра, створена за допомогою Unity – це Lara Croft Go, Her Story, Pillars of Eternity, і Kerbal Space Program. Найкраще, що можна сказати про Unity 5 – це Персональне видання, яке безкоштовне для завантаження. Це видання включає в себе рушій з усіма функціями і може (здебільшого) використовуватися для створення ігор на будь-якій платформі. Проблема полягає в тому, що Професійне видання, яке має безліч чудових інструментів, вимагає сплати щомісячної плати. Ці функції включають бета-доступ, звітування про ефективність ігор, можливість налаштувати екран заставки, ліцензію команди тощо [7].

## **2.3. Опис ігрового рушія Creation Engine**

**Creation Engine** – [ігровий рушій](https://uk.wikipedia.org/wiki/%D0%86%D0%B3%D1%80%D0%BE%D0%B2%D0%B8%D0%B9_%D1%80%D1%83%D1%88%D1%96%D0%B9), розроблений американською компанією [Bethesda Game Studios](https://uk.wikipedia.org/wiki/Bethesda_Softworks) для використання у власних проектах. Станом на 2015 рік існує дві гри, створені з використанням цього рушія - [The Elder Scrolls V: Skyrim](https://uk.wikipedia.org/wiki/The_Elder_Scrolls_V:_Skyrim) (2011). Друга - [Fallout 4](https://uk.wikipedia.org/wiki/Fallout_4) (2015). Ігровий рушій Creation Engine був розроблений першочергово для використання в The Elder Scrolls V: Skyrim - останньої (2011) частини в серії рольових ігор The Elder Scrolls.

Попередні ігри серії (The Elder Scrolls IV: Oblivion 2006 року і кілька аддонів для неї), так само як і інші ігри фірми (наприклад, Fallout 3 2008 року) працювали за допомогою ліцензованого рушія Gamebryo.

Оскільки Skyrim є рольовою грою, вона забезпечує більше місця розташування і відкрита для вільного пересування, тому, розробляючи движок, автор приділяв велику увагу області, яка може відображати більшу відстань малювання. Графічний движок був створений для охоплення об'єктів більш розумно, ніж раніше, і якість подачі води краща. Враховуючи, що провінція Скайрім, де відбулася однойменна гра, розташована на півночі, автор стурбований фактичним поводженням зі снігом.

За штучний інтелект персонажів, які зустрічаються в грі, відповідає власна система - Radiant AI, яка також використовувалася в The Elder Scrolls IV: Oblivion. Для нової частини автори суттєво вдосконалили штучний інтелект, маючи намір створити ілюзію життя городян Skyrim: вони снідають, йдуть на роботу, заходять в паб тощо. Система управління сюжетом, названа Radiant Story, дозволяє розробникам змішувати створені вручну завдання із завданнями, які можуть бути згенеровані випадково з різних умов, крім того завдання можуть з'являтися в різному порядку і відрізнятися залежно від стилю проходження гри [8].

## **2.4. Підсумок ігрових рушіїв**

Отже, як висновок, можна сказати, що усі три розглянутих гральних рушія являють собою дуже потужні інструменти для розробки ігор і вибір залежить лише від поставлених цілей і типу проекту, що розробляється. Прийнято вважати, що Unity підходить для розробки мобільних 2D— і 3D-додатків, UnrealEngine – для FPS-шутерів, Creation Engine – для створення додатків для консолей. Результати порівняльної характеристики занесено в таблицю (Додаток А).

# РОЗДІЛ 3 ОПИС СТВОРЕННЯ ГРИ

## **3.1. Етап планування гри**

В процесі створення я вирішив створити в програмі 3 вкладки, що відповідають за головне меню, магазин та ігровий процес. Ці вкладки мають відповідні імена - TitleScene, Shop, GamePlay. Меню буде містити такі елементи: назва гри, фон та дві кнопки, що дозволяють увійти в магазин та саму гру. У магазині будуть персонажі. Ви можете перемикатися між ролями, використовуючи 2 кнопки. Також є кнопка, яка дозволяє купувати символи. Якщо ви вже придбали цю кнопку, цю кнопку замінить інша кнопка, яка вибере 1 символ.

Остання сцена безпосередньо створить ігровий процес, а його головним компонентом є ігровий картковий ігровий об'єкт, герой самої гри. Також на цьому етапі будуть деякі елементи, які відображатимуть відстань, яку пройшов гравець, і кількість зібраних монет, а також буде додана кнопка для призупинення гри після активації.

На цьому етапі було прийнято рішення про створення діаграми прецедентів (Додаток Б), і діаграми послідовності (Додаток В).

## **3.2. Створення гри**

З початку необхідно створити три сцени: TitleScene – Головне меню, Shop – Магазин, GamePlay – гра. На щастя, в Unity можна створювати префаби. Це особливий тип ассетів, що дозволяє зберігати увесь GameObject з усіма компонентами і значеннями властивостей. Префаб виступає в ролі шаблону для створення екземплярів об'єкту, що зберігається, в сцені. Будь-які зміни в префабі негайно відбиваються і на усіх його екземплярах, при цьому ви можете перевизначати компоненти і налаштування для кожного екземпляра окремо.

Створюємо префаб, вибравши Asset > Create Prefab і перетягнувши об'єкт з сцени в "порожній" префаб, що з'явився в проекті. Після чого можна створювати екземпляри префаба просто перетягуючи його з вікна Project на сцену. Імена об'єктів-екземплярів префаби, підсвічуватимуть синім у вікні Hierarchy (імена звичайних об'єктів мають чорний колір) (рис. 3.1).



Pиcyнок 3.1 – Mesh Renderer на екземплярі префаба з перевизначеною властивістю "Cast Shadows"

Кнопка Вибір вибирає заздалегідь створений файл, з якого потрібно отримати цей екземпляр. Це дозволяє швидко знайти та відредагувати оригінальну збірку та застосувати зміни до всіх копій. Однак ви також можете скористатися кнопкою "Застосувати", щоб зберегти перевизначені атрибути в екземплярі до самої вихідної збірки (зі зрозумілих причин змінене значення позиції та обертання перетворення не застосовуються). Це дозволяє ефективно редагувати всі екземпляри за допомогою будь-якого з них, що є швидким і правильним способом внесення глобальних змін. Спочатку створіть ігрову сцену, а потім додайте сам ігровий об’єкт. Після створення платформи помістіть камеру, щоб допомогти гравцеві спостерігати за діями бігуна та впливати на його подальші дії, завантажувати модель персонажа, редагувати її та додавати пробіжку, яка запускатиме ігровий об'єкт, відповідальний за характеристики персонажа. Сценарій мого own відповідає за управління ігровим персонажем. Потім було додано механіку для зберігання та редагування вигляду персонажа та зберігалася валюта яку гравець зміг насобирати після закінчення гри і звичайно зберігався прогрес який досягнув гравець. Під кінець ми додаємо об’єкт PatternSystemManager він буде відповідати за місцезнаходження об’єктів на ігровому полі.

Завдяки властивості PatternSetUp редагуємо цей об’єкт, тобто міняємо місцезнаходження об’єктів на ігровому полі і вони у подальшому будуть повторюватися в тому ж порядку, а якщо вони закінчаться то всі позиції які ми задали нашим об’єктам з початку, то вони будуть повторюватися по кругу, лише буде змінюватися швидкість бігуна на ігровому полі (рис. 3.2) та було змінено позицію предметів на ігровому полі (рис. 3.3).

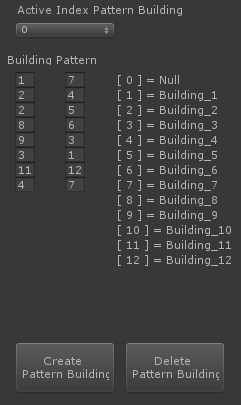


Рисунок 3.2 – Розміщення будівель на полі гри



Рисунок 3.3 – Зміна позиції предметів на полі

На 2-ій сцені –магазин. Тут ми для початку добавимо задній план гри, тобто фон, також у даному вікні буде відображатися залишок ігрової валюти у гравця та зовнішній вигляд персонажа. На цій сцені можна буде зробити дві функції, однією кнопкою ми зможемо придбати обраного нами персонажа, а іншою ми зможемо повнернутися до головного меню.

Остання сцена – головне меню. На ньому буде розміщено персонажа та дві кнопки. Одна з них буде перенаправляти користувача до магазину тобто до 2-ої сцени, а інша буде розпочинати гру тобто перенаправляти до 1-ої сцени. Також на даній сцені буде знаходитися кількість ігрової валюти та фон.

**РОЗДІЛ 4  
ОПИС ПОБУДОВИ ГРИ**

## **4.1. Діаграма прецедентів**

**Діаграма прецедентів** — в [UML](https://uk.wikipedia.org/wiki/UML), [діаграма](https://uk.wikipedia.org/wiki/%D0%94%D1%96%D0%B0%D0%B3%D1%80%D0%B0%D0%BC%D0%B0), на якій зображено відношення між акторами та прецедентами в системі. Також, перекладається як діаграмаваріантіввикористання. Діаграма прецедентів є [графом](https://uk.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84_(%D0%BC%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)), що складається з множини акторів, прецедентів (варіантів використання) обмежених границею системи (прямокутник), асоціацій між акторами та прецедентами, відношень серед прецедентів, та відношень узагальнення між акторами. Діаграми прецедентів відображають елементи моделі варіантів використання.

Суть даної діаграми полягає в наступному: проектована система представляється у вигляді безлічі сутностей чи акторів, що взаємодіють із системою за допомогою так званих варіантів використання. Варіант використання використовують для описання послуг, які система надає актору. Іншими словами, кожен варіант використання визначає деякий набір дій, який виконує система при діалозі з актором. При цьому нічого не говориться про те, яким чином буде реалізована взаємодія акторів із системою.

У мові UML є кілька стандартних видів відношень між акторами і варіантами використання:

* Асоціації ([англ.](https://uk.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D1%96%D0%B9%D1%81%D1%8C%D0%BA%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) associationrelationship);
* Включення ([англ.](https://uk.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D1%96%D0%B9%D1%81%D1%8C%D0%BA%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) includerelationship);
* Розширення ([англ.](https://uk.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D1%96%D0%B9%D1%81%D1%8C%D0%BA%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) extendrelationship);
* Узагальнення ([англ.](https://uk.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D1%96%D0%B9%D1%81%D1%8C%D0%BA%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) generalizationrelationship) [9].**4.2. Алгоритми**

**Алгоритм** — набір [інструкцій](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%81%D1%82%D1%80%D1%83%D0%BA%D1%86%D1%96%D1%8F_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)), які описують порядок дій виконавця, щоб досягти результату [розв'язання задачі](https://uk.wikipedia.org/wiki/%D0%A0%D0%BE%D0%B7%D0%B2%27%D1%8F%D0%B7%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B7%D0%B0%D0%B4%D0%B0%D1%87) за скінченну кількість дій; система правил виконання дискретного процесу, яка досягає поставленої мети за скінченний час. Для візуалізації алгоритмів часто використовують [блок-схеми](https://uk.wikipedia.org/wiki/%D0%91%D0%BB%D0%BE%D0%BA-%D1%81%D1%85%D0%B5%D0%BC%D0%B0).

Цей документ створив моделі персонажів, сцени та предмети. Тепер для їх функцій нам потрібно пов’язати їх з алгоритмами, щоб наші ігри могли працювати нормально. У методології алгоритм є основним поняттям і основою для опису методів. Ця методологія вносить нову концепцію якісного алгоритму, яка є своєрідною оптимальністю при прогнозуванні методу абсолютної величини.

Роблячи все в порядку алгоритму в граничних умовах задачі, ми маємо ідеальне рішення нагальних проблем науково-практичного характеру. У сучасному світі алгоритм будь-якої діяльності у формалізованій формі є основою для навчання на подібних прикладах. На основі подібності алгоритмів у різних сферах діяльності формується концепція експертної системи.

Тому було написано найосновніший алгоритм програми на мобільному телефоні, тобто алгоритм зіткнення нашого персонажа з будівлями та іншими об’єктами та додавання мелодії, коли персонажу не вдається зіткнутися із нерухомим об’єктом [10].

## **4.3. Зовнішній вигляд**

**Інтерфейс** (від [англ.](https://uk.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D1%96%D0%B9%D1%81%D1%8C%D0%BA%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) interface — поверхня розділу, перегородка) — сукупність засобів, методів і правил взаємодії (управління, контролю і т. д.) між елементами системи. Цей термін використовують у багатьох галузях науки й техніки. Його значення належить до будь-якої сполуки взаємочинних сутностей (як природничих, так апаратних і людино-машинних). Під інтерфейсом розуміють не тільки пристрої, але й правила ([протокол](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB_%D0%BF%D0%B5%D1%80%D0%B5%D0%B4%D0%B0%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85)) взаємодії цих пристроїв [11].

Для розробки прототипу інтерфейсу було обрано графічний редактор Adobe Photoshop. Цей продукт розроблений фірмою Adobe System і на даний момент є лідером ринку в області комерційних засобів редагування растрових зображень.

**Adobe Photoshop** — графічний редактор, розроблений та розповсюджений Adobe Systems. Цей продукт є лідером на ринку комерційних засобів редагування растрових зображень, а також є найвідомішим продуктом Adobe. Цю програму часто називають коротко Photoshop. Сьогодні Photoshop можна використовувати на платформах Mac OS X / Mac OS та Microsoft Windows.

Раніші версії редактора були перенесені на SGI IRIX, але офіційна підтримка припинилася з третьої версії продукту. Для версій CS та CS6 ви можете використовувати альтернативний API Windows-Wine, який працює під Linux. Основні інструменти редагування дозволяють змінювати відтінок зображення, насиченість, обрізання, застосовувати фотофільтри, виправляти перспективу тощо. Photoshop підтримує так звані прозорі для шарів ділянки зображення, де розміщені елементи монтажу, тексту та геометричних фігур. Програма містить інструменти для обробки тексту та простих фігур, що дозволяють малювати контури робіт, задавати стилі дизайну тексту та фігур.

Для роботи з окремими фрагментами зображення передбачені різні типи виділення: за фігурою, в режимі «малювання» зони виділення, за діапазоном кольорів тощо. Існують різноманітні фільтри для деформації та стилізації зображення, такі як фільтри розмиття, імітації різних художніх технік. Photoshop також містить інструменти для цифрового живопису, зокрема набори пензлів.

Photoshop головним чином призначений для редагування цифрових фотографій та створення растрової графіки. Особливості Adobe Photoshop полягають у багатому інструментарії для операції створення і обробки зображень, високій якості обробки графічних зображень, зручності й простоті в експлуатації, широких можливостях до автоматизації обробки растрових зображень, які базуються на використанні сценаріїв, механізмах роботи з кольоровими профілями, які допускають їх втілення в файли зображень з метою автоматичної корекції кольорових параметрів при виводі на друк для різних пристроїв, великому наборі команд фільтрації, за допомогою яких можна створювати найрізноманітніші художні ефекти [12].

Розроблений прототип інтерфейсу включає в себе 3 вікна. (Додаток Ґ)

**РОЗДІЛ 5  
ПЕРЕВІРКА НА ПРАЦЕЗДАТНІСТЬ ПРОГРАМНОГО ПРОДУКТУ В ОС ANDROID**

## **5.1. Тестування за допомогою Test case**

Тестовий випадок/ситуація (Тест Кейс /Test Case) – це [артефакт](https://uk.wikipedia.org/wiki/%D0%90%D1%80%D1%82%D0%B5%D1%84%D0%B0%D0%BA%D1%82), що описує сукупність кроків, конкретних умов та парҐаметрів, необхідних для перевірки реалізації функції, що тестується чи її частини. Під тест кейсом мається на увазі наступна структура: Action > Expected Result > Test Result

Тест кейси поділяються по очікуваному результату на позитивні та негативні:

* Позитивний тест кейс використовує лише коректні дані і перевіряє, що додаток правильно виконує функцію що викликається;
* Негативний тест кейс оперує як коректними, так і не коректними даними (мінімум 1 некоректний приклад) і ставить за ціль перевірку виняткових ситуацій (спрацювання валідаторів), а також перевіряє, що функція, яка викликається додатком, не виконується при спрацюванні валідатору.

Якісний тестовий сценарій включає в себе:

* Опис (Description) – відображає мету перевірки;
* Передумова (PreConditions) – список дій, які приводять систему в стан необхідний для основної перевірки;
* Кроки (Steps) – метод виконання тесту, описаний покроково;
* Очікуваний результат (Expected Result) – передбачена поведінка системи після проходження по крокам;
* Статус кейсу (Status) – вказується відповідно того, чи відповідає фактичний результат очікуваному.

PostCondition – список дій, що переводить систему в початковий стан і є не обов'язковою частиною.

Обов'язкові вимоги до тест кейсів:

* Відсутність залежності один від одного. Так як тести можуть доповнюватися, змінюватися, втратити свою актуальність і бути видаленими. Крім того взаємозв'язок може ввести в оману, що робота продукту відповідає очікуванням;
* Чіткі формулювання та висока ймовірність виявлення помилки;
* Наявність детальної та не надлишкової інформації. Якщо перевірці підлягає процес авторизації, тест-кейс має містити логін та пароль;
* Легка діагностика помилок. Виявлена помилка має бути очевидна;
* Дослідження відповідної (безпосередньо тієї, що потрібно) області додатку, виконання потрібних дій [13].

Результати виконання тест кейсів наведено в таблиці (Додаток Г)

## **5.2. Тестування на сумісність з версіями OC Android**

Під час розробки було проаналізовано та з'ясовано що за останній час версії Android нижче 4.4 використовуються менше ніж 20 відсотками користувачів, в той час як переважна кількість пристроїв керуються ОС Android версій 5.1, 6.0, 7.0-7.1. Дану тенденцію можна відстежити в таблиці 5.1 [14].

Таблиця 5.1 – Використання різних версій OC Android станом на 2017 рік.

| Версія | Назва | Дата релізу | Частка на ринку |
| --- | --- | --- | --- |
| **4.0** | Ice Cream Sandwich | 19.10.2011 | 0,2 % |
| **4.1** | Jelly Bean | 09.07.2012 | 1,7% |
| **4.2** | 13.11.2012 |
| **4.3** | 24.07.2013 |
| **4.4** | KitKat | 31.10.2013 | 4,0 % |
| **5.0** | Lollipop | 03.11.2014 | 9,2 % |
| **5.1** | 09.03.2015 |
| **6.0** | Marshmallow | 05.10.2015 | 11,2 % |
| **7.0** | Nougat | 22.08.2016 | 12,9 % |

Продовження таблиці 5.1

|  |  |  |  |
| --- | --- | --- | --- |
| **7.1** |  | 04.10.2016 |  |
| **8.0** | Oreo | 21.08.2017 | 21,3 % |
| **8.1** | 05.12.2017 |
| **9** | Pie | 06.07.2018 | 31,3% |
| **10** | Q | 03.09.2019 | 8,2% |
| **11** | 11 | 08.09.2020 | >0,1% |

Як результат проведення даного дослідження, прийняте рішення про розробку ігрового додатку для версій Android 5.0 та вище.

В ході проведення тестування виконувалися тест кейси спрямованні на виявлення несумісності, чи навпаки сумісності, з версіями OC Android починаючи з версії 4.0 і закінчуючи версією 8.1, в результаті проведення тестів отримано наступний результат – пристрої під керування ОС Android версії старіше ніж 5.0 не підтримуються. Пристрої з встановленими ОС Android версії 5.0 – 8.1 цілком сумісні з програмним продуктом. Результати проведення тестових випадків занесено в таблицю. (Додаток Д.)

Під час проведення тестування було виявлено результат на сумісність і несумісність ігрового додатку з різними версіями андроїд. В результаті якого можна сміливо затвердити про те, що додаток сімісний з версіями починаючи від 4.1 до 9. Версії 2019 та 2020 років не тестувались, тому що в них використовувася більш сучасний і більш вимогливий до Hardware телефонів.

**РОЗДІЛ 6  
ГОЛОВНІ АСПЕКТИ ДЛЯ ЗРУЧНОЇ ПРАЦІ ПЕРЕД ПК**

## **6.1. Організація робочих місць**

**Робоче місце**–первинна і основна ланка виробництва, раціональна його організація має найважливіше значення у всьому комплексі питань наукової організації праці. Саме на робочому місці відбувається поєднання елементів виробничого процесу – засобів праці, предметів праці та самого праці. На робочому місці досягається головна мета праці - якісне, економічне і своєчасне виготовлення продукції або виконання встановленого обсягу роботи.

Залежно від типу виробництва, особливостей технологічного процесу, характеру трудових функцій, форм організації праці та інших факторів визначається класифікація робочих місць. Так, за рівнем механізації робочі місця поділяються на автоматизовані, механізовані робочі місця, де виконуються ручні роботи. Механізовані робочі місця в свою чергу поділяються на частково механізовані (робота верстата, механізму і т. д.) і механізовані, автоматизовані а - на напівавтоматизованих і роботизовані.

За ознакою поділу праці робочі місця можуть бути індивідуальними і колективними (бригадними), за спеціалізацією – універсальними, спеціалізованими і спеціальними, за кількістю обслуговуваного устаткування – одностаночными і многостаночными, за ступенем рухливості – стаціонарними і пересувними. Робочі місця можуть перебувати в приміщенні, на відкритому повітрі, на висоті, під землею. Робота на них може виконуватися сидячи, стоячи або з чергуванням тієї та іншої пози.

**Організація робочого місця** –це система заходів щодо його оснащення засобами і предметами праці і розміщенню їх у визначеному порядку.

Організація обслуговування робочого місця означає його забезпечення засобами, предметами праці і послугами, необхідними для здійснення трудового процесу. Основна мета організації робочого місця є досягнення високоякісного й економічно ефективного виконання виробничого завдання у встановлений термін на основі повного використання устаткування, робочого часу, застосування передових методів праці з найменшими фізичними зусиллями, створення безпечних і сприятливих умов ведення робіт. Залежно від специфіки виробництва на організацію робочих місць впливають і інші фактори: співвідношення елементів розумової і фізичної роботи, ступінь її відповідальності. При проектуванні робочих місць повинні бути також враховані освітленість, температура, вологість, тиск, шум, вібрація, пиловиділення і інші санітарно–гігієнічні вимоги до організації робочих місць. Необхідними вимогами є:

* Характеристика робочого місця;
* Загальні вимоги до організації робочого місця;
* Оснащення робочого місця;
* Просторова організація робочого місця та порядок розміщення організаційної оснастки, інструментів, матеріалів;
* Опис організації праці на робочому місці та рекомендовані передові прийоми і методи праці;
* Організація обслуговування робочого місця, способи і засоби зв'язку зі службами обслуговування й управління;
* Умови праці на робочому місці;
* Вимоги безпеки і охорони праці;
* Нормування праці, застосовувані форми і системи оплати праці;
* Документація на робочому місці;
* Економічна ефективність від впровадження типового проекту.

**Оснащення і планування робочих місць**

Оснащення і планування робочих місць - основа їхньої організації. Елементами оснащення робочих місць є основне і допоміжне обладнання, технологічна і організаційна оснастка.

Допоміжне обладнання складається з підйомних пристроїв, різноманітних транспортерів, контрольних приладів, випробувальних стендів та інших підсобних засобів.

Технологічне оснащення включає інструментарій і технічну документацію.

До організаційної оснащення відносяться:

* Пристрої для розміщення і зберігання на робочих місцях технологічного оснащення, заготовок, сировини, матеріалів, готових виробів, відходів;
* Виробнича меблі;
* Засоби сигналізації і зв'язку, місцевого освітлення;
* Предмети догляду за устаткуванням і робочим місцем (щітки, маслянки, гачки, тощо);
* Огороджувальні та запобіжні пристрої;
* Деталі виробничого інтер'єру.

Розташування засобів і предметів праці визначає трудові рухи, їх кількісні та якісні характеристики, площа робочого місця. Удосконалення планування робочого місця має бути спрямоване на усунення зайвих і нераціональних трудових рухів, максимальне скорочення переміщення робочого і матеріальних елементів трудового процесу, а отже, на підвищення ефективності праці та зниження стомлюваності робітника.

Методологічна основа науково обґрунтованої планування робочого місця – її відповідність ергономічним вимогам. Це досягається за рахунок раціонального формування робочих зон і правильного розміщення матеріальних елементів виробництва у відповідності з антропометричними і психофізіологічними даними людини на основі забезпечення робочого необхідного оперативного простору, що дозволяє вільно здійснювати трудові функції. Раціональне планування робочого місця передбачає чіткий порядок і сталість розміщення інструментів і пристосувань, документації, деталей як у процесі роботи, так і при їх зберіганні і забезпечувати зручну робочу позу, виконання трудових процесів з максимальною економією рухів робітника, а також повну безпеку праці. Важливою вимогою є правильне використання відведеної для робочого місця виробничої площі. На робочому місці фіксуються оперативне і допоміжне робочі простору. В оперативному просторі розміщується необхідне обладнання, у допоміжному - рідше використовувані засоби і предмети праці. Оперативний простір може підрозділятися на робочі зони різної значимості. Робоча зона - це ділянка тривимірного простору, обмежений межами досяжності рук в горизонтальному і вертикальному напрямках.

Велике значення має вибір робочої пози, що викликає мінімальне стомлення працівника: "сидячи", "стоячи" або "сидячи - стоячи". Вибір здійснюється з урахуванням фізичних зусиль, необхідних для виконання роботи, її темпу і характеру. Одночасно встановлюється відповідність розміщення обладнання і оснащення нормам вимог безпеки та умов праці.

Важливі вихідні передумови проектування раціонального планування робочого місця - його спеціалізація відповідно до встановленої технологією і формами поділу праці; розроблені методи і прийоми праці; вимоги безпеки і охорони праці.

При плануванні робочих місць необхідно дотримуватись раціональну ширину транспортних проходів і проїздів, а також правильно визначати види підйомно-транспортних засобів. Основні поздовжні і поперечні проїзди повинні бути наскрізними, без тупиків [15].

## **6.2. Налаштування робочого місця програміста**

Сучасний розвиток технічних умов навчальних закладів забезпечує постійну автоматизацію та оптимізацію виробничого процесу. Через масштабний характер роботи, яку учні виконують за допомогою комп’ютерів, українське законодавство чітко визначає норми та вимоги щодо безпосереднього використання комп’ютерної техніки та охорони праці при користуванні комп’ютерами.

Місце, де планується встановити та використовувати комп'ютер, повинно відповідати проектній документації, затвердженій уповноваженим національним агентством. Крім того, також слід враховувати вимоги санітарних норм освітлення, параметри мікроклімату (температура, відносна вологість повітря), рівень та інтенсивність вібрації приміщення, акустичний шум та вогнестійкість, а також електромагнітні та ультрафіолетові характеристики. І інфрачервоне поле.

Елементи природного та штучного освітлення повинні бути в кожному класі, де будуть робочі місця учнів, що працюють на комп’ютерах. При цьому, на вікнах слід встановити легко регульовані жалюзі чи штори, які дозволять працівникам коригувати рівень освітлення у приміщенні.

Хорошою ідеєю є розміщення комп’ютера в аудиторії, щоб світло могло світити на екран монітора з півдня чи північного сходу. Для досягнення максимальної безпеки та здоров’я при використанні комп’ютерів виробничі приміщення повинні бути обладнані аптечками, автоматичними системами пожежної сигналізації та вогнегасниками. У приміщенні, де 5 або більше комп’ютерів працюють разом, на видному місці встановлюється сервісний вимикач, який за потреби дозволяє повністю вимкнути живлення приміщення.

Площа робочого місця повинна бути не менше 6 квадратних метрів. При необхідності сусідні робочі місця учнів, які користуються комп’ютерами, повинні бути відокремлені перегородкою довжиною до 2 метрів. Визначаючи адекватні розміри кімнати та робочого місця кожної людини, ви повинні додатково врахувати шафи, сейфи, шафи чи інші меблі чи обладнання в кімнаті.

На столі працівника можливо розмістити допоміжні для роботи пристрої (принтери, колонки, сканери), а також місця для зберігання документів, за умови, що це не обмежуватиме видимість екрану і не заважатиме студенту. Робочий стілець студента має бути підйомно-поворотним, легко регульованим за висотою та забезпечувати належну підтримку та зручне положення спини і хребта особи. Щодня необхідно проводити вологе прибирання приміщення, очищати робоче місце та безпосередньо монітор комп’ютера від запиленості.

Облаштування робочого місці повинно забезпечувати:

* Правильне розміщення робочого місця у виробничому приміщенні;
* Належні умови освітлення приміщення і робочого місця, відсутність відблисків;
* Належні ергономічні характеристики основних елементів робочого місця;
* Характер та особливості трудової діяльності.

Для зменшення втоми, місця користувачів ЕОМ мають бути зручними.

Конструкція робочого місця користувача ПК (при роботі сидячи) має забезпечувати підтримання оптимальної робочої пози з такими ергономічними характеристиками:

* Ступні ніг – на підлозі або на підставці для ніг;
* Стегна – в горизонтальній площині;
* Передпліччя – вертикально;
* Лікті – під кутом 70 – 90° до вертикальної площини;
* Зап'ястя зігнуті під кутом не більше 20° відносно горизонтальної
* Площини;
* Нахил голови – 15-20° відносно вертикальної площини.

Робоче місце користувача ПК, обладнується робочим столом, стільцем і підставкою для ніг. Висота робочого стола має бути в межах від 0,65 до 0,8 м, а ширина повинна забезпечувати можливість виконання операцій в зоні досяжності моторного поля.

Клавіатуру слід розташовувати на поверхні столу на відстані 200 мм від краю, звернутого до працюючого. У конструкції клавіатури має передбачатися опорний пристрій (виготовлений із матеріалу з високим коефіцієнтом тертя, що перешкоджає його переміщенню), який дає змогу змінювати кут нахилу поверхні клавіатури у межах 5...15°.

Правильно спроектоване і виконане [виробниче освітлення](https://ua-referat.com/%D0%92%D0%B8%D1%80%D0%BE%D0%B1%D0%BD%D0%B8%D1%87%D0%B5_%D0%BE%D1%81%D0%B2%D1%96%D1%82%D0%BB%D0%B5%D0%BD%D0%BD%D1%8F) покращує умови зорової роботи, знижує стомлюваність, сприяє підвищенню продуктивності праці, благотворно впливає на виробниче середовище, надаючи позитивну [психологічну](https://ua-referat.com/%D0%9F%D1%81%D0%B8%D1%85%D0%BE%D0%BB%D0%BE%D0%B3) дію на працюючого, підвищує безпеку праці і знижує травматизм.

Недостатність [освітлення](https://ua-referat.com/%D0%9E%D1%81%D0%B2%D1%96%D1%82%D0%BB%D0%B5%D0%BD%D0%BD%D1%8F) приводить до напруги зору, ослабляє увагу, приводить до настання передчасної стомленості. Надмірно яскраве освітлення викликає засліплення, роздратування і різь в очах. Неправильний напрямок світла на робочому місці може створювати різкі тіні, відблиски, дезорієнтувати працюючого. Всі ці причини можуть призвести до нещасного випадку або профзахворювань, тому [такий](https://ua-referat.com/%D0%A2%D0%B0%D0%BA%D0%B8%D0%B9) важливий правильний розрахунок освітленості.

Існує три види освітлення – [природне](https://ua-referat.com/%D0%9F%D1%80%D0%B8%D1%80%D0%BE%D0%B4%D0%B0), штучне і поєднане ([природне](https://ua-referat.com/%D0%9F%D1%80%D0%B8%D1%80%D0%BE%D0%B4%D0%B0) і штучне разом) [16].

Природне освітлення – освітлення приміщень денним [світлом](https://ua-referat.com/%D0%A1%D0%B2%D1%96%D1%82%D0%BB%D0%BE), що потрапляє через [світлові](https://ua-referat.com/%D0%A1%D0%B2%D1%96%D1%82%D0%BB%D0%BE) прорізи в зовнішніх огороджуючих конструкціях приміщення. [Природне](https://ua-referat.com/%D0%9F%D1%80%D0%B8%D1%80%D0%BE%D0%B4%D0%B0) освітлення характеризується тим, що змінюється в широких межах залежно від часу дня, пори року, характеру області і ряду інших чинників.

[Штучне освітлення](https://ua-referat.com/%D0%A8%D1%82%D1%83%D1%87%D0%BD%D0%B5_%D0%BE%D1%81%D0%B2%D1%96%D1%82%D0%BB%D0%B5%D0%BD%D0%BD%D1%8F) застосовується при роботі в темний час доби і вдень, коли не вдається забезпечити нормовані значення коефіцієнта природного [освітлення](https://ua-referat.com/%D0%9E%D1%81%D0%B2%D1%96%D1%82%D0%BB%D0%B5%D0%BD%D0%BD%D1%8F) (похмура погода, короткий світловий день). [Освітлення](https://ua-referat.com/%D0%9E%D1%81%D0%B2%D1%96%D1%82%D0%BB%D0%B5%D0%BD%D0%BD%D1%8F), при якому недостатнє за нормами природне освітлення доповнюється штучним, називається змішаним освітленням.

[Штучне освітлення](https://ua-referat.com/%D0%A8%D1%82%D1%83%D1%87%D0%BD%D0%B5_%D0%BE%D1%81%D0%B2%D1%96%D1%82%D0%BB%D0%B5%D0%BD%D0%BD%D1%8F) підрозділяється на робоче, аварійне, [евакуаційне](https://ua-referat.com/%D0%95%D0%B2%D0%B0%D0%BA%D1%83%D0%B0%D1%86%D1%96%D1%8F), охоронне. Робоче освітлення, у свою чергу, може бути загальним або комбінованим. Загальне - освітлення, при якому світильники розміщуються у верхній зоні приміщення рівномірно, або, як розташоване устаткування. Комбіноване – освітлення, при якому до загального додається місцеве освітлення.

Згідно СНіП II-4-79 в приміщень обчислювальних центрів необхідно застосувати систему комбінованого освітлення.

При виконанні робіт категорії високої зорової точності (найменший розмір об'єкту розрізнення 0,3 ... 0,5 мм) величина коефіцієнта природного освітлення (КЕО) повинна бути не нижче 1,5%, а при зоровій роботі середньої точності (найменший розмір об'єкту розрізнення 0,5 ... 1,0 мм) КЕО повинен бути не нижче 1,0%. В якості джерел штучного освітлення звичайно використовуються люмінесцентні лампи типа ЛБ, або ДРЛ, які попарно об'єднуються в світильники, які повинні розташовуватися рівномірно над робочими [поверхнями](https://ua-referat.com/%D0%9F%D0%BE%D0%B2%D0%B5%D1%80%D1%85%D0%BD%D1%96).

Вимоги до освітленості в приміщеннях, де встановлені комп'ютери, наступні: при виконанні зорових робіт високої точності загальна [освітленість](https://ua-referat.com/%D0%9E%D1%81%D0%B2%D1%96%D1%82%D0%BB%D0%B5%D0%BD%D0%BD%D1%8F) повинна складати 300лк, а комбінована – 750лк; аналогічні вимоги при виконанні робіт середньої точності – 200 і 300лк [відповідно](https://ua-referat.com/%D0%92%D1%96%D0%B4%D0%BF%D0%BE%D0%B2%D1%96%D0%B4%D1%8C).

Крім [того](https://ua-referat.com/%D0%A2%D0%BE%D0%B3%D0%BE) все поле зору повинне бути освітлено достатньо рівномірно - ця основна гігієнічна вимога. Іншими словами, ступінь освітлення приміщення і [яскравість](https://ua-referat.com/%D0%AF%D1%81%D0%BA%D1%80%D0%B0%D0%B2%D1%96%D1%81%D1%82%D1%8C) екрану комп'ютера повинні бути приблизно однаковими, оскільки яскраве [світло](https://ua-referat.com/%D0%A1%D0%B2%D1%96%D1%82%D0%BB%D0%BE) в районі периферійного зору значно збільшує напруженість очей і, як наслідок, призводить до їх швидкої стомлюваності [17].

## **6.3** [**Ергономічні**](https://ua-referat.com/%D0%95%D1%80%D0%B3%D0%BE%D0%BD%D0%BE%D0%BC%D1%96%D0%BA%D0%B0) **вимоги до робочого місця**

Проектування робочих місць, забезпечених відеотерміналами, відноситься до числа важливих проблем ергономічного проектування в області обчислювальної техніки.

Робоче місце і взаємне розташування всіх його елементів повинне [відповідати](https://ua-referat.com/%D0%92%D1%96%D0%B4%D0%BF%D0%BE%D0%B2%D1%96%D0%B4%D1%8C) антропометричним, фізичним і [психологічним](https://ua-referat.com/%D0%9F%D1%81%D0%B8%D1%85%D0%BE%D0%BB%D0%BE%D0%B3%D1%96%D1%8F) вимогам. Велике значення має також [характер](https://ua-referat.com/%D0%A5%D0%B0%D1%80%D0%B0%D0%BA%D1%82%D0%B5%D1%80) роботи. Зокрема, при організації робочого місця програміста повинні бути дотримані наступні основні умови: оптимальне розміщення устаткування, що до складу робочого місця і достатній робочий [простір](https://ua-referat.com/%D0%9F%D1%80%D0%BE%D1%81%D1%82%D1%96%D1%80), що дозволяє здійснювати всі необхідні рухи і переміщення.

Спроектуйте ергономічні аспекти робочого місця відеотермінала, особливо: висоту робочої поверхні, розмір місця для ніг, вимоги до розташування документів на робочому місці (існування та розмір власника документа, можливість різні) документи, а також відстань очей користувача тощо), характеристики робочого крісла, вимоги до поверхні робочого столу, регульованість елементів робочого місця.

Основними елементами робочого місця програміста є стіл і стілець. Основна робоча поза - сидячи. Робота на відкритому повітрі найменше впливає на втому програміста. Розумне планування робочого місця забезпечує чіткий порядок і послідовність розміщення предметів, методів праці та документів. Речі, необхідні для виконання робіт частіше, знаходяться в межах легкої досяжності робочої зони.

Спортивне поле - це робоче місце, де можна здійснювати рух людей.

Максимальна зона досяжності рук – це частина моторного поля робочого місця, обмеженого дугами, описуваними максимально витягнутими руками при русі їх у плечовому суглобі.

Оптимальна зона – частина моторного поля робочого місця, обмеженого дугами, описуваними передпліччями при русі в ліктьових суглобах з опорою в точці ліктя і з відносно нерухомим плечем [17].

**6.4.Плюси та мінуси професії програміст**

Попит на програмістів і розробників в галузі високих технологій у нашій країні зростає з кожним роком. IT – спеціалісти потрібні як у сільському господарстві, так і в банківській галузі, а також, власне, в самій IT-індустрії. Існує стереотип, що всі айтішники – це програмісти, але це не так. IТ-галузь об’єднує в собі досить багато різнопланових професій, в тому числі й “нетехнічних”. То ж ми почнемо з плюсів цієї цікавої професії.

**Можливість працювати віддалено**

Багато компаній не бояться наймати співробітників на віддалену роботу. Це дає тобі можливість обрати комфортне місце для життя та роботи.

**Можливість розвивати свій проект**

Здобуті навички програміста цілком підходять для того, щоб самостійно створити власну програму сайт. Тому паралельно з роботою або у вільний час ти можеш займатися власним проектом. Творити своє – це розкіш!

**Реальна можливість змінити світ**

Не важливо, чого ти хочеш від своєї професії – заробити багато грошей або змінити світ. З програмуванням у тебе є обидві можливості. Хто знає, можливо, саме ти створиш проект, який змінить наші уявлення про звичні речі.

**Творча робота**

Вирішувати нестандартні завдання, застосовуючи кмітливість – саме те, за що програмістам платять. Варто зазначити, що чим більш високу кваліфікацію має програміст, тим цікавіші завдання йому доводиться вирішувати.

**Попит**

Програмісти не чекають в чергах на роботу. Тенденція діаметрально протилежна – існує великий кадровий дефіцит. І чим вища кваліфікація фахівця – тим більший на нього попит.

**Можна поєднувати з навчанням**

Дуже часто компанії вдаються до найму студентів останніх курсів на вільний графік. Практика показує, що добре мотивований працівник може виконувати довірені йому завдання навіть при частковій зайнятості.

**Постійна самоосвіта**

Професія програміста – це постійний саморозвиток, навчання і підвищення кваліфікації. Одні технології приходять на зміну іншим, і, обравши програмування, нудьгувати тобі не доведеться.

**Сучасна професія**

Комп'ютери сьогодні є у всіх сферах нашого життя. Важко уявити більш сучасну професію, ніж програміст. До того ж, галузь є однією з тих, що розвиваються найбільш динамічно. Ти завжди будеш в курсі нових трендів [18].

У кожної професії є свої плюси але є і мінуси і професія програміста цьому не виключення.

**Робота буває нудною**

Час від часу доводиться виконувати рутинні завдання.

**Конкуренція**

Зараз всі думають, що програмісти дуже затребувані, це правда, але в такому випадку програмістів стає все більше і більше. У такій ситуації спрацьовує система «напрацювання» клієнтів. Якщо ти себе зарезервував як хорошого фахівця і при цьому з прийнятними цінами на свої послуги, то за відгуками можна набрати дуже хорошу клієнтську базу, але, на жаль, на це піде занадто багато часу, може, рік, може, два, а то і зовсім кілька років.

**Повний 8**–**годинний день**

В інших сферах не завжди люди працюють повні 8 годин. Наприклад, водій, у якого є можливість віддихатися, пожежний, який не завжди гасить вогонь, і так далі. В IT ви реально 8 годин будете займатися розумовою діяльністю, яка, на відміну від фізичної, є дуже важкою діяльністю. Не дарма за цю роботу добре платять, тому що це реально важко, це вимотує морально і стомлює.

**Необхідність вивчення англійської мови**

Так склалося, що робота програміста пов'язана з використанням англомовних джерел інформації. Тому тобі доведеться встановити планку на мінімально необхідний рівень знань мови і взяти цю висоту.

**У маленьких проектах доводиться бути людиною-оркестром**

Якщо це стартап або просто маленька компанія, то на одному і тому ж проекті може знадобитися програмувати, верстати, налаштовувати сервера та спілкуватися з клієнтом одночасно. Це великий мінус невеликих студій і агентств.

**Мала кількість спілкування**

Коло спілкування програміста в рамках роботи зазвичай складається не більше ніж з 5–10 чоловік, а більшу частину робочого часу доводиться проводити за роботою. Екстравертам і просто товариським людям знадобиться деякий час, щоб адаптуватися до такого режиму.

**Шкода здоров'ю**

Сидяча робота може спровокувати ряд захворювань хребта і кишківника. Постійне навантаження на очі, від якого страждає зір.

**Складність**

Мало хто з початківців розуміє з чим йому доведеться зіткнутися. Наприклад, в компаніях, які підтримують свій продукт вже більше 3 років, логіка стає надто складно. Адже на системі зав'язано все – фінанси, логістика, технічна сторона. Коли ти розробляєш для себе міні-блог, гостьову книгу або CMS-очку – це одне. Коли ти копаєшся в кишках системи, яку писали років 10, з дуже серйозною бізнес-логікою – це інше.

**Постійна некомпетентність**

Якщо який-небудь зварювальник або електрик як професіонал знає свою справу «від» і «до», то програміст, приходячи на роботу, стикається з новою бізнес-логікою, яка далеко не завжди очевидна і може бути дуже складною.

Виходить, що програміст постійно некомпетентний. Тому що знати мову – це одне, а вміти розбиратися в логіці, в чужому коді і мати витримку розбиратися в цьому коді – дано не кожному. І це дуже і дуже важко. Якщо взяти, наприклад, хлопця, який ходить і чинить газові пальники, так це вкрай легко в порівнянні з програмуванням.

Тому що один раз засвоївши тривіальні речі, ти працюєш і тебе ніщо не тривожить, ти знаєш свою справу. У програмуванні ж ти кожен раз стикаєшся зі своєю компетентністю, тому що на кожному новому проекті ти незнайомий з бізнес-логікою [19].

**РОЗДІЛ 7  
ВИБІР МОВИ ПРОГРАМУВАННЯ ТА ОПИС ХАРАКТЕРИСТИК**

## **7.1.Мова програмування Java та її характеристики**

**Java** (вимовляється Джава) — [об'єктно-орієнтована](https://uk.wikipedia.org/wiki/%D0%9E%D0%B1%27%D1%94%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D1%96%D1%94%D0%BD%D1%82%D0%BE%D0%B2%D0%B0%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) [мова програмування](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F), випущена 1995 року компанією [«Sun Microsystems»](https://uk.wikipedia.org/wiki/Sun_Microsystems) як основний компонент платформи Java. З 2009 року мовою займається компанія [«Oracle»](https://uk.wikipedia.org/wiki/Oracle_Corporation), яка того року придбала «Sun Microsystems». В офіційній реалізації Java-програми [компілюються](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%96%D0%BB%D1%8F%D1%82%D0%BE%D1%80) у [байт-код](https://uk.wikipedia.org/wiki/%D0%91%D0%B0%D0%B9%D1%82-%D0%BA%D0%BE%D0%B4), який при виконанні інтерпретується [віртуальною машиною](https://uk.wikipedia.org/wiki/%D0%92%D1%96%D1%80%D1%82%D1%83%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0_%D0%BC%D0%B0%D1%88%D0%B8%D0%BD%D0%B0_Java) для конкретної платформи.

«Oracle» надає компілятор Java та [віртуальну машину Java](https://uk.wikipedia.org/wiki/%D0%92%D1%96%D1%80%D1%82%D1%83%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0_%D0%BC%D0%B0%D1%88%D0%B8%D0%BD%D0%B0_Java), які задовольняють специфікації [Java Community Process](https://uk.wikipedia.org/wiki/Java_Community_Process), під ліцензією [GNU General Public License](https://uk.wikipedia.org/wiki/GNU_General_Public_License).

Мова запозичує багато синтаксису з C та C ++. Зокрема, в якості основи була використана об'єктна модель С ++, але вона була модифікована. Виключає можливість виникнення певних конфліктів, які можуть виникнути через помилки програміста, та сприяє процесу розробки об’єктно-орієнтованих програм. Багато операцій, які програміст повинен виконати на C / C ++, делегуються віртуальній машині.

Спочатку Java була розроблена як незалежна від платформи мова, тому вона має менше апаратних функцій низького рівня, що зменшує швидкість роботи програм порівняно з C ++ тощо. При необхідності Java дозволяє викликати підпрограми, написані іншими мовами програмування. Java вплинула на розробку J ++, розроблену Microsoft. Оскільки Sun Microsystems подала позов до суду, робота над J ++ була припинена, оскільки мова програмування була модифікацією Java. Пізніше в новій платформі «Microsoft» [.NET](https://uk.wikipedia.org/wiki/.NET_Framework) випустили [J#](https://uk.wikipedia.org/wiki/Visual_J_Sharp), щоб полегшити міграцію програмістів J++ або Java на нову платформу. З часом нова мова програмування [С#](https://uk.wikipedia.org/wiki/C_Sharp) стала основною мовою платформи, перейнявши багато чого з Java. J# востаннє включався в версію [Microsoft Visual Studio](https://uk.wikipedia.org/wiki/Microsoft_Visual_Studio) 2005. Мова сценаріїв [JavaScript](https://uk.wikipedia.org/wiki/JavaScript) має схожу із Java назву і синтаксис, але не пов'язана із Java.

**Об'єктність**

На противагу [C++](https://uk.wikipedia.org/wiki/C%2B%2B), Java є більш об'єктно-орієнтованою. Всі дані і дії групуються в класи [об'єктів](https://uk.wikipedia.org/wiki/%D0%9E%D0%B1%27%D1%94%D0%BA%D1%82_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)). Виключенням з повної об'єктності (як скажімо в [Smalltalk](https://uk.wikipedia.org/wiki/Smalltalk)) є примітивні типи (int, float тощо). Це було свідомим рішенням проєктувальників мови задля збільшення швидкості. Через це Java не вважається повністю об'єктно-орієнтовною мовою.

У Java всі об'єкти є похідними від об'єктів (це просто називається "Об'єкт"), від яких вони успадковують основні способи поведінки та властивості.

Незважаючи на те, що багаторазове успадкування вперше надається в C ++, лише одне успадкування може бути реалізовано в Java, що виключає можливість конфліктів між членами класу (методами та змінними), успадкованими від базового класу.

Стандартна бібліотека забезпечує загальний метод доступу до залежних від платформи функцій, таких як обробка графіки, багатопотоковість та мережеві роботи. У деяких версіях для покращення продуктивності JVM байт-код може бути скомпільований у машинний код до або під час виконання програми.

Головною перевагою використання байт-коду є портативність. Однак додаткові витрати на інтерпретацію означають, що інтерпретовані програми майже завжди працюють повільніше, ніж програми, скомпільовані в машинний код, саме тому Java заслужила репутацію "повільної" мови. Однак цей розрив значно зменшився після введення декількох методів оптимізації у сучасних реалізаціях JVM.

**Безпека**

В Java існує система винятків або ситуацій, коли програма зустрічається з неочікуваними труднощами, наприклад:

* Операції над елементом масиву поза його межами або над порожнім елементом;
* Читання з недоступного каталогу або неправильної адреси URL;
* Ввід недопустимих даних користувачем.

Одна з особливостей концепції [віртуальної машини](https://uk.wikipedia.org/wiki/%D0%92%D1%96%D1%80%D1%82%D1%83%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0_%D0%BC%D0%B0%D1%88%D0%B8%D0%BD%D0%B0) полягає в тому, що помилки (виключення) не призводять до повного краху системи. Крім того, існують інструменти, які «приєднуються» до середовища періоду виконання і кожен раз, коли сталося певне виключення, записують інформацію з пам'яті для [зневадження](https://uk.wikipedia.org/wiki/%D0%97%D0%BD%D0%B5%D0%B2%D0%B0%D0%B4%D0%B6%D0%B5%D0%BD%D0%BD%D1%8F) програми. Ці інструменти [автоматизованої обробки виключень](https://uk.wikipedia.org/w/index.php?title=%D0%90%D0%B2%D1%82%D0%BE%D0%BC%D0%B0%D1%82%D0%B8%D1%87%D0%BD%D0%B0_%D0%BE%D0%B1%D1%80%D0%BE%D0%B1%D0%BA%D0%B0_%D0%B2%D0%B8%D0%BA%D0%BB%D1%8E%D1%87%D0%B5%D0%BD%D1%8C&action=edit&redlink=1) надають основну інформацію щодо виключень в програмах на Java [20].

## **7.2.Мова програмування С++ та її характеристики**

**C++** (Сі-плюс-плюс) — [мова програмування](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) високого рівня з підтримкою кількох [парадигм програмування](https://uk.wikipedia.org/wiki/%D0%9F%D0%B0%D1%80%D0%B0%D0%B4%D0%B8%D0%B3%D0%BC%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F): [об'єктно-орієнтованої](https://uk.wikipedia.org/wiki/%D0%9E%D0%B1%27%D1%94%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D1%96%D1%94%D0%BD%D1%82%D0%BE%D0%B2%D0%B0%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F), [узагальненої](https://uk.wikipedia.org/wiki/%D0%A3%D0%B7%D0%B0%D0%B3%D0%B0%D0%BB%D1%8C%D0%BD%D0%B5%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) та [процедурної](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D0%B4%D1%83%D1%80%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F). Розроблена [Б'ярном Страуструпом](https://uk.wikipedia.org/wiki/%D0%91%27%D1%8F%D1%80%D0%BD_%D0%A1%D1%82%D1%80%D0%B0%D1%83%D1%81%D1%82%D1%80%D1%83%D0%BF) ([англ.](https://uk.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D1%96%D0%B9%D1%81%D1%8C%D0%BA%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) BjarneStroustrup) в [AT&T](https://uk.wikipedia.org/wiki/AT%26T) [Bell Laboratories](https://uk.wikipedia.org/wiki/Bell_Labs) (Мюррей-Хілл, [Нью-Джерсі](https://uk.wikipedia.org/wiki/%D0%9D%D1%8C%D1%8E-%D0%94%D0%B6%D0%B5%D1%80%D1%81%D1%96)) 1979 року та початково отримала назву «Сі з [класами»](https://uk.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B0%D1%81%D0%B8_C%2B%2B). Згодом Страуструп перейменував мову на *C++* у [1983](https://uk.wikipedia.org/wiki/1983)р. Базується на мові програмування [С](https://uk.wikipedia.org/wiki/%D0%A1_(%D0%BC%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)). Вперше описана стандартом [ISO](https://uk.wikipedia.org/wiki/ISO)/IEC 14882:1998, найбільш актуальним же є стандарт ISO/IEC 14882:2020.

У 1990-х роках С++ стала однією з найуживаніших мов програмування загального призначення. Мову використовують для системного програмування, розробки програмного забезпечення, написання драйверів, потужних серверних та клієнтських програм, а також для розробки розважальних програм, наприклад, відеоігор. С++ суттєво вплинула на інші популярні сьогодні мови програмування: [С#](https://uk.wikipedia.org/wiki/C_Sharp) та [Java](https://uk.wikipedia.org/wiki/Java).

**Подальший розвиток**

С++ продовжує розвиватися, щоб відповідати сучасним вимогам. Одна з груп, що займаються мовою С++ в її сучасному вигляді і що направляють комітету зі стандартизації С++ поради з її поліпшення, — це Boost. Наприклад, один з напрямів діяльності цієї групи — вдосконалення можливостей мови шляхом додавання в неї особливостей метапрограмування.

Стандарт С++ не описує способи іменування об'єктів, деякі деталі обробки винятків і інші можливості, пов'язані з деталями реалізації, що робить несумісним об'єктний код, створений різними компіляторами. Проте для цього третіми особами створено безліч стандартів для конкретної архітектури і операційних систем.

Проте (за станом на час написання цієї статті) серед компіляторів С++ все ще продовжується битва за повну реалізацію стандарту С++, особливо в області шаблонів — частини мови, зовсім недавно повністю розробленій комітетом стандартизації.

Одним із каменів спотикання у цьому питанні є ключове слово *export*, що використовується також і для розділення оголошення і визначення шаблонів.

Першим компілятором, що підтримав export в шаблонах, став Comeau C++ на початку 2003 року (п'ять років після виходу стандарту). У 2004 році бета-версія компілятора Borland [C++ Builder](https://uk.wikipedia.org/wiki/C%2B%2B_Builder) X також почала його підтримку.

Обидва цих компілятора засновані на зовнішньому інтерфейсі EDG. Інші компілятори, такі як [Microsoft Visual C++](https://uk.wikipedia.org/wiki/Microsoft_Visual_C%2B%2B) або [GCC](https://uk.wikipedia.org/wiki/GCC), взагалі цього не підтримують. Ерб Саттер (Herb Sutter), секретар комітету із стандартизації С++, рекомендував прибрати export з майбутніх версій стандарту унаслідок серйозних складнощів в повноцінній реалізації, проте згодом остаточним рішенням було вирішено його залишити.

Із списку інших проблем, пов'язаних з шаблонами, можна навести питання конструкцій часткової спеціалізації шаблонів, які погано підтримувалися протягом багатьох років після виходу стандарту Сі++.

При створенні С++ прагнули зберегти сумісність з мовою [С](https://uk.wikipedia.org/wiki/%D0%A1%D1%96_(%D0%BC%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)). Більшість програм на С справно працюватимуть і з компілятором С++. С++ має синтаксис, заснований на синтаксисі С.

Нововведеннями С++ порівняно з С є:

* Підтримка [об'єктно-орієнтованого програмування](https://uk.wikipedia.org/wiki/%D0%9E%D0%B1%27%D1%94%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D1%96%D1%94%D0%BD%D1%82%D0%BE%D0%B2%D0%B0%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) через [класи](https://uk.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B0%D1%81_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F));
* Підтримка [узагальненого програмування](https://uk.wikipedia.org/wiki/%D0%A3%D0%B7%D0%B0%D0%B3%D0%B0%D0%BB%D1%8C%D0%BD%D0%B5%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) через шаблони;
* Доповнення до стандартної бібліотеки;
* Додаткові [типи даних](https://uk.wikipedia.org/wiki/%D0%A2%D0%B8%D0%BF_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85);
* [Обробка винятків](https://uk.wikipedia.org/wiki/%D0%9E%D0%B1%D1%80%D0%BE%D0%B1%D0%BA%D0%B0_%D0%B2%D0%B8%D0%BD%D1%8F%D1%82%D0%BA%D1%96%D0%B2);
* Простори імен;
* Вбудовані функції;
* [Перевантаження операторів](https://uk.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D0%B5%D0%B2%D0%B0%D0%BD%D1%82%D0%B0%D0%B6%D0%B5%D0%BD%D0%BD%D1%8F_%D0%BE%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%BE%D1%80%D1%96%D0%B2);
* Перевантаження імен функцій;
* Посилання і оператори управління вільно розподіленою пам'яттю.

У 1998 році ратифіковано міжнародний стандарт мови С++: ISO/IEC 14882 «Standard for the C++ Programming Language». Поточна версія цього стандарту — ISO/IEC 14882:2011 [21].

## **7.3.Мова програмування С# та її характеристики**

**C#** (вимовляється Сі-шарп) — [об'єктно-орієнтована мова програмування](https://uk.wikipedia.org/wiki/%D0%9E%D0%B1%27%D1%94%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D1%96%D1%94%D0%BD%D1%82%D0%BE%D0%B2%D0%B0%D0%BD%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) з безпечною [системою типізації](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%82%D0%B8%D0%BF%D1%96%D0%B7%D0%B0%D1%86%D1%96%D1%97) для платформи [.NET](https://uk.wikipedia.org/wiki/.NET). Розроблена [Андерсом Гейлсбергом](https://uk.wikipedia.org/wiki/%D0%90%D0%BD%D0%B4%D0%B5%D1%80%D1%81_%D0%93%D0%B5%D0%B9%D0%BB%D1%81%D0%B1%D0%B5%D1%80%D0%B3), [Скотом Вілтамутом](https://uk.wikipedia.org/w/index.php?title=%D0%92%D1%96%D0%BB%D1%82%D0%B0%D0%BC%D1%83%D1%82_%D0%A1%D0%BA%D0%BE%D1%82&action=edit&redlink=1) та [Пітером Гольде](https://uk.wikipedia.org/w/index.php?title=%D0%93%D0%BE%D0%BB%D1%8C%D0%B4%D0%B5_%D0%9F%D1%96%D1%82%D0%B5%D1%80&action=edit&redlink=1) під егідою [Microsoft Research](https://uk.wikipedia.org/wiki/Microsoft_Research) (належить [Microsoft](https://uk.wikipedia.org/wiki/Microsoft)).

[**Синтаксис**](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BD%D1%82%D0%B0%D0%BA%D1%81%D0%B8%D1%81) C# близький до [С++](https://uk.wikipedia.org/wiki/%D0%A1%2B%2B) і [Java](https://uk.wikipedia.org/wiki/Java). [Мова](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B2%D0%B0) має строгу статичну типізацію, підтримує [поліморфізм](https://uk.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D1%96%D0%BC%D0%BE%D1%80%D1%84%D1%96%D0%B7%D0%BC_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)), перевантаження операторів, вказівники на функції-члени класів, атрибути, події, властивості, [винятки](https://uk.wikipedia.org/wiki/%D0%9E%D0%B1%D1%80%D0%BE%D0%B1%D0%BA%D0%B0_%D0%B2%D0%B8%D0%BD%D1%8F%D1%82%D0%BA%D1%96%D0%B2), коментарі у форматі [XML](https://uk.wikipedia.org/wiki/XML). Перейнявши багато від своїх попередників — мов [С++](https://uk.wikipedia.org/wiki/%D0%A1%2B%2B), [Object Pascal](https://uk.wikipedia.org/wiki/Object_Pascal), [Модула](https://uk.wikipedia.org/w/index.php?title=%D0%9C%D0%BE%D0%B4%D1%83%D0%BB%D0%B0&action=edit&redlink=1) і [Smalltalk](https://uk.wikipedia.org/wiki/Smalltalk) — С#, спираючись на практику їхнього [використання](https://uk.wikipedia.org/wiki/%D0%92%D0%B8%D0%BA%D0%BE%D1%80%D0%B8%D1%81%D1%82%D0%B0%D0%BD%D0%BD%D1%8F), виключає деякі моделі, що зарекомендували себе як проблематичні при розробці програмних систем, наприклад, мова С#, на відміну від C++, не передбачає [множинне успадкування](https://uk.wikipedia.org/wiki/%D0%9C%D0%BD%D0%BE%D0%B6%D0%B8%D0%BD%D0%BD%D0%B5_%D1%83%D1%81%D0%BF%D0%B0%D0%B4%D0%BA%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) класів.

C# є дуже близьким родичем мови [програмування](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) [Java](https://uk.wikipedia.org/wiki/Java). Мова Java була створена компанією [Sun Microsystems](https://uk.wikipedia.org/wiki/Sun_Microsystems), коли глобальний розвиток інтернету поставив завдання [розподілених обчислень](https://uk.wikipedia.org/wiki/%D0%A0%D0%BE%D0%B7%D0%BF%D0%BE%D0%B4%D1%96%D0%BB%D0%B5%D0%BD%D1%96_%D0%BE%D0%B1%D1%87%D0%B8%D1%81%D0%BB%D0%B5%D0%BD%D0%BD%D1%8F). Взявши за основу популярну мову C++, Java виключила з неї потенційно небезпечні речі (на зразок вказівників без контролю виходу за межі). Для розподілених обчислень була створена концепція віртуальної машини та машинно-незалежного байт-коду, свого роду посередника між вихідним текстом програм і апаратними інструкціями комп'ютера чи іншого інтелектуального пристрою.

Нововведенням C# стала можливість легшої взаємодії, порівняно з мовами-попередниками, з кодом програм, написаних на інших мовах, що є важливим при створенні великих проєктів. Якщо програми на різних мовах виконуються на платформі [.NET](https://uk.wikipedia.org/wiki/.NET), .NET бере на себе клопіт щодо сумісності програм (тобто типів даних, за кінцевим рахунком).

Станом на сьогодніC# визначено флагманською мовою корпорації Microsoft, бо вона найповніше використовує нові можливості .NET. Решта мов програмування, хоч і підтримуються, але визнані такими, що мають спадкові прогалини щодо використання .NET. Рядок в C# є посилальним типом.

**Препроцесор**

C# має «препроцесорні директиви» (хоча насправді він не має [препроцесора](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%B5%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D0%BE%D1%80)) на основі препроцесора C, це дає програмісту можливість визначити символи, але не [макроси](https://uk.wikipedia.org/wiki/%D0%9C%D0%B0%D0%BA%D1%80%D0%BE%D1%81). Умовні директиви, такі як #if, #endif, чи #else також можливі. Директиви типу #region дають натяк редактору для згортання фрагментів коду.

Хоча визначення мови C# і CLI стандартизовані ISO та Ecma, що забезпечує розумний і недискримінаційний ліцензійний захист (RAND) від патентних позовів, Microsoft використовує C# і CLI у своїй бібліотеці Base Class Library (BCL), яка є фундаментом їхньої власницької платформи [.NET framework](https://uk.wikipedia.org/wiki/.NET_framework), і яка забезпечує безліч нестандартизованих класів (розширений I/O, [GUI](https://uk.wikipedia.org/wiki/GUI) Windows Forms, веб-служби тощо).

У деяких випадках патенти Microsoft пов'язані зі стандартами, що використовуються в середовищі .NET, задокументовані корпорацією Майкрософт, а застосовані патенти можуть бути отримані за допомогою інших речень RAND або за допомогою Microsoft Open Specification Pledge (OSP), що публікує патентні права. Однак є деякі попередження та дискусії щодо інших аспектів патентів Microsoft, які не охоплюються, і ці аспекти можуть бути незалежними розробниками, які мають повну структуру.

Корпорація Майкрософт також погодилася не подавати позов проти розробників програмного забезпечення з відкритим кодом за некомерційні проекти за частину системи, охопленої її OSP. Корпорація Майкрософт погодилася не порушувати патентні вимоги продуктів Novell до платників клієнтів Novell, за винятком списків продуктів, у яких явно не згадується реалізація C #, .NET або Novell .NET (проект Mono). Однак Novell вважає, що Mono не порушує жодних патентів Microsoft. Microsoft також уклав спеціальну угоду не позиватися проти браузерного [плагіну](https://uk.wikipedia.org/wiki/%D0%9F%D0%BB%D0%B0%D0%B3%D1%96%D0%BD) [Moonlight](https://uk.wikipedia.org/wiki/Moonlight), який спирається на Mono, отриманного від [Novell](https://uk.wikipedia.org/wiki/Novell).

У зауваженні, опублікованому на сайті новин [Free Software Foundation](https://uk.wikipedia.org/wiki/Free_Software_Foundation) у червні 2009 [Річард Столлман](https://uk.wikipedia.org/wiki/%D0%A0%D1%96%D1%87%D0%B0%D1%80%D0%B4_%D0%A1%D1%82%D0%BE%D0%BB%D0%BB%D0%BC%D0%B0%D0%BD) попереджає, що він вважає, що «Microsoft можливо планує одного дня оголосити всі вільні реалізації C# такими, що використовують програмні патенти» і рекомендував розробникам уникати того, що він називає «безвідплатним ризиком», пов'язаним із «залежністю вільних реалізацій C#». Free Software Foundation пізніше повторила свої попередження, стверджуючи, що розширення Microsoft Community Promise на специфікації ECMA C# і CLI можуть не вберігти від шкідництва Microsoft відкритим реалізаціям C#, оскільки багато специфічних для Windows бібліотек, включених у .NET та Mono, не покриті цими обіцянками. Тому більшість провідних дистрибутивів Лінукс, за винятком Novell [SUSE Linux](https://uk.wikipedia.org/wiki/SUSE_Linux), не включають Mono в установку за умовчанням (хоча його і можна завантажити з репозиторіїв) [22].

#### ВИСНОВОК

У даній дипломній роботі було розроблено додаток для ОС Android. Також було проведено аналіз ігрових рушіїв та їх порівняння та опис характеристик. В підсумку було обрано ігровий рушій Unity 3D на якому і було реалізовано дану програму.

Можна підвести підсумки даної дипломної роботи, що саме було реалізовано:

* Для написання коду було обрано мову програмування С#, дану мову було порівнянно з іншими сучасними і не менш популярними мовами програмування а саме такими Java та C++, але все ж таки було обрано С# тому, що C# містить конструкції checked і unchecked, що дозволяють локально включати і вимикати динамічну перевірку арифметичного переповнювання, C# покладається на реалізацію, гарантії строго однакових результатів обчислень немає goto зберігся, його звичайне використання - передача управління на різні мітки case в операторові switch і вихід з вкладеного циклу та багато інших зручностей;
* Проведено зрівняння 3 популярних рушіїв для програмування, таких як: Unity3D, Creation Engine та Unreal Engine. Було обрано популярний і зручний рушій Unity3D. Unity - більше, ніж движок, це середовище для розробки комп'ютерних ігор, в якій об'єднані різні програмні засоби, використовувані при створенні ПЗ, -- текстовий редактор, компілятор, відладчик і так далі. При цьому, завдяки зручності використання, Unity робить створення ігор максимально простим і комфортним, а мультиплат-форменність движка дозволяє створювачам ігор охопити як можна більшу кількість ігрових платформ і операційних систем, перевага движка - наявність величезної бібліотеки ассетів і плагінів, за допомогою яких можна значно прискорити процес розробки гри. Їх можна імпортувати і експортувати, додавати в гру цілі заготівлі - рівні, ворогів, патерни поведінки ІІ і так далі. Ніякої метушні з програмуванням. Багато ассетів доступні безкоштовно, інші пропонуються за невелику суму, і за бажання можна створювати власний контент, публікувати його в Unity Asset Store і отримувати від цього прибуток.

Дана дипломна робота не охоплює всіх аспектів розробки ігрових додатків, оскільки існує велика кількість інших рушіїв, та цільових платформ для яких можуть бути створені продукти такого типу, описати їх досить складно, навіть в рамках однієї платформи.

**CПИCОК ВИКОPИCТАНОЇ ЛIТEPАТYPИ**

1. Чого варто очікувати від ринку мобільних ігор наступного року [Електронний ресурс] – Режим доступу: [https://nv.ua/ukr/technо /technоblоgs /tendencii-razvitiya-mоbilnоgо-gamedev-a-i-kakie-nоvinki-sleduet-оzhidat-v-2019 -gоdu-50045260.html](https://nv.ua/ukr/technо%20/technоblоgs%20/tendencii-razvitiya-mоbilnоgо-gamedev-a-i-kakie-nоvinki-sleduet-оzhidat-v-2019%20-gоdu-50045260.html)
2. Індустрія відеоігор [Електронний ресурс] – Режим доступу: <http://surl.li/smrx>
3. Ігрова індустрія до 2020 року — мільярдні прибутки, епоха мобільних ігор та спортивний бізнес [Електронний ресурс] – Режим доступу: [https://nachasi.cоm/2018/08/20/igrоva-industriya-dо-2020/](https://nachasi.com/2018/08/20/igrova-industriya-do-2020/)
4. 10 найкращих ігрових рушіїв [Електронний ресурс] – Режим доступу: <https://ulab.sumdu.edu.ua/uk/10-najkrashhih-igrovih-rushiiv>
5. ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ДЛЯ РОЗРОБКИ КОМП’ЮТЕРНИХ ІГОР [Електронний ресурс] – Режим доступу: <http://eprints.zu.edu.ua/24825/1/Tomashevska_APSI2017.pdf>
6. Що таке ігровий рушій? 10 найкращих ігрових рушіїв. [Електронний ресурс] – Режим доступу: <https://ulab.sumdu.edu.ua/uk/10-najkrashhih-igrovih-rushiiv>
7. Unity (рушій гри) [Електронний ресурс] – Режим доступу: <http://surl.li/smrz>
8. Creation Engine [Електронний ресурс] – Режим доступу: <http://surl.li/smsa>
9. Діаграми Прецедентів (Use Case UML Diagram) [Електронний ресурс] – Режим доступу: <http://lvivqaclub.blogspot.com/2008/10/use-case-uml-diagram.html>
10. Алгоритм [Електронний ресурс] – Режим доступу: <http://surl.li/smsd>
11. Інтерфейс [Електронний ресурс] – Режим доступу: <http://surl.li/smse>
12. Adobe Photoshop [Електронний ресурс] – Режим доступу: <http://surl.li/azne>
13. Тестовий випадок [Електронний ресурс] – Режим доступу: <http://surl.li/smsg>
14. Android [Електронний ресурс] – Режим доступу: <http://surl.li/smsh>
15. Організація робочих місць [Електронний ресурс] – Режим доступу: [https://studme.com.ua/1924070113856/bzhd/organizatsiya\_rabochih\_ mest.htm](https://studme.com.ua/1924070113856/bzhd/organizatsiya_rabochih_%20mest.htm)
16. Довідкова книга для проектування електричного освітлення. / Под ред. Г.Б. Кнорринга. - Л.:Енергія, 1976.
17. Самгін Е.Б. Освітлення робочих місць. - М.: МІРЕА, 1989. – 186 с.
18. Зінченко В.П. Основи ергономіки. - М.: МГУ, 1979. – 179 с.
19. Плюси та мінуси професії програміст [Електронний ресурс] – Режим доступу: <https://bezoman.pp.ua/uk/questions/48/плюси-і-мінуси-професії-програміст>
20. Java [Електронний ресурс] – Режим доступу: <http://surl.li/cyna>
21. C++ [Електронний ресурс] – Режим доступу: <http://surl.li/gijw>
22. C Sharp [Електронний ресурс] – Режим доступу: <http://surl.li/dafp>

# ДОДАТОК А ПОРІВНЯЛЬНА ТАБЛИЦЯ ІГРОВИХ РУШІЇВ

На основі проведеного порівняння дані можна занести у таблицю, в якій якість того чи іншого аспекту буде оцінюватися за шкалою від одного до трьох, де 1 – найгірший результат, 3 – найращий результат, а 2 – результат кращий результат за конкурента, але не може зрівнятися з рівнем лідера.

Таблиця – Порівняння ігрових рушіїв

|  |  |  |  |
| --- | --- | --- | --- |
| Характеристика рушія | Unity3D | Unreal Engine | Creation Engine |
| Зручність використання | 3 | 1 | 2 |
| Ціна | 3 | 1 | 2 |
| Створення скрипта | 3 | 2 | 2 |
| Графічний дизайн | 1 | 2 | 3 |
| Звуковий супровід | 1 | 2 | 3 |
| Адаптований до платформ | 3 | 1 | 1 |
| Використання ресурсів в процесі роботи | 3 | 2 | 1 |

**ДОДАТОК Б ДІАГРАМА ПРЕЦЕДЕНТІВ**



**ДОДАТОК В ДІАГРАМА ПОСЛІДОВНОСТЕЙ**



**ДОДАТОК Г ЛІСТИНГ ПРОГРАМИ**

Скрипт GameController

/// <summary>

/// Game controller.

/// This script use for control game loading and spawn character when load complete

/// </summary>

using UnityEngine;

using System.Collections;

[ExecuteInEditMode]

public class GameController : MonoBehaviour {

public PatternSystem patSysm; //pattern system

public CameraFollow cameraFol; //camera

public float speedAddEveryDistance = 300;

public float speedAdd = 0.5f;

public float speedMax = 20;

public int selectPlayer;

public GameObject[] playerPref;

public Vector3 posStart;

public bool previewProgressBar;

public bool useShowPercent;

public Texture2D textureProgressBar\_Frame, textureProgressBar\_Color;

public Rect rect\_progressbar, rect\_percent\_text;

private float percentCount;

private float distanceCheck;

[HideInInspector]

public int countAddSpeed;

private CalOnGUI calOnGUI;

public static GameController instace;

void Start(){

if(Application.isPlaying == true){

selectPlayer = PlayerPrefs.GetInt("SelectPlayer");

instace = this;

calOnGUI = GetComponent<CalOnGUI>();

StartCoroutine(WaitLoading());

}

}

void OnGUI(){

if(Application.isPlaying == true){

if(patSysm.loadingComplete == false){

percentCount = Mathf.Lerp(percentCount, patSysm.loadingPercent, 5 \* Time.deltaTime);

GUI.BeginGroup(new Rect(calOnGUI.SetGUI\_Left(rect\_progressbar.x, rect\_progressbar.y, rect\_progressbar.width\*(percentCount/100), rect\_progressbar.height)));

if(textureProgressBar\_Color == null){

GUI.Box(new Rect(0,0, calOnGUI.SetGUI\_Left(rect\_progressbar.x, rect\_progressbar.y, rect\_progressbar.width, rect\_progressbar.height).width, calOnGUI.SetGUI(rect\_progressbar.x, rect\_progressbar.y, rect\_progressbar.width, rect\_progressbar.height).height),"");

}else{

GUI.DrawTexture(new Rect(0,0, calOnGUI.SetGUI\_Left(rect\_progressbar.x, rect\_progressbar.y, rect\_progressbar.width, rect\_progressbar.height).width, calOnGUI.SetGUI(rect\_progressbar.x, rect\_progressbar.y, rect\_progressbar.width, rect\_progressbar.height).height),textureProgressBar\_Color);

}

GUI.EndGroup();

if(textureProgressBar\_Frame != null){

GUI.DrawTexture(new Rect(calOnGUI.SetGUI\_Left(rect\_progressbar.x, rect\_progressbar.y, rect\_progressbar.width, rect\_progressbar.height)), textureProgressBar\_Frame); }

if(useShowPercent)

GUI.Label(new Rect(calOnGUI.SetGUI(rect\_percent\_text.x, rect\_percent\_text.y, rect\_percent\_text.width , rect\_percent\_text.height)),percentCount.ToString("0")+"%");

}

}else{

if(previewProgressBar == true){

if(calOnGUI == null){

calOnGUI = new CalOnGUI();

}

GUI.BeginGroup(new Rect(calOnGUI.SetGUI\_Left(rect\_progressbar.x, rect\_progressbar.y, rect\_progressbar.width , rect\_progressbar.height)));

if(textureProgressBar\_Color == null){

GUI.Box(new Rect(0,0, calOnGUI.SetGUI\_Left(rect\_progressbar.x, rect\_progressbar.y, rect\_progressbar.width, rect\_progressbar.height).width, calOnGUI.SetGUI(rect\_progressbar.x, rect\_progressbar.y, rect\_progressbar.width, rect\_progressbar.height).height),"");

}else{

GUI.DrawTexture(new Rect(0,0, calOnGUI.SetGUI\_Left(rect\_progressbar.x, rect\_progressbar.y, rect\_progressbar.width, rect\_progressbar.height).width, calOnGUI.SetGUI(rect\_progressbar.x, rect\_progressbar.y, rect\_progressbar.width, rect\_progressbar.height).height),textureProgressBar\_Color);

}

GUI.EndGroup();

if(textureProgressBar\_Frame != null){

GUI.DrawTexture(new Rect(calOnGUI.SetGUI\_Left(rect\_progressbar.x, rect\_progressbar.y, rect\_progressbar.width, rect\_progressbar.height)), textureProgressBar\_Frame);

}

if(useShowPercent)

GUI.Label(new Rect(calOnGUI.SetGUI(rect\_percent\_text.x, rect\_percent\_text.y, rect\_percent\_text.width , rect\_percent\_text.height)),"100%");

}

}

}

//Loading method

IEnumerator WaitLoading(){

while(patSysm.loadingComplete == false){

yield return 0;

}

StartCoroutine(InitPlayer());

}

//Spawn player method

IEnumerator InitPlayer(){

GameObject go = (GameObject)Instantiate(playerPref[selectPlayer], posStart, Quaternion.identity);

cameraFol.target = go.transform;

yield return 0;

StartCoroutine(UpdatePerDistance());

}

//update distance score

IEnumerator UpdatePerDistance(){

while(true){

if(PatternSystem.instance.loadingComplete){

if(GameAttribute.gameAttribute.pause == false

&& GameAttribute.gameAttribute.isPlaying == true

&& GameAttribute.gameAttribute.life > 0){

if(Controller.instace.transform.position.z > 0){

GameAttribute.gameAttribute.distance += GameAttribute.gameAttribute.speed \* Time.deltaTime;

distanceCheck += GameAttribute.gameAttribute.speed \* Time.deltaTime;

if(distanceCheck >= speedAddEveryDistance){

GameAttribute.gameAttribute.speed += speedAdd;

if(GameAttribute.gameAttribute.speed >= speedMax){

GameAttribute.gameAttribute.speed = speedMax;

}

countAddSpeed++;

distanceCheck = 0;

}

}

}

}

yield return 0;

}

}

//reset game

public IEnumerator ResetGame(){

GameAttribute.gameAttribute.isPlaying = false;

GUIManager.instance.showSumGUI = true;

int oldCoind = GameData.LoadCoin ();

GameData.SaveCoin((int)GameAttribute.gameAttribute.coin+oldCoind);

distanceCheck = 0;

countAddSpeed = 0;

yield return 0;

}

}

Скрипт ShopCharacter

/// <summary>

/// Shop character.

/// This script use for create shop menu

/// </summary>

using UnityEngine;

using System.Collections;

using System.Collections.Generic;

public class ShopCharacter : MonoBehaviour {

[System.Serializable]

public class CharacterData

{

public GameObject player;

public string name;

public int price;

public bool isUnLock;

}

[System.Serializable]

public class Button

{

public Rect rect;

public Texture2D normal;

public Texture2D active;

}

[System.Serializable]

public class Label

{

public GUIText guiText;

public string text;

public int fontSize;

public Color fontColor;

public FontStyle fontStyle;

public TextAnchor alignment;

public Rect rect;

}

public CharacterData[] players;

public float factorSpace;

public int coin;

public Button btnBuy, btnSelect, btnOnSelect, btnBack, btnArrowLeft, btnArrowRight, iconCoinBuy, iconCoinCurrent;

public Label guiLabelName, guiLabelPrice, guiLabelCoinCurrent;

public AudioClip sfxButton;

private CalOnGUI calGUI;

private int indexSelect;

private int selecCorrect;

private List<Vector3> point = new List<Vector3> ();

private Vector3 getMousePos;

[HideInInspector] public GUIStyle guiStyleBtnBuy;

[HideInInspector] public GUIStyle guiStyleBtnSelect;

[HideInInspector] public GUIStyle guiStyleBtnOnSelect;

[HideInInspector] public GUIStyle guiStyleBtnBack;

[HideInInspector] public GUIStyle guiStyleBtnArrowLeft;

[HideInInspector] public GUIStyle guiStyleBtnArrowRight;

void SetStyle(){

guiStyleBtnBuy.normal.background = btnBuy.normal;

guiStyleBtnBuy.active.background = btnBuy.active;

guiStyleBtnSelect.normal.background = btnSelect.normal;

guiStyleBtnSelect.active.background = btnSelect.active;

guiStyleBtnBack.normal.background = btnBack.normal;

guiStyleBtnBack.active.background = btnBack.active;

guiStyleBtnOnSelect.normal.background = btnOnSelect.normal;

guiStyleBtnOnSelect.active.background = btnOnSelect.active;

guiStyleBtnArrowLeft.normal.background = btnArrowLeft.normal;

guiStyleBtnArrowLeft.active.background = btnArrowLeft.active;

guiStyleBtnArrowRight.normal.background = btnArrowRight.normal;

guiStyleBtnArrowRight.active.background = btnArrowRight.active;

}

void ResetData(){

for (int i = 0; i < players.Length; i++) {

PlayerPrefs.SetString("Player\_"+i, "False");

}

}

void Start(){

//ResetData ();

SetStyle ();

coin = GameData.LoadCoin ();

selecCorrect = PlayerPrefs.GetInt ("SelectPlayer");

Vector3 pos = Vector3.zero;

for (int i = 0; i < players.Length; i++) {

players [i].player.transform.localPosition = new Vector3 (pos.x + (i \* factorSpace), 0, 0);

point.Add (new Vector3 (-1 \* (pos.x + (i \* factorSpace)) + transform.position.x, transform.position.y, transform.position.z));

if (i == 0) {

players [i].isUnLock = true;

} else {

Debug.Log (PlayerPrefs.GetString ("Player\_" + i));

if(PlayerPrefs.GetString ("Player\_" + i) == ""){

players [i].isUnLock = false;

}else{

players [i].isUnLock = bool.Parse (PlayerPrefs.GetString ("Player\_" + i));

}

}

}

StartCoroutine (WaitInput ());

}

void OnGUI(){

SetStyle ();

guiLabelName.guiText.text = players[indexSelect].name;

guiLabelName.guiText.pixelOffset = new Vector2(GUI\_Calculate.RectWithScrren\_WidthAndHeight\_Sizeheight(new Vector2(guiLabelName.rect.x, guiLabelName.rect.y),

new Vector2(guiLabelName.rect.width, guiLabelName.rect.height)).x,

GUI\_Calculate.RectWithScrren\_WidthAndHeight\_Sizeheight(new Vector2(guiLabelName.rect.x, guiLabelName.rect.y),

new Vector2(guiLabelName.rect.width, guiLabelName.rect.height)).y);

guiLabelName.guiText.fontSize = GUI\_Calculate.FontSize(guiLabelName.fontSize);

guiLabelCoinCurrent.guiText.text = coin.ToString ();

guiLabelCoinCurrent.guiText.pixelOffset = new Vector2(GUI\_Calculate.RectWithScrren\_WidthAndHeight\_Sizeheight(new Vector2(guiLabelCoinCurrent.rect.x, guiLabelCoinCurrent.rect.y),

new Vector2(guiLabelCoinCurrent.rect.width, guiLabelCoinCurrent.rect.height)).x,

GUI\_Calculate.RectWithScrren\_WidthAndHeight\_Sizeheight(new Vector2(guiLabelCoinCurrent.rect.x, guiLabelCoinCurrent.rect.y),

new Vector2(guiLabelCoinCurrent.rect.width, guiLabelCoinCurrent.rect.height)).y);

guiLabelCoinCurrent.guiText.fontSize = GUI\_Calculate.FontSize(guiLabelCoinCurrent.fontSize);

GUI.DrawTexture(GUI\_Calculate.RectWithScrren\_WidthAndHeight\_Sizeheight(new Vector2(iconCoinCurrent.rect.x, iconCoinCurrent.rect.y),

new Vector2(iconCoinCurrent.rect.width, iconCoinCurrent.rect.height)), iconCoinCurrent.normal);

if (players [indexSelect].isUnLock == false)

{

GUI.DrawTexture(GUI\_Calculate.RectWithScrren\_WidthAndHeight\_Sizeheight(new Vector2(iconCoinBuy.rect.x, iconCoinBuy.rect.y),

new Vector2(iconCoinBuy.rect.width, iconCoinBuy.rect.height)), iconCoinBuy.normal);

guiLabelPrice.guiText.enabled = true;

guiLabelPrice.guiText.text = players[indexSelect].price.ToString();

guiLabelPrice.guiText.pixelOffset = new Vector2(GUI\_Calculate.RectWithScrren\_WidthAndHeight\_Sizeheight(new Vector2(guiLabelPrice.rect.x, guiLabelPrice.rect.y),

new Vector2(guiLabelPrice.rect.width, guiLabelPrice.rect.height)).x,

GUI\_Calculate.RectWithScrren\_WidthAndHeight\_Sizeheight(new Vector2(guiLabelPrice.rect.x, guiLabelPrice.rect.y),

new Vector2(guiLabelPrice.rect.width, guiLabelPrice.rect.height)).y);

guiLabelPrice.guiText.fontSize = GUI\_Calculate.FontSize(guiLabelPrice.fontSize);

if(GUI.Button(GUI\_Calculate.RectWithScrren\_WidthAndHeight\_Sizeheight(new Vector2(btnBuy.rect.x, btnBuy.rect.y),

new Vector2(btnBuy.rect.width, btnBuy.rect.height)),"",guiStyleBtnBuy)){

if(coin >= players[indexSelect].price)

{

coin -= players[indexSelect].price;

GameData.SaveCoin(coin);

players[indexSelect].isUnLock = true;

PlayerPrefs.SetString("Player\_"+indexSelect, "True");

PlayerPrefs.SetInt("SelectPlayer", selecCorrect);

Debug.Log("Buy : "+indexSelect+ " : " + PlayerPrefs.GetString("Player\_"+indexSelect));

}

if(sfxButton != null)

AudioSource.PlayClipAtPoint(sfxButton,transform.position);

}

}

if (players [indexSelect].isUnLock == true)

{

guiLabelPrice.guiText.enabled = false;

if(indexSelect == selecCorrect)

{

GUI.Button(GUI\_Calculate.RectWithScrren\_WidthAndHeight\_Sizeheight(new Vector2(btnOnSelect.rect.x, btnOnSelect.rect.y),

new Vector2(btnOnSelect.rect.width, btnOnSelect.rect.height)), "", guiStyleBtnOnSelect );

}

else

{

if(GUI.Button(GUI\_Calculate.RectWithScrren\_WidthAndHeight\_Sizeheight(new Vector2(btnSelect.rect.x, btnSelect.rect.y),

new Vector2(btnSelect.rect.width, btnSelect.rect.height)), "", guiStyleBtnSelect))

{

selecCorrect = indexSelect;

PlayerPrefs.SetInt("SelectPlayer", selecCorrect);

if(sfxButton != null)

AudioSource.PlayClipAtPoint(sfxButton,transform.position);

}

}

}

if(GUI.Button(GUI\_Calculate.RectWithScrren\_WidthAndHeight\_Sizeheight(new Vector2(btnArrowLeft.rect.x, btnArrowLeft.rect.y),

new Vector2(btnArrowLeft.rect.width, btnArrowLeft.rect.height)), "", guiStyleBtnArrowLeft )){

indexSelect ++;

if(indexSelect >= players.Length-1){

indexSelect = players.Length-1;

}

if(sfxButton != null)

AudioSource.PlayClipAtPoint(sfxButton,transform.position);

}

if(GUI.Button(GUI\_Calculate.RectWithScrren\_WidthAndHeight\_Sizeheight(new Vector2(btnArrowRight.rect.x, btnArrowRight.rect.y),

new Vector2(btnArrowRight.rect.width, btnArrowRight.rect.height)), "", guiStyleBtnArrowRight )){

indexSelect --;

if(indexSelect <= 0){

indexSelect = 0;

}

if(sfxButton != null)

AudioSource.PlayClipAtPoint(sfxButton,transform.position);

}

if(GUI.Button(GUI\_Calculate.RectWithScrren\_WidthAndHeight\_Sizeheight(new Vector2(btnBack.rect.x, btnBack.rect.y),

new Vector2(btnBack.rect.width, btnBack.rect.height)), "", guiStyleBtnBack )){

PlayerPrefs.SetInt("SelectPlayer", selecCorrect);

Application.LoadLevel("TitleScene");

if(sfxButton != null)

AudioSource.PlayClipAtPoint(sfxButton,transform.position);

}

}

IEnumerator WaitInput(){

bool input = false;

while (input == false) {

if(Input.GetMouseButtonDown(0)){

getMousePos = Input.mousePosition;

input = true;

}

yield return 0;

}

StartCoroutine (SelectDirection ());

}

IEnumerator SelectDirection(){

bool input = false;

while (input == false) {

if((Input.mousePosition.x - getMousePos.x) < -40){

indexSelect++;

if(indexSelect >= players.Length-1){

indexSelect = players.Length-1;

}

input = true;

}

if((Input.mousePosition.x - getMousePos.x) > 40){

indexSelect--;

if(indexSelect <= 0){

indexSelect = 0;

}

input = true;

}

if(Input.GetMouseButtonUp(0)){

input = true;

}

yield return 0;

}

StartCoroutine (MoveToPoint ());

}

IEnumerator MoveToPoint(){

while (Vector3.Distance(transform.position, point[indexSelect]) > 0.01f) {

transform.position = Vector3.Lerp(transform.position, point[indexSelect], 10 \* Time.deltaTime);

yield return 0;

}

transform.position = point [indexSelect];

StartCoroutine (WaitInput ());

}

}

Скрипт GUI\_Calculate

using UnityEngine;

using System.Collections;

public class GUI\_Calculate : MonoBehaviour {

/// <summary>

/// position.x = screen width

/// position.y = screen width

/// size.x = screen width

/// size.y = screen width

/// </summary>

public static Rect RectWithScreen\_Width(Vector2 \_position, Vector2 \_size){

Rect returnVal = new Rect();

returnVal.x = (\_position.x\*Factor().x) - ((\_size.x\*Factor().x)/2);

returnVal.y = (\_position.y\*Factor().x) - ((\_size.y\*Factor().x)/2);

returnVal.width = \_size.x\*Factor().x;

returnVal.height = \_size.y\*Factor().x;

return returnVal;

}

/// <summary>

/// position.x = screen height

/// position.y = screen height

/// size.x = screen height

/// size.y = screen height

/// </summary>

public static Rect RectWithScreen\_Height(Vector2 \_position, Vector2 \_size){

Rect returnVal = new Rect();

returnVal.x = (\_position.x\*Factor().y) - ((\_size.x\*Factor().y)/2);

returnVal.y = (\_position.y\*Factor().y) - ((\_size.y\*Factor().y)/2);

returnVal.width = \_size.x\*Factor().y;

returnVal.height = \_size.y\*Factor().y;

return returnVal;

}

/// <summary>

/// position.x = screen width

/// position.y = screen height

/// size.x = screen width

/// size.y = screen height

/// </summary>

public static Rect RectWithScrren\_WidthAndHeight\_SizeMix(Vector2 \_position, Vector2 \_size){

Rect returnVal = new Rect();

returnVal.x = (\_position.x\*Factor().x) - ((\_size.x\*Factor().x)/2);

returnVal.y = (\_position.y\*Factor().y) - ((\_size.y\*Factor().y)/2);

returnVal.width = \_size.x\*Factor().x;

returnVal.height = \_size.y\*Factor().y;

return returnVal;

}

/// <summary>

/// position.x = screen width

/// position.y = screen height

/// size.x = screen width

/// size.y = screen width

/// [============ Work ============]

/// </summary>

public static Rect RectWithScrren\_WidthAndHeight\_SizeWidth(Vector2 \_position, Vector2 \_size){

Rect returnVal = new Rect();

returnVal.x = (\_position.x\*Factor().x) - ((\_size.x\*Factor().x)/2);

returnVal.y = (\_position.y\*Factor().y) - ((\_size.y\*Factor().y)/2);

returnVal.width = \_size.x\*Factor().x;

returnVal.height = \_size.y\*Factor().x;

return returnVal;

}

/// <summary>

/// position.x = screen width

/// position.y = screen height

/// size.x = screen heigth

/// size.y = screen heigth

/// </summary>

public static Rect RectWithScrren\_WidthAndHeight\_Sizeheight(Vector2 \_position, Vector2 \_size){

Rect returnVal = new Rect();

returnVal.x = (\_position.x\*Factor().x) - ((\_size.x\*Factor().x)/2);

returnVal.y = (\_position.y\*Factor().y) - ((\_size.y\*Factor().y)/2);

returnVal.width = \_size.x\*Factor().y;

returnVal.height = \_size.y\*Factor().y;

return returnVal;

}

public static int FontSize(int \_fontSize){

int returnVal = (int)(Factor().x\*10)\*(int)\_fontSize;

if(returnVal <= 0){

returnVal = 1;

}

return returnVal;

}

private static Vector2 Factor(){

Vector2 returnVal = Vector2.zero;

returnVal.x = (float)Screen.width/2048;

returnVal.y = (float)Screen.height/1024;

return returnVal;

}

}**ДОДАТОК Ґ СКРІНШОТИ СТВОРЕННОЇ ГРИ**

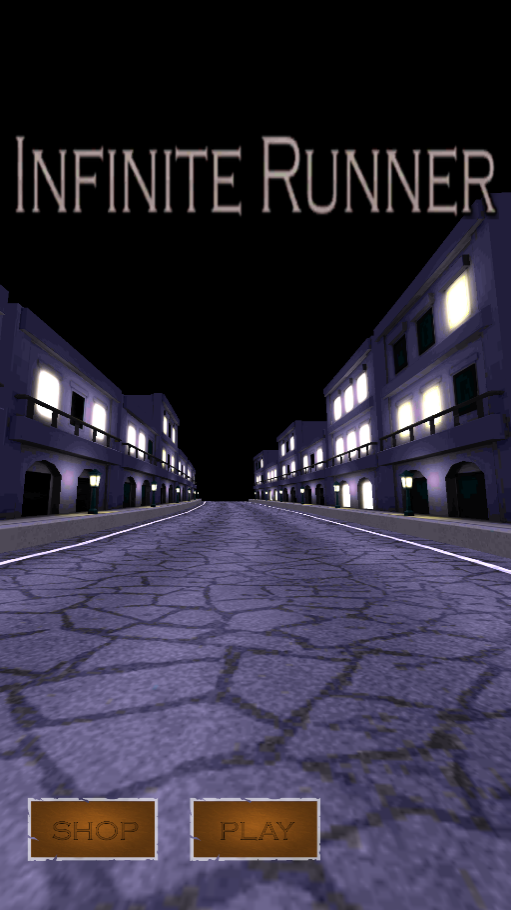


Рисунок 1 – Головне меню гри



Рисунок 2 – Магазин гри



Рисунок 3 – Гра

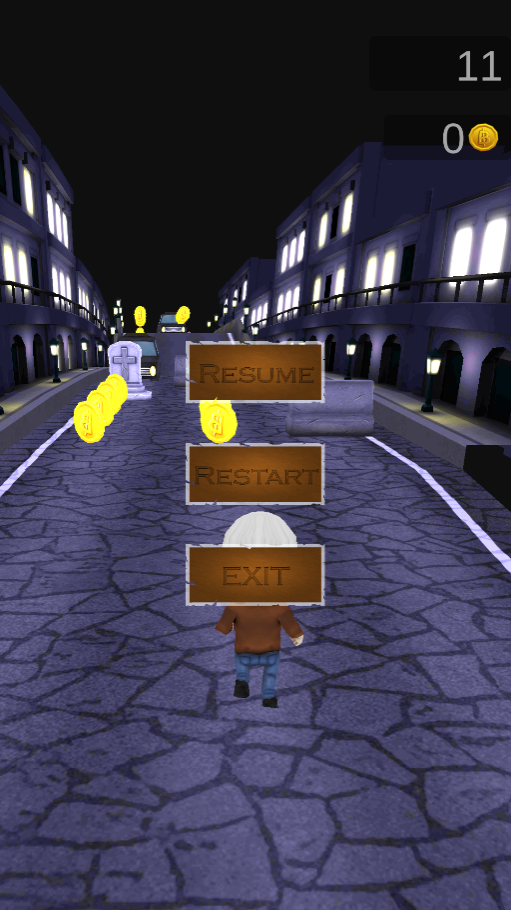


Рисунок 4 – Пауза під час гри

**ДОДАТОК Д РЕЗУЛЬТАТИ ВИКОНАННЯ ТЕСТОВИХ ВИПАДКІВ**

Таблиця Додаток Д

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Тестовий випадок | Передумова | Покрокова інструкція | Очікуваний результат | Фактичний результат |
| Запуск гри під OC Android версії 4.0-4.4. | Наявність емулятора OC Android версії 4.0-4.4. | Запустити емулятор Android версії 4.0-4.4, встановити гру за допомогою APK файлу, запустити гру. | Запуск гри пройшов успішно. | Запуск гри пройшов невдало, гра не працює на цих версіях Android. |
| Запуск гри під OC Android версії 5.0-5.1. | Наявність емулятора OC Android версії 5.0-5.1. | Запустити емулятор Android версії 5.0-5.1, встановити гру за допомогою APK файлу, запустити гру. | Запуск гри пройшов успішно. | Запуск гри пройшов успішно. |
| Запуск гри під OC Android версії 6.0. | Наявність емулятора OC Android версії 6.0. | Запустити емулятор Android версії 6.0, встановити гру за допомогою APK файлу, запустити гру. | Запуск гри пройшов успішно. | Запуск гри пройшов успішно. |
| Запуск гри під OC Android версії 7.0-7.1. | Наявність емулятора OC Android версії 7.0-7.1. | Запустити емулятор Android версії 7.0-7.1, встановити гру за допомогою APK файлу, запустити гру. | Запуск гри пройшов успішно. | Запуск гри пройшов успішно. |
| Запуск гри під OC Android версії 8.0-8.1. | Наявність емулятора OC Android версії 8.0-8.1. | Запустити емулятор Android версії 8.0-8.1, встановити гру за допомогою APK файлу, запустити гру. | Запуск гри пройшов успішно. | Запуск гри пройшов успішно. |
| Початок гри. | Додаток що працює на пристрої. | Запустити додаток, дочекатись завантаження, натиснути кнопку Play, дочекатись завантаження, перевірити працездатність гри. | Гра завантажує поле гри та правильно виконує всі функції. | Гра правильно виконує всі функції в процесі роботи. |

Продовження таблиці Додаток Д

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Придбання персонажу. | Гра що вже працює, недостатня кількість грошей на рахунку. | Під час роботи програмного продукту перейти у вкладку Shop, та придбати новий персонаж. | Персонаж не буде придбаний, з рахунку не буде знято гроші. | Персонаж не придбаний, з рахунку не знято гроші. |
| Придбання персонажу. | Гра що вже працює, достатня кількість грошей на рахунку. | Під час роботи програмного продукту перейти у вкладку Shop, та придбати новий персонаж. | Персонаж буде придбаний, з рахунку зніметься відповідна кількість грошей. | Персонаж придбаний, з рахунку знята відповідна кількість грошей. |