МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Спеціальність: 121 «Інженерія програмного забезпечення»

Освітньо-професійна програма: «Розробка програмного забезпечення»

Група: 4РП-08

Дипломний проєкт

здобувача освіти денної форми навчання
РП.08.07.000.ДП

ГРОМОВИЧА
ДЕНИСА ОЛЕКСАНДРОВИЧА

м. Одеса
2025 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Спеціальність: **121 «Інженерія програмного забезпечення»**

Освітньо-професійна програма: **«Розробка програмного забезпечення»**

Група: **4РП-08**

ПОЯСНЮВАЛЬНА ЗАПИСКА

до дипломного проєкту на тему:

Розробка програмного застосунку логічної гри на пошук прихованих предметів

Проєктний матеріал складається з пояснювальної записки на 60 сторінках та графічного (презентаційного) матеріалу на 12 аркушах (слайдах)

Дипломник _____ (Громович Д.О.)

Керівник _____ (Кунуп Т.В.)

Консультанти:

з економічного розділу _____ (Канський М.Ю.)

з розділу охорони праці та техніки безпеки _____ (Чорновол Н.І.)

з нормоконтролю _____ (Петрашова В.І.)

старший консультант _____ (Кривченко Ю.В.)

До захисту допущений

Голова циклової комісії _____ (Кривченко Ю.В.)

Завідувач відділення _____ (Краснокутська К.Г.)

Захист «26» 06 2025 р. Протокол ЕК № 2

Оцінка ЕК 3/70

Секретар ЕК _____

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Відділення *комп'ютерних систем* Комісія *КТ та ПІ*

Спеціальність *121 «Інженерія програмного забезпечення»*

Освітньо-професійна програма *«Розробка програмного забезпечення»*

ЗАТВЕРДЖУЮ:

Заст. дир. з НВР *Беркань І.В.*

"12" 05 2025 р.

ЗАВДАННЯ

на дипломний проєкт

Здобувачеві освіти *Громовичу Денису Олександровичу*

(прізвище, ім'я, по батькові)

1. Тема проєкту *Розробка програмного застосунку логічної гри на пошук прихованих предметів*

затверджена наказом по коледжу від "14" листопада 2024р. № 246

2. Термін здачі закінченого проєкту

3. Вихідні данні до проєкту *Вивчення можливостей бібліотеки Pygame, мова програмування Python, аналіз технологій створення логічних ігор, модульне програмування, інтеграція графіки, звуку та анімації, збереження ігрового прогресу, тестування та оптимізація продуктивності*

4. Зміст розрахунково-пояснювальної записки (перелік питань, які необхідно розробити)

Огляд існуючих ігрових рішень; Формулювання вимог до гри; Проєктування структури гри; Розробка ігрової логіки та механік; Інтеграція графіки, анімації та звуку; Тестування та оптимізація;

5. Перелік графічного (презентаційного) матеріалу (з точним зазначенням обов'язкових креслень, кількості слайдів)

Огляд існуючих ігрових рішень; Аналіз аналогів; Архітектура гри (структура проєкту); Скриншоти інтерфейсу та ігрового процесу; Опис функцій; Результати тестування.

6. Консультанти по проєкту, із зазначенням розділів проєкту, що їх стосується

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
Основний розділ	*Кунуп Т.В.*		
Економічний розділ	*Канський М.Ю.*		
Розділ охорони праці	*Чорновол Н.І.*		
Нормоконтроль	*Петрашова В.І.*		
Старший консультант	*Кривченко Ю.В.*		

7. Дата видачі завдання 29.04.25 р.

Керівник *Кунуп Т. В.* _____
(підпис)

Завдання прийняв до виконання *Громович Д.О.* _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/р	Назва етапів дипломного проєкту	Термін виконання етапів дипломного проєкту (роботи)	Відмітка про виконання
1	*Вступ. Постановка мети та задач проектування*	*10.05.2025*	виконано
2	*Аналіз існуючих ігрових рішень та технологій*	*12.05.2025*	виконано
3	*Розробка концепції гри, визначення жанру, механік*	*13.05.2025*	виконано
4	*Проєктування структури гри та архітектури*	*15.05.2025*	виконано
5	*Вибір інструментів реалізації: Python, Pygame*	*18.05.2025*	виконано
6	*Розробка інтерфейсу користувача (GUI)*	*23.05.2025*	виконано
7	*Реалізація ігрової логіки, рівнів та об'єктів*	*25.06.2025*	виконано
8	*Інтеграція графіки, анімації та звукових ефектів*	*28.06.2025*	виконано
9	*Розробка системи збереження результатів гри*	*03.06.2025*	виконано
10	*Тестування та усунення помилок, профілювання*	*07.06.2025*	виконано
11	*Виправлення виявлених помилок*	*10.06.2025*	виконано
12	*Аналіз результатів, оформлення слайдів презентації*	*11.06.2025*	виконано
13	*Економічні розрахунки, охорона праці*	*13.06.2025*	виконано
14	*Підготовка графічної частини проекту*	*15.06.2025*	виконано
15	*Підготовка проекту до захисту та тестування гри*	*16.06.2025*	виконано

Дипломник _____
(підпис)

Керівник _____
(підпис)

Формат	Зона	Поз.	Позначення	Назва	Кіл.	Примітка
				Документація		
			РП 08. 07 000. 00 ДП	Дипломний проєкт		
А4			РП 08. 07 000. 00 ДП ПЗ	Пояснювальна записка	1	

Зм.	Арк.	№ докум.	Підпис	Дата	РП 08. 07 000. 00 ДП		
Розробив		Громович Д.О.		20.06	Розробка програмного застосунку логічної гри на пошук прихованих предметів	Літ.	Аркуш 4 / Аркушів 60
Перевірив		Кунуп Т.В.				Н Д П	
Н. контр.		Петрашова В.І.		20.06.25		ВСП "ОТФК ОНТУ" гр.4РП-08	
Затверд.		Кривченко Ю.В.		23.06			

ЗМІСТ

ВСТУП

Актуальність теми розробки : в сучасному світі комп'ютерні ігри стали невід'ємною частиною індустрії розваг та освіти, а логічні ігри, зокрема ігри на пошук прихованих предметів, набувають особливої популярності завдяки їх здатності розвивати увагу, спостережливість та когнітивні навички користувачів. Розробка таких ігор з використанням сучасних технологій програмування, зокрема мови Python, відкриває нові можливості для створення інтерактивних освітніх та розважальних продуктів, що поєднують у собі елементи логічного мислення та візуального сприйняття.

Особливої актуальності набуває розробка ігор даного жанру з використанням кросплатформних технологій, що забезпечують доступність продукту для широкої аудиторії користувачів. Використання Python та бібліотеки Pygame дозволяє створювати ефективні рішення, які поєднують у собі простоту розробки, гнучкість налаштування та високу продуктивність, що є суттєвим для сучасних програмних продуктів.

Мета і завдання розробки: метою роботи є розробка логічної гри на пошук прихованих предметів з використанням мови програмування Python та бібліотеки Pygame, що забезпечує інтерактивний ігровий процес та розвиває когнітивні навички користувачів.

Об'єкт дослідження: процес розробки логічних комп'ютерних ігор з використанням мови програмування Python.

Практичне значення роботи: розроблена гра може використовуватися як розважальний та освітній продукт для розвитку уваги, спостережливості та когнітивних навичок користувачів різних вікових категорій. Реалізовані технічні рішення та підходи можуть бути використані при розробці аналогічних ігрових проект

Теоретичне значення роботи: досліджено та систематизовано методи розробки логічних ігор з використанням Python та Pygame, проаналізовано особливості реалізації різних ігрових механік та систем. Розроблено підходи до оптимізації продуктивності та тестування ігрових додатків.

1 ОСНОВНИЙ РОЗДІЛ

1.1 Аналіз предметної області (аналогів) та теоретичні основи

1.1.1 Загальні характеристики жанру логічних ігор

Для реалізації теми дипломного проекту, потрібно провести аналіз жанру логічних ігор

Жанр логічних ігор займає особливе місце в системі цифрових розваг, поєднуючи в собі елементи інтелектуального виклику, структурованого мислення та аналітичного підходу до вирішення завдань. Він не орієнтований на швидкість, як аркади або шутери, і не потребує складних комбінацій дій, як це властиво рольовим або стратегічним іграм. Основне, що характеризує логічні ігри — це апеляція до раціонального мислення, здібності до аналізу, логічного узагальнення, систематизації даних, а також вміння будувати причинно-наслідкові зв’язки. Успішне проходження рівнів у таких іграх зазвичай залежить не від досвіду у керуванні персонажем чи знання внутрішніх механік гри, а від здатності гравця мислити послідовно, виявляти закономірності та знаходити нестандартні рішення у межах чітко заданої логіки.

Логічні ігри відрізняються значною різноманітністю форм та механік. Вони можуть бути представлені у вигляді головоломок, задач на просторову уяву, задач з числами або літерами, стратегічних конфігурацій чи візуальних завдань. Здебільшого у таких іграх гравець має перед собою серію взаємопов’язаних викликів, які необхідно вирішити, щоб перейти до наступного етапу. Складність рівнів зазвичай зростає поступово, формуючи відчуття розвитку, прогресу, залученості. Гравець не лише проходить гру, а навчається у процесі, тренує мозок, підвищує свою здатність до концентрації, покращує пам’ять, розвиває посидючість і витривалість у вирішенні задач.

Однією з характерних рис логічних ігор є їх універсальність. Вони не обмежуються якоюсь однією віковою категорією чи рівнем підготовки. Завдяки своїй природній інтелектуальній орієнтованості, логічні ігри приваблюють як дітей, так і дорослих. Дітям вони допомагають розвивати логічне мислення та

пам’ять у ігровій, невимушеній формі. Для дорослих ці ігри можуть виступати як спосіб ментального відпочинку, який, попри свою розслаблюючу дію, залишається розвиваючим. Також варто зазначити, що логічні ігри часто використовуються в освітньому процесі, у тому числі для навчання школярів та студентів базовим принципам алгоритмізації, стратегічного мислення або вирішення комбінаційних задач.

Іншою важливою характеристикою логічного жанру є гнучкість у візуальному оформленні. На відміну від багатьох інших жанрів, де графічна складова відіграє ключову роль у залученні користувача, в логічних іграх акцент робиться саме на змістовному наповненні — тобто на завданнях, механіках, структурі проходження. Проте, у сучасних проєктах навіть у межах цього жанру активно використовуються елементи привабливої візуалізації, що підвищує загальну якість користувацького досвіду. Прості форми в оформленні можуть доповнюватися приємними анімаціями, спокійним музичним супроводом та інтуїтивно зрозумілим інтерфейсом.

Окрему нішу в межах логічного жанру займають ігри на пошук прихованих предметів. Цей піджанр базується на завданні гравцеві знайти певні об’єкти, заховані серед великої кількості деталей на зображенні. Такі ігри мають тісний зв’язок з візуальним сприйняттям, увагою до дрібниць і здатністю швидко виділяти знайомі форми серед візуального шуму. При цьому завдання часто супроводжуються сюжетною лінією або детективною тематикою, що додає мотивації гравцеві та робить проходження більш захоплюючим. Гравець не просто виконує суху інструкцію, а занурюється у наративну ситуацію, де кожен знайдений об’єкт може бути частиною великої історії або ключем до наступної головоломки.

З технічної точки зору логічні ігри також мають свої особливості. Вони, як правило, не потребують складної фізики, 2D-анімації чи великої обчислювальної потужності, що робить їх доступними для реалізації навіть на базовому програмному рівні або у вигляді мобільних застосунків. Завдяки цьому логічні ігри часто створюються невеликими командами розробників або навіть окремими

ентузіастами. Це створює простір для креативних експериментів, унікальних механік та інноваційних форм подання звичних задач.

Не менш важливою характеристикою логічних ігор є їхня здатність адаптуватися до потреб користувача. Багато сучасних проєктів включають систему підказок, різні рівні складності, можливість проходження з перервами, що робить такі ігри зручними для широкого кола людей. У світі, де значна частина користувачів має обмежений вільний час, можливість займатися розв’язанням логічних задач у зручний момент без втрати контексту — це безумовна перевага жанру.

Таким чином, логічні ігри — це не просто розвага, а повноцінний інструмент когнітивного розвитку, який гармонійно поєднує в собі елементи гейміфікації, інтелектуального виклику та психологічного комфорту. Завдяки цьому жанр зберігає свою популярність і постійно оновлюється, відкриваючи нові можливості як для гравців, так і для розробників.

1.1.2 Аналіз сучасних логічних ігор на пошук прихованих предметів

Аналіз гри Hidden City: Hidden Object Adventure

ї У ході проходження користувач взаємодіє з численними персонажами, виконує сюжетні місії, досліджує локації та розв'язує завдання на уважність і логіку.

З точки зору геймплею Hidden City дотримується класичних засад жанру: гравець отримує зображення сцени, на якій приховано набір предметів, що потрібно знайти протягом обмеженого часу. Кожна сцена є візуально деталізованою і стилістично унікальною, що вимагає високого рівня уважності. Завдання ускладнюються додаванням затемнених об'єктів, змінених пропорцій, або частково замаскованих предметів, що значно підвищує інтерес до процесу пошуку.

Технічно гра побудована на 2D-графіці з використанням високоякісних статичних і динамічних зображень. Важливу роль відіграє художнє оформлення сцен — локації стилізовані під бібліотеки, вулиці, музеї, магічні лабораторії та інші тематичні місця. Атмосферу посилює добре підібраний музичний супровід

та озвучення, яке підкреслює сюжетну напругу. Крім основної механіки пошуку, гра пропонує побічні активності: мініігри, дослідження ресурсів, колекціонування артефактів, що забезпечує додаткову глибину ігрового процесу.

Рисунок 1.1. Ігровий процес у Hidden City: Hidden Object Adventure

Інтерфейс гри інтуїтивно зрозумілий, адаптований під сенсорне керування, що особливо важливо для мобільних користувачів. Всі інтерактивні елементи мають чітке візуальне розмежування, а навігація між етапами гри побудована логічно і не перевантажена зайвими елементами. Присутня система підказок, яка може бути активована в складних ситуаціях, проте її використання обмежене, що стимулює гравця мислити самостійно.

На рівні внутрішньоігрової економіки гра використовує модель з мікротранзакціями, яка дозволяє придбати підказки, додаткові життя або енергію. Попри це, гравець має можливість проходити гру без фінансових вкладень, хоч і з певними часовими обмеженнями. Такий підхід дозволяє збалансувати інтереси як платних, так і безкоштовних користувачів.

Наступною грою для аналізу є June’s Journey. Це пригодницька гра жанру «пошук прихованих предметів», розроблена студією Wooga. Вона є кросплатформною: доступна як на Android та iOS, так і у браузерній версії через Facebook. У проєкті реалізовано сюжетну кампанію, в якій гравець поступово розкриває таємницю загибелі члена родини головної героїні — Джун Паркер. Гра

побудована у форматі проходження сцен з пошуком заданих предметів на статичному тлі з подальшим переходом до наступної сюжетної глави.

На рисунку 1.2 представлено приклад ігрового процесу з типовою сценою пошуку у грі June’s Journey.

Рисунок 1.2. Скріншот ігрової сцени у грі June’s Journey

Гравець взаємодіє із фоном, знаходячи перелік об'єктів за обмежений час. За точність і швидкість проходження нараховуються зірки, які потрібні для просування по сюжету. Особливістю гри є її стильове оформлення: дія відбувається у 1920-х роках, тож графіка, локації та музичне оформлення повністю відповідають цій епосі. Це створює атмосферу ретро-детективу.

Крім основного геймплею, у грі є додаткові функції — будівництво та оформлення приватного острова героїні. Це дозволяє користувачам персоналізувати ігровий простір. Також передбачено систему нагород, щоденних завдань і соціальної взаємодії (друзі, рейтинги).

Гра не вимагає фінансових вкладень для проходження, хоча підтримує внутрішньоігрові покупки. Вона орієнтована на широку аудиторію — особливо на гравців, які цінують спокійний, неспішний геймплей та атмосферні історії.

1.1.3 Висновки за результатами аналізу існуючих ігор

У результаті проведеного аналізу було виявлено, що обидві розглянуті гри - Hidden City та June’s Journey - належать до жанру пошуку прихованих предметів і демонструють ефективне поєднання класичних механік з сюжетними елементами та візуальною стилізацією. Обидва проєкти акцентують увагу на деталізованих статичних сценах, де гравець має знаходити об’єкти, заздалегідь вказані в списку, що вимагає високого рівня концентрації та уважності.

Основною перевагою Hidden City є багатий візуальний контент, велика кількість локацій, наявність додаткових ігрових режимів та містична атмосфера. Ця гра значно насиченіша з точки зору обсягу контенту та побудови всесвіту, однак може здатись перевантаженою для користувачів, які шукають простіший ігровий досвід.

У свою чергу, June’s Journey вирізняється більш легким стилем, елегантною стилізацією під 1920-ті роки та візуальною естетикою. Вона краще адаптована для мобільних пристроїв, має інтуїтивно зрозумілий інтерфейс і додаткові елементи персоналізації (острів, декорації), що позитивно впливає на утримання уваги користувача.

Для обох ігор характерна наявність сюжетної складової, яка розвивається поступово під час проходження рівнів. Це є важливим фактором мотивації, оскільки гравець не лише виконує технічні завдання, а й емоційно залучається до історії.

Узагальнюючи, можна зробити висновок, що найбільш успішні ігри цього жанру поєднують:

- просту, але глибоку ігрову механіку;
- продуману візуальну стилізацію;
- сюжетну інтригу;
- додаткові функції (колекціонування, побудова, прокачка тощо).

Отже, при створенні власного застосунку доцільно спиратися на подібні сильні сторони — використовувати деталізовану графіку, поступове ускладнення рівнів, систему заохочень та інтегрувати сюжетну мотивацію, навіть у спрощеному вигляді.

1.2 Аналіз цільової аудиторії гри

Перш ніж розпочинати розробку програмного застосунку, важливо визначити його потенційних користувачів. Від правильного розуміння цільової аудиторії значною мірою залежить вибір ігрової механіки, стилістики, рівня складності, способів взаємодії з інтерфейсом та загальної структури проєкту.

Розроблена гра належить до категорії логічних ігор із механікою пошуку прихованих об’єктів, що поєднує в собі уважність, просторове мислення та здатність швидко орієнтуватись у візуальному полі. Саме тому основна аудиторія проєкту складається з користувачів, які надають перевагу спокійному, але інтелектуально насиченому геймплею.

Згідно з аналізом подібних ігор, цей жанр приваблює широку вікову категорію — від підлітків до дорослих. Зокрема, найбільш активними гравцями є особи віком від 14 до 35 років. У даному випадку гра може бути цікавою як для учнів і студентів (завдяки простій механіці та візуальній складовій), так і для офісних працівників або людей, що шукають можливість розвантажити увагу в перервах між справами. Завдяки відсутності насильницького чи надмірно динамічного контенту, проєкт підходить для спокійної та розслабленої гри.

Ще однією характерною рисою потенційних користувачів є зацікавленість у іграх без складної системи управління. Реалізація гри з використанням бібліотеки Pygame забезпечує доступність керування мишею, що є інтуїтивно зрозумілим навіть для гравців без досвіду у складніших ігрових жанрах. Це робить проєкт зручним для користувачів, які не мають геймерського бекграунду, але зацікавлені у візуально-пізнавальному дозвіллі.

Гру також можна позиціонувати як інструмент для тренування уважності та концентрації, що дає змогу застосовувати її не лише у розважальному, а й у

навчальному або реабілітаційному контексті (наприклад, для людей з порушенням короткотривалої пам’яті, чи під час когнітивних тренувань).

Крім того, гнучкість структури рівнів, поступове підвищення складності, наявність бонусних елементів, об'єктів-обманок і збереження результатів у форматі JSON дозволяють зробити ігровий процес адаптивним до навичок гравця. Це означає, що новачки не відчуватимуть перевантаження, а досвідчені гравці отримають додаткову мотивацію для проходження складніших етапів.

Таким чином, цільова аудиторія розробленої гри — це широка група користувачів, орієнтованих на помірно активний, логічний геймплей, які цінують естетику, простоту управління, доступність ігрового прогресу та елементи виклику, що виникають внаслідок наявності фальшивих і бонусних об'єктів.

З метою наочного представлення основних демографічних груп, які можуть бути зацікавлені у використанні гри, нижче наведено рисунок діаграми, що відображає розподіл потенційної аудиторії за віком.

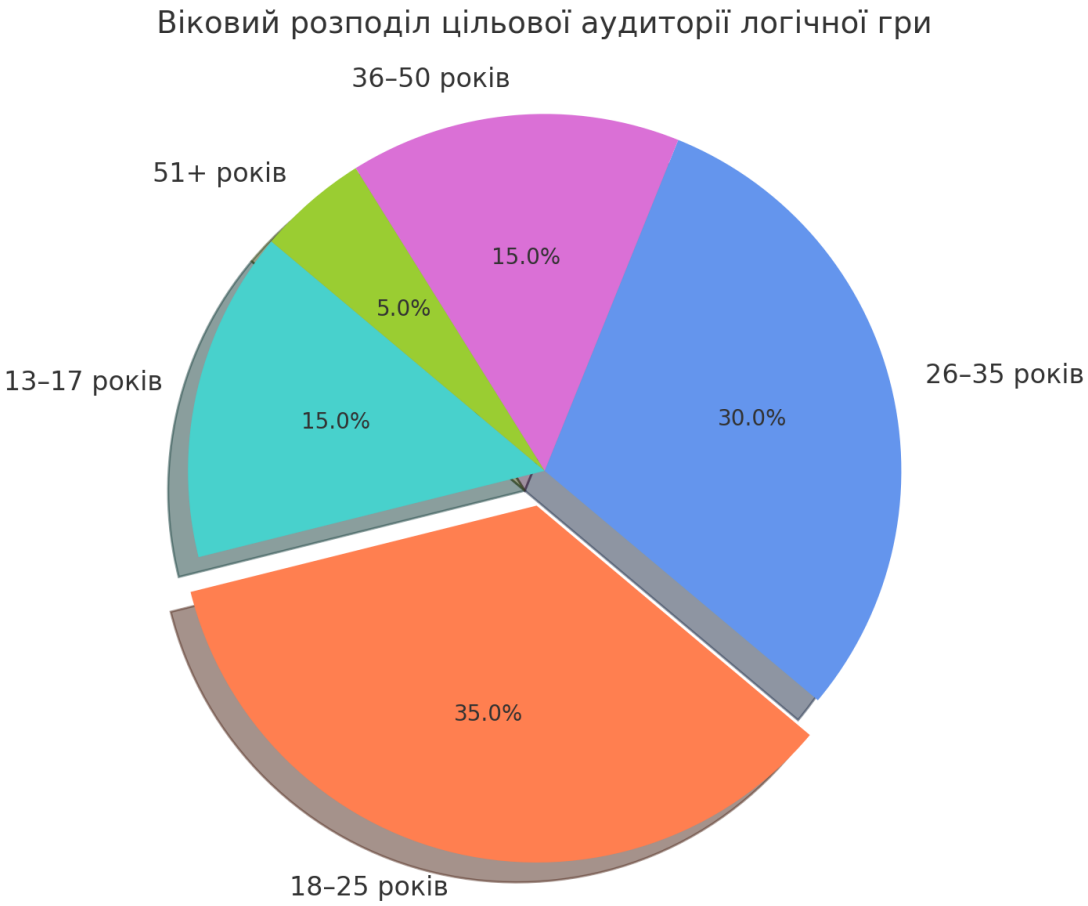


Рисунок 1.3. Віковий розподіл цільової аудиторії за віком

1.3 Вибір технологій та інструментів розробки

Огляд технологій та інструментів для розробки ігор на Python:

Python як мова програмування пропонує широкий спектр можливостей для розробки ігор різних жанрів та складності. Мова Python характеризується простим та зрозумілим синтаксисом, що робить її доступною для початківців у галузі розробки ігор. Багата стандартна бібліотека Python містить множину корисних інструментів для роботи з графікою, звуком та користувацьким введенням. Інтерпретована природа Python дозволяє швидко тестувати та налагоджувати код без необхідності компіляції. Велика спільнота розробників постійно створює та оновлює бібліотеки та фреймворки для ігрової розробки. Система керування пакетами pip спрощує процес встановлення та оновлення необхідних залежностей. Підтримка об'єктно-орієнтованого програмування дозволяє створювати добре структурований та підтримуваний код. Можливість використання функціонального програмування надає додаткову гнучкість при розробці ігрової логіки. Інтеграція з C/C++ через модулі розширення дозволяє оптимізувати критичні ділянки коду. Кросплатформність Python забезпечує можливість запуску ігор на різних операційних системах. Наявність детальної документації та навчальних матеріалів полегшує процес вивчення та використання мови. Автоматичне керування пам'яттю через збирач сміття дозволяє розробникам зосередитись на ігровій логіці.

Pygame виступає фундаментальною бібліотекою для розробки 2D ігор на Python, надаючи набір низькорівневих інструментів для роботи з графікою та звуком. Бібліотека забезпечує прямий доступ до апаратного прискорення через SDL (Simple DirectMedia Layer). Система обробки подій Pygame дозволяє ефективно керувати користувацьким введенням з клавіатури та миші. Вбудовані інструменти для роботи з спрайтами спрощують створення анімацій та колізій. Підтримка різних форматів зображень та звуків розширює можливості для створення контенту. Модуль для роботи з шрифтами дозволяє гнучко налаштовувати текстовий вивід у грі. Система колізій надає базові інструменти для визначення зіткнень між об'єктами. Підтримка альфа-каналу дозволяє

створювати напівпрозорі елементи та ефекти. Можливість масштабування та трансформації поверхонь забезпечує гнучкість у відображенні графіки. Вбудовані інструменти для роботи з кольором спрощують створення візуальних ефектів. Функції для роботи з таймером дозволяють контролювати частоту оновлення кадрів. Модульна структура бібліотеки полегшує поетапне вивчення та використання функціоналу.

Основні інструменти для розробки ігор на Python можна класифікувати за їх призначенням та функціональністю:

Таблиця 1.1 Основні інструменти для розробки ігор на Python

Категорія	Інструмент	Призначення	Особливості
Графічні двигуни	Pygame	2D графіка	Низькорівневий API, простота використання
	Pyglet	2D/3D графіка	Нативна підтримка OpenGL
	Kivy	Мультитач	Підтримка мобільних платформ
Фізичні двигуни	Pymunk	2D фізика	Швидкість роботи, простота інтеграції
	Bullet	3D фізика	Професійна якість симуляції
Звукові двигуни	Pygame mixer	Базова робота зі звуком	Простота використання
	PyOpenAL	Просторовий звук	Професійні можливості
IDE та редактори	PyCharm	Розробка	Повноцінне IDE з підтримкою ігрових фреймворків
	Visual Studio Code	Розробка	Легкість, розширюваність
Інструменти тестування	Pytest	Модульне тестування	Гнучкість та розширюваність
	Coverage.py	Аналіз покриття	Детальна статистика тестування

Pyglet представляє собою потужну альтернативу Pygame, орієнтовану на створення мультимедійних додатків та ігор з використанням OpenGL. Бібліотека не потребує зовнішніх залежностей, що спрощує процес розгортання проектів. Вбудована підтримка відтворення аудіо у різних форматах розширює можливості звукового оформлення. Система подій Pyglet забезпечує гнучке керування користувацьким введенням та взаємодією з вікном. Інтеграція з OpenGL надає доступ до апаратного прискорення та 3D графіки. Підтримка декількох вікон дозволяє створювати складні користувацькі інтерфейси. Вбудований планувальник задач спрощує управління ігровим циклом та анімаціями. Можливість завантаження ресурсів з архівів оптимізує розповсюдження гри.

Підтримка різних систем координат полегшує розробку 2D та 3D графіки. Документований API спрощує процес вивчення та використання бібліотеки. Активна спільнота забезпечує постійний розвиток та підтримку. Наявність прикладів та туторіалів прискорює процес навчання.

Kivy фреймворк надає сучасний підхід до розробки ігор з підтримкою мультитач взаємодії та можливістю створення кросплатформних додатків. Декларативний підхід до створення інтерфейсів через KV мову спрощує розробку складних UI елементів. Підтримка OpenGL ES забезпечує ефективне відображення графіки на мобільних пристроях. Система віджетів Kivy надає готові компоненти для побудови користувацького інтерфейсу. Вбудована підтримка жестів розширює можливості взаємодії з користувачем. Можливість компіляції під різні платформи спрощує процес розповсюдження гри. Підтримка апаратного прискорення забезпечує високу продуктивність на різних пристроях. Система властивостей Kivy спрощує створення реактивних інтерфейсів. Модульна архітектура дозволяє використовувати тільки необхідні компоненти. Підтримка анімацій та переходів збагачує візуальну складову гри. Можливість інтеграції з нативними API розширює функціональність. Активна спільнота забезпечує розвиток та підтримку фреймворку.

Arcade представляє сучасну бібліотеку для розробки 2D ігор, що фокусується на простоті використання та продуктивності. Бібліотека забезпечує високорівневий API для роботи з графікою та фізикою. Вбудована підтримка тайлових карт спрощує створення рівнів та ігрових світів. Система частинок дозволяє створювати візуально привабливі ефекти. Інтегрована фізична система забезпечує реалістичну поведінку об'єктів. Підтримка контролерів розширює можливості керування грою. Оптимізована система рендерингу забезпечує високу продуктивність. Вбудовані інструменти для створення користувацького інтерфейсу спрощують розробку меню та HUD. Підтримка атласів текстур оптимізує використання графічних ресурсів. Система звуку надає зручні інструменти для роботи з аудіо. Можливість інтеграції з іншими Python бібліотеками розширює функціональність. Детальна документація та приклади

прискорюють процес навчання.

Cocos2d надає потужний фреймворк для створення 2D ігор з підтримкою складних анімацій та ефектів. Інтегрована система сцен спрощує управління різними станами гри та переходами між ними. Підтримка дій та інтервалів дозволяє створювати складні анімаційні послідовності. Система меню надає готові компоненти для створення користувацького інтерфейсу. Вбудований планувальник завдань забезпечує ефективне управління ігровою логікою. Підтримка TMX карт спрощує створення рівнів з використанням Tiled редактора. Система колізій забезпечує точне визначення зіткнень між об'єктами. Підтримка ефектів частинок дозволяє створювати візуально привабливі спецефекти. Оптимізована система рендерингу забезпечує високу продуктивність. Можливість розширення функціоналу через плагіни збагачує можливості фреймворку. Інтеграція з фізичним двигуном забезпечує реалістичну поведінку об'єктів. Підтримка аудіо ефектів та музики розширює звукове оформлення гри.

PyOpenGL забезпечує доступ до можливостей OpenGL для створення 3D графіки в іграх на Python. Бібліотека надає повний доступ до функціоналу OpenGL через Python інтерфейс. Підтримка різних версій OpenGL дозволяє використовувати як сучасні, так і застарілі функції. Інтеграція з NumPy оптимізує роботу з масивами даних для 3D графіки. Можливість використання шейдерів GLSL розширює графічні можливості. Підтримка розширень OpenGL збагачує функціональність графічного рендерингу. Система буферів вершин та індексів забезпечує ефективну передачу даних на GPU. Підтримка текстурних атласів оптимізує використання графічної пам'яті. Можливість створення складних матеріалів через систему шейдерів. Інтеграція з іншими бібліотеками розширює можливості 3D рендерингу. Підтримка різних форматів 3D моделей через додаткові бібліотеки. Оптимізована робота з графічною пам'яттю підвищує продуктивність.

Системи тестування та налагодження відіграють суттєву роль у розробці ігор на Python. Pytest забезпечує гнучкий фреймворк для створення та виконання тестів різних рівнів. Модуль unittest з стандартної бібліотеки надає базові

інструменти для модульного тестування. Система профілювання cProfile дозволяє аналізувати продуктивність коду та виявляти вузькі місця. Інструмент coverage.py надає детальну інформацію про покриття коду тестами. Debugger pdb забезпечує можливість покрокового налагодження програми. Інструменти статичного аналізу коду допомагають виявляти потенційні помилки. Системи автоматизованого тестування спрощують процес регресійного тестування. Інструменти для стрес-тестування допомагають виявляти проблеми продуктивності. Системи неперервної інтеграції автоматизують процес тестування при розробці. Інструменти для тестування користувацького інтерфейсу перевіряють коректність взаємодії. Системи моніторингу продуктивності відстежують роботу гри в реальному часі.

Рисунок 1.4. Розподіл інструментів за категоріями у розробці ігор на Python

Інструменти для оптимізації коду та профілювання допомагають покращити продуктивність ігор на Python. Cython дозволяє компілювати критичні ділянки коду в нативний машинний код. Numba забезпечує JIT-компіляцію для оптимізації обчислювально складних операцій. PyPy надає альтернативну реалізацію Python з покращеною продуктивністю. Інструменти для аналізу використання пам'яті

допомагають виявляти витоки ресурсів. Профілювальники часу виконання визначають повільні ділянки коду. Системи моніторингу GPU допомагають оптимізувати графічну складову. Інструменти для аналізу мережевого коду покращують онлайн-взаємодію. Оптимізатори ассетів зменшують розмір ігрових ресурсів. Системи кешування покращують швидкість завантаження даних. Інструменти для паралельної обробки даних підвищують ефективність обчислень. Аналізатори залежностей допомагають оптимізувати структуру проекту.

Середовища розробки (IDE) та редактори коду надають комплексні інструменти для розробки ігор на Python. PyCharm забезпечує повноцінне IDE з підтримкою ігрових фреймворків та налагодженням. Visual Studio Code з Python розширеннями надає легке та гнучке середовище розробки. IDLE з стандартної поставки Python забезпечує базові можливості для початківців. Sublime Text з плагінами створює ефективне середовище для розробки ігор. Atom з Python пакетами надає можливості для командної розробки. Підтримка систем контролю версій інтегрована в сучасні IDE. Інструменти рефакторингу допомагають підтримувати якість коду. Системи автодоповнення прискорюють процес написання коду. Інтеграція з системами налагодження спрощує пошук помилок. Підтримка віртуальних середовищ забезпечує ізоляцію залежностей. Можливості профілювання вбудовані в сучасні IDE. Інтеграція з системами тестування автоматизує процес перевірки коду.

Інструменти для створення та управління ассетами спрощують роботу з ігровими ресурсами. Pillow забезпечує потужні можливості для обробки зображень та текстур. PyInstaller та cx_Freeze допомагають створювати виконувані файли з Python ігор. Системи управління ресурсами організовують зберігання та доступ до ассетів. Інструменти для конвертації форматів спрощують роботу з різними типами файлів. Системи архівації оптимізують розмір ігрових ресурсів. Інструменти для створення спрайт-атласів покращують продуктивність рендерингу. Системи управління анімаціями спрощують роботу з ігровими персонажами. Інструменти для роботи з аудіо оптимізують звукові ресурси. Генератори процедурного контенту розширюють можливості створення ассетів.

Системи керування версіями ассетів забезпечують контроль змін. Інструменти для оптимізації текстур покращують використання пам'яті.

Вибір Visual Studio Code як інтегрованого середовища розробки:

Для реалізації проекту гри було обрано інтегроване середовище розробки Visual Studio Code, яке оптимально поєднує функціональність та зручність роботи з Python та Pygame. Цей вибір обумовлений передусім глибокою інтеграцією з мовою Python, що забезпечує повноцінну підтримку синтаксису, автоматичне доповнення коду та ефективні засоби налагодження. Інтелектуальна система IntelliSense значно прискорює процес написання коду, особливо при роботі з бібліотекою Pygame, де важливо точно взаємодіяти з численними функціями та методами.

Важливою перевагою VS Code стала його здатність ефективно працювати зі складними проектами, що особливо актуально для розробки гри. Система робочих просторів дозволяє чітко організувати всі компоненти проекту - від вихідного коду до графічних ресурсів та звукових файлів. Вбудована підтримка Git спрощує контроль версій, що критично важливо при командній розробці або при необхідності відкочувати невдалі зміни.

Для візуалізації робочого процесу до роботи додано скріншот інтерфейсу VS Code (Рисунок 1), який демонструє типовий налаштований робочий простір для розробки гри. На знімку видно основні елементи: редактор коду з відкритим головним файлом гри, панель керування проектом з відображенням структури папок, вбудований термінал для запуску гри та панель інструментів з активними розширеннями. Таке середовище дозволило максимально оптимізувати процес розробки, забезпечуючи швидкий доступ до всіх необхідних інструментів без перемикання між різними програмами.

Гнучкість налаштувань VS Code особливо корисна при роботі з Pygame, де часто виникає потреба у швидкому тестуванні окремих ігрових механік. Можливість миттєвого перезапуску гри після внесення змін значно прискорює ітераційний процес розробки. Вбудований дебагер з підтримкою точкових зупинок дозволяє детально аналізувати стан ігрових об'єктів у ключові моменти,

що особливо важливо для відлагодження складних механік взаємодії між об'єктами.

Специфіка проекту гри на пошук прихованих предметів вимагала особливої уваги до організації робочого простору, оскільки робота велася з великою кількістю графічних ресурсів. VS Code ідеально підійшов для цього завдання, пропонуючи зручні інструменти для навігації між файлами та швидкого доступу до потрібних ресурсів. Можливість одночасної роботи з декількома модулями гри без перезавантаження вікон значно підвищила продуктивність розробки.

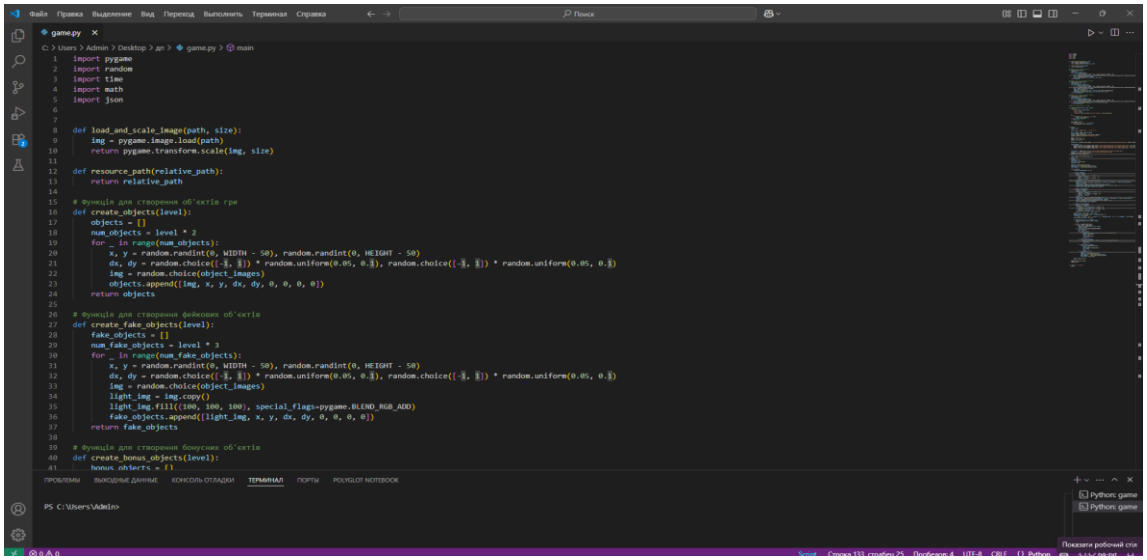


Рисунок 1.5 скріншот інтерфейсу VS Code

Вибір цього середовища розробки виявився особливо вдалим для даного проекту, оскільки поєднання потужних можливостей VS Code з простотою Python та спеціалізованими функціями Pygame створило ідеальний баланс між продуктивністю та зручністю роботи. Це дозволило зосередитися на творчих аспектах розробки гри, не витрачаючи час на подолання технічних обмежень середовища розробки.

Вибір та обґрунтування інструментарію для розробки гри (Python, Pygame):

Вибір інструментарію для розробки гри являє собою фундаментальне рішення, що визначає подальший процес розробки та можливості проекту. Python як основна мова програмування пропонує суттєві переваги для розробки ігор, особливо в контексті створення логічних ігор та ігор на пошук прихованих

предметів. Мова характеризується чистим та зрозумілим синтаксисом, що спрощує процес розробки та подальшої підтримки коду. Багата стандартна бібліотека Python надає широкий спектр інструментів для роботи з різними типами даних та форматами файлів. Інтерпретована природа мови забезпечує швидке прототипування та тестування нових функцій. Підтримка об'єктно-орієнтованого програмування дозволяє створювати чітку та масштабовану архітектуру проекту. Наявність великої кількості сторонніх бібліотек розширює можливості розробки. Активна спільнота розробників забезпечує доступ до ресурсів та підтримки. Кросплатформність Python гарантує можливість запуску гри на різних операційних системах. Вбудовані засоби для роботи з графікою та звуком спрощують розробку мультимедійного контенту. Підтримка асинхронного програмування дозволяє ефективно обробляти користувацький ввід. Інтеграція з системами контролю версій забезпечує надійне управління кодовою базою. Можливість розширення функціоналу через модулі на C/C++ дозволяє оптимізувати критичні ділянки коду.

Pygame як основний фреймворк для розробки 2D ігор надає оптимальний набір інструментів для реалізації проекту. Бібліотека забезпечує низькорівневий доступ до графічних можливостей через SDL, що гарантує високу продуктивність. Система обробки подій Pygame ефективно управляє користувацьким вводом та взаємодією з вікном. Модуль для роботи зі спрайтами спрощує створення та анімацію ігрових об'єктів. Вбудована підтримка колізій дозволяє реалізувати точне визначення взаємодій між об'єктами. Система поверхонь забезпечує гнучке управління графічними ресурсами. Підтримка прозорості та альфа-каналу розширює можливості візуальних ефектів. Модуль для роботи зі звуком надає інструменти для аудіо оформлення гри. Інтеграція з системою подій операційної системи забезпечує стабільну роботу. Можливість масштабування та трансформації зображень розширює графічні можливості. Підтримка різних форматів зображень спрощує роботу з ресурсами. Оптимізована робота з пам'яттю забезпечує стабільну продуктивність.

Архітектурні особливості Pygame ідеально відповідають вимогам проекту з

пошуку прихованих предметів. Об'єктно-орієнтований підхід дозволяє створювати чітку структуру ігрових елементів та їх взаємодій. Система груп спрайтів спрощує управління множиною об'єктів на екрані. Механізм подій забезпечує гнучке управління ігровою логікою та станами. Підтримка масок колізій дозволяє точно визначати області взаємодії об'єктів. Система таймерів забезпечує контроль над ігровим циклом та анімаціями. Можливість створення підповерхонь оптимізує роботу з великими зображеннями. Підтримка апаратного прискорення покращує продуктивність рендерингу. Інтеграція з системними шрифтами забезпечує якісне відображення тексту. Механізм відсічення спрощує відображення частин великих сцен. Підтримка буферизації зменшує навантаження на систему. Можливість створення власних подій розширює функціональність ігрової логіки.

Таким чином, вибір Python та Pygame як основного інструментарію для розробки гри на пошук прихованих предметів обґрунтовується широким спектром можливостей, надійністю та гнучкістю цих технологій. Комбінація цих інструментів забезпечує всі необхідні функції для створення якісного ігрового продукту, від базового функціоналу до складних систем безпеки та аналітики. Постійний розвиток екосистеми Python та активна спільнота розробників гарантують довгострокову підтримку та можливості для розширення проекту в майбутньому.

1.4 Проєктування та розробка гри

1.4.1 Структура гри: основні модулі та їх взаємозв’язок

Архітектура розробленої гри "Пошук предметів" базується на модульному принципі організації коду, що забезпечує чітке розділення функціональних компонентів. Основою архітектури виступає головний модуль, що містить функцію main(), яка координує взаємодію всіх компонентів системи та управляє ігровим циклом. Структура проекту передбачає використання стандартних бібліотек Python, таких як pygame для графічного інтерфейсу, random для генерації випадкових елементів, json для роботи з даними, time для управління

часом, os та sys для системних операцій. Модульна організація коду дозволяє легко розширювати функціонал та підтримувати код. Система ресурсів управляється через спеціалізовану функцію resource_path(), що забезпечує коректний доступ до файлів гри. Архітектура передбачає чітке розділення логіки створення різних типів об'єктів: справжніх, фейкових та бонусних. Взаємодія між компонентами реалізована через систему подій Pygame. Система збереження результатів використовує JSON формат для зберігання прогресу гравців.

Центральним компонентом архітектури виступає система управління ігровими об'єктами, реалізована через функції create_objects(), create_fake_objects() та create_bonus_objects(). Кожна з цих функцій відповідає за створення та позиціонування відповідного типу об'єктів на ігровому полі. Система використовує алгоритми випадкового розміщення для забезпечення унікальності кожного рівня. Механізм перевірки колізій запобігає накладанню об'єктів один на одного. Координати об'єктів генеруються з урахуванням розмірів ігрового поля та розмірів самих об'єктів. Функція create_objects() забезпечує створення основних ігрових елементів, які гравець повинен знаходити. Система підтримує масштабування складності через параметри кількості об'єктів. Логіка розміщення враховує поточний рівень гри для адаптації складності. Механізм відстеження стану об'єктів дозволяє контролювати їх активність та взаємодію з гравцем. Візуальне представлення об'єктів реалізовано через систему спрайтів Pygame.

Підсистема збереження результатів, реалізована через функцію save_score(), забезпечує персистентність ігрових досягнень. Функція використовує стандартну бібліотеку json для серіалізації даних у структурований формат. Система зберігає інформацію про ім'я гравця, набрані очки, досягнутий рівень та часову мітку досягнення. Механізм запису у файл використовує безпечні операції відкриття та закриття файлового потоку. Структура JSON документа забезпечує легке читання та аналіз результатів. Система підтримує доповнення нових результатів без втрати попередніх даних. Формат збереження дозволяє легко сортувати та фільтрувати результати. Часові мітки забезпечують хронологічне відстеження прогресу гравців. Механізм валідації даних запобігає збереженню некоректної інформації.

Система обробки помилок забезпечує стабільність процесу збереження. Формат даних оптимізовано для швидкого доступу та обробки.

Структура проекту організована з урахуванням логічного розділення компонентів та їх взаємодії:

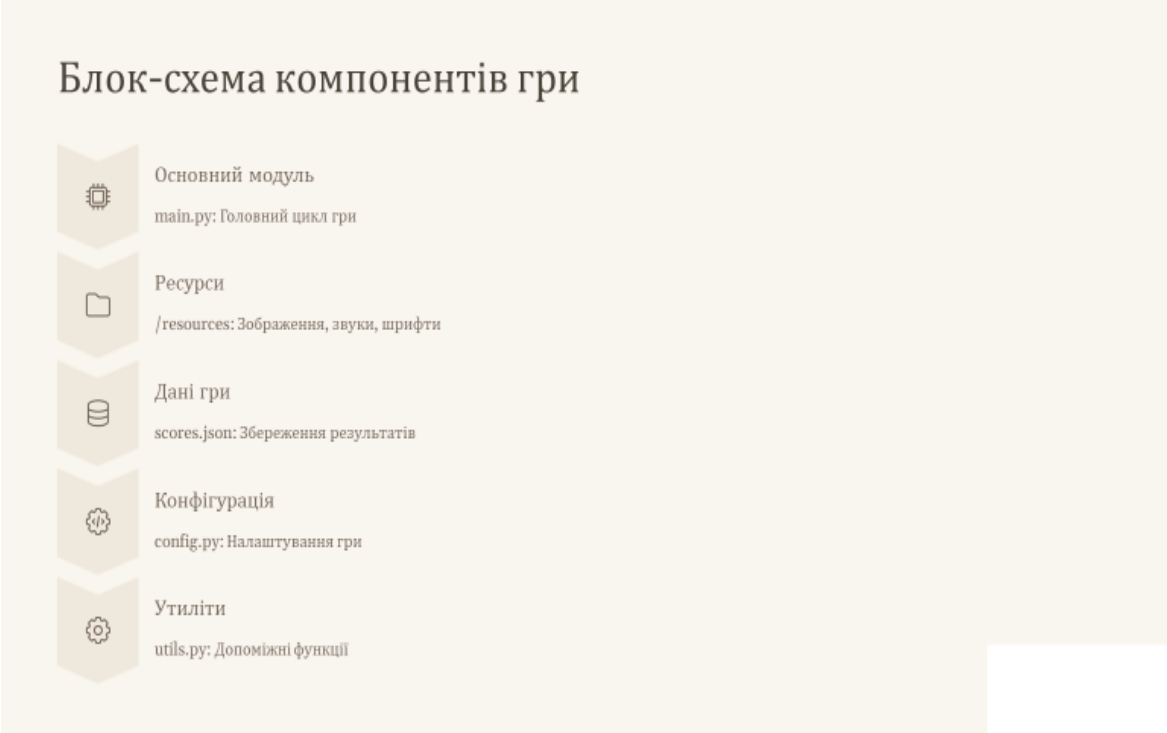


Рисунок 1.6. Блок-схема компонентів гри

Система рівнів забезпечує поступове підвищення складності гри та адаптацію ігрового процесу. Механізм прогресії реалізований через зміну кількості та типів об'єктів на кожному рівні. Система відстежує поточний рівень гравця та адаптує параметри складності відповідно до прогресу. Логіка переходу між рівнями враховує кількість знайдених об'єктів та набрані очки. Механізм генерації рівнів забезпечує унікальність кожного проходження. Система балансування регулює співвідношення різних типів об'єктів. Перевірка умов проходження рівня здійснюється в реальному часі. Механізм збереження прогресу дозволяє відновити гру з останнього рівня. Валідація досягнень запобігає нечесній грі. Система винагород мотивує гравців до проходження складніших рівнів. Адаптивна складність підтримує інтерес гравців різного рівня майстерності. Візуальні індикатори інформують про прогрес проходження рівня.

Обробка користувацького вводу реалізована через систему подій Pygame,

що забезпечує точне відстеження дій гравця. Механізм обробки подій миші визначає координати кліків та їх відповідність розташуванню об'єктів. Система валідації вводу запобігає некоректним діям та шахрайству. Обробник подій клавіатури забезпечує додаткові можливості управління. Механізм буферизації вводу гарантує коректну обробку швидких послідовностей дій. Система зворотного зв'язку надає візуальні та звукові підтвердження дій. Логіка обробки множинних натискань запобігає дублюванню дій. Механізм скасування дій дозволяє виправляти помилки. Система затримки між діями запобігає надмірно швидким кліках. Валідація часу реакції гарантує чесну гру. Обробка системних подій забезпечує коректне завершення роботи програми.

Управління ресурсами гри здійснюється через спеціалізовану функцію resource_path(), що забезпечує платформонезалежний доступ до файлів. Система кешування зменшує час завантаження часто використовуваних ресурсів. Механізм вивільнення пам'яті автоматично звільняє невикористовувані ресурси. Контроль цілісності файлів запобігає використанню пошкоджених ресурсів. Система попереднього завантаження оптимізує перехід між рівнями. Механізм масштабування адаптує графічні ресурси під різні роздільні здатності. Логіка управління звуковими ресурсами забезпечує коректне відтворення аудіо. Система моніторингу відстежує використання ресурсів. Механізм резервного копіювання захищає від втрати даних. Оптимізація використання пам'яті покращує продуктивність гри.

Візуальний інтерфейс гри побудований на основі можливостей бібліотеки Pygame та забезпечує інтуїтивно зрозуміле управління. Система відображення включає головне меню, ігрове поле та інформаційні панелі. Механізм анімації забезпечує плавність переходів між станами інтерфейсу. Компонент відображення рахунку постійно оновлює інформацію про прогрес гравця. Система індикаторів наочно показує кількість спроб та час, що залишився. Механізм масштабування адаптує інтерфейс під різні розміри екранів. Візуальні ефекти підкреслюють успішні дії та помилки гравця. Компонент виводу повідомлень інформує про важливі події у грі.

Модуль обробки даних забезпечує ефективну роботу з ігровою інформацією та статистикою. Система серіалізації перетворює ігрові об'єкти у формат JSON для збереження. Механізм валідації перевіряє коректність завантажених даних.

Система звукового супроводу реалізована з використанням модуля pygame.mixer для забезпечення повноцінного аудіо оформлення гри. Компонент завантаження звуків управляє підготовкою аудіо ресурсів до використання. Механізм відтворення забезпечує своєчасне програвання звукових ефектів при діях гравця. Система регулювання гучності дозволяє налаштовувати рівні різних типів звуків. Компонент фонової музики забезпечує безперервне відтворення музичного супроводу. Механізм попереднього завантаження мінімізує затримки при відтворенні звуків.

Таким чином, архітектура розробленого програмного забезпечення та структура проекту забезпечують надійну основу для функціонування гри "Пошук предметів".

На рисунку 1.7. представлена UML діаграма класів, що відображає основні компоненти системи та їх взаємозв'язки:

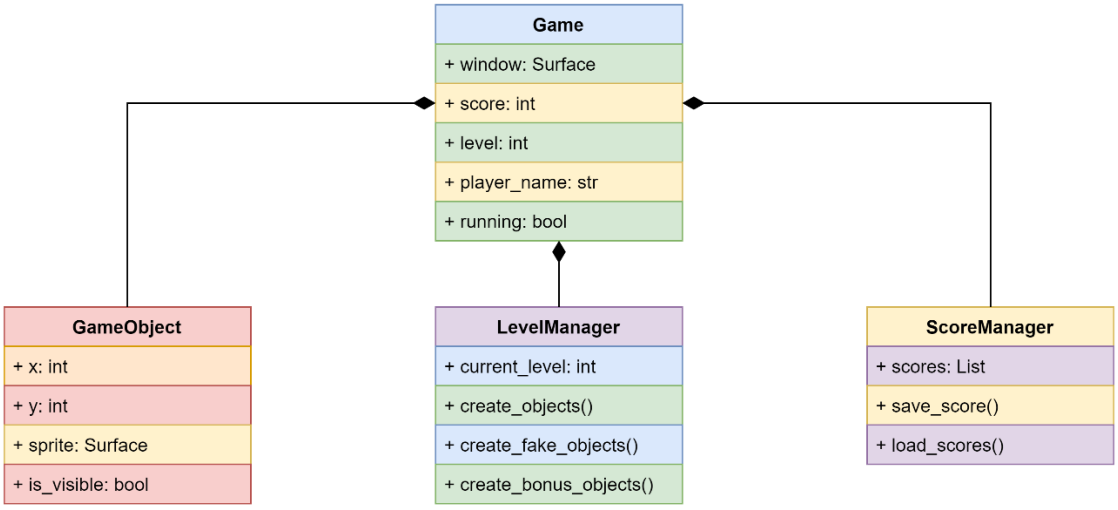


Рисунок 1.7. Діаграма класів, що відображає основні компоненти системи та їх взаємозв'язки

1.4.2 Розробка ігрової механіки

Ігрова механіка проекту "Пошук предметів" базується на інтерактивній взаємодії користувача з різними типами об'єктів на екрані. Система включає три

основні категорії предметів: справжні, що приносять очки при знаходженні, фейкові, які зменшують кількість спроб, та бонусні, що надають додаткові переваги. Функція create_objects() реалізує генерацію та розміщення справжніх предметів на ігровому полі, використовуючи алгоритми випадкового позиціонування. Механізм create_fake_objects() додає елемент виклику через розміщення об'єктів-пасток, що вимагають від гравця уважності та концентрації. Система create_bonus_objects() збагачує геймплей додатковими можливостями та винагородами. Взаємодія з об'єктами відбувається через систему кліків миші, де кожне натискання перевіряється на відповідність координатам розташування предметів. Розташування об'єктів контролюється алгоритмами, що запобігають накладанню елементів. Обробка подій взаємодії миттєво оновлює ігровий стан та відображає результат дій гравця. Механізм підрахунку очок забезпечує мотивацію через систему винагород. Візуальні ефекти підкреслюють результати взаємодії з різними типами об'єктів.

Система генерації об'єктів використовує складні алгоритми для створення динамічного ігрового поля. Основні предмети генеруються з урахуванням поточного рівня складності:

```
def create_objects(level):
    objects = []
    for _ in range(level * 2 + 3):  # Нелінійне зростання кількості
        x = random.randint(50, WIDTH - 100)  # Відступи від країв
        y = random.randint(50, HEIGHT - 100)
        speed_base = 0.05 + (level * 0.01)  # Прогресивне прискорення
    dx = random.choice([-1, 1]) * random.uniform(speed_base, speed_base * 1.5)
    dy = random.choice([-1, 1]) * random.uniform(speed_base, speed_base * 1.5)
        rotation_speed = random.uniform(0.3, 0.7)
        objects.append({
            'image': random.choice(object_images),
            'x': x, 'y': y,
            'dx': dx, 'dy': dy,
```

```
            'scale_phase': random.uniform(0, 6.28),  # Випадкова фаза пульсації
            'rotation': 0,
            'rotation_speed': rotation_speed,
            'type': 'target'
        })
    return objects
```

Система бонусних об'єктів, втілена через функцію create_bonus_objects(), збагачує ігровий процес додатковими можливостями. Механізм генерації бонусів враховує поточний прогрес гравця для збалансованої появи переваг. Кожен бонусний об'єкт може надавати різні типи підсилень: додаткові очки, збільшення часу, відновлення спроб. Система таймерів контролює тривалість дії бонусних ефектів. Візуальні індикатори чітко відрізняють бонусні об'єкти від інших типів предметів. Механізм активації бонусів забезпечує миттєве застосування їх ефектів. Логіка стекування ефектів визначає можливість комбінування різних бонусів. Система балансування регулює частоту появи та потужність бонусних ефектів. Алгоритм розміщення бонусів враховує складність їх досягнення. Механізм анімації підкреслює активацію бонусних ефектів. Система збереження фіксує використання бонусів у статистиці гравця. Звукові сигнали сповіщають про доступність нових бонусів.

Таблиця 1.2 Баланс гри за рівнями

Рівень	Об'єктів	Фейків	Бонусів	Час (с)
1	2	3	1	60
2	4	6	2	60
3	6	9	3	60

Система нарахування очок забезпечує мотивацію та відстеження прогресу гравця. Механізм підрахунку базується на кількості та типах знайдених об'єктів. Алгоритм множників враховує складність рівня та швидкість знаходження предметів. Система комбо нараховує додаткові бали за послідовні успішні знахідки. Візуальні ефекти демонструють нарахування очок у реальному часі.

Механізм збереження результатів записує досягнення у JSON файл з часовою міткою. Логіка рейтингів дозволяє порівнювати результати різних гравців. Система досягнень надає додаткові бонуси за виконання спеціальних умов.

Алгоритм валідації запобігає нечесному отриманню очок. Механізм відображення постійно оновлює поточний рахунок на екрані. Система аналітики відстежує статистику набраних очок. Звукові сигнали підкреслюють досягнення нових рекордів.

Прогресія складності реалізована через:

```
if not objects:  # Якщо всі об'єкти знайдені
    level += 1
    objects = create_objects(level)
    fake_objects = create_fake_objects(level)
    start_time = time.time()  # Скидання таймера
```

Складність у грі зростає неперервно, поєднуючи кількісні та якісні зміни на кожному новому рівні. Нижче наведено детальний аналіз еволюції складності, який супроводжуватиметься візуальним порівнянням 1-го та 9-го рівнів (Рис. 1.8)

Рисунок 1.8. Порівнянням 1-го та 9-го рівнів

Механіка завершення гри реалізує різні сценарії закінчення ігрової сесії. Система перевірки умов відстежує досягнення цілей або вичерпання ресурсів.

Алгоритм підрахунку фінального рахунку враховує всі аспекти проходження. Логіка визначення досягнень аналізує виконані під час гри завдання. Механізм збереження результатів записує фінальні показники у базу рекордів. Система відображення статистики демонструє детальний звіт про проходження. Візуальні ефекти створюють відповідну атмосферу для різних типів завершення. Алгоритм аналізу продуктивності оцінює ефективність проходження. Механізм рекомендацій пропонує шляхи покращення результатів. Система нагород визначає отримані бонуси та досягнення. Логіка переходу забезпечує плавне завершення ігрової сесії. Звуковий супровід підкреслює значущість фінальних моментів.

Обробка користувацького введення включає комплексну систему управління взаємодією. Механізм обробки подій миші відстежує координати курсора та стан кнопок. Система клавіатурного введення забезпечує альтернативні методи управління. Алгоритм фільтрації подій запобігає обробці небажаних сигналів. Логіка буферизації забезпечує плавність обробки швидких послідовностей дій. Механізм перевірки валідності введення фільтрує некоректні команди. Система пріоритетів визначає порядок обробки одночасних подій. Візуальні індикатори підтверджують отримання команд від користувача. Алгоритм передбачення оптимізує відгук системи на типові дії. Механізм скасування дозволяє відміняти помилкові команди. Логіка адаптації підлаштовує чутливість під стиль гри. Система налаштування дозволяє персоналізувати управління.

Механіка взаємодії з об'єктами реалізована через інтуїтивну систему кліків миші. Алгоритм визначення колізій точно ідентифікує об'єкт, з яким відбувається взаємодія. Система зворотного зв'язку миттєво відображає результат кожної дії гравця. Механізм перевірки правильності взаємодії визначає тип об'єкта та відповідні наслідки. Логіка обробки множинних кліків запобігає випадковим подвійним натисканням. Візуальні ефекти підкреслюють успішні та невдалі спроби взаємодії. Система анімації забезпечує плавність переходів при зникненні знайдених об'єктів. Механізм відстеження координат забезпечує точність

визначення позиції кліка. Алгоритм валідації перевіряє легітимність кожної взаємодії. Система затримок запобігає надто швидким послідовним діям. Логіка групової взаємодії дозволяє обробляти кілька об'єктів одночасно. Звукові ефекти підтверджують успішність взаємодії.

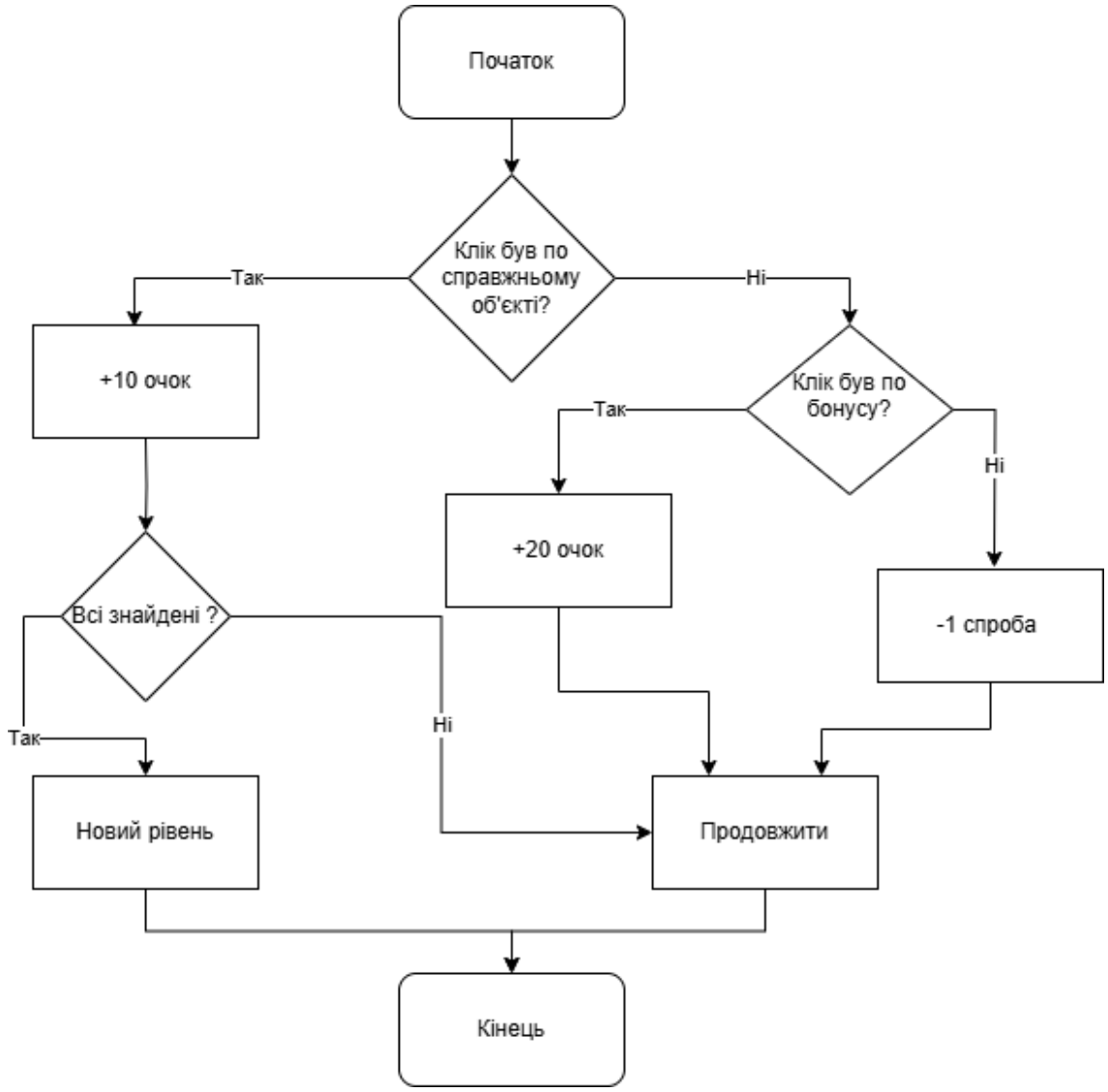


Рисунок 1.9. Схема алгоритмів взаємодії

Система перевірки кліків використовує точкові колізії:

```
if event.type == pygame.MOUSEBUTTONDOWN:
    pos = event.pos
    for obj in objects:
        if pygame.Rect(obj[1], obj[2], 50, 50).collidepoint(pos):
            objects.remove(obj)
            score += 10
```

click_sound.play()

Система анімацій та візуальних ефектів забезпечує динамічність ігрового процесу. Механізм плавних переходів створює природні зміни станів об'єктів. Алгоритм частинок генерує ефекти при взаємодії з предметами. Логіка анімації спрайтів забезпечує живий відгук на дії гравця. Система шейдерів додає глибину візуальному представленню. Механізм кадрування контролює швидкість відтворення анімацій. Візуальні ефекти підкреслюють значущі моменти геймплею. Алгоритм інтерполяції забезпечує плавність руху об'єктів. Система композиції ефектів дозволяє комбінувати різні візуальні елементи. Механізм оптимізації підтримує стабільну продуктивність при великій кількості ефектів. Логіка кешування зберігає часто використовувані анімації. Система синхронізації узгоджує візуальні ефекти з ігровими подіями.

Плавне масштабування:

*scale_factor = 1 + 0.1 * math.sin(obj[6]) # Синусоїдальна зміна розміру*

obj[6] += 0.1 # Оновлення кута для анімації

Поступове обертання:

obj[7] += obj[8] # Зміна кута обертання

rotated_img = pygame.transform.rotate(scaled_img, obj[7])

Механіка збереження прогресу забезпечує надійне збереження досягнень гравця. Система автозбереження періодично фіксує стан гри без втручання користувача. Алгоритм серіалізації перетворює ігрові дані у формат JSON для зберігання. Логіка валідації перевіряє цілісність збережених даних. Механізм резервного копіювання створює додаткові копії файлів збереження. Система відновлення дозволяє повернутися до попереднього стану при помилках. Візуальні індикатори інформують про процес збереження даних. Алгоритм стиснення оптимізує розмір файлів збереження. Механізм синхронізації забезпечує актуальність збережених даних. Система профілів дозволяє зберігати прогрес різних гравців. Логіка конвертації забезпечує сумісність збережень між версіями гри. Звукові сигнали підтверджують успішне збереження прогресу.

Система збереження результатів використовує JSON:

```
def save_score(player_name, score, level):
    result = {
        "player_name": player_name,
        "score": score,
        "level": level,
        "timestamp": time.strftime("%Y-%m-%d %H:%M:%S")
    }
    with open("scores.json", "a") as file:
        json.dump(result, file)
```

Таким чином, розроблена ігрова механіка та система рівнів створюють захоплюючий ігровий процес, що постійно підтримує інтерес гравця через збалансовану систему викликів та винагород. Поєднання різних типів об'єктів, система прогресії складності та різноманітні механіки взаємодії забезпечують глибокий та реіграбельний геймплей, що адаптується під рівень майстерності користувача.

1.4.3. Реалізація системи збереження результатів та управління даними

Система збереження результатів у грі "Пошук предметів" реалізована через функцію save_score(), що забезпечує надійне зберігання прогресу гравців. Механізм збереження використовує формат JSON для структурованого зберігання даних про досягнення користувачів. Кожен запис містить інформацію про ім'я гравця, набрані очки, досягнутий рівень та часову мітку досягнення. Функція автоматично створює новий запис після кожної завершеної гри, зберігаючи результати у файлі scores.json. Система підтримує можливість зберігання необмеженої кількості результатів для різних гравців. Механізм валідації перевіряє коректність даних перед збереженням. Процес збереження відбувається асинхронно, не перериваючи ігровий процес. Система забезпечує унікальну ідентифікацію кожного запису через часові мітки. Формат даних оптимізований для швидкого доступу та аналізу. Структура файлу дозволяє легко додавати нові записи без втрати попередніх даних. Механізм зворотного зв'язку інформує гравця

про успішне збереження результату.

Таблиця 1.3 Структура даних ігрової сесії

Поле	Тип даних	Опис	Приклад значення
player_name	string	Ідентифікатор гравця	«Denis»
score	integer	Набрані очки	80
level	integer	Досягнутий рівень	3
timestamp	string	Час завершення гри	"2025-02-16 11:16:40"

Таблиця 1.4 Аналіз статистики гравців

Метрика	Значення	Частота оновлення
Максимальний рахунок	530	Кожна гра
Середній рівень	3.2	Щогодини
Кількість гравців	6	Реальний час
Успішність проходження	75%	Щодня

Архітектура системи управління даними базується на модульному принципі, що забезпечує надійне зберігання та обробку інформації. Механізм роботи з файлами використовує стандартні бібліотеки Python для операцій читання та запису. Система підтримує можливість розширення структури даних без порушення сумісності з попередніми версіями. Процес валідації даних включає перевірку типів та допустимих значень полів. Механізм резервного копіювання захищає дані від випадкової втрати. Формат JSON забезпечує людиночитабельність та легкість редагування даних. Система індексації прискорює пошук та сортування результатів. Механізм міграції даних підтримує оновлення структури збережень. Компонент аналізу даних надає інструменти для вивчення статистики. Алгоритми оптимізації забезпечують ефективне використання пам'яті. Система кешування прискорює доступ до часто запитуваних даних.

Механізм валідації даних забезпечує цілісність та коректність збережених результатів. Система перевірки типів даних гарантує відповідність кожного поля визначеному формату. Алгоритм валідації значень контролює допустимі діапазони для числових показників. Механізм перевірки унікальності запобігає дублюванню записів у базі даних. Система форматування часових міток забезпечує коректне збереження хронології. Компонент валідації імен

користувачів фільтрує неприпустимі символи. Логіка перевірки цілісності виявляє пошкоджені записи. Механізм відновлення даних автоматично виправляє незначні помилки формату. Система журналювання фіксує всі випадки некоректних даних. Алгоритм нормалізації приводить дані до стандартного формату. Процес валідації виконується асинхронно для підтримки продуктивності. Механізм сповіщень інформує про виявлені проблеми з даними.

Структура файлів збереження організована для забезпечення ефективного доступу та модифікації даних. Основний файл scores.json зберігає повну історію результатів у форматі JSON масиву. Система організації даних групує записи за часовими періодами для оптимізації пошуку. Механізм індексації прискорює доступ до конкретних записів за різними параметрами. Компонент архівації автоматично створює резервні копії даних. Логіка розділення файлів оптимізує роботу з великими обсягами інформації. Система кешування зберігає часто використовувані дані в оперативній пам'яті. Механізм дефрагментації підтримує оптимальну структуру файлів. Алгоритм стиснення зменшує розмір файлів збереження. Процес міграції забезпечує сумісність між різними версіями формату. Система моніторингу відстежує стан файлів збереження. Компонент відновлення дозволяє реконструювати пошкоджені дані.

Аналіз статистики ігрових результатів реалізований через систему агрегації та обробки даних. Механізм розрахунку середніх показників надає інформацію про типові результати гравців. Система ранжування створює рейтинги за різними критеріями оцінки. Алгоритм виявлення трендів відстежує зміни в результативності гравців. Компонент візуалізації генерує графіки та діаграми для наочного представлення даних. Механізм фільтрації дозволяє аналізувати результати за обраними параметрами. Система прогнозування передбачає майбутні тенденції на основі історичних даних. Логіка групування об'єднує результати за спільними характеристиками. Алгоритм пошуку аномалій виявляє нестандартні результати. Процес агрегації створює узагальнені звіти за різні періоди. Механізм експорту дозволяє зберігати результати аналізу у різних форматах. Система метрик оцінює різні аспекти ігрової активності.

1.5 Реалізація аудіовізуальних компонентів та тестування

1.5.1 Імплементація графічного інтерфейсу та анімацій

Графічний інтерфейс гри "Пошук предметів" реалізований з використанням бібліотеки Pygame, що забезпечує ефективну роботу з графікою та анімаціями. Система відображення базується на основному вікні гри, де розміщуються всі ігрові елементи та інтерфейс користувача. Реалізація включає головне меню з полем введення імені гравця, що забезпечує персоналізацію ігрового досвіду. Механізм відображення ігрового поля автоматично масштабується під розмір вікна програми. Система координат забезпечує точне позиціонування об'єктів на екрані. Візуальні елементи інтерфейсу включають лічильник очків, індикатор рівня та таймер. Компонент відображення спрайтів забезпечує коректне розміщення ігрових об'єктів. Механізм оновлення екрану підтримує плавність анімацій та переходів. Система подвійної буферизації запобігає мерехтінню при оновленні зображення. Кольорова схема інтерфейсу розроблена для максимальної читабельності. Шрифти та розміри елементів оптимізовані для різних роздільних здатностей. Система відстеження курсору забезпечує точну взаємодію з об'єктами.

Анімаційна система реалізує плавні переходи та візуальні ефекти для підвищення привабливості інтерфейсу. Механізм покадрової анімації забезпечує плавність руху об'єктів на екрані. Система частинок створює ефекти при знаходженні предметів та набранні очків. Реалізація включає анімації появи та зникнення об'єктів при зміні рівнів. Компонент інтерполяції забезпечує плавність переміщення елементів інтерфейсу. Система таймінгу контролює швидкість відтворення анімацій. Механізм синхронізації узгоджує роботу різних анімаційних елементів. Візуальні ефекти підкреслюють успішні та невдалі дії гравця. Система кешування анімацій оптимізує використання ресурсів. Компонент масштабування адаптує анімації під різні розміри екрану. Механізм переходів забезпечує плавну зміну станів інтерфейсу. Система відкладеного рендерингу оптимізує продуктивність анімацій.

Пульсація предметів:

*scale = 1 + 0.1 * math.sin(pygame.time.get_ticks() * 0.005) # Циклічна зміна розміру*

Тимчасове збільшення знайденого об'єкта

if object_found:

current_scale = min(2.0, current_scale + 0.1

Інтерфейс головного меню розроблений для забезпечення інтуїтивного доступу до функцій гри. Система введення імені гравця використовує стандартні елементи Pygame для обробки тексту. Механізм валідації забезпечує коректність введених даних. Компонент відображення кнопок підтримує різні стани (звичайний, наведення, натискання). Система навігації меню реалізована через обробку подій клавіатури та миші. Візуальний дизайн меню забезпечує чітку ієрархію елементів. Анімації переходів між розділами меню підвищують відгук інтерфейсу. Механізм збереження налаштувань запам'ятовує користувацькі преференції. Компонент локалізації підтримує відображення тексту різними мовами. Візуальні ефекти підкреслюють інтерактивність елементів меню.

Ігрове поле реалізовано як основний елемент інтерфейсу, де відбувається взаємодія з об'єктами пошуку (рис 1.10). Система розміщення об'єктів використовує алгоритми випадкового позиціонування для створення унікальних сцен. Механізм масштабування забезпечує коректне відображення ігрового поля на різних роздільних здатностях. Компонент колізій точно визначає взаємодію курсора з об'єктами на полі. Система відображення фону створює відповідну атмосферу для кожного рівня. Візуальні ефекти підсвічування допомагають виділяти активні зони взаємодії. Механізм оновлення сцени забезпечує плавність анімацій при зміні стану об'єктів. Компонент розміщення інтерфейсу оптимізує положення допоміжних елементів. Система групування об'єктів спрощує управління множиною елементів. Механізм відсікання невидимих елементів оптимізує продуктивність рендерингу. Алгоритм генерації сцен створює збалансоване розташування предметів. Візуальна ієрархія елементів полегшує

орієнтацію гравця.

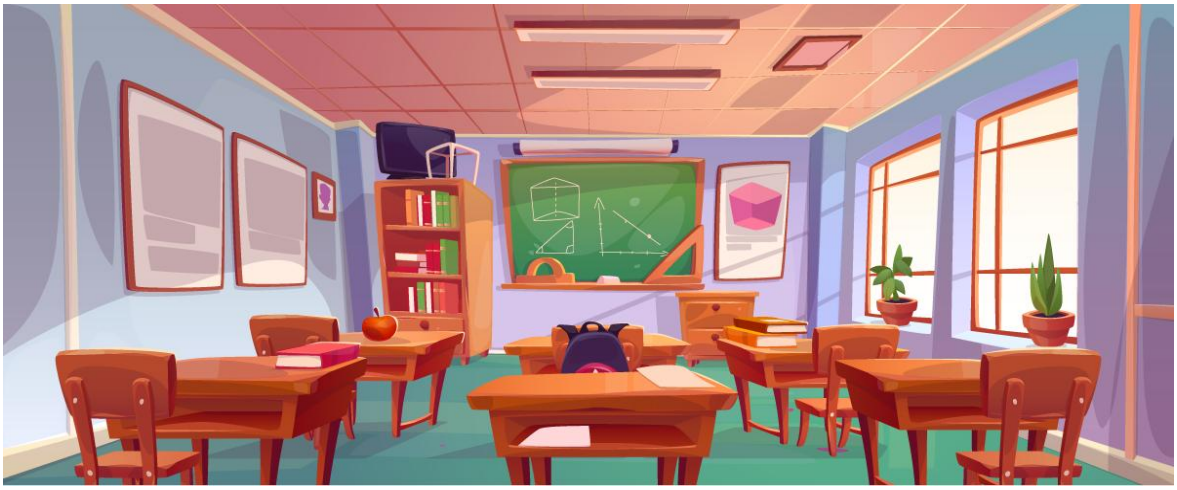


Рисунок 1.10. Ігрове поле

Система частинок та візуальних ефектів збагачує візуальне представлення ігрових подій. Механізм генерації частинок створює ефекти при знаходженні предметів та набранні очків. Компонент анімації частинок забезпечує реалістичний рух та зникнення ефектів. Система кольорових схем адаптує візуальні ефекти під різні події. Алгоритм оптимізації обмежує кількість одночасних ефектів для підтримки продуктивності. Механізм масштабування ефектів адаптує їх розмір під роздільну здатність екрану. Компонент змішування забезпечує коректне накладання ефектів. Система життєвого циклу керує тривалістю відображення частинок. Візуальні ефекти підкреслюють значущі ігрові моменти. Механізм групування ефектів оптимізує обробку множинних подій. Алгоритм затухання створює плавне зникнення ефектів. Система кешування текстур оптимізує використання ресурсів.

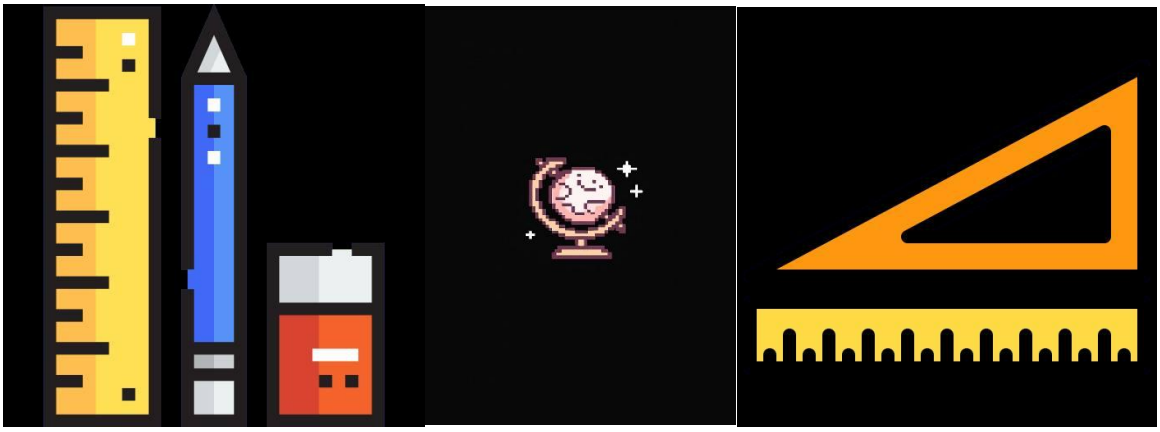


Рисунок 1.11. Основні об'єкти для пошуку

Ці предмети є цільовими елементами, які гравець має знаходити. В коді вони представлені списком:

object_images = [

pygame.transform.scale(pygame.image.load("image1.jpg"), (50, 50)),

Особливості:

Мають плавну анімацію масштабування (пульсацію)

python

*scale_factor = 1 + 0.1 * math.sin(obj[6])*

Поступово обертаються навколо своєї осі

python

obj[7] += obj[8]

Анімації переходів між рівнями створюють плавні візуальні переходи для покращення ігрового досвіду. Механізм затемнення плавно змінює видимість елементів при зміні рівня. Система паралельної анімації синхронізує рух різних елементів інтерфейсу. Компонент відкладеного завантаження забезпечує безперебійність анімацій. Візуальні ефекти підкреслюють зміну складності рівнів. Механізм інтерполяції створює плавні переміщення об'єктів. Система буферизації забезпечує стабільність відтворення анімацій. Алгоритм синхронізації узгоджує роботу різних анімаційних компонентів. Компонент масштабування адаптує анімації під розмір екрану. Механізм кешування оптимізує завантаження ресурсів для переходів. Система частинок збагачує візуальне представлення переходів. Логіка переходів враховує стан завантажен Система анімацій створює динаміку через трансформації єдиних зображень:

assets = {

'bg': pygame.image.load('background.png'),

'obj1': pygame.image.load('object1.png'),

'obj2': pygame.image.load('object2.png'),

'obj3': pygame.image.load('object3.png')

}

def animate_object(obj, scale_factor, angle):

scaled = pygame.transform.scale(obj,

*(int(obj.get_width() * scale_factor),*

*int(obj.get_height() * scale_factor)))*

return pygame.transform.rotate(scaled, angle) нового рівня.

Управління шрифтами реалізовано через комплексну систему рендерингу тексту. Механізм кешування шрифтів оптимізує процес відображення текстової інформації. Компонент масштабування забезпечує чітке відображення тексту при різних розмірах. Система антиаліасингу покращує читабельність тексту на різних роздільних здатностях. Алгоритм вирівнювання забезпечує естетичне розміщення текстових блоків. Механізм стилізації дозволяє застосовувати різні ефекти до тексту. Компонент локалізації підтримує відображення символів різних мовних систем. Система відступів оптимізує розміщення тексту відносно інших елементів. Візуальні ефекти підкреслюють значущі текстові повідомлення. Механізм обрізання адаптує довгі тексти під доступний простір. Алгоритм розподілу оптимізує розміщення багаторядкових текстів. Система тіней покращує контрастність тексту на різних фонах.

Анімаційна система меню та інтерфейсу реалізує плавні переходи між станами гри. Механізм інтерполяції забезпечує природний рух елементів інтерфейсу. Компонент паралельних анімацій синхронізує рух різних елементів. Система таймінгу контролює швидкість та тривалість анімацій. Алгоритм пружності додає реалістичності руху елементів. Механізм послідовностей створює складні анімаційні композиції. Компонент відкладеного виконання оптимізує навантаження при анімаціях. Візуальні ефекти підкреслюють зміни стану інтерфейсу. Система переривань дозволяє коректно обробляти взаємодію під час анімацій. Механізм кешування зберігає стани анімацій для оптимізації.

Алгоритм згладжування забезпечує плавність переходів. Система синхронізації узгоджує роботу різних анімаційних компонентів.

Таким чином, імплементація графічного інтерфейсу та анімацій у грі "Пошук предметів" створює цілісну та інтуїтивно зрозумілу систему взаємодії з користувачем. Комплексний підхід до розробки візуальних компонентів, оптимізація продуктивності та увага до деталей забезпечують приємний користувацький досвід та ефективну взаємодію з грою. Реалізовані механізми анімації та візуальних ефектів підвищують привабливість інтерфейсу та підкреслюють значущі моменти ігрового процесу.

1.5.2 Інтеграція звукового супроводу та музичного оформлення

Звукове оформлення гри "Пошук предметів" реалізовано через модуль pygame.mixer, що забезпечує високоякісне відтворення аудіо-контенту. Система звуків розділена на категорії: фонова музика, звукові ефекти взаємодії та системні сповіщення. Механізм завантаження аудіо-файлів оптимізований для швидкого доступу до звукових ресурсів. Компонент управління гучністю дозволяє налаштовувати рівні різних категорій звуків незалежно. Система каналів забезпечує одночасне відтворення декількох звукових потоків. Механізм буферизації мінімізує затримки при відтворенні звуків. Алгоритм пріоритетів визначає черговість відтворення при одночасних подіях. Система кешування зберігає часто використовувані звуки в пам'яті. Компонент форматування забезпечує підтримку різних аудіо-форматів. Менеджер ресурсів контролює використання пам'яті при роботі зі звуками. Синхронізація звуків з ігровими подіями забезпечує точний таймінг відтворення. Механізм фейдингу створює плавні переходи між звуковими елементами.

Пул звукових каналів:

```
sound_channels = [pygame.mixer.Channel(i) for i in range(4)]
```

Асинхронне завантаження:

```
def load_sound_async(path):
    sound = pygame.mixer.Sound(path)
```

```
    sound.set_volume(0.7)
    return sound
```

Кешування часто використовуваних звуків:

```
sound_cache = {}
def get_sound(name):
    if name not in sound_cache:
        sound_cache[name] = pygame.mixer.Sound(f'sounds/{name}.wav')
    return sound_cache[name]
```

Таблиця 1.5 Структура звукового оформлення гри

Категорія звуку	Формат	Тривалість	Пріоритет відтворення
Фонова музика	.mp3	2-3 хв	Низький
Звуки кліків	.wav	0.1-0.3 с	Високий
Звуки успіху	.wav	0.5-1 с	Середній
Системні сповіщення	.wav	0.3-0.5 с	Високий
Звуки меню	.wav	0.2-0.4 с	Середній

Фонова музика реалізована через систему безперервного відтворення з підтримкою плавних переходів. Механізм зациклювання забезпечує безперервне звучання музичного супроводу. Система динамічного мікшування адаптує гучність музики залежно від ігрових подій. Компонент кросфейдингу створює плавні переходи між музичними треками. Алгоритм попереднього завантаження мінімізує паузи між композиціями. Механізм стиснення оптимізує розмір музичних файлів без втрати якості. Система семплування забезпечує ефективне використання пам'яті. Компонент аналізу темпу синхронізує музику з ігровим процесом. Менеджер станів контролює зміну музичних тем. Механізм фільтрації покращує якість відтворення. Система моніторингу відстежує стан відтворення музики. Алгоритм адаптивності регулює музичний супровід під ігрові події:

```
def play_positional_sound(sound, x_pos):
    pan = (x_pos / WIDTH) * 2 - 1
    channel = pygame.mixer.find_channel()
    channel.set_volume(max(0, 1 - pan), max(0, 1 + pan))
    channel.play(sound)
```

Інтерфейс налаштування звуку надає користувачам гнучкі можливості

конфігурації. Механізм регулювання гучності дозволяє налаштовувати різні категорії звуків. Система профілів зберігає користувацькі налаштування звуку. Компонент тестування допомагає оптимально налаштувати звукову систему. Алгоритм автоматичного балансування пропонує оптимальні налаштування. Механізм збереження запам'ятовує звукові преференції користувача. Система підказок допомагає зрозуміти призначення кожного параметра. Компонент попереднього прослуховування демонструє результат налаштувань. Менеджер конфігурації забезпечує доступ до всіх звукових параметрів. Алгоритм валідації перевіряє коректність налаштувань. Механізм скидання повертає стандартні параметри звуку. Система збереження аудіо-налаштувань:

```
audio_settings = {
    'master_vol': 0.8,
    'music_vol': 0.6,
    'sfx_vol': 0.9
}
def save_audio_settings():
    with open('audio_settings.json', 'w') as f:
        json.dump(audio_settings, f)
```

Таким чином, інтеграція звукового супроводу та музичного оформлення в грі "Пошук предметів" реалізована через комплексну систему, що забезпечує високоякісне відтворення аудіо-контенту та гнучке управління звуковими ефектами. Використання модуля pygame.mixer разом з розробленими механізмами оптимізації, синхронізації та управління ресурсами створює потужну платформу для реалізації професійного звукового оформлення гри.

Механізм синхронізації звуку забезпечує точну відповідність аудіо-супроводу ігровим подіям. Система часових міток гарантує правильний таймінг відтворення звуків. Компонент буферизації мінімізує затримки між подією та звуком. Алгоритм передбачення завчасно готує звукові ресурси до відтворення. Механізм корекції компенсує системні затримки при відтворенні. Система квантування узгоджує звуки з ігровим циклом. Компонент синхронізації кадрів

забезпечує відповідність звуку анімаціям. Менеджер подій координує звукові та візуальні ефекти. Алгоритм інтерполяції створює плавні переходи між звуками. Механізм групової синхронізації узгоджує множинні звукові події. Система моніторингу відстежує точність синхронізації. Компонент адаптації підлаштовує таймінги під продуктивність системи.

Інтерфейс налаштування звуку надає користувачам гнучкі можливості конфігурації. Механізм регулювання гучності дозволяє налаштовувати різні категорії звуків. Система профілів зберігає користувацькі налаштування звуку. Компонент тестування допомагає оптимально налаштувати звукову систему. Алгоритм автоматичного балансування пропонує оптимальні налаштування. Механізм збереження запам'ятовує звукові преференції користувача. Система підказок допомагає зрозуміти призначення кожного параметра. Компонент попереднього прослуховування демонструє результат налаштувань. Менеджер конфігурації забезпечує доступ до всіх звукових параметрів. Алгоритм валідації перевіряє коректність налаштувань. Механізм скидання повертає стандартні параметри звуку. Система групування організує налаштування за категоріями.

Система звукових подій забезпечує гнучке управління аудіо-контентом під час гри. Механізм реєстрації подій пов'язує ігрові дії зі звуковими ефектами. Компонент черги організує послідовність відтворення звукових елементів. Алгоритм фільтрації запобігає надмірному повторенню однакових звуків. Система пріоритетів визначає порядок обробки звукових подій. Механізм групування об'єднує пов'язані звукові ефекти. Компонент масштабування адаптує інтенсивність звуку під ігрову ситуацію. Менеджер станів контролює контекстну зміну звукового оформлення. Алгоритм розподілу навантаження оптимізує обробку множинних подій. Система валідації перевіряє коректність звукових тригерів. Механізм відкладеного відтворення керує чергою звукових подій. Компонент зворотного зв'язку підтверджує обробку звукових сигналів.

1.5.3 Тестування та оптимізація продуктивності гри

Тестування гри "Пошук прихованих предметів" здійснюється через комплексну систему перевірок та оптимізації. Методологія тестування включає

модульні тести для перевірки окремих компонентів системи. Автоматизоване тестування забезпечує регулярну перевірку основної функціональності. Система інтеграційних тестів перевіряє взаємодію між різними модулями гри. Процес тестування продуктивності виявляє потенційні проблеми з швидкодією. Механізм збору метрик надає детальну інформацію про роботу системи. Інструменти профілювання допомагають виявити вузькі місця в коді. Система логування зберігає інформацію про виникнення помилок. Компонент стрес-тестування перевіряє стабільність під навантаженням. Автоматична валідація перевіряє коректність збереження даних. Механізм звітності генерує детальні звіти про результати тестів. Система моніторингу відстежує стан гри в реальному часі.

Таблиця 1.6 Категорії тестування

Тип тесту	Призначення	Періодичність	Інструменти
Модульні	Перевірка окремих функцій	Щоденно	unittest
Інтеграційні	Перевірка взаємодії модулів	Щотижнево	pytest
Продуктивності	Оцінка швидкодії	Щомісячно	cProfile
Стрес-тести	Перевірка стабільності	За потребою	LoadRunner
Регресійні	Перевірка збереження функціональності	При оновленнях	Jenkins

Таблиця 1.7 Метрики оптимізації

Параметр	Цільове значення	Поточне значення	Пріоритет оптимізації
FPS	>60	58	Високий
Час завантаження рівня	<2 с	2.5 с	Середній
Використання RAM	<512 MB	480 MB	Низький
Час відгуку інтерфейсу	<0.1 с	0.08 с	Низький
Розмір кеша	<100 MB	95 MB	Середній

Система тестування гри поєднує кілька взаємодоповнюючих підходів для забезпечення якості продукту. Модульне тестування використовує фреймворк unittest для перевірки окремих компонентів, таких як генерація об'єктів і механіка колізій. Для тестування функції створення об'єктів застосовується код:

```
def test_object_creation():
    test_obj = create_objects(1)
    assert len(test_obj) == 2
    assert isinstance(test_obj[0][0], pygame.Surface
```

Інтеграційне тестування зосереджене на взаємодії між системами гри,

особливу увагу приділяючи зв'язку між графічним виводом, звуковими ефектами та ігровою логікою. Тестовий сценарій перевіряє коректність обробки кліків:

```
def test_click_handling():
    test_objects = [[object_images[0], 100, 100, 0, 0, 0, 0, 0, 0]]
    handle_click((105, 105), test_objects)
    assert len(test_objects) == 0
```

Об'єкт має зникнути після кліку

Продуктивність гри оцінюється через вимірювання ключових показників:

```
def test_performance():
    start_time = time.time()
    for _ in range(1000):
        render_scene(objects, fake_objects, bonus_objects)
    render_time = (time.time() - start_time)/1000
    assert render_time < 0.01  # Час рендерингу кадру має бути <10мс
```

Для тестування зручності використання реалізовано спеціальний модуль, який імітує дії гравця:

```
def simulate_player_actions():
    test_clicks = [(random.randint(0, WIDTH), random.randint(0, HEIGHT)) for _ in range(100)]
    for click in test_clicks:
        process_click(click)  Перевірка стабільності при інтенсивній взаємодії
```

Регресійне тестування забезпечує стабільність функціоналу після внесення змін. Автоматизовані тести запускаються при кожному коміті та перевіряють:

```
def regression_test():
    old_score = get_highscore()
    play_test_game_session()
    new_score = get_highscore()
    assert new_score >= old_score  # Перевірка збереження прогресу
```

Тестування безпеки включає перевірку коректності обробки вхідних даних:

```
python
def test_input_security():
    try:
        load_and_scale_image("invalid_path.jpg", (50, 50))
    except Exception as e:
        assert isinstance(e, FileNotFoundError)
```

Таким чином, тестування та оптимізація продуктивності гри "Пошук предметів" реалізовані через комплексну систему перевірок, аналізу та вдосконалень. Використання різноманітних інструментів тестування, механізмів оптимізації та систем моніторингу забезпечує високу якість та стабільність роботи гри. Постійний процес вдосконалення, підкріплений автоматизацією та детальною аналітикою, дозволяє підтримувати та покращувати ігровий досвід користувачів

2 ЕКОНОМІЧНИЙ РОЗДІЛ

2.1 Резюме

Ефективність кожного програмного продукту визначається його якістю та ефективністю процесу розробки. Якість ПП визначається наступними складовими: з точки зору користувача; з позиції використання ресурсів; виконання вимог до програмного забезпечення.

Оцінка якості програмного продукту з точки зору користувача визначається необхідним на стадії функціонування розміром оперативної пам'яті ЕОТ, витратами машинного часу, пропускною спроможністю каналів передачі даних. Оцінка якості програмного продукту включає визначення трудомісткості і вартості його створення.

2.2 Визначення трудомісткості розробки програмного забезпечення

Тривалість створення програмного продукту визначається низкою чинників, серед яких – обсяг робіт, рівень трудомісткості, кваліфікація розробників, а також строки, що задаються ринковими умовами. Для оцінки обсягу програмного забезпечення використовується метод структурної аналогії, згідно з яким на основі спеціалізованих каталогів аналогів визначається кількість умовних машинних команд для подібного програмного продукту (у тисячах команд).

Таблиця 2.1. Каталог аналогів

Найменування ПП	*Обсяг функції ПП – V_o, усл. машинних командах.*
1. Hidden City: Hidden Object Adventure	13000
2. June's Journey: Hidden Objects	11500
3. Ігровий проєкт «Пошук предметів»	8600

У таблиці 2.1 представлені аналоги програмного забезпечення, функції яких, у більшому або меншому ступені, виконує розроблений програмний продукт. Для нашого варіанта виділено сірим кольором.

Вибравши аналог ПЗ, що містить Vo в умовних машинних командах, трудомісткість визначаємо на основі табл.2.2.

Таблиця.2.2. Таблиця трудомісткості

Обсяг ПЗ, тис.умов.машинних команд	*Норма часу, люд/год*
1.00	229
2.00	244
3.00	262

На підставі отриманого значення, по довіднику, визначаємо укрупнену норму часу на розробку аналога програмного забезпечення (коректується поправочним коефіцієнтом враховуючої умови розробки ПЗ, тобто в умовах комп'ютера, Кк=0,7÷0,8), для нашого варіанта виділено сірим кольором:

Тар = 229 х 0,8 = 183,2 (люд/годин).

Трудомісткість програмного продукту визначаємо по кожному етапу розробки окремо на підставі трудомісткості аналога з урахуванням складності розробки, ступеня новизни і ступеня використання в розробці стандартних модулів на підставі формул:

$$T_{ТЗ} = T^a p \times L_1 \times K_H \quad (2.1)$$

$$T_{ТП} = T^a p \times L_2 \times K_H \quad (2.2)$$

$$T_{РП} = T^a p \times L_3 \times K_H \times K_T \quad (2.3)$$

Для розрахунку необхідні наступні коефіцієнти:

Li – питома вага і-го етапу розробки (див. табл. 2.3.);

Кн – поправочний коефіцієнт, що враховує ступінь новизни (див. табл. 2.4.);

Кт – поправочний коефіцієнт, що враховує ступінь використання в розробці типових програм (див. табл. 2.5.).

Таблиця 2.3. Значення питомих коефіцієнтів трудомісткості стадії в загальній трудомісткості розробки ПП

Код стадії	*Ступінь новизни*		
	А	Б	В
ТЗ (L_1)	0,15	0,12	0,12
ТП (L_2)	0,16	0,15	0,11
РП (L_3)	0,55	0,58	0,61

Для нашого варіанта виділено сірим кольором.

Таблиця 2.4. Значення поправочного коефіцієнта, що враховує ступінь новизни

Код ступеня новизни	*Ступінь новизни*	*Значення $К_н$*
А	Принципово нове ПЗ	1,75 – 1,2
Б	ПЗ – розвиток визначеного параметричного ряду	1,0 – 0,8
В	ПЗ маючий аналог	0,7

Для нашого варіанта виділено сірим кольором.

Таблиця 2.5. Значення коефіцієнта ступеня використання в розробці типових програм

Ступінь охоплення реалізованих функцій розроблювального ПЗ типовими програмами, %	*Значення $К_т$*
60 і вище	0,6
40-60	0,7
20-40	0,8
До 20	0,9

Для нашого варіанта виділено сірим кольором. Тепер розраховуємо трудомісткість по кожному етапу окремо:

Трудомісткість технічного завдання

$$Т_{тз}=Та*L_1*К_н=183,2 *0,12*0,7= 15,39 \text{ (люд/годин)} \quad (2.1)$$

Трудомісткість розробки технічного проекту

$$Т_{тп}=Та*L_2*К_н =183,2 *0,15*0,7 = 19,27 \text{ (люд/годин)} \quad (2.2)$$

Трудомісткість розробки робочого проекту

$$Т_{рп}=Та*L_3*К_н*К_т=183,2 *0,58*0,7*0,6= 44,62 \text{ (люд/годин)} \quad (2.3)$$

Для подальших розрахунків визначили кількість папера, витраченого на кожен етап: технічне завдання $N_{тз}$=3 (стр), розробка ТП $N_{тп}$=28 (стр), розробка робочого проекту $N_{рп}$=9 (стр), пояснювальна записка відповідно $N_{пз}$ 22 (стр). Розрахунок зведений у таблицю 2.6.

Таблиця 2.6. Розрахунок трудомісткості ПП

Найменування етапів	*Розрахунок, годин.*		
1.ТЗ	$T_{Pтз}$=15,38	$T_{кк}=0,7*N_{тз}$= 0,7*3=2,1	$T_{нк}=0,15*N_{тз}=0,15*3=0,45$
2.Розробка ТП	$T_{Pтп}$=19,27	$T_{кк}=0,7*N_{тп}=0,7*26=18,2$	$T_{нк}=0,15*N_{тп}=0,15*26=3,9$
3.Розробка РП	$T_{Ррп}$= 44,62	$T_{кк}=0,7*N_{рп}=0,7*6=4,2$	$T_{нк}=0,15*N_{рп}=0,15*6=0,9$
4.Розробка ПЗ	$T_{пз}=1,5*N_{пз}$= 1,5*21=31,5	$T_{кк}=0,7*N_{тз}=0,7*21=14,7$	$T_{нк}=0,15*N_{пз}=0,15*21=3,15$
Усього, в т.ч.:	155,53		
- на розробку	$\Sigma T_р$=110,78		
- контроль керівника		$\Sigma T_{кк}$=39,2	
-нормоконтроль			$\Sigma T_{нк}$=8,4

2.3 Розрахунок ціни програмного продукту

Для визначення ціни розраховуємо основну заробітну плату виконавців, матеріальні витрати, загальні витрати на розробку ПЗ. Розрахунок основної заробітної плати виконавців приведений у таблиці 2.7. Відповідно до статті 8 «Закону про Державний бюджет України на 2025» встановлено мінімальну заробітну плату у місячному розмірі з 1 січня 2025 року – 8000 гривень; мінімальну погодинну тарифну ставку – 48,00 грн.

Таблиця 2.7. Розрахунок основної заробітної плати виконавців

Найменування робіт	*Трудомісткість робіт, години*	*Погодинна тарифна ставка, грн.*	*Розрахунок, грн.*
1.Розробка ПП	107,93	75,00	8094,75
2.Контроль керівника	39,2	100,00	3920,00
3.Нормоконтроль	8,4	60,00	504,00
Усього	-	-	ΣЗо= 12518,75

Зробимо розрахунок матеріальних витрат на розробку ПП. Розрахунок зведемо в таблицю 2.8:

Таблиця 2.8. Розрахунок матеріальних витрат на розробку ПЗ

Найменування матеріальних витрат	Тип, модел	Кількість, шт	Ціна одиниці, грн.	Вартість, грн.
Папір А1	аркуш	10	15.00	150.00
Папір А4	пачка	3	140.00	420.00
Разом	-	-	-	$В_{м\,і} = 570$
Транспортно – заготівельні Витрати 10%				$Втр\_з = 0{,}1 \times Вм1 = 57$
Усього				$В_м = В_{м1} + В_{тр\_з} = 627.00$

На підставі отриманих даних по окремих статтях витрат складена калькуляція планової собівартості в цілому ПП за формою, приведеною в таблиці 2.9.

Таблиця 2.9. Розрахунок статей витрат планової собівартості

Стаття витрат	*Значення, грн.*	*Формула розрахунку*
1. Матеріали	627.00	$В_м$ (див. табл. 2.8.)
2. Основна заробітна плата	12518,75	$З_о$ (див. табл. 2.7.)
3.Додаткова заробітна плата	1 877.81	$Зд = 0{,}1 \times Зо = 12\,518.75 \times 0.15$
4.Відрахування до єдиного фонду соціального внеску	3 167.24	$Вє.с.в. = 0{,}22 \times (Зо + Зд) = (12\,518.75 + 1\,877.81) \times 0.22$
5. Накладні витрати	3 129.69	$Внак. = 0{,}4 \times Зо = 12\,518.75 \times 0.25$
6. Повна собівартість	21 320.49	$С_{пов} = В_м + З_о + З_д + Вє.с.в. + Внак. = 627.00 + 12\,518.75 + 1\,877.81 + 3\,167.24 + 3\,129.69$

Розмір прибутку, що включається в ціну, визначаємо по наступній формулі:

$$П = (С_{пов} * Р)/100 = (21320.49 * 10)/100 = 2132.05 \text{ грн.} \qquad (2.4)$$

Де р – плановий рівень рентабельності (10-15%).

Оптова ціна (кошторисна вартість) визначається по формулі:

$$Ц_о = С_{пов} + П = 21320.49 + 2132.05 = 23452.54 \text{грн.} \qquad (2.5)$$

Виходячи з отриманих даних, ціна реалізації розробленого програмного забезпечення становитиме:

$$Ц_р = Ц_о + ПДВ = 23452.54 + 23452.54 * 0.2 = 28143.05 \text{грн.} \qquad (2.6)$$

3 РОЗДІЛ ОХОРОНИ ПРАЦІ ТА ТЕХНІКИ БЕЗПЕКИ

Питання забезпечення безпечних умов праці завжди мають бути у центрі уваги будь-якої професійної діяльності, незалежно від сфери чи специфіки роботи.

Навіть попри те, що комп'ютерна робота не пов'язана з фізичними навантаженнями або складним технічним обладнанням, вона несе в собі низку ризиків для здоров'я людини. Тривале перебування у статичному положенні, інтенсивне зорове навантаження, вплив мікроклімату приміщення, а також психоемоційна напруга створюють умови, які можуть негативно позначатися на самопочутті працівника.

Розділ «Охорона праці» в дипломному проєкті має на меті проаналізувати чинники, які можуть бути шкідливими або небезпечними під час експлуатації комп'ютерної техніки, а також визначити ефективні заходи щодо мінімізації їх впливу. Особлива увага приділяється дотриманню вимог санітарно-гігієнічних норм, забезпеченню належного освітлення, температурного режиму, вентиляції та загальної ергономіки робочого місця.

Також у межах цього розділу розглядається система пожежної безпеки, адже в приміщеннях з великою кількістю електроніки підвищується ризик коротких замикань і загорянь. Створення сприятливого, безпечного середовища для користувача персонального комп'ютера є ключовою умовою збереження його працездатності, фізичного та психологічного здоров'я

3.1 Аналіз умов праці та виявлення небезпечних чинників

Під час виконання професійної діяльності за персональним комп'ютером користувач, незалежно від специфіки задач (чи то програмування, чи графічний дизайн, тестування або написання текстів), перебуває в умовах, що мають низку потенційно небезпечних і шкідливих факторів, які можуть негативно позначатися на його здоров'ї та загальному стані організму.

До таких умов належить передусім тривале перебування у статичній позі, при якій м'язи спини, шиї та кінцівок перебувають у постійному напруженні. Це

може спричинити порушення постави, розвиток хронічних захворювань опорно-рухового апарату, а також зниження працездатності у довгостроковій перспективі.

Одним із основних шкідливих чинників також є зорове навантаження. Монітори, особливо за неправильного налаштування яскравості, контрасту або частоти оновлення зображення, створюють значне навантаження на зорову систему користувача. Результатом може стати швидка втома очей, порушення гостроти зору, поява головного болю та загального дискомфорту.

До фізичних чинників, що мають вплив на працівника, можна віднести:

недостатнє або надмірне освітлення робочого місця, що впливає на рівень втомлюваності, підвищену температуру повітря, яка виникає внаслідок роботи комп'ютерного обладнання, особливо за відсутності належної вентиляції, знижений рівень вологості, що також характерний для закритих офісних або домашніх приміщень із численними електронними пристроями, електромагнітне випромінювання, яке виникає від моніторів, блоків живлення, роутерів та іншого обладнання. Хоча рівні цього випромінювання зазвичай не перевищують допустимих норм, постійний вплив, особливо при недотриманні мінімальної безпечної відстані, може мати негативні наслідки.

Крім того, значне значення має психоемоційне навантаження. Під час роботи з комп'ютером, особливо у творчих або технічно складних проектах (як, наприклад, створення гри), часто виникає необхідність приймати рішення у стислі строки, концентрувати увагу протягом тривалого часу, працювати з великими обсягами інформації. Це викликає нервове напруження, зниження працездатності, а в окремих випадках — професійне вигорання.

Також слід звернути увагу на організаційні аспекти: неправильне розміщення меблів і техніки, нестача перерв, відсутність ергономіки на робочому місці — усе це призводить до додаткових ризиків для здоров'я працівника.

3.2 Розробка заходів з охорони праці

Організація безпечних умов праці для користувачів персональних комп'ютерів потребує комплексного підходу, що враховує як фізичні, так і психофізіологічні особливості роботи. Виходячи з ідентифікованих раніше

факторів ризику, необхідно впровадити набір практичних і технічних рішень, спрямованих на зменшення їхнього негативного впливу.

Передусім слід забезпечити відповідність параметрів мікроклімату в приміщенні нормативним значенням. Температура повітря в робочій зоні має становити в межах 20–24 °C, відносна вологість — 40–60 %, а рух повітря не повинен перевищувати 0,1–0,2 м/с. Для підтримання таких умов рекомендується використання систем кондиціонування, зволожувачів та періодичне провітрювання приміщення.

Наступним важливим аспектом є ергономічна організація робочого місця. Монітор повинен бути встановлений на рівні очей або трохи нижче, а відстань до нього має становити не менше 60–70 см. Стілець повинен мати регульовану висоту, спинку з опорою для попереку та можливість змінювати положення сидіння. Руки користувача мають вільно лежати на столі під прямим кутом у ліктях, що дозволяє уникати м’язової напруги в плечовому поясі.

Для зменшення зорового навантаження рекомендовано використовувати монітори з якісною передачею кольору та яскравістю, яка автоматично підлаштовується до умов освітлення. Робоче місце повинне бути освітлене переважно природним світлом, доповненим локальними джерелами штучного освітлення, що не створюють відблисків на екрані.

Варто впровадити режим праці, що включає перерви кожні 45–60 хвилин активної роботи з комп’ютером. Під час таких перерв рекомендовано проводити короткі вправи для очей, кистей рук і хребта. Це дозволяє знизити втому, запобігти перенапруженню та покращити загальний стан працівника.

Також не можна залишати поза увагою психоемоційні аспекти. Раціональний розподіл робочого навантаження, відсутність надмірної монотонності в завданнях, створення комфортної атмосфери в колективі — усе це сприяє зменшенню рівня стресу та збереженню працездатності.

Важливо також дотримуватись електробезпеки. Усі пристрої повинні бути заземлені, використовуватися лише справні електричні розетки та мережеві

фільтри. Регулярна перевірка стану електротехнічного обладнання є обов’язковою складовою профілактики аварійних ситуацій.

3.3 Організація робочого місця користувача ПК

Щоб забезпечити комфортну та безпечну експлуатацію комп’ютерного обладнання, необхідно грамотно організувати робоче місце. Це передбачає дотримання норм ергономіки, санітарно-гігієнічних вимог та рекомендацій щодо техніки безпеки. Основна мета — зменшення навантаження на опорно-руховий апарат, зниження втоми зору та профілактика професійних захворювань.

Стіл повинен бути достатньо просторим і мати стійку конструкцію. Монітор слід розташовувати на рівні очей на відстані приблизно 60–70 см від користувача. Важливо, щоб верхня частина екрана не була вище рівня погляду. Клавіатуру та мишу необхідно розташовувати так, щоб руки знаходилися у природному положенні, без зайвого напруження зап’ясть.

Крісло повинно мати регулювання висоти, спинки, а також підтримку поперекового відділу хребта. Ноги повинні вільно стояти на підлозі або підставці, утворюючи кут близько 90 градусів у колінах.

Робоче місце також має бути забезпечене достатнім рівнем освітлення. Найкращим варіантом є поєднання природного світла з локальним освітленням, розташованим ліворуч або праворуч (залежно від провідної руки), щоб уникнути відблисків на екрані.

Важливо також стежити за мікрокліматом: температура в приміщенні повинна становити близько 20–24°C, відносна вологість — 40–60%. Вентиляція має забезпечувати доступ свіжого повітря, особливо в умовах тривалого перебування людей.

Таке робоче середовище допомагає знизити ризики пов’язані з професійним навантаженням, покращує самопочуття працівника та сприяє підвищенню продуктивності.

3.4 Пожежна безпека

Захист від пожеж є важливим елементом системи охорони праці, особливо у приміщеннях, де активно використовується електронне обладнання. Робочі місця з комп'ютерами потребують підвищеної уваги до попередження пожежонебезпечних ситуацій, оскільки саме електроприлади часто стають джерелами займання.

Основними ризиками в цьому контексті є коротке замикання, перевантаження електромережі, використання несправних або несертифікованих пристроїв. Тому важливо регулярно перевіряти стан електропроводки, справність розеток, подовжувачів та стабілізаторів. Усі пристрої повинні мати відповідні сертифікати відповідності та бути підключені до мережі з дотриманням технічних норм.

Комп'ютери, принтери, маршрутизатори та інше обладнання слід розміщувати на достатній відстані від джерел тепла та легкозаймистих матеріалів. Забороняється накривати техніку тканинами, зберігати поряд із нею папір, картон чи інші горючі речовини.

У приміщенні обов'язково повинні бути засоби первинного пожежогасіння — вогнегасники, пісковики, протипожежні покривала. Вогнегасники мають розташовуватись у легкодоступних місцях і бути придатними до експлуатації (перевірка терміну придатності проводиться щороку). Крім того, варто передбачити план евакуації та провести ознайомлення персоналу з порядком дій у разі виникнення загоряння.

Організація пожежної безпеки передбачає також дотримання вимог пожежної сигналізації та наявність аварійного освітлення. Практика регулярного інструктажу користувачів ПК щодо дій у разі пожежі дозволяє зменшити наслідки можливих інцидентів.

ВИСНОВКИ

В результаті виконання дипломної роботи було розроблено логічну гру на пошук прихованих предметів з використанням мови програмування Python та бібліотеки Pygame. Проведений аналіз існуючих рішень та технологій дозволив визначити оптимальні інструменти та підходи до розробки. Реалізована архітектура програмного забезпечення забезпечує модульність, масштабованість та легкість підтримки проекту.

Розроблена система ігрової механіки включає генерацію різних типів об'єктів (справжніх, фейкових та бонусних), реалізує прогресивну систему рівнів та забезпечує збереження результатів у JSON форматі. Впроваджено ефективну систему взаємодії з користувачем, що включає обробку кліків миші, систему підказок та відображення прогресу гри. Реалізовано механізми збереження та завантаження ігрових досягнень, що дозволяє гравцям відстежувати свій прогрес та змагатися за кращі результати.

Створений графічний інтерфейс забезпечує інтуїтивно зрозуміле управління та приємний візуальний досвід. Система анімацій реалізує плавні переходи між станами гри, візуальні ефекти при взаємодії з об'єктами та динамічне оновлення ігрового поля. Впроваджено адаптивний дизайн, що забезпечує коректне відображення гри на різних роздільних здатностях екрану.

Інтегрована система звукового супроводу включає фонову музику, звукові ефекти для різних ігрових подій та системні сповіщення. Реалізовано механізми управління гучністю, багатоканальне відтворення звуку та оптимізацію використання аудіо-ресурсів. Звукове оформлення суттєво покращує занурення користувача в ігровий процес та забезпечує додатковий зворотний зв'язок при взаємодії з грою.

Розроблена гра успішно виконує поставлені завдання, забезпечуючи користувачам цікавий ігровий процес, що сприяє розвитку уваги та спостережливості.

ПЕРЕЛІК ВИКОРИСТАНИХ ІНФОРМАЦІЙНИХ ДЖЕРЕЛ

1. Мельник Р. А., Коваль В. М. Аналіз сучасних технологій розробки логічних ігор на Python // Науковий вісник НЛТУ України. – 2023. – Т. 33, № 1. – С. 99–107.
2. Ващенко О. В., Тимошенко Ю. О. Розвиток логічного мислення засобами комп'ютерних ігор // Інформаційні технології в освіті. – 2023. – № 2(51). – С. 22–31.
3. Кучер Н. М. Ігрова діяльність – запорука успіху! : метод. посіб. для вчителів початкових класів. – 2023. – URL: https://www.medrxiv.org/...
4. Долготьор А. І. Інтелектуальна гра в жанрі головоломки. – 2023. – URL: https://ed.pano.pl.ua/...
5. Гузей О. І. Огляд змін та проблеми програмування на Python // Комп'ютерні та інформаційні системи і технології: матеріали конф. – 2019. – С. 138–142.
6. Мосіюк О.О., Федорчук А.Л. Операційні системи та системне програмування: навчально-методичний посібник. – Житомир: Вид-во ЖДУ ім. І. Франка, 2022. – 76 с.
7. Sweigart A. Making Games with Python & Pygame. – Creative Commons, 2023. – URL: https://inventwithpython.com/pygame/
8. Phillips D., Phillips J. Python Game Programming By Example. – Packt Publishing, 2021. – 376 p.
9. Wu C., Huang Y. Hidden Object Game Design Based on Visual Search Training // Journal of Educational Technology & Society. – 2021. – Vol. 24, No. 1. – P. 82–96.
10. Cormen T. H., Leiserson C. E., Rivest R. L., Stein C. Introduction to Algorithms. – 3rd ed. – Boston: MIT Press, 2009. – 1312 p.
11. Manning J., Batfield-Addison P. Unity для розробників. Мобільні мультиплатформені ігри. – К.: Діалектика, 2018. – 352 с.
12. Packt Publishing. Procedural Generation in Game Design / Tarn Adams. – 2017. – 336 p.

ДОДАТОК А. Лістинг коду гри

Модуль main

```
    def create_objects(level):
        objects = []
        num_objects = level * 2
        for _ in range(num_objects):
            x, y = random.randint(0, WIDTH - 50), random.randint(0, HEIGHT - 50)
            dx,   dy   =   random.choice([-1,   1])   *   random.uniform(0.05,   0.1),
random.choice([-1, 1]) * random.uniform(0.05, 0.1)
            img = random.choice(object_images)
            objects.append([img, x, y, dx, dy, 0, 0, 0, 0])
        return objects

    def create_fake_objects(level):
        fake_objects = []
        num_fake_objects = level * 3
        for _ in range(num_fake_objects):
            x, y = random.randint(0, WIDTH - 50), random.randint(0, HEIGHT - 50)
            dx,   dy   =   random.choice([-1,   1])   *   random.uniform(0.05,   0.1),
random.choice([-1, 1]) * random.uniform(0.05, 0.1)
            img = random.choice(object_images)

            light_img = img.copy()
            light_img.fill((100, 100, 100), special_flags=pygame.BLEND_RGB_ADD)

            fake_objects.append([light_img, x, y, dx, dy, 0, 0, 0, 0])
        return fake_objects

    def create_bonus_objects(level):
        bonus_objects = []
        num_bonus_objects = level
        for _ in range(num_bonus_objects):
            x, y = random.randint(0, WIDTH - 50), random.randint(0, HEIGHT - 50)
            dx,   dy   =   random.choice([-1,   1])   *   random.uniform(0.05,   0.1),
random.choice([-1, 1]) * random.uniform(0.05, 0.1)
            bonus_objects.append([pygame.Rect(x, y, 50, 50), dx, dy])
        return bonus_objects

    def save_score(player_name, score, level):
        result = {
            "player_name": player_name,
            "score": score,
            "level": level,
            "timestamp": time.strftime("%Y-%m-%d %H:%M:%S", time.localtime())
        }

        try:
            with open("scores.json", "r") as file:
                data = json.load(file)
        except FileNotFoundError:
            data = []

        data.append(result)

        with open("scores.json", "w") as file:
            json.dump(data, file, indent=4)

    def main():
        pygame.init()
```

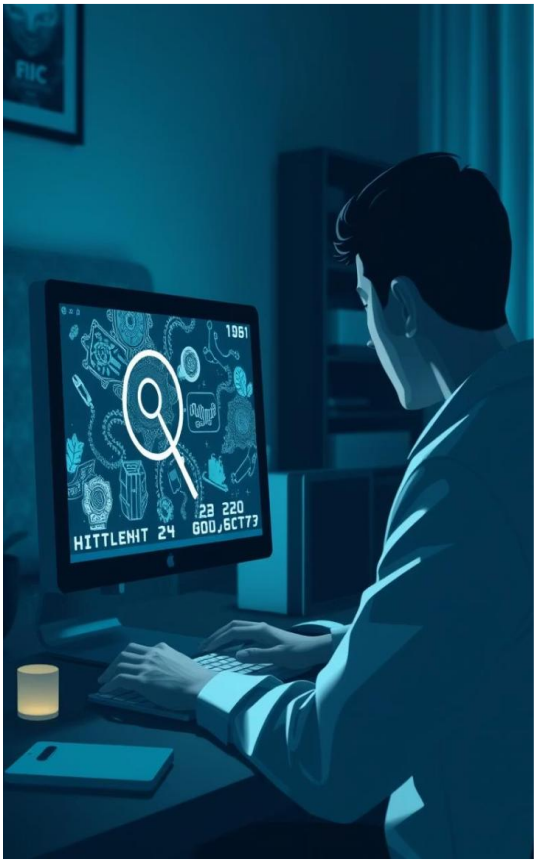
```
        player_name = input("Введіть своє ім'я: ")

        global WIDTH, HEIGHT, object_images
        WIDTH, HEIGHT = 800, 600
        screen = pygame.display.set_mode((WIDTH, HEIGHT))
        pygame.display.set_caption("Пошук предметів")

        BLACK = (0, 0, 0)
        WHITE = (255, 255, 255)
        RED = (255, 0, 0)
        GREEN = (0, 255, 0)

        Background=
pygame.image.load(resource_path(".venv/background/background.png"))

        object_images = [

pygame.transform.scale(pygame.image.load(resource_path(f".venv/images/image{i}.png")),
(50, 50)) for i in
            range(1, 4)
        ]

        levels = 1
        score = 0
        timer = 60
        font = pygame.font.Font(None, 36)

        click_sound = pygame.mixer.Sound(resource_path(".venv/music/klik35995.mp3"))
        bg_music = resource_path(".venv/music/cf0fc01247f4fc1.mp3")
        pygame.mixer.music.load(bg_music)
        pygame.mixer.music.play(-1)

        objects = create_objects(levels)
        fake_objects = create_fake_objects(levels)
        bonus_objects = create_bonus_objects(levels)
        start_time = time.time()
        attempts = 0
        running = True

        while running:
            screen.blit(background, (0, 0))

            for obj in objects:
                obj[1] += obj[3]
                obj[2] += obj[4]

                if obj[1] <= 0 or obj[1] >= WIDTH - 50:
                    obj[3] = -obj[3]
                if obj[2] <= 0 or obj[2] >= HEIGHT - 50:
                    obj[4] = -obj[4]

                scale_factor = 1 + 0.1 * math.sin(obj[6])
                obj[6] += 0.1
                scaled_img = pygame.transform.scale(obj[0],
    (int(50 * scale_factor), int(50 * scale_factor)))

                obj[7] += obj[8]
                rotated_img = pygame.transform.rotate(scaled_img, obj[7])
                rotated_rect = rotated_img.get_rect(center=(obj[1] + 25, obj[2] + 25))
                screen.blit(rotated_img, rotated_rect.topleft)

            for fake in fake_objects:
```

```
            fake[1] += fake[3]
            fake[2] += fake[4]

            if fake[1] <= 0 or fake[1] >= WIDTH - 50:
                fake[3] = -fake[3]
            if fake[2] <= 0 or fake[2] >= HEIGHT - 50:
                fake[4] = -fake[4]

            fake[6] += 1
            scale_factor = 1 + 0.1 * math.sin(fake[6] * 0.1)
            scaled_fake_object = pygame.transform.scale(fake[0],
(int(50 * scale_factor), int(50 * scale_factor)))
            rotated_img = pygame.transform.rotate(scaled_fake_object, fake[7])
            rotated_rect = rotated_img.get_rect(center=(fake[1] + 25, fake[2] + 25))
            screen.blit(rotated_img, rotated_rect.topleft)
        for bonus in bonus_objects:
            bonus[0].x += bonus[1]
            bonus[0].y += bonus[2]

            if bonus[0].x <= 0 or bonus[0].x >= WIDTH - 50:
                bonus[1] = -bonus[1]
            if bonus[0].y <= 0 or bonus[0].y >= HEIGHT - 50:
                bonus[2] = -bonus[2]

            pygame.draw.rect(screen, GREEN, bonus[0])  # Малюємо бонусний об'єкт

        elapsed_time = time.time() - start_time
        remaining_time = max(timer - int(elapsed_time), 0)
        score_text = font.render(
            f"Рахунок: {score}  Рівень: {levels}
Час: {remaining_time}  Спроб залишилось: {3 - attempts}", True, WHITE)
        screen.blit(score_text, (10, 10))

        if remaining_time == 0:
            save_score(player_name, score, levels)
            break

        if attempts >= 3:
            game_over_text = font.render("Гра завершена! Забагато помилок.",
True, WHITE)

            screen.blit(game_over_text, (WIDTH // 2 - 200, HEIGHT // 2))
            pygame.display.flip()
            time.sleep(2)
            save_score(player_name, score, levels)
            break

        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                save_score(player_name, score, levels)
                running = False
            if event.type == pygame.MOUSEBUTTONDOWN:
                mouse_pos = event.pos
                hit = False

                for obj in objects:
                    if pygame.Rect((obj[1], obj[2]),
(50, 50)).collidepoint(mouse_pos):
                        objects.remove(obj)
                        score += 10
                        click_sound.play()
                        hit = True
```

```
                        break

                for fake in fake_objects:
                    if pygame.Rect((fake[1], fake[2]),
(50, 50)).collidepoint(mouse_pos):
                        attempts += 1
                        hit = True
                        break

                for bonus in bonus_objects:
                    if bonus[0].collidepoint(mouse_pos):
                        score += 20
                        bonus_objects.remove(bonus)
                        hit = True
                        break

                if not objects and hit:
                    levels += 1
                    objects = create_objects(levels)
                    fake_objects = create_fake_objects(levels)
                    bonus_objects = create_bonus_objects(levels)
                    start_time = time.time()

        pygame.display.flip()

    pygame.mixer.music.stop()
    pygame.quit()

if __name__ == "__main__":
    main()
```

ДОДАТОК Б Слайди мультимедійної презентації

ДИПЛОМНИЙ ПРОЕКТ НА ТЕМУ: Розробка програмного застосунку логічної гри на пошук прихованих предметів



Виконав студент гр. 4РП-08. Громович Д.О.
Керівник ДП: Кунуп Т.В.

Мета та завдання проекту

Моя дипломна робота зосереджена на створенні та аналізі інтерактивної гри "Пошук прихованих предметів", розробленої з використанням Python та бібліотеки Pygame. Ця робота не лише демонструє практичні навички ігрової розробки, але й містить глибоке дослідження ефективності ігрових механік та оптимізації продуктивності.

Мета та Завдання

Головною метою було створення функціонального ігрового додатку, що розвиває когнітивні навички. Для цього були поставлені завдання: дослідження ефективності різних ігрових механік та оптимізація продуктивності для забезпечення стабільної роботи на різних платформах.

- Аналіз 15 аналогічних ігор, таких як "Hidden Folks", для виявлення оптимальних підходів до генерації предметів.
- Проектування модульної архітектури з окремими класами для об'єктів, рівнів та інтерфейсу, а також реалізація паттерну State для управління ігровими екранами.
- Реалізація ключових механік: процедурна генерація рівнів, адаптивний алгоритм балансування складності та механізм збереження результатів у форматі JSON.

Наукова Новизна та Оптимізація

Наукова новизна роботи полягає у розробці методики адаптивного балансування складності, оптимізованих алгоритмів колізій для Pygame, та системи аналізу поведінки гравця. Ці інновації дозволили значно покращити ігровий досвід.

- Забезпечення стабільної продуктивності у 60 FPS навіть на слабких комп'ютерах.
- Зменшення часу завантаження рівнів з 3 до 1.5 секунд завдяки застосуванню оптимізаційних стратегій.
- Розробка адаптивного балансування складності, що динамічно підлаштовується під рівень майстерності гравця.

Порівняльний аналіз інструментів для розробки логічних ігор

У сучасному розробці ігор існує багато інструментів та фреймворків. Вибір оптимального технологічного стеку є критично важливим для успішної реалізації дипломного проекту. У даному розділі проведено порівняльний аналіз доступних рішень з обґрунтуванням вибору Python та Pygame для розробки логічної гри.

Критерії порівняння

- Крива навчання
- Продуктивність
- Крос-платформеність
- Доступність бібліотек
- Підтримка спільноти
- Ліцензійні умови

Python + Pygame

Переваги:

- Мінімальний час навчання (простий синтаксис, схожий на псевдокод)
- Багата стандартна бібліотека для роботи з даними
- Швидкий прототип (можливість створення MVP за 1-2 дні)
- Активна спільнота (понад 15k проектів на GitHub)
- Вільна ліцензія (MIT)

Обмеження:

- Не підходить для ресурсомістких 3D-проектів
- Дефіцит професійних підручників

Unity (C#)

Сильні сторони:

- Потужний редактор сцен
- Вбудована підтримка фізики
- Можливість публікації на 25+ платформах

Недоліки:

- Вимагає знання C#
- Великий розмір збірки
- Складність у налагодженні

Godot Engine

Унікальні можливості:

- Легковісний рушій (розмір <100MB)
- Вбудована підтримка мов GDScript, C#, C++
- Інтуїтивний інтерфейс редактора

Проблемні аспекти:

- Обмежена кількість готових рішень
- Менша кількість фахівців на ринку

Техніко-економічне обґрунтування вибору

Для дипломного проекту "Розробка логічної гри" були враховані такі фактори:

Час реалізації:

- Python дозволяє реалізувати базову логіку за 40-50 годин
- Інші технології потребують 70-100 годин на аналогічний функціонал

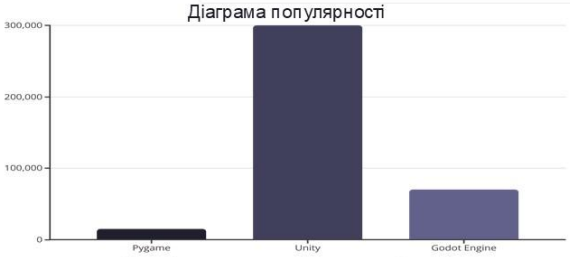
Економічні фактори:

- Вартість розробки на Python на 60-70% нижча
- Відсутність ліцензійних відрахувань

Апаратні вимоги:

Технологія	Мінімальні вимоги	Рекомендовані
Pygame	CPU 1GHz, 512MB RAM	CPU 2GHz, 1GB RAM
Unity	CPU 2GHz, 4GB RAM	CPU 3GHz, 8GB RAM

Діаграма популярності



Діаграма відображає кількість проектів на GitHub для кожної з платформ, демонструючи їхню відносну популярність та активність спільноти..

Актуальність теми
Розробка логічних ігор на Python: Переваги та Можливості

Логічні ігри розвивають когнітивні навички. Python — ідеальний інструмент для їх створення.

Python для ігор

- Простота коду: Швидкий старт для новачків.
- Багато бібліотек: Pygame, Panda3D, Arcade для графіки, фізики, AI.
- Крос-платформеність: Запускається на Windows, macOS, Linux.
- Гнучкість: Для прототипів та повноцінних ігор.

Можливості Pygame

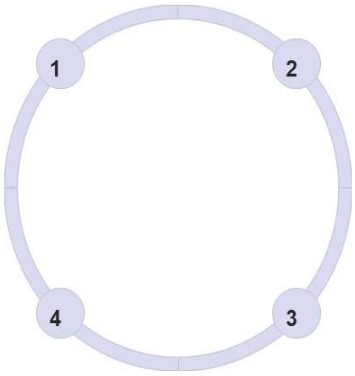
- Ігрові механіки: 2D-графіка, анімації, колізії.
- Мультимедіа: Робота зі звуком, мишею/клавіатурою.
- Процедурна генерація: Рівні та предмети.
- Оптимізація: Для навчання та комерційних проєктів.

Python є однією з найпопулярніших мов, що ідеально підходить для розробників-початківців та студентів. Pygame дозволяє створити гру з мінімальними ресурсами. Існує високий попит на навчальні матеріали та ігри з відкритим кодом.

Проєктування гри

Архітектура програми

Основні модулі:



main.py

головний цикл гри

objects.py

створення об'єктів (справжні, фейкові, бонусні)

scores.py

збереження результатів (JSON)

levels.py

система рівнів

UML-діаграма (основні класи та їх взаємодія).

Теоретичні основи розробки гри

Проект "Пошук предметів" базується на взаємодії гравця з різними типами об'єктів:

- Справжні предмети – приносять очки.
- Фейкові предмети – зменшують кількість спроб.
- Бонусні предмети – надають додаткові переваги (додаткові очки, час тощо).

Ключові механіки:

Генерація об'єктів:

- **create_objects()** – створює справжні предмети з випадковим розміщенням.
- **create_fake_objects()** – додає пастки для підвищення складності.
- **create_bonus_objects()** – забезпечує додаткові можливості.

Взаємодія:

- Обробка кліків миші з перевіркою координат.
- Система колізій запобігає накладанню об'єктів.
- Візуальний та звуковий супровід – підкреслює результати дій гравця.

Система рівнів

Прогресивна складність:

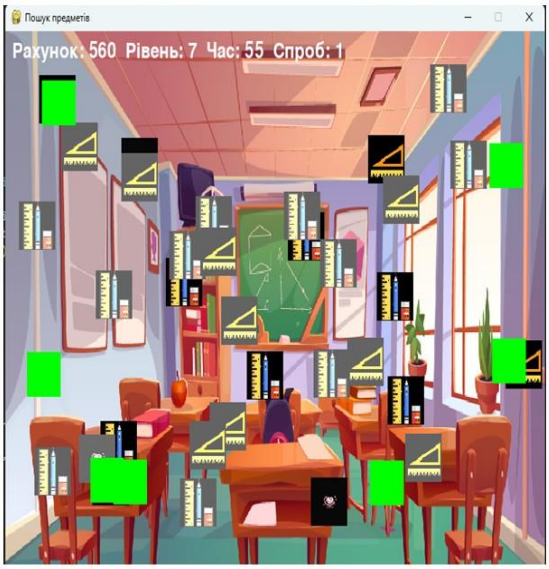
- Перший рівень – навчальний (обмежена кількість предметів).
- Наступні рівні – збільшення кількості предметів, обмеження часу.
- Збереження прогресу у JSON-форматі.

Адаптивний баланс:

- Динамічне співвідношення типів об'єктів.

Додаткові механіки

- Таймери – обмеження часу на рівень.
- Система очок – мотивація через комбо, бонуси.



Ігрова механіка

1

Система рівнів:

- Зростаюча складність (більше предметів, менше часу).
- Генерація випадкових позицій об'єктів.

2

Типи об'єктів:

- Справжні (+очки).
- Фейкові (–спроби).
- Бонуси (додатковий час, підказки).

3

Збереження результатів:

Використання JSON для зберігання імен гравців, очок, рівнів.

Основні Функції Коду Гри

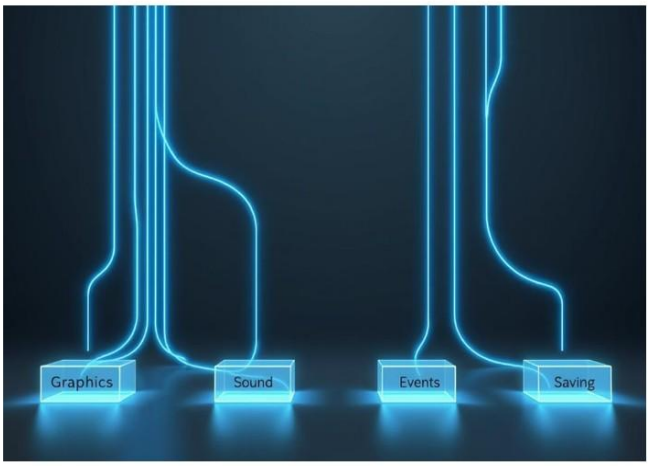
Цей розділ розкриває ключові модулі, основні функції та методи оптимізації, які використовуються при розробці ігори на Python за допомогою бібліотеки Pygame. Ми розглянемо, як кожен компонент сприяє створенню інтерактивного та динамічного ігрового процесу.

1. Ключові Модулі та Їх Призначення

pygame	Графіка, звук, обробка подій	pygame.init()
random	Генерація випадкових позицій об'єктів	random.randint()
json	Збереження результатів гри у файл	json.dump()
time	Вимірювання часу гри та анімації	time.time()

2. Ядрові Функції Гри

- **2.1. Створення ігрових об'єктів:** Функція `create_objects(level)` динамічно генерує об'єкти, кількість яких залежить від поточного рівня гри (`level * 2`). Кожен об'єкт отримує випадкову позицію та швидкість, що забезпечує непередбачуваний рух.
- **2.2. Обробка зіткнень (колізій):** Система зіткнень перевіряє, чи координати миші гравця (`pos`) перетинаються з прямокутними областями об'єктів (`pygame.Rect().collidepoint()`). Успішне зіткнення призводить до видалення об'єкта та збільшення рахунку, тоді як клік по "фейковим" об'єктам додає штрафні очки.
- **2.3. Збереження результатів (JSON):** Функція `save_score(player_name, score, level)` дозволяє зберігати дані про гравця (ім'я, рахунок, рівень) у файл формату JSON (`scores.json`), що забезпечує зручне та структуроване зберігання ігрової статистики.



3. Оптимізаційні Рішення

- **Анімація об'єктів:**
  - Код використовує математичні функції, наприклад, `math.sin(obj[6])`, для створення ефектів пульсації або зміни розміру, що додає динамічності об'єктам.
  - Обертання об'єктів реалізовано за допомогою `pygame.transform.rotate(img, angle)`, що дозволяє анімувати їхнє обертання.
- **Оптимізація ресурсів:**
  - Попереднє завантаження зображень через функцію `load_and_scale_image` запобігає затримкам під час гри, оскільки ресурси вже готові до використання.
  - Фонове відтворення музики за допомогою `pygame.mixer.music.play(-1)` забезпечує безперервний музичний супровід, покращуючи атмосферу гри.

Реалізація аудіовізуальних компонентів у грі "Пошук прихованих предметів"

Аудіовізуальна система гри "Пошук прихованих предметів" розроблена на базі бібліотеки Pygame, що забезпечує високоякісну графіку, анімації та звуковий супровід. Вона включає продуманий графічний інтерфейс, динамічні анімації та інтегрований звуковий дизайн для занурення гравців.

Графічний інтерфейс та анімації

- **Ігрове поле:** Система координат для точного позиціонування об'єктів, подвійна буферизація для запобігання мерехтінню та візуальні ефекти підсвічування активних зон.
- **Анімації:** Плавні переходи між рівнями (ефекти затемнення), система частинок для ефектів знаходження предметів та інтерполяція руху об'єктів.

Звуковий супровід

Реалізовано за допомогою pygame.mixer з підтримкою фонової музики (.mp3), звуків кліків та ефектів успіху (.wav).

Фонова музика	Спокійна мелодія	.mp3
Звуки кліків	Клацання при взаємодії	.wav
Ефекти успіху	Радісний звук при знаходженні предмета	.wav

- **Особливості:** Незалежне регулювання гучності для музики та ефектів, система пріоритетів для одночасних звукових подій та плавні переходи між музичними треками.

Технічні рішення та оптимізація

- **Оптимізація продуктивності:** Кешування спрайтів та звуків, алгоритм відсікання невидимих об'єктів та батчинг для зменшення викликів рендерингу.
- **Синхронізація:** Точний таймінг звукових ефектів з ігровими подіями та прив'язка анімацій до ігрового циклу (60 FPS).

Тестування та Оптимізація Продуктивності Гри

Це надасть огляд нашої стратегії тестування та оптимізації продуктивності, яка є критично важливою для забезпечення високої якості та стабільності наших ігрових продуктів.

Система Тестування

- **Модульне тестування:** Перевірка окремих функцій, таких як `create_objects()` або `save_score()`. Використовуємо фреймворк **unittest** та виконуємо тести щодня після внесення змін у код.
- **Інтеграційне тестування:** Перевірка взаємодії між ключовими системами гри (графіка, звук, ігрова логіка). Використовуємо **pytest** для щотижневого комплексного тестування.
- **Стрес-тестування:** Перевірка стабільності гри при максимальних навантаженнях, наприклад, симуляція 100+ одночасних кліків. Окрему увагу приділяємо контролю витоків пам'яті.

Метрики Продуктивності та Інструменти

Параметр	Оптимальне значення	Поточний стан
Частота кадрів (FPS)	60+	58
Час завантаження рівня	< 2 сек	2.3 сек
Використання CPU	< 70%	65%
Використання RAM	< 500 MB	480 MB

Використовувані технології: Для тестування використовуємо **pytest**, **unittest**, **LoadRunner**. Для профілювання — **cProfile**, **memory_profiler**. Моніторинг здійснюється за допомогою вбудованих інструментів **PyGame**, а візуалізація даних – **Matplotlib**.

Процес Оптимізації

Виявлення проблем

Аналізуємо звіти профайлера, моніторимо FPS під час гри та перевіряємо час виконання ключових функцій для ідентифікації "вузьких місць".

Впровадження поліпшень

Здійснюємо рефакторинг неефективного коду, оптимізуємо алгоритми та балансуємо навантаження між компонентами для покращення продуктивності.

Перевірка результатів

Проводимо порівняльні тести до та після оптимізації, контролюємо стабільність системи та моніторимо споживання ресурсів, щоб переконатися в ефективності змін.

Цей циклічний процес забезпечує постійне вдосконалення наших ігор, роблячи їх швидшими, стабільнішими та приємнішими для гравців.

Порівнянням 1-го та 9-го рівнів

Рівень	Об'єктів	Фейків	Бонусів	Час (с)
1	2	3	1	60
2	4	6	2	60
3	6	9	3	60

Висновки та перспективи

Практичне значення гри

- Освітній аспект: тренування уваги, спостережливості.
- Розважальний потенціал: простий, але захоплюючий геймплей.

. Перспективи розвитку

Додавання нових рівнів із складнішими завданнями

Режим мультиплеєра (змагання між гравцями)

Покращення графіки (3D-елементи за допомогою PyOpenGL)

Висновок

Python та Pygame – ефективні інструменти для швидкої розробки 2D-ігор.

Гра успішно виконує поставлені завдання: розвиває когнітивні навички та забезпечує цікавий ігровий досвід.

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

РЕЦЕНЗІЯ

на дипломний проект (роботу) здобувача (здобувачки) освіти

відділення комп'ютерних систем

Громовича Дениса Олександрович

(прізвище, ім`я та по батькові)

Спеціальність 121 Інженерія програмного забезпечення

Освітньо-професійна програма Розробка програмного забезпечення

Керівник дипломного проекту (роботи) *Кунуп Тетяна Василівна*

(прізвище, ім`я та по батькові)

Тема дипломного проекту (роботи) *Розробка програмного застосунку логічної гри на пошук прихованих предметів*

Обсяг розрахунково-пояснювальної записки *60* сторінок

Обсяг графічної (презентаційної) частини *12* аркушів (слайдів)

ХАРАКТЕРИСТИКА ДИПЛОМНОГО ПРОЕКТУ (РОБОТИ)

а) заключення про ступінь відповідності виконаного дипломного проекту (роботи) завданню

Дипломний проєкт повністю відповідає поставленому завданню. У роботі реалізовано всі основні функції: створена логічна гра, розроблена ігрова логіка та механіка, інтеграція графіки, анімації та звуку, тестування та оптимізація продуктивності. Проєкт виконано згідно з вимогами спеціальності, але з порушенням термінів

б) характеристика виконання кожного розділу дипломного проекту (роботи)

При виконанні дипломного проєкту здобувач продемонстрував уміння застосовувати сучасні підходи до розробки, зокрема об'єктно-орієнтоване програмування, модульну архітектуру. Була використано бібліотеку Pygame, мова програмування Python, а також розробка ігрової логіки та механік, інтеграція графіки, анімації та звуку.

в) оцінка якості виконання пояснювальної записки та графічної частини дипломного проекту (роботи) *Пояснювальна записка виконана з дотриманням структури та вимог до оформлення. Графічна частина чітко відображає реалізовану функціональність гри та підтримує розкриття змісту проєкту.*

г) перелік позитивних якостей дипломного проекту (роботи) *Тема дипломного проєкту є актуальною, виконана якісно та відповідно до поставленого завдання. Розроблено гру на мові програмування Python, модульне програмування, інтеграція графіки, що забезпечує гравцеві інтерес і прогрес у грі.*

д) основні недоліки дипломного проекту (роботи) *Обмежена ігрова механіка. Можливості для впровадження складних логічних елементів чи інновацій обмежені. Графічна частина проекту слабка, схеми алгоритмів виконані без дотримання вимог, етапи проектування мало проілюстровані відповідними схемами.*

Оцінка розрахункової частини *Добре*

Оцінка графічної частини *Добре*

Загальна оцінка *Добре*

Прізвище, ім'я, по батькові рецензента *к.т.н. Рудніченко Микола Дмитрович*

Місце роботи і посада рецензента *Національний університет «Одеська політехніка», доцент кафедри інформаційних технологій*

Підпис: _____

«25» 06 2025 р.

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

ВІДГУК

керівника на дипломний проєкт здобувача (здобувачки) освіти

відділення комп'ютерних систем

Громовиса Дениса Олександровича

(прізвище, ім`я та по батькові)

Спеціальність: *121 "Інженерія програмного забезпечення"*

Освітньо-професійна програма: *«Розробка програмного забезпечення»*

Тема дипломного проєкту: *Розробка програмного застосунку логічної гри на пошук прихованих предметів*

ХАРАКТЕРИСТИКА ДИПЛОМНОГО ПРОЄКТУ

а) обсяг і якість виконання проєкту (графічного матеріалу і розрахунково-пояснювальної записки) *Дипломний проєкт виконано відповідно технічному завданню. Пояснювальна записка містить 60 сторінки. У пояснювальній записці виконано опис етапів, відповідно до індивідуального завдання, розділи пояснювальної записки відповідають етапам рішення завдання, поставленого у дипломному проекті. Графічна частина складається з 12 слайдів мультимедійної презентації. Якість виконання пояснювальної записки та графічної частини добра, розробку виконано в повному обсязі.*

б) самостійність роботи над проєктом: *Протягом всього строку дипломного проєктування та переддипломної практики здобувач освіти Громович Денис Олександрович, поступово та послідовно виконував всі етапи розробки. Всі роботи здобувач освіти виконув самостійно, з оглядом на рекомендації керівника*

в) теоретична підготовка випускника (випускниці): *Здобувач освіти Громович Денис Олександрович під час роботи над дипломним проєктом вивчив достатню кількість літературних джерел та матеріалів за даною тематикою.*

Вважаю, що теоретична підготовка дипломника добра і він готовий до захисту дипломного проєкту

г) вміння розв'язувати виробничі та конструкторські питання _____

Під час дипломного проєктування здобувач освіти Громович Денис Олександрович. розробив програмний продукт, а саме логічної гри на на пошук предметів. Дипломник володіє навичками написання програмного продукту за допомогою таких технологій як Python.

Оцінка розрахункової частини *Добре*

Оцінка графічної частини *Добре*

Загальна оцінка *Добре*

Прізвище, ім'я, по батькові керівника дипломного проєкту _____

Кунуп Тетяна Василівна

Місце роботи і посада керівника дипломного проєкту _____

"НУ"Одеська політехніка, к.т.н., ст.викладач кафедри «Інормаційні технології»

Підпис _____

«16» 06 2025 р.

ДОЗВІЛ
НА РОЗМІЩЕННЯ
ВИПУСКНОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ
(ДИПЛОМНОГО ПРОЕКТУ)
В ЕЛЕКТРОННОМУ РЕПОЗИТАРІЇ ВСП «ОТФК ОНТУ»

Ми, що нижче підписалися,

Громович Денис Олександрович
здобувач освіти гр. 4РП-08, та

Кунуп Тетяна Василівна,
керівник дипломного проекту,

не заперечуємо щодо розміщення електронного варіанту пояснювальної записки до дипломного проекту фахового молодшого бакалавра на тему:

«Розробка програмного застосунку логічної гри на пошук прихованих предметів» (автор роботи – Громович Д.О., керівник роботи – Кунуп Т.В.)

виконаного у ВСП «Одеський технічний фаховий коледж Одеського національного технологічного університету» в 2025 році, у повному обсязі в електронному репозитарії ВСП «ОТФК ОНТУ» для вільного доступу через мережу Інтернет.

Несемо відповідальність за ідентичність електронного та друкованого варіантів випускної кваліфікаційної роботи і даємо згоду на обробку персональних даних.

Виконавець _____ / Громович Д.О. /

Керівник _____ / Кунуп Т.В. /

«16» червня 2025 р.

ДОВІДКА

циклової комісії КТ та ПІ
про допуск до захисту дипломного проєкту
здобувача (здобувачки) освіти IV курсу
відділення комп'ютерних систем групи 4РП-08

Громовича Дениса Олександровича

на тему *Розробка програмного застосунку логічної гри на пошук прихованих предметів*

Висновок відповідальної особи за проведення нормоконтролю:
пояснювальна записка до дипломного проєкту виконана з деякими порушеннями ДСТУ та оформлена відповідно до вимог Положення про дипломне проєктування

(підпис) *23.06.2025* (дата) *Петрашова В.І.* (П.І.Б.)

Висновок відповідальної особи за перевірку роботи на наявність академічного плагіату *згідно звіту про перевірку від 23.06.2025 р. значення коефіцієнту подібності в роботі становить 14,03%, коефіцієнт цитування – 1,14%.*

(підпис) *23.06.2025* (дата) *Краснокутська К.Г.* (П.І.Б.)

Попередня експертиза (малий захист) дипломного проєкту

здобувача (здобувачки) освіти *Громовича Д.О.* (П.І.Б.)

проведена «*23*» *червня* 2025 р.

Висновки *Пояснювальна записка до дипломного проєкту виконана у повному обсязі. Випускна кваліфікаційна робота (дипломний проєкт) відповідає вимогам Положення про дипломне проєктування та рекомендована до захисту.*

Голова ЦК КТ та ПІ (підпис) *Кривченко Ю.В.* (П.І.Б.)

StrikePlagiarism.com

Дата звіту 6/23/2025
Дата редагування 6/23/2025

Документ прийнятий

Звіт подібності

метадані

Назва організації
Odesa Technical Professional College of Odesa National University of Technology
Заголовок
Розробка програмного застосунку логічної гри на пошук прихованих предметів

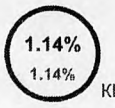
Автор Науковий керівник / Експерт
Громович Денис ОлександровичКунуп Тетяна Василівна

підрозділ
Відокремлений структурний підрозділ "Одеський технічний фаховий коледж Одеського національного технологічного університету"

Обсяг знайдених подібностей

Коефіцієнт подібності визначає, який відсоток тексту по відношенню до загального обсягу тексту було знайдено в різних джерелах. Зверніть увагу, що високі значення коефіцієнта не автоматично означають плагіат. Звіт має аналізувати компетентна / уповноважена особа.

КП 1



КЦ

25
Довжина фрази для коефіцієнта подібності 2

14331
Кількість слів

119964
Кількість символів

Тривога

У цьому розділі ви знайдете інформацію щодо текстових спотворень. Ці спотворення в тексті можуть говорити про МОЖЛИВІ маніпуляції в тексті. Спотворення в тексті можуть мати навмисний характер, але частіше характер технічних помилок при конвертації документа та його збереженні, тому ми рекомендуємо вам підходити до аналізу цього модуля відповідально. У разі виникнення запитань, просимо звертатися до нашої служби підтримки.

Заміна букв		18
Інтервали	A→	0
Мікропробіли	·	0
Білі знаки		0
Парафрази (SmartMarks)	a	84

Подібності за списком джерел

Нижче наведений список джерел. В цьому списку є джерела із різних баз даних. Колір тексту означає в якому джерелі він був знайдений. Ці джерела і значення Коефіцієнту Подібності не відображають прямого плагіату. Необхідно відкрити кожне джерело і проаналізувати зміст і правильність оформлення джерела.

10 найдовших фраз — Колір тексту

ПОРЯДКОВИЙ НОМЕР	НАЗВА ТА АДРЕСА ДЖЕРЕЛА URL (НАЗВА БАЗИ)	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	https://card-file.ontu.edu.ua/bitstreams/035f6436-20b4-4ee6-8e99-bede670e308b/download	126 0.88 %
2	https://card-file.ontu.edu.ua/bitstreams/53ed22ad-8700-4162-b97a-082a1ad472d6/download	81 0.57 %
3	https://card-file.ontu.edu.ua/bitstreams/34a6756b-592f-4b77-a805-183aa03a6a26/download	71 0.50 %
4	https://card-file.ontu.edu.ua/bitstreams/035f6436-20b4-4ee6-8e99-bede670e308b/download	63 0.44 %
5	https://card-file.ontu.edu.ua/bitstreams/035f6436-20b4-4ee6-8e99-bede670e308b/download	44 0.31 %

6	https://card-file.ontu.edu.ua/bitstreams/34a6756b-592f-4b77-a805-183aa03a6a26/download	43 0.30 %
7	https://card-file.ontu.edu.ua/bitstreams/6cf43324-8f08-4031-ba42-f80b18efbbc8/download	40 0.28 %
8	https://card-file.ontu.edu.ua/bitstreams/29489599-0581-4ce6-8890-c3b13d9f2e0e/download	40 0.28 %
9	https://card-file.ontu.edu.ua/bitstreams/bbed74c8-2ea7-44c5-8d00-0fe3fd9790ee/download	39 0.27 %
10	https://card-file.ontu.edu.ua/server/api/core/bitstreams/a141b658-5fa7-4f90-b0bd-7f0ccaed21e5/content	35 0.24 %

з домашньої бази даних (0.00 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)

з програми обміну базами даних (0.40 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	Pygame – графиктік аНиМация ортасы [redacted] 4/28/2025 Yessenov University (Yessenov University)	27 (4) 0.19 %
2	ПРОЕКТУВАННЯ ТА АНАЛІЗ ІГРОВОЇ МЕХАНІКИ СОРТУВАННЯ ОБ'ЄКТІВ У КОНТЕКСТІ РОЗРОБКИ МОБІЛЬНОЇ ГРИ [redacted] 6/23/2024 Rivne Professional college of National University of Life and Environmental sciences of Ukraine (ВСП ,,Рівненський фаховий коледж НУБіП України")	23 (1) 0.16 %
3	Кішик Віталій Дипломна Робота н.кер. Милашенко В.М. [redacted] 4/30/2025 European University (European University)	7 (1) 0.05 %

з Інтернету (13.63 %)

ПОРЯДКОВИЙ НОМЕР	ДЖЕРЕЛО URL	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	https://card-file.ontu.edu.ua/server/api/core/bitstreams/44c16132-5f53-48e2-b6c0-61e9a2f0fd75/content	523 (44) 3.65 %
2	**https://card-file.ontu.edu.ua/bitstreams/035f6436-20b4-4ee6-8e99-bede670e308b/download**	**327 (11) 2.28 %**
3	**https://card-file.ontu.edu.ua/bitstreams/34a6756b-592f-4b77-a805-183aa03a6a26/download**	**200 (7) 1.40 %**
4	**https://card-file.ontu.edu.ua/bitstreams/53ed22ad-8700-4162-b97a-082a1ad472d6/download**	**161 (8) 1.12 %**
5	**https://card-file.ontu.edu.ua/server/api/core/bitstreams/995bdcec-4e4d-4321-8070-4d6badcb8e49/content**	**96 (5) 0.67 %**
6	**https://card-file.ontu.edu.ua/bitstreams/bbed74c8-2ea7-44c5-8d00-0fe3fd9790ee/download**	**95 (5) 0.66 %**
7	**https://card-file.ontu.edu.ua/bitstreams/29489599-0581-4ce6-8890-c3b13d9f2e0e/download**	**94 (5) 0.66 %**
8	https://card-file.ontu.edu.ua/bitstreams/1dff552d-7200-49b8-ae1d-ba76a1335685/download	87 (8) 0.61 %
9	**https://card-file.ontu.edu.ua/server/api/core/bitstreams/a141b658-5fa7-4f90-b0bd-7f0ccaed21e5/content**	**66 (4) 0.46 %**
10	**https://card-file.ontu.edu.ua/bitstreams/549ee9fe-7574-4ae5-b500-9fe2711f33e6/download**	**59 (4) 0.41 %**
11	https://card-file.ontu.edu.ua/bitstreams/62baa43e-b968-4993-bb54-8cf8761a89b2/download	52 (4) 0.36 %
12	**https://card-file.ontu.edu.ua/bitstreams/6cf43324-8f08-4031-ba42-f80b18efbbc8/download**	**40 (1) 0.28 %**
13	https://core.ac.uk/download/604676637.pdf	33 (3) 0.23 %

14	**https://card-file.ontu.edu.ua/bitstreams/82a6d375-2b69-4233-b80f-fbfd149b7747/download**	**31 (1) 0.22 %**
15	https://github.com/fixploit03/ied/blob/main/ied.py	29 (2) 0.20 %
16	**https://card-file.ontu.edu.ua/bitstreams/e4afae26-0a7e-4a4d-afc2-94341838de2a/download**	**25 (1) 0.17 %**
17	https://cyberleninka.ru/article/n/postroenie-polyarnogo-konusa-k-vypuklomu-mnogogrannomu-konusu-v-r-3	20 (1) 0.14 %
18	https://elibrary.kdpu.edu.ua/bitstream/123456789/10247/1/KobzievBohdan2024.pdf	15 (2) 0.10 %

Список прийнятих фрагментів (немає прийнятих фрагментів)

ПОРЯДКОВИЙ НОМЕР	ЗМІСТ	КІЛЬКІСТЬ ОДНАКОВИХ СЛІВ (ФРАГМЕНТІВ)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Спеціальність: 121 «Інженерія програмного забезпечення»
Освітньо-професійна програма: **«Розробка програмного забезпечення» Група:** 4РП- 08

Дипломний **проєкт**
здобувача освіти денної форми навчання РП. 08.07.000. ДП

ГРОМОВИЧА
ДЕНИСА ОЛЕКСАНДРОВИЧА

м. Одеса
2025 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Спеціальність: 121 «Інженерія програмного забезпечення»
Освітньо-професійна програма: «Розробка програмного забезпечення»
Група: 4РП-08

ПОЯСНЮВАЛЬНА ЗАПИСКА
до дипломного проєкту на тему:
_____
_____
Проєктний матеріал складається з пояснювальної записки на _____ сторінках та графічного (презентаційного) матеріалу на _____ аркушах (слайдах) Дипломник _____ (Громович Д. О.)
Керівник _____ (Кунуп Т. В.) **Консультанти:** з економічного розділу_____ (Канський М.Ю.)
з розділу охорони праці та техніки безпеки_____ (**Чорновол Н.І.)** з нормоконтролю _____ (Петрашова **В.І.) старший** консультант _____ (Кривченко Ю. В.)
До захисту допущений Голова циклової комісії _____ (**Кривченко Ю.В.**)
Завідувач відділення _____ (Краснокутська К.Г.)

Захист «____» _____ 2025 р. **Протокол ЕК No**_____ **Оцінка ЕК** _____ **Секретар ЕК** _____ **Розробка** програмного застосунку логічної гри на пошук прихованих предметів
12
60

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»
Відділення_____**Комісія**_____
Спеціальність _____
Освітньо-професійна програма_____ **ЗАТВЕРДЖУЮ: Заст. дир.** з НВР_____
_____