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

Навчально-науковий
інститут електроенергетики
(інститут)
Факультет інформаційних технологій
(факультет)
Кафедра інформаційних технологій та комп'ютерної інженерії
(повна назва)

ПОЯСНЮВАЛЬНА ЗАПИСКА
кваліфікаційної роботи ступеня бакалавра

студента Чалої Карини Дмитрівни
(ПІБ)
академічної групи 126-20-1
(шифр)
спеціальності 126 Інформаційні системи та технології
(код і назва спеціальності)
за освітньо-професійною програмою Інформаційні системи та технології
(офіційна назва)
на тему «Розробка ігрового застосунку для розвитку дітей мовою Python»
(назва за наказом ректора)

Керівники	**Прізвище, ініціали**	**Оцінка за шкалою**		**Підпис**
		рейтинговою	**інституційною**	
кваліфікаційної роботи	доц. Сергєєва К.Л.			
розділів:				
Аналіз стану області рішення задачі	доц. Сергєєва К.Л.			
Проектне рішення	доц. Сергєєва К.Л.			

Рецензент	доц. Ширін А.Л.			

Нормоконтролер	Коротенко Г.М.			

Дніпро
2024

ЗАТВЕРДЖЕНО:
завідувач кафедри
інформаційних технологій
та комп'ютерної інженерії
(повна назва)
_____ Гнатушенко В.В.
(підпис) (прізвище, ініціали)

«_____» _____ 2024 року

ЗАВДАННЯ
на кваліфікаційну роботу
ступеню ___бакалавра_____
(бакалавра, спеціаліста, магістра)

Студенту _____Чалій К.Д.____ **академічної групи** __126-20-1_____
(прізвище та ініціали) (шифр)
спеціальності _____126 «Інформаційні системи та технології»_____
(код і назва спеціальності)
за освітньо-професійною програмою _«Інформаційні системи та технології»_
(офіційна назва)
на тему__«Розробка ігрового застосунку для розвитку дітей мовою Python» __,
затверджену наказом ректора НТУ «Дніпровська політехніка» від 23.05.2024 № 469-с

Розділ	**Зміст**	**Термін виконання**
Аналіз стану області рішення задачі	Використання мови програмування Python для розробки навчальних ігрових застосунків для дітей, її переваги. Інструменти для розробки ігор, як Pygame, існуючі ігрові додатки для дітей з їхніми перевагами та недоліками, а також сучасні тенденції у створенні ігрових застосунків.	15.04.2024
Проектне рішення	Проектне рішення ігрового застосунку для розвитку дітей, з акцентом на вивчення математики через гру, та на створенні цікавого і мотивуючого інструменту для ефективного навчання молодших школярів.	20.06.2024

Завдання видано _____ __Сергєєва К.Л.__
(підпис керівника) (прізвище, ініціали)

Дата видачі _____

Дата подання до екзаменаційної комісії __03.07.2024__

Прийнято до виконання _____ ____Чала К.Д.___
(підпис студента) (прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 87 с., 21 рис., 5 додатки, 10 джерел.

Об’єкт розробки: ігровий застосунок, призначений для розвитку математичних навичок у дітей шляхом інтерактивного навчання.

Мета розробки: створення інтерактивного ігрового застосунку, спрямованого на покращення математичних навичок у дітей шляхом надання їм можливості вивчати та практикувати математичні завдання через ігрове середовище, що стимулює учнів до активного навчання та розвитку.

У вступі йдеться про актуальність та важливість використання ігрових застосунків для навчання математики дітей у сучасному світі, де технології грають ключову роль. Ігрові методи навчання привертають увагу дітей та сприяють розвитку їхніх когнітивних і соціальних навичок.

У першому розділі йдеться про використання мови програмування Python для розробки навчальних ігрових застосунків для дітей. Обговорюються переваги Python, включаючи його кросплатформну сумісність, велике співтовариство розробників і викладачів, а також ресурси для навчання. Також розглядаються інструменти для розробки ігор на Python, наприклад, Pygame,. Описуються існуючі ігрові додатки для розвитку дітей, їхні переваги та недоліки, а також сучасні тенденції у створенні ігрових застосунків.

У другому розділі наведено проектне рішення ігрового застосунку, призначеного для розвитку дітей. Ідея цього застосунку – вивченння математики та заохочення дітей способом гри. Головна мета проекту полягає в створенні такого ігрового застосунку, який не лише сприятиме ефективному навчанню математики, але й буде цікавим і мотивуючим для молодших школярів.

Практична цінність цієї кваліфікаційної роботи полягає в розробці ефективного ігрового застосунку, що сприяє навчанню дітей математики. Використання Python і відповідних бібліотек (таких як Pygame) забезпечує створення інтерактивного, захоплюючого ігрового середовища, яке мотивує дітей до вивчення математичних концепцій через гру. Робота пропонує

інноваційний підхід до навчання, який поєднує розвагу та освіту, що сприяє розвитку когнітивних і соціальних навичок дітей.

Проект можна успішно впровадити у школах, дитячих садках, позашкільних установах, бібліотеках, на домашніх уроках, в літніх таборах, на освітніх платформах і мобільних додатках. Це сприятиме інтерактивному навчанню математики, роблячи процес цікавим і доступним для дітей незалежно від контексту та місця.

ІГРОВИЙ ЗАСТОСУНОК, РОЗВИТОК ДІТЕЙ, ГРА, МОВА ПРОГРАМУВАННЯ, PYTHON, PYGAME.

ABSTRACT

Explanatory note: 87 p., 21 fig., 5 appendices, 10 sources.

Object of development: a game application designed to develop children's mathematical skills through interactive learning.

Purpose of development: to create an interactive game application aimed at improving children's mathematical skills by providing them with the opportunity to learn and practice mathematical tasks through a game environment that stimulates students to actively learn and develop.

The introduction discusses the relevance and importance of using game-based applications for teaching math to children in today's world, where technology plays a key role. Game-based learning methods capture children's attention and contribute to the development of their cognitive and social skills.

The first chapter introduces the use of the Python programming language to develop educational game applications for children. It discusses the benefits of Python, including its cross-platform compatibility, large community of developers and educators, and learning resources. Python game development tools, such as Pygame, are also discussed. Existing game applications for children's development, their advantages and disadvantages, as well as current trends in game development are described.

The second section presents a design solution for a game application designed for children's development. The idea of this application is to teach math and encourage children through play. The main goal of the project is to create a game application that will not only contribute to effective math learning, but will also be interesting and motivating for younger students.

The practical value of this qualification work is the development of an effective game application that promotes children's learning of mathematics. The use of Python and relevant libraries (such as Pygame) ensures the creation of an interactive, engaging gaming environment that motivates children to learn math concepts through play. The work offers an innovative approach to learning that combines entertainment and

education, which contributes to the development of children's cognitive and social skills.

The project can be successfully implemented in schools, kindergartens, out-of-school institutions, libraries, home lessons, summer camps, educational platforms and mobile applications. It will promote interactive math learning, making the process interesting and accessible to children regardless of context and location.

GAME APPLICATION, CHILD DEVELOPMENT, GAME, PROGRAMMING LANGUAGE, PYTHON, PYGAME.

ЗМІСТ

ВСТУП

У сучасному світі, насиченому технологіями, навчання через ігри стає все більш актуальним і ефективним методом освіти, особливо коли йдеться про навчання математиці. Ігрові застосунки не лише залучають увагу дітей, але і сприяють розвитку їхніх когнітивних та соціальних навичок.

Актуальністю теми є зростаюча цифровізація суспільства вимагає нових підходів до навчання. Ігрові технології дозволяють створити цікавий та динамічний освітній процес, де кожна дитина може власноруч вивчати математику, розвивати креативність та вирішувати завдання в ігровому форматі.

У процесі розробки використано різноманітні інструменти та технології для забезпечення функціональності, інтерактивності та привабливості ігрового застосунку. До основних компонентів розробки належить мова програмування Python з бібліотекою Pygame для створення інтерактивного ігрового середовища, графіки та анімацій. Це використано для створення захоплюючого та ефективного навчального середовища, яке сприятиме підвищенню зацікавленості дітей у вивченні математики.

Каліфікаційна робота спрямована на розробку ігрового застосунку, призначеного для навчання математики дітей. Основна мета полягає в створенні інтерактивного середовища, яке за допомогою ігрових завдань, головоломок та інтерактивних ситуацій сприяє покращенню навичок у математиці. Робота включає аналіз сучасних підходів до ігрового навчання, дослідження психологічних аспектів використання ігор у навчанні, технології розробки і тестування ефективності застосунку.

Практична частина передбачала реалізацію проекту, його тестування та апробацію в освітніх установах з метою підтвердження його педагогічної ефективності. Висновки роботи будуть спрямовані на узагальнення результатів дослідження та розробку рекомендацій для подальшого використання ігрових технологій у навчальному процесі.

РОЗДІЛ 1
АНАЛІЗ СТАНУ ОБЛАСТІ РІШЕННЯ ЗАДАЧІ

Преамбула

У першому розділі кваліфікаційної роботи подано огляд існуючих проектів для розвитку дітей, їх переваги та недоліки. Розробка освітніх ігрових застосунків для дітей є важливою та актуальною задачею у сучасному світі. Вони не лише надають розваги, але й допомагають у розвитку різноманітних навичок, таких як логічне мислення, швидкість реакції та здатність до вирішення математичних задач. Аналіз стану області рішення задачі включає в себе огляд поточних рішень, оцінку їх ефективності та виявлення можливостей для вдосконалення.

Зі зростанням технологічного прогресу, навчальні методи зазнають значних змін. Традиційні підходи поступаються місцем інтерактивним, цифровим методам, які можуть краще залучити дітей і підтримати їх інтерес до навчання. Гейміфікація освіти, зокрема через використання освітніх ігор, стає все більш популярною.

Також у розділі було оглянуто, що треба врахувати при розробці ігрових застосунків для дітей, зокрема вік і рівень розвитку дитини, предметну область, безпеку та приватність, якість контенту та ефективність навчального матеріалу. Також і тестування та оцінка ефективності застосунків є критично важливими етапами, що допомагають визначити, наскільки успішно застосунок виконує свої завдання з навчання та розвитку дітей.

Завдяки перегляду доступних джерел та літератури, а також практичного використання ігрових застосунків у різних освітніх контекстах. Перш за все, ігрові додатки демонструють значний потенціал у розвитку когнітивних, соціальних та емоційних навичок дітей. Це може допомагти у вивченні нових матеріалів, сприяти розвитку логічного мислення, покращувати пам'ять і концентрацію.

1.1. Вступ до розробки ігрового застосунку для розвитку дітей

Починаючи шлях розробки ігрових додатків за допомогою Python, важливо провести ретельний пошук існуючих існуючих ІТ-рішень. Цей крок гарантує, що розробники знають про доступні інструменти, бібліотеки та фреймворки, які оптимізовують процес розробки та покращують кінцевий продукт. Досліджуючи поточний стан рішень у сфері розробки ігор на Python, розробники використовують існуючі ресурси для вирішення проблем, підвищення ефективності та створення інноваційних та привабливих ігрових програм.

1.1.1 Використання Python

Python – це універсальна та широко використовувана мова програмування, яка добре підходить для створення навчальних програм. Його високорівнева, інтерпретована природа та чіткий синтаксис роблять його ідеальним вибором для початківців, тоді як його великі бібліотеки та модулі забезпечують потужність і гнучкість, необхідні для більш складних програм.

У контексті розробки навчальної ігрової програми для дітей Python пропонує кілька переваг. По-перше, його інтерактивний та захоплюючий характер робить його чудовим вибором для створення веселих і навчальних ігор. Крім того, великі бібліотеки та фреймворки Python, такі як Pygame і Pyglet, надають необхідні інструменти для створення високоякісної графіки, звуку та інших функцій гри.

Python — це гнучка мова загального призначення, яка повністю підтримує як процедурне, так і об’єктно-орієнтоване програмування. Завдяки вбудованим та стороннім пакетам підходить для виконання широкого спектра завдань [1].

Крім того, Python має велике та активне співтовариство розробників і викладачів, а це означає, що є численні ресурси, доступні для тих, хто хоче створювати освітні програми. Ці ресурси включають навчальні посібники, книги, курси та форуми, де розробники можуть ділитися ідеями, ставити запитання та співпрацювати над проектами.

Ще однією перевагою використання Python для освітніх програм є його кросплатформна сумісність. Це означає, що програми, розроблені на Python,

запускаються в різних операційних системах, включаючи Windows, Mac і Linux, що полегшує розповсюдження та обмін освітніми ресурсами.

Хоча Python має багато переваг, одним з його потенційних недоліків може бути швидкість виконання коду. На відміну від компільованих мов, Python спочатку компілюється у внутрішній байт-код, який потім інтерпретується. Це може призвести до того, що програми Python працюють повільніше, ніж аналогічні програми, написані на мовах, таких як C.

Однак важливо зазначити, що сучасні комп'ютери мають значну обчислювальну потужність. У більшості випадків швидкість розробки стає більш важливим фактором, ніж швидкість виконання. А Python, завдяки своїй простоті та лаконічності, дозволяє розробляти програми значно швидше.

Також Python легко розширюється модулями, написаними на C або C++. Такі модулі використовуються для виконання ресурсоємних частин програми, що значно покращує загальну продуктивність.

Python є чудовим вибором для створення навчальних програм, особливо в контексті розробки ігрових програм для дітей. Його універсальність, потужність і гнучкість у поєднанні з великими бібліотеками та активною спільнотою роблять його ідеальним вибором для розробників, які прагнуть створювати цікаві та ефективні освітні ресурси.

Python надає безліч готових бібліотек для різних цілей: наукові обчислення, обробка зображень і даних, машинне навчання та інші. Це робить її універсальним інструментом для будь-якої сфери та завдання [5].

Завдяки пошуку існуючих IT-рішень розробник має багатство знань і досвіду спільноти Python, що дозволяє бути в курсі останніх тенденцій і технологій у розробці ігор.

1.1.2. Інструменти для розробки ігор на Python

Pygame: Pygame – це набір модулів для Python, які дозволяють легко створювати ігри. Він має зручний API і може використовуватися для створення різноманітних ігор для дітей, від аркадних до логічних. Pygame є одним з найпопулярніших інструментів для розробки ігор у Python. Він надає доступ до

різноманітних функцій для роботи з графікою, звуком та управлінням мишею та клавіатурою. Pygame є простим у використанні та дозволяє створювати як 2D, так і деякі типи 3D ігор.

Turtle Graphics: Модуль Turtle у Python – це інструмент для навчання програмуванню, який дозволяє керувати віртуальною "черепашкою". Він ідеально підходить для створення простих ігор, які розвивають логіку та творчість дітей.

Використання GUI бібліотек: Python має багато бібліотек для створення графічного інтерфейсу користувача (GUI), таких як Tkinter, PyQt, або Kivy. Ці бібліотеки можуть бути використані для створення ігор зі своїм інтерфейсом, що дозволяє створювати ігри з різноманітними завданнями і викликами для дітей.

Веб-розробка: Python також може бути використаний для створення веб-ігор з використанням фреймворків, таких як Django або Flask. Це може включати ігри, які дозволяють дітям взаємодіяти одне з одним через Інтернет або навіть навчальні ігри, які грають у веб-браузері.

Навчальні проекти: Створення невеликих ігор може бути частиною навчального процесу для дітей, де вони можуть вивчати основи програмування, в той же час розвиваючи креативність і логічне мислення. Наприклад, створення простої гри "Вгадай число" або "Змійка" може бути цікавим та корисним досвідом.

При розробці ігор у мові програмування Python існує ще й декілька технологій та інструментів, які допомагають створювати різноманітні та цікаві ігрові застосунки. Деякі з них:

Godot Engine з GDScript: Godot Engine – це безкоштовний та відкритий ігровий движок, який має свою власну мову програмування GDScript, що схожа на Python. З GDScript створюються складні та потужні ігри, використовуючи можливості Godot Engine.

Panda3D: Panda3D – це бібліотека для розробки 3D ігор, яка надає широкі можливості для створення графічно складних проектів. Вона підтримує Python

як мову програмування та надає доступ до різних функцій для роботи з 3D графікою, анімацією та фізикою.

Arcade: Arcade – це ще одна бібліотека для розробки ігор у Python, яка спрощує створення 2D ігор. Вона має простий та зрозумілий інтерфейс, що дозволяє швидко створювати ігрові проекти для початківців та досвідчених розробників.

Kivy: Kivy – це бібліотека для розробки мультимедійних додатків, включаючи ігри, з використанням Python. Вона надає можливості для роботи з сенсорними екранами, анімацією та графікою, що робить її популярним вибором для створення ігор для мобільних платформ.

Ці технології та інструменти дозволяють розробникам створювати різноманітні та захоплюючі ігри у мові програмування Python. Вони мають різні особливості та функціонал, що дозволяє розробникам вибирати найбільш підходящий інструмент для своїх потреб.

Створення ігор для розвитку дітей у Python – це захоплюючий спосіб вивчати програмування, який може сприяти їхньому розвитку у різних аспектах.

Авжеж, сучасний світ динамічно розвивається, і все більшу роль у ньому відіграють інформаційні технології. Це твердження справедливе і для сфери освіти, де все частіше використовуються комп'ютерні та мобільні ігри для розвитку дітей.

Ігрові додатки є потужним інструментом для навчання та розвитку дітей. Їх використовують для розвитку когнітивних навичок, таких як логіка, пам'ять, увага та вирішення проблем. Ігри допомогають дітям розвинути соціальні та емоційні навички, такі як співпраця, спілкування та самоконтроль.

Мова програмування Python є популярним вибором для розробки ігрових додатків. Python проста у вивченні та використанні, має велику бібліотеку модулів та фреймворків для розробки ігор.

1.1.3. Використання Python у створенні навчальних застосунків

Створення інтерактивних веб-сайтів: Python може використовуватися для розробки веб-сайтів з інтерактивними вправами, візуалізаціями даних та навчальними іграми.

Розробка мобільних додатків: Python може використовуватися для створення мобільних додатків для iOS та Android, які допомагають учням вивчати різні предмети.

Розробка навчальних ігор: Python може використовуватися для створення ігрових застосунків, які роблять процес навчання більш цікавим та захоплюючим.

Аналіз даних: Python може використовуватися для аналізу даних про успішність учнів та для персоналізації навчання. Створення візуалізацій даних: Python може використовуватися для створення інтерактивних візуалізацій даних, які допомагають учням краще зрозуміти складні концепції.

Простота Python у вивченні, універсальність та широкий спектр бібліотек роблять його ідеальним вибором для розробників, які прагнуть зробити процес навчання більш ефективним та цікавим.

Python використовується для різних цілей: для створення ігор і веб-застосунків, розробки внутрішніх інструментів для різноманітніх проектів. Мова також широко застосовується в науковій області для досліджень і розв’язування прикладних завдань.

Є безліч сучасних ігрових додатків, які використовуються для розвитку дітей.

1.2. Готові проекти для розвитку дітей

На основі Python та Pygame існує безліч готових проектів, спрямованих на розвиток дітей різного віку. Pygame – набір кросплатформенних модулів для мови програмування Python, створений для розробки відеоігор [7]. Ці проекти поєднують в собі розвагу із навчанням, сприяючи розвитку різних навичок через ігровий процес.

Один із типових проектів — це ігри, що сприяють розвитку логічного мислення. Вони можуть включати головоломки, лабіринти або завдання на збір предметів, що допомагають дітям розвивати стратегічне та просторове мислення.

Інші проекти фокусуються на розвиток математичних навичок через ігрові завдання, такі як гра з арифметичними операціями або вправи на розвиток обчислювальної швидкості. Це дозволяє дітям вдосконалювати навички математики в ігровій формі, що робить процес навчання більш цікавим і залучаючим.

1.2.1. Існуючі застосунки для розвитку дітей, їх переваги та недоліки

Khan Academy Kids: Цей додаток пропонує безліч ігор та вправ для розвитку когнітивних та соціальних навичок дітей.

Наш набір захоплюючих навчальних ігор дає дітям змогу розвивати необхідні навички для формування цифрового майбутнього [8].

Ця програма охоплює математику, читання, письмо, логіку, соціальний та емоційний розвиток, а також творчі ігри. Зараз ця програма викладається тільки англійською мовою [3].

Переваги: Велика кількість ігор та вправ, доступність українською мовою.

Недоліки: Деякі ігри не дають дітям знань з певної предметної області.

DragonBox Algebra 5+: Цей додаток використовує ігри, щоб допомогти дітям навчитися алгебрі.

Переваги: Ефективний метод навчання алгебрі. Недоліки: Недоступний українською мовою.

Toonia: Цей додаток пропонує безліч ігор та вправ для розвитку логічного мислення та вирішення проблем.

Переваги: Розвиває логічне мислення та вирішення проблем. Недоліки: Немає чіткої структури навчання.

Рисунок 1.1 – Гра Khan Academy Kids

Kodable: Цей додаток використовує ігри, щоб допомогти дітям навчитися основам програмування.

Переваги: Допомагає дітям навчитися основам програмування.

Недоліки: Недоступний українською мовою, платний.

Рисунок 1.2 – Гра Kodable

Ці та інші ігрові додатки свідчать про те, що існує великий потенціал для використання ігор у розвитку дітей. Звісно існує багато прикладів створення ігор на мові програмування Python, вони діляться на такі категорії:

Ігри на пам'ять: Ці ігри допомагають дітям розвивати пам'ять та концентрацію уваги. Прикладом може бути гра "Simon Says", де діти повинні повторювати дії, які їм говорить віртуальний помічник.

Ігри на логіку: Ці ігри допомагають дітям розвивати логічне мислення та навички вирішення проблем. Прикладом може бути гра "Tic-Tac-Toe", де діти повинні розташувати три свої фігури в ряд, щоб перемогти.

Ігри на творчість: Ці ігри допомагають дітям розвивати уяву та творчі здібності. Прикладом може бути гра "Storytelling", де діти можуть створювати свої власні історії за допомогою віртуальних персонажів та декорацій.

Storyteller — це дуже проста і зрозуміла гра-головоломка. Гравцям надається бібліотека персонажів і сцен, а також завдання зібрати їх разом, щоб розповісти історії на теми кохання, зради та помсти [6].

Рисунок 1.3 – Гра Storytelling

Незважаючи на те, що існує багато ігрових додатків для розвитку дітей, все ще є потреба в нових додатках. Звісно, існують і проблеми з сучасними ігровими додатками. Ці проблеми описують так:

Не всі ігри є освітніми: Деякі ігри просто розважальні і не дають дітям жодних знань чи навичок.

Не всі ігри відповідають віку дітей: Деякі ігри занадто складні для маленьких дітей, а інші занадто прості для старших дітей.

Не всі ігри доступні українською мовою: Це може зробити їх недоступними для дітей, які не володіють англійською мовою.

В сучасному світі ігрові додатки стають все більш популярними інструментами для розвитку дітей.

1.2.2. Сучасні тенденції у створенні ігрових застосунків

Персоналізація: Ігри стають більш персоналізованими, адаптуючись до індивідуальних потреб та здібностей кожної дитини. Це робить процес навчання більш ефективним та захоплюючим.

Інтерактивність: Ігри стають більш інтерактивними, використовуючи сенсорні екрани, доповнену реальність та інші технології, які дозволяють дітям активно досліджувати та взаємодіяти з віртуальним світом.

Соціальний аспект: Ігри стають більш соціальними, даючи можливість дітям спілкуватися та співпрацювати один з одним в рамках ігрового процесу.

Gamification: Принципи гейміфікації (використання ігрових механік в неігрових контекстах) все частіше використовуються в освітніх застосунках для підвищення мотивації та залучення дітей до навчання.

STEM-освіта: Ігрові додатки все частіше використовуються для викладання STEM-дисциплін (наука, технології, інженерія, математика), роблячи їх більш доступними та цікавими для дітей. Ігрові технології є складовою частиною педагогічних технологій, однією з унікальних форм навчання, яка дозволяє зробити цікавою і захоплюючою не тільки роботу учнів на творчо-пошуковому рівні, але і буденні кроки по вивченню навчальних предметів [10].

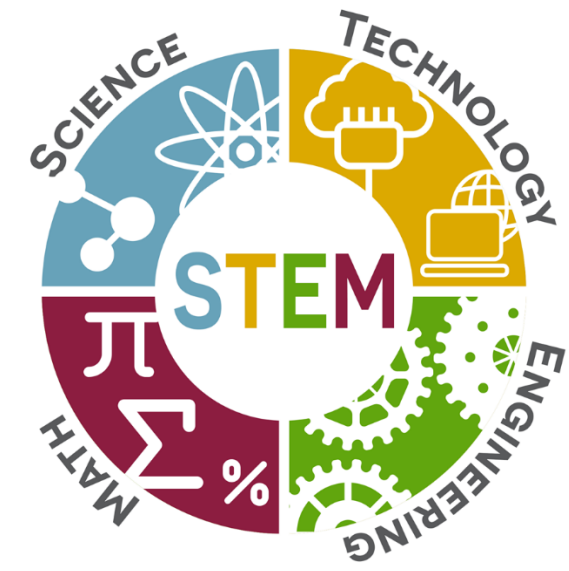

Рисунок 1.4 – STEAM-освіта

Розвиток емоційного інтелекту: Ігри стають більш спрямованими на розвиток емоційного інтелекту дітей, включаючи такі навички, як емпатія, самоконтроль та самоусвідомлення.

Безпека та приватність: Розробники ігрових додатків все більше уваги приділяють питанням безпеки та приватності дітей, використовуючи відповідні технології та політики.

Доступність: Ігрові додатки стають більш доступними для дітей з різними потребами, включаючи дітей з інвалідністю.

Дослідження та оцінка: Все більше досліджень присвячено вивченню ефективності ігрових додатків для розвитку дітей, що дозволяє розробникам вдосконалювати свої продукти.

Зростання ринку: Ринок ігрових додатків для розвитку дітей стрімко зростає, що свідчить про високий попит на ці продукти з боку батьків та освітян.

Одним з найважливіших є принципи гейміфікації, в діяльності з дітьми використовуються для стимулювання їхньої мотивації, підвищення зацікавленості та активності у навчанні та розвитку. Основні принципи гейміфікації, які застосовуються у діяльності з дітьми, включають:

Цілі та досягнення: Встановлення конкретних цілей та досягнень, які діти можуть отримати через участь у діяльності.

Змагальний дух та співпраця: Створення змагальних та співпрацюючих елементів у діяльності, які стимулюють дітей до досягнення кращих результатів та співпраці з однолітками.

Нагороди та визнання: Використання системи нагород та визнання, таких як бейджі, медалі, лідерські дошки тощо, для підвищення мотивації дітей до досягнення цілей та виконання завдань.

Прогрес та розвиток: Відстеження прогресу та розвитку кожного учасника діяльності та надання зворотного зв'язку, що допомагає дітям бачити свій успіх та досягнення.

Індивідуалізація та адаптація: Надання можливостей для індивідуального розвитку та адаптації діяльності до потреб кожного учасника, щоб кожна дитина могла знайти в ній виклик та інтерес.

Емоційне залучення: Створення захоплюючого та емоційно залученого середовища, яке сприяє активному участю дітей та позитивному відношенню до навчання та розвитку.

Гейміфікація в діяльності з дітьми може бути ефективним інструментом для підвищення їхньої мотивації, зацікавленості та результативності у навчанні та розвитку. Важливо враховувати індивідуальні особливості та потреби кожної дитини та створювати гейміфіковані активності, які відповідають їхнім інтересам та можливостям.

Ігрові додатки не повинні повністю замінювати традиційні методи навчання. Їх слід використовувати як доповнення до інших освітніх ресурсів.

Під час створення ігрового застосунку треба звернути увагу на:

- Вік дитини
- Рівень розвитку дитини
- Предметна область, яку хочеться розвинути
- Безпека та приватність
- Якість контенту
- Ефективність

Ігрові додатки є потужним інструментом для розвитку дітей, якщо їх використовувати з розумом.

1.3 Оцінка та валідація ігрового застосунку

Оцінка та валідація ігрового застосунку є важливим етапом у процесі його розробки. Цей етап дозволяє перевірити функціональність, продуктивність та відповідність вимогам перед випуском програми на ринок чи використанням у навчальних цілях.

1.3.1. Тестування

Тестування та оцінка ефективності застосунку з точки зору розвитку дитини є важливим етапом у процесі розробки ігрового застосунку для дітей.

Оцінка ефективності допомагає визначити, наскільки успішно застосунок виконує свої основні завдання щодо навчання та розвитку дітей. Можна визначити деякі кроки, які можна виконати для тестування та оцінки ефективності застосунку:

Пілотне тестування: Перед використанням застосунку на широкій аудиторії дітей, важливо провести пілотне тестування на невеликій групі дітей. Це дозволяє виявити можливі проблеми, помилки та недоліки у роботі застосунку.

Збір даних: Під час тестування важливо збирати дані про взаємодію дітей з застосунком, їхні досягнення та реакцію на навчальний контент. Це може бути здійснено за допомогою анкет, спостережень та відгуків від батьків та вчителів.

Оцінка навчальних результатів: Порівнювати досягнення дітей, які використовували застосунок, з тими, хто не користувався ним. Вимірюйте зміни у рівні знань, вмінь та навичок учнів, щоб оцінити ефективність застосунку як засобу навчання.

Аналіз співпраці та соціального взаємодії: Важливо враховувати не лише навчальні результати, але й взаємодію дітей під час користування застосунком. Спостерігайте за їхнім спілкуванням, співпрацею та розвитком соціальних навичок.

Залучення експертів: Попросіть педагогів, психологів або фахівців у галузі розвитку дітей оцінити ефективність застосунку з погляду їхньої експертної думки та професійного досвіду.

Врахування відгуків користувачів: Збирайте відгуки від користувачів застосунку, включаючи дітей, батьків та педагогів. Вони можуть надати корисну інформацію щодо користування застосунком та його впливу на розвиток дитини.

1.3.2. Загальна оцінка ефективності застосунку

Загальна оцінка ефективності застосунку має враховувати як навчальні результати, так і загальний вплив на розвиток дітей з різних аспектів. Тестування та оцінка повинні бути проведені систематично та об'єктивно з метою покращення якості застосунку та його відповідності потребам користувачів.

Не треба забувати про відсутність ігрових рішень на базі Python, – це є однією з проблем, які виникають у контексті розвитку ігрових застосунків для дітей. Деякі причини цієї відсутності є такими:

Обмеженість у готових рішеннях: Існує обмежена кількість готових ігрових рішень на базі Python порівняно з іншими мовами програмування, такими як Unity (C#) або Unreal Engine (C++).

Неадекватність для складних ігор: Python часто використовується для створення простих або середньо складних ігор, але не завжди підходить для створення складних та графічно потужних ігор, оскільки його продуктивність може бути обмеженою.

Менший ринок для Python-ігор: У порівнянні з іншими мовами програмування, такими як C# або C++, ринок ігор, розроблених з використанням Python, менший, що може зробити його менш привабливим для розробників ігор.

Брак спеціалізованих ігрових двигунів: Існують спеціалізовані ігрові двигуни, такі як Unity або Unreal Engine, які надають більше можливостей для розробки ігор, ніж загальнопризначені бібліотеки Python, такі як Pygame чи Kivy.

Рисунок 1.5 – Логотип Pygame

Низька продуктивність для деяких типів ігор: Для деяких типів ігор, особливо для графічно важких або ігор з великою кількістю обчислень, Python може мати низьку продуктивність порівняно з іншими мовами програмування.

У зв'язку з цим, виникає необхідність в подальшому дослідженні та розвитку ігрових рішень на базі Python, а також у вивченні шляхів оптимізації та покращення продуктивності для створення більш складних та потужних ігор.

Необхідність розвитку дітей через ігри є важливим аспектом сучасної педагогіки та психології розвитку. Ігри відіграють ключову роль у процесі навчання та розвитку дітей з кількох причин:

Залучення у процес навчання: Ігровий підхід до навчання дозволяє дітям бути активно залученими до навчального процесу. Гра створює стимул до вивчення та виконання завдань, оскільки діти сприймають навчальні матеріали як частину цікавого та захоплюючого процесу.

Розвиток різних навичок: Ігри можуть сприяти розвитку різних аспектів особистості дитини, включаючи когнітивні (логіка, пам'ять, увага), моторні (координація рухів), соціальні (спілкування, співпраця) та емоційні (толерантність до стресу, управління емоціями) навички.

Стимулювання творчості та креативності: Багато ігор стимулюють творчість та креативність дітей, оскільки вони можуть експериментувати, вирішувати проблеми та виявляти нові підходи до вирішення завдань.

Підвищення мотивації та самодисципліни: Участь у викликах ігор сприяє формуванню мотивації до досягнення цілей та виявленню самодисципліни в процесі їх досягнення.

Вивчення важливих життєвих навичок: Деякі ігри допомагають дітям вивчати важливі навички для успішного функціонування у сучасному світі, такі як фінансова грамотність, критичне мислення та прийняття рішень.

Також використання штучного інтелекту (ШІ) у дитячих іграх може відкривати широкі можливості для покращення досвіду гри, навчання та розвитку дітей. , які використовуються для застосування ШІ у дитячих іграх:

Підганяння складності гри: ШІ може адаптувати складність гри до рівня вмінь та здібностей кожної дитини. Наприклад, в іграх з головоломками ШІ може адаптувати рівень складності завдань в залежності від успішності гравця.

Персоналізований навчальний процес: ШІ може аналізувати здібності та потреби кожної дитини та надавати індивідуалізовані завдання та поради для покращення навичок та знань. Створення інтелектуальних ворогів або партнерів: У багатьох іграх використовується ШІ для створення інтелектуальних ворогів або партнерів, які реагують на дії гравця та приймають відповідні рішення.

Покращення реалістичності та іммерсії: ШІ може допомагати створювати більш реалістичне середовище гри, де персонажі реагують на дії гравця більш природним чином.

Автоматизоване управління гравцями-ботами: ШІ може керувати поведінкою віртуальних гравців-ботів у мультиплеєрних іграх, що дозволяє створювати більш динамічні та викликаючі ігрові ситуації.

Навчання розв'язанню проблем та прийняттю рішень: ШІ може моделювати різні ситуації та допомагати дітям розвивати навички розв'язання проблем та прийняття рішень у безпечній ігровій обстановці.

Використання ШІ у дитячих іграх може допомагати не лише покращити розваги, а й зробити їх більш освітніми та розвиваючими. Важливо забезпечити етичне та відповідальне використання ШІ у дитячому контенті, а також враховувати потреби та особливості кожної вікової групи дітей.

Ігри є потужним інструментом для навчання та розвитку дітей, оскільки вони не лише надають можливість вивчати нові знання, але й сприяють розвитку комплексу різних навичок та якостей, які є важливими для успішного життя у сучасному світі.

1.3.3. Технічні вимоги та можливості

Дуже важливим є технічні вимоги та можливості, вони можуть варіюватися в залежності від специфіки проекту та потреб цільової аудиторії. Однак, ось загальні технічні вимоги та можливостей:

Мова програмування Python: Основним інструментом для розробки буде мова програмування Python. Python є високорівневою мовою, яка дозволяє швидко та ефективно реалізувати ігровий функціонал.

Ігрові двигуни та бібліотеки: Для створення ігрового застосунку можна використовувати різноманітні ігрові двигуни та бібліотеки, такі як Pygame, Pyglet, Panda3D, або використовувати вбудовані інструменти для реалізації графічного інтерфейсу.

Графічний двигун та анімація: Використання графічного двигуна та можливостей анімації дозволяє створювати привабливі та захоплюючі ігрові середовища.

Звукове супроводження: Додавання звукового супроводу до ігрового застосунку може підвищити іммерсію та привабливість для гравців. Для цього можна використовувати бібліотеки для роботи зі звуком, такі як Pygame.mixer або Pydub.

Інтерактивність та взаємодія: Розробка механік взаємодії та інтерактивних елементів дозволяє створити цікаві та захоплюючі геймплейні ситуації, що сприяють активному навчанню та розвитку. Інтеграція з штучним інтелектом: Використання штучного інтелекту дозволяє створювати інтелектуальних ворогів або партнерів, персоналізовані поради та рекомендації, а також адаптувати гру до індивідуальних потреб гравців.

Мобільність: Якщо потрібно, ігровий застосунок може бути адаптований для мобільних пристроїв, що розширює його досяжність та доступність для широкого кола користувачів.

Доступність: Важливо також забезпечити, щоб ігровий застосунок був доступний для дітей з різними потребами, включаючи можливості для налаштування інтерфейсу, адаптивний геймплей та підтримку веб-доступності.

Використання ігор для навчання та розвитку дітей має значний потенціал у підвищенні мотивації, зацікавленості та результативності навчання.

Це вказує на важливість використання принципів гейміфікації, а також інтеграції штучного інтелекту у дитячі ігри для створення персоналізованих, цікавих та розвиваючих ігрових досвідів. Переваги використання мови програмування Python для розробки ігор полягають у її простоті та доступності, а також у великій кількості бібліотек та інструментів, які полегшують процес розробки.

Необхідно звернути увагу на індивідуальні потреби та особливості дітей під час розробки ігрових застосунків, а також забезпечити високий рівень безпеки та конфіденційності в їхній взаємодії з грою. Додаткові дослідження та

розробка практичних рішень у цій області можуть сприяти подальшому вдосконаленню та розвитку ігрових застосунків для розвитку дітей мовою Python.

Звісно, існує певний дефіцит ігрових застосунків для розвитку дітей, розроблених мовою програмування Python. Хоча Python є популярним інструментом для розробки програмного забезпечення, особливо в освітніх цілях, ігрові застосунки для дітей на цій мові все ще залишаються досить обмеженими.

1.4. Користь ігрового застосунку

Робота над розробкою ігрового застосунку для розвитку дітей мовою Python має велике значення. Вона може допомогти заповнити прогалину на ринку ігор для дітей, а також надати нові можливості для навчання та розвитку через ігри на базі Python.

Слід зазначити, що розвиток ігрових застосунків для дітей мовою Python може сприяти не лише їхньому навчанню, а й стимулювати творчість, логічне мислення та проблемне вирішення. Гра як інструмент розвитку може сприяти формуванню цінних навичок, таких як співпраця, комунікація та критичне мислення, що є важливими у сучасному світі.

Крім того, розробка ігрових застосунків відкриває можливості для дослідження нових підходів до навчання та розвитку дітей. Шляхом експериментування з різними методами гейміфікації, інтеграції інтерактивних елементів та адаптації гри до індивідуальних потреб користувачів можна досягти значного прогресу у використанні ігор як засобу навчання.

Розробка ігрових застосунків для дітей мовою Python може бути відмінним інструментом для залучення дітей до програмування та комп'ютерних наук. Це може стимулювати їх інтерес до STEM-освіти та підготовити їх до майбутніх професійних викликів у сфері технологій.

Розробка ігрового застосунку для розвитку дітей мовою Python є актуальною та перспективною задачею, яка може мати значний позитивний вплив на навчання та розвиток дітей у сучасному світі.

1.4.1 Перспективи та можливості розробки ігор на Python

Розробка ігрових застосунків на Python – це динамічно розвиваюча галузь з безліччю можливостей. Завдяки своїй простоті, гнучкості та широкому спектру доступних інструментів Python є чудовим вибором для розробників, які хочуть створювати захоплюючі ігри для різних платформ. З постійним розвитком нових технологій, таких як штучний інтелект, віртуальна та доповнена реальність та хмарні ігри, майбутнє розробки ігор на Python виглядає дуже перспективно.

Дослідження існуючих ігрових рішень на Python підтвердило, що ця мова програмування дозволяє ефективно втілювати різноманітні ідеї та концепції в ігровому середовищі. Багато успішних проектів вже було створено з використанням Python, і вони продовжують привертати увагу користувачів та розвиватися.

Крім того, Python має велику та активну спільноту розробників, яка надає значну підтримку та ресурси для вирішення проблем і вдосконалення ігрових проектів. Це забезпечує певну стабільність та можливості для росту для тих, хто працює у цій області.

Висновок

Після проведеного аналізу стану галузі розробки ігрових застосунків на мові Python можна зробити наступні висновки: Перш за все, виявлено значну активність та постійний розвиток цієї галузі. Python вже давно визнаний як одна з переважних мов програмування для розробки ігор, завдяки своїй простоті в освоєнні, гнучкості та багатству наявних бібліотек і фреймворків.

Отже, на основі проведеного аналізу можна зробити висновок, що розробка ігрового застосунку на мові Python має значний потенціал для успішної реалізації та впровадження інноваційних ідей у сфері геймдеву. Успіх у цій галузі залежить від творчості, технічної компетентності та здатності до співпраці з іншими членами спільноти, але умови для досягнення успіху вже створені.

Python має свої обмеження, особливо в контексті швидкодії великих та ресурсомістких ігор. Хоча він і є досить ефективним для створення прототипів

та невеликих проектів, для великих ігрових проектів часто використовують мови програмування з більшою продуктивністю, такі як C++ або Java.

Також, варто враховувати зростаючу конкуренцію в галузі розробки ігор. З кожним роком все більше розробників виходить на цей ринок, що може зробити важчим вирішення проблеми виділення вашого продукту серед конкурентів.

Проте, не зважаючи на ці виклики, перспективи розробки ігрових застосунків на мові Python залишаються обіцяючими. Вона продовжує залишатися однією з найпопулярніших мов для початківців та досвідчених розробників, завдяки своїй простоті та потужності. Знаючи це, можна зробити висновок, що успішна реалізація ігрового застосунку на мові Python потребує уважного планування, творчості та відданості, але вона також має великий потенціал для досягнення успіху в цій динамічній та захоплюючій галузі.

Успішна реалізація ігрового застосунку на мові Python також потребує уваги до взаємодії з користувачами та їхнього залучення до процесу розробки. Важливо забезпечити зручний інтерфейс користувача, який би сприяв залученню гравців та забезпечував їм зручність у використанні застосунку. При цьому слід враховувати різноманітність аудиторії та їхні індивідуальні потреби та вподобання.

Треба розглянути впровадження елементів соціальної взаємодії у грі, таких як мультиплеєрні режими або можливість обміну результатами гри та враженнями між користувачами. Це дозволить створити спільноту навколо вашого застосунку та підвищити його популярність.

Загалом, успішна реалізація ігрового застосунку на мові Python вимагає від розробників не лише технічної компетентності, але й уміння взаємодіяти з аудиторією, творчості та стратегічного мислення. Завдяки цьому, проект матиме всі шанси на успіх та визнання у геймдев-індустрії.

Розробка ігрових застосунків на мові Python є перспективним напрямом, який пропонує широкі можливості для інновацій та творчості. Подальші дослідження та розвиток технологій у цій області сприятимуть покращенню

якості та розширенню функціональності ігрових продуктів, роблячи їх доступнішими та привабливішими для широкого кола користувачів.

Підводячи підсумок, розробка ігрового застосунку для розвитку дітей мовою Python має потенціал стати важливим інструментом у навчальному та розвивальному процесі дітей. Дослідження та розробка в цій області можуть сприяти покращенню якості освіти та розвитку дітей, забезпечуючи їм цікаві та ефективні ігрові досвіди, які сприяють їхньому зростанню та навчанню.

РОЗДІЛ 2
ПРОЕКТНІ РІШЕННЯ

2.1. Основна ідея проекту

Sweet math – це ігровий застосунок, який допомагає дітям вивчати математику.

У цьому застосунку є кілька рівнів складності, які відповідають віковій групі дитини. У грі гравець керує бджілкою, яка повинна знищувати квітки, відповідаючи на математичні питання. Якщо гравець відповідає правильно, він отримує бали. Якщо гравець помиляється, він втрачає серця. Якщо гравець втратить всі серця, гра закінчується.

У грі також є можливість вибору рівня складності та вихід у головне меню.

2.1.1 Об’єкти та ідея

Головний персонаж:

Гравець керує бджілкою, яка може рухатися вгору і вниз по екрану. Метою є уникання зіткнень з квітками та іншими перешкодами.

Квітки:

Ці об'єкти рухаються з правого боку екрану до лівого. Кожен з них має випадкове числове значення, яке може бути відповіддю на математичне питання, відображене на екрані.

Математичні питання та відповіді:

На екрані відображається математичне питання з варіантами відповідей, які представлені на квітках. Гравець повинен вибрати правильний варіант, стріляючи у відповідну квітку, щоб пролетіти далі.

Рахунок:

Відображає кількість балів, які гравець заробив під час гри. Бали надаються за кожен успішний постріл у правильну квітку.

Управління:

Гравець може керувати бджілкою за допомогою клавіш на клавіатурі: вверх, вниз, стріляти. Також є можливість натискати клавішу ENTER для виходу в головне меню.

Завершення гри і головне меню:

Після того як гравець втрачає всі життя (серця), гра завершується, і відображається головне меню, де можна розпочати нову гру або вийти з програми.

Метою ігрового застосунку є розвиваток та перевірка математичних навичок користувача через інтерактивне взаємодію з гравцем.

2.2. Функціональність та тестування ігрового застосунку

Функціональність та тестування ігрового застосунку є критичним етапом у його розробці, спрямованим на забезпечення якості, надійності та коректної роботи програми під час використання користувачами. Першим кроком у розробці є визначення основних функціональних вимог до ігрового застосунку. Це можуть бути ігрові механіки, система керування, інтерфейс користувача, обробка введення користувача та інші ключові аспекти.

2.2.1. Найменування

Проект має назву "Sweet math" («Солодка математика») – це ігровий застосунок для вивчення математики дітей.

Цей ігровий застосунок може застосовуватись у освіті, розвагах, дитячих іграх.

2.2.2. Область застосування

Цей проект може бути використаний як освітня гра для дітей, що допомагає вивчати математику через інтерактивний ігровий процес. Гравець може вдосконалювати свої навички множення, вибираючи правильні відповіді на квітках.

2.2.3 Тестування реакції і швидкості

Гра дозволяє гравцеві вдосконалювати реакційні навички та швидкість реагування, уникаючи зіткнень з перешкодами (квітками) та одночасно стріляючи у них.

В цілому, цей проект може служити не лише для розваг, але і для навчання.

2.3. Підстави для розробки

Сучасне суспільство знаходиться на етапі стрімкої цифровізації, яка охоплює всі сфери життя, включаючи освіту. Впровадження технологій в освітній процес дозволяє зробити його більш ефективним, цікавим та доступним для різних категорій учнів. Одним з перспективних напрямків є використання ігрових технологій у навчанні, зокрема для розвитку математичних навичок у дітей.

2.3.1 Освітня потреба

Розвиток математичних навичок: Багато дітей мають труднощі з засвоєнням математичних знань у традиційній шкільній формі. Гра допоможе зробити навчання цікавим та інтерактивним.

2.3.2 Ігрова терапія

Використання ігор у терапевтичних цілях для дітей, які мають труднощі з навчанням або соціалізацією. "Sweet math" може стати ефективним інструментом для педагогів та психологів.

2.3.3. Стимулювання інтересу до наук

Гра може сприяти ранньому інтересу до наук, технологій, інженерії та математики (STEM), що важливо для розвитку сучасного суспільства.

2.3.4 Ефективність навчання

Діти краще запам’ятовують інформацію, коли вона подається в ігровій формі. Ігровий формат допомагає утримувати увагу та зацікавленість дітей.

Додаток може адаптуватися до рівня знань та швидкості навчання кожної дитини, що робить навчальний процес більш ефективним.

2.3.5 Потреби батьків та вчителів

Батьки отримують інструмент, який допоможе їхнім дітям навчатися в цікавій формі, що також сприяє спільному проведенню часу.

Учителі можуть використовувати додаток як додатковий навчальний ресурс на уроках математики

2.3.6. Внесок у розвиток освіти.

Розробка таких проектів сприяє загальному розвитку освіти та підвищенню рівня знань у суспільстві.

2.3.7. Доступність освіти

Гра може бути доступною для дітей з різних соціальних та економічних верств, що сприяє рівноправному доступу до освіти.

2.3.8. Використання сучасних технологій

Використання мобільних додатків та інтерактивних технологій у навчанні стає все більш популярним, а розробка таких продуктів відповідає тенденціям ринку.

2.4. Призначення розробки

Призначенням розробки є створення інтерактивного ігрового застосунку, що сприятиме розвитку математичних навичок у дітей молодшого та середнього шкільного віку. Застосунок це не тільки інструментом навчання, але й засобом підвищення мотивації до вивчення математики через залучення дітей у захопливий ігровий процес.

2.4.1 Освітня мета

Покращення математичних навичок: Основна мета розробки – допомогти дітям покращити свої математичні навички через інтерактивні та цікаві ігрові завдання.

За допомогою ігор можна вивчати багато шкільних предметів і тем зі шкільної програми. Також цей метод можна використовувати для подання додаткового матеріалу з різних предметів або розвитку різних навичок [9].

Підготовка до школи: Застосунок може використовуватися як інструмент для підготовки дошкільнят до навчання у школі, зокрема для засвоєння базових математичних понять.

2.4.2. Мотивація та залучення

Залучення дітей до навчання: Гра допомагає утримувати увагу та зацікавленість дітей, що сприяє більш ефективному навчанню.

Мотивація через гру: Ігровий формат сприяє мотивації дітей до вивчення математики, роблячи процес навчання захоплюючим і веселим.

2.4.3. Розвиток когнітивних навичок

Розвиток логічного мислення: Гра стимулює розвиток логічного мислення, вирішення проблем та аналітичних навичок у дітей.

Покращення пам'яті та уваги: Виконання різних завдань і рішень математичних задач сприяє покращенню пам'яті та концентрації уваги.

2.4.4. Адаптивне навчання

Індивідуальний підхід: Додаток може адаптувати складність завдань відповідно до рівня знань кожної дитини, що дозволяє забезпечити індивідуальний підхід до навчання.

Моніторинг прогресу: Вчителі та батьки можуть стежити за прогресом дитини, що дозволяє коригувати навчальний процес відповідно до потреб дитини.

2.4.5. Підтримка педагогів і батьків

Додатковий навчальний ресурс: Застосунок може використовуватися вчителями як додатковий ресурс на уроках математики або під час домашнього навчання.

Зручність для батьків: Батьки отримують ефективний інструмент для допомоги дітям у вивченні математики в домашніх умовах.

2.4.6. Соціальна відповідальність

Доступність якісної освіти: Проект спрямований на забезпечення доступу до якісної освіти для дітей з різних соціальних та економічних верств.

Рівноправність в освіті: Забезпечення рівних можливостей для всіх дітей у навчанні математики через інноваційні та інтерактивні методи.

2.4.7. Популяризація STEM-напрямків

Заохочення інтересу до науки і техніки: Гра може стати першим кроком до зацікавленості дітей у науці, технологіях, інженерії та математиці (STEM), що є важливим для їхнього майбутнього. STEM-освіта запроваджується в умовах інтеграції усіх видів освіти: формальної, неформальної, інформальної [4].

2.4.8. Забезпечення розваги та навчання

Баланс між розвагою та навчанням: Гра забезпечує баланс між розвагою та навчанням, що робить процес вивчення математики приємним і ефективним.

Ці аспекти призначення розробки підкреслюють важливість створення ігрового застосунку, який допоможе дітям не лише освоїти математичні навички, але й розвивати логічне мислення, пам'ять, увагу та загальний інтерес до науки.

2.5. Основні вимоги до розробки та інтерфейсу

Розробка інтерактивного ігрового застосунку для розвитку математичних навичок у дітей вимагає дотримання низки ключових вимог. Ці вимоги покликані забезпечити не тільки ефективне навчання, але й створити привабливе, мотивуюче та доступне середовище для користувачів.

2.5.1. Функціональні вимоги

Головний екран і меню:

Відображення головного меню з кнопкою "Грати".

Можливість вибору рівня складності перед початком гри.

Ігровий процес:

Підтримка ігрового поля з рухомими об'єктами (квітки, бджілка).

Рух бджілки вгору і вниз за допомогою клавіш управління.

Генерація випадкових числових операцій на квітках.

Перевірка відповідей гравця на квітки (стріляння кулями).

Відображення поточного рахунку.

Обробка зіткнень та взаємодія об'єктів:

Визначення умов зіткнення бджілки з квіткою.

Реакція на зіткнення: втрата життя гравцем (якщо зіткнення відбулося).

Видалення квіток та куль при зіткненні з відповідними об'єктами.

Управління грою:

Можливість паузи і виходу до головного меню.

Обробка кліків мишею для виходу з гри.

Візуальні ефекти та звуки:

Відтворення звуків під час стрільби кулями та зіткнень.

Анімація знищення квіток і відображення візуальних ефектів при попаданні.

Підтримка різних рівнів складності:

Зміна швидкості руху бджілки відповідно до обраного рівня.

Зміна діапазону випадкових числових операцій в залежності від рівня.

Збереження стану гри:

Оновлення поточних налаштувань гри (рахунок, рівень, стан об'єктів) після кожної гри.

Можливість зберігання найкращих результатів.

2.5.2. Нефункціональні вимоги

Нефункціональні вимоги визначають якісні характеристики системи, які не є безпосередньо пов'язаними з її функціональністю.

Ефективність та продуктивність:

Гра повинна зберігати стабільну кількість кадрів на секунду (FPS) навіть при інтенсивному використанні графічних ефектів.

Мінімізація затримок і простою при обробці зіткнень та графічного оновлення.

Надійність:

Зменшення ймовірності програмних помилок і збоїв під час гри.

Адекватне керування винятками та відновленням гри після помилок.

Сумісність:

Сумісність гри з різними операційними системами (Windows, macOS, Linux).

Оптимізація під різні типи графічних адаптерів і їх параметри.

Зручність використання:

Інтуїтивний і зрозумілий інтерфейс користувача (меню, управління, взаємодія з об'єктами).

Підтримка клавіатурного та мишиного управління, а також можливість використання геймпадів.

Документація та підтримка:

Наявність документації для користувачів і розробників (інструкції).

Можливість швидкого відгуку на звернення користувачів щодо проблем та питань.

Естетичність:

Привабливий дизайн ігрового середовища.

Відповідність стандартам графічного дизайну ігрової індустрії.

2.5.5. Вимоги до інтерфейсу

Вимоги до інтерфейсу важливі для забезпечення зручного і ефективного взаємодії користувача з грою.

Головне меню:

Меню з можливістю вибору рівня складності (1-4).

Ігрове вікно:

Відображення основної інформації: рахунок, рівень, кількість життів гравця.

Відповідність графічного оформлення тематиці гри (пасіка, квітки, вулики тощо).

Інтерфейс для управління гравцем (клавіші або миша для переміщення).

Елементи гри:

Відображення квіток з числовими значеннями та відповідними варіантами відповідей.

Відображення куль і їх рух на екрані.

Анімації та звукові ефекти для дії взаємодії (попадання тощо).

Кінцева гра (Game Over):

Відображення результату гри: набрані бали, досягнутий рівень.

Можливість рестарту гри або повернення до головного меню.

Загальні вимоги до інтерфейсу:

Інтуїтивний дизайн інтерфейсу, легкість навігації для користувача.

Співвідношення розмірів і графічний стиль елементів інтерфейсу для зручності використання на різних роздільних здатностях екранів.

Достатній контраст кольорів для забезпечення читабельності тексту та інших елементів.

Адаптивність:

Підтримка різних типів управління: клавіатура, миша.

Можливість адаптації інтерфейсу під різні операційні системи і пристрої.

Ці вимоги допоможуть забезпечити, що гра матиме зручний, привабливий і функціональний інтерфейс, який сприятиме задоволенню користувачів під час гри.

2.6. Поєднання Python і Pygame: створення ігрових застосунків.

Python та Pygame втілюють у собі безмежні можливості для створення захоплюючих ігрових застосунків. Python, як мова програмування, відома своєю простотою та потужністю, що дозволяє розробникам швидко і ефективно втілювати ідеї в життя. У поєднанні з Pygame, бібліотекою для розробки ігор на Python, вони утворюють міцну комбінацію, яка сприяє створенню якісних та інноваційних ігрових проектів.

2.6.1 Python як мова програмування

Простота та зрозумілість: Python має чистий та лаконічний синтаксис, що робить його легким для вивчення та розуміння, особливо для початківців.

Широкий вибір бібліотек: Python має велику кількість бібліотек для різних потреб, що спрощує розробку графічних програм, таких як ігри.

Підтримка спільноти: Python має велику та активну спільноту розробників, яка надає підтримку, документацію та різноманітні рішення для розв'язання проблем.

2.6.2 Pygame для створення ігрового застосунку

Простота використання: Pygame є простим у використанні фреймворком для розробки ігор, який дозволяє швидко створювати ігрові програми.

Широкі можливості: Pygame надає доступ до графіки, звуку та управління, що дозволяє реалізувати різноманітні ігрові механіки.

Підтримка спільноти: Pygame також має активну спільноту розробників, яка надає багато прикладів, документацію та підтримку для новачків.

Pygame і бібліотека SDL переносяться на різні платформи та пристрої, вони повинні визначати та працювати з абстракціями для різних реалій апаратного забезпечення [2].

2.6.3 Об'єднання Python та Pygame

Ідеальна комбінація: Python та Pygame в поєднанні надають можливість швидко та ефективно розробляти графічні ігри з нескладною логікою.

Підтримка мови програмування: Використання Python дозволяє легко реалізувати математичні алгоритми та генерацію завдань для навчального аспекту гри.

Загалом, вибір Python та Pygame для розробки ігрового застосунку обумовлений їхньою простотою використання, масштабованістю, підтримкою спільноти та широкими можливостями створення ігрових додатків.

2.7 Огляд проекту

"Sweet math" – це інтерактивна гра для дітей, призначена для розвитку математичних навичок через веселу ігрову діяльність. Гра поєднує елементи навчання та розваг, що робить процес навчання привабливим для дітей.

Архітектурою та технологією є мова програмування Python та ігровий рушій Pygame, графікою є статичні зображення для фонових сцен, анімації для персонажів та об'єктів, існують звукові ефекти для різних дій (вистріл, натискання кнопок). Інтерфейс: Графічний інтерфейс з кнопками для навігації в меню, показом балів, питань та інших елементів гри.

Основні компоненти це:

Меню: Головне меню, меню налаштувань, екран з інструкціями.

Ігрове поле: Основний ігровий екран з персонажем (бджілка), мішень (квіточки) та об'єктами (серця, кулі).

Система рівнів: Чотири рівні складності, які змінюють діапазон чисел у завданнях.

Система очок: Відображення набраних очок, правильні відповіді на завдання збільшують кількість очок.

Запитання та відповіді: Випадкові математичні завдання на множення з трьома варіантами відповідей.

Колізії: Перевірка зіткнень між персонажем, квіткою та кулями.

2.7.1 Блок-схема та uml-діаграма.

Блок-схема послідовності дій у програмі написаній з використанням бібліотеки Pygame.

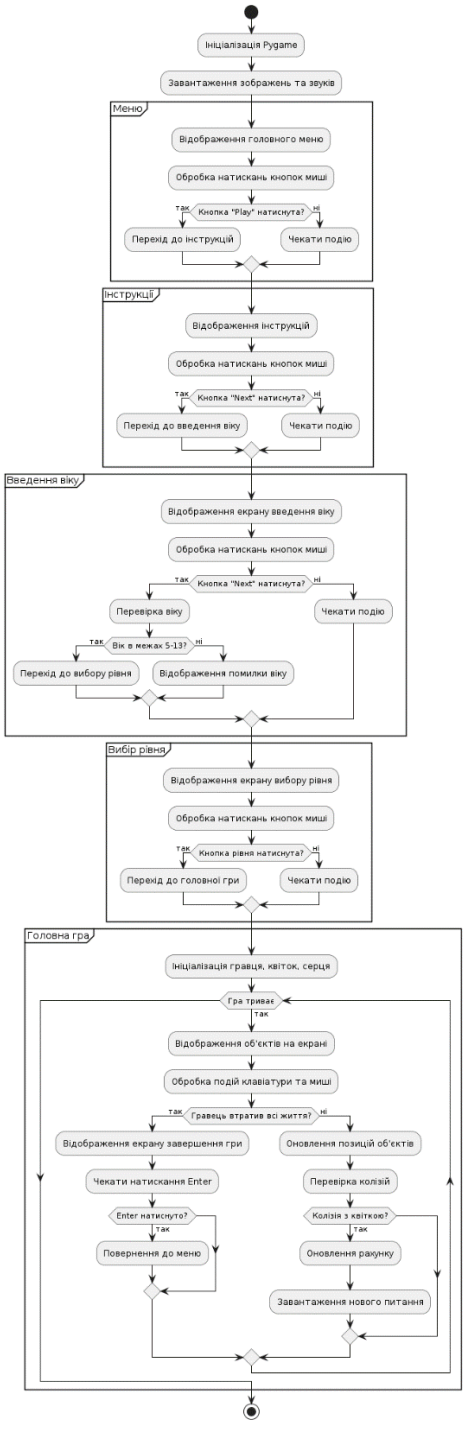

Рисунок 2.1 – Блок-схема проекту

Опис блоків та функціоналу:

Ініціалізація Pygame: Запускається процес ініціалізації бібліотеки Pygame, що необхідно для роботи програми.

Завантаження зображень та звуків: Після ініціалізації завантажуються всі необхідні для гри зображення та звукові файли.

Меню

Відображення головного меню: На екран виводиться головне меню гри.

Обробка натискань кнопок миші: Програма обробляє натискання кнопок миші.

Якщо натиснута кнопка "Play":

Переходить до інструкцій.

Якщо ні:

Чекає подію (нове натискання).

Інструкції

Відображення інструкцій: На екран виводяться інструкції до гри.

Обробка натискань кнопок миші: Програма обробляє натискання кнопок миші.

Якщо натиснута кнопка "Next":

Переходить до введення віку.

Якщо ні:

Чекає подію.

Введення віку

Відображення екрану введення віку: На екран виводиться форма для введення віку.

Обробка натискань кнопок миші: Програма обробляє натискання кнопок миші.

Якщо натиснута кнопка "Next":

Перевірка віку.

Якщо вік в межах 5-13 років:

Переходить до вибору рівня.

Якщо ні:

Відображається помилка віку.

Якщо ні:

Чекає подію.

Вибір рівня

Відображення екрану вибору рівня: На екран виводиться меню вибору рівня.

Обробка натискань кнопок миші: Програма обробляє натискання кнопок миші.

Якщо натиснута кнопка рівня:

Переходить до головної гри.

Якщо ні:

Чекає подію.

Головна гра

Ініціалізація гравця, квіток, сердець: Програма ініціалізує всі необхідні об'єкти для гри, такі як гравець, квітки та серця.

Гра триває?: Перевірка, чи триває гра.

Якщо так:

Відображаються об'єкти на екрані.

Обробка подій клавіатури та миші.

Гравець втратив всі життя?

Якщо так:

Відображається екран завершення гри.

Чекає натискання Enter.

Якщо натиснуто Enter:

Повернення до меню.

Якщо ні:

Оновлення позицій об'єктів.

Перевірка колізій.

Якщо колізія з квіткою:

Оновлення рахунку.

Завантаження нового питання.

Якщо ні:

Завершення гри.

Ця блок-схема представляє логіку роботи гри, включаючи етапи завантаження, відображення меню та інструкцій, введення віку, вибору рівня, ігрового процесу та завершення гри.

Діаграма класів застосунку:

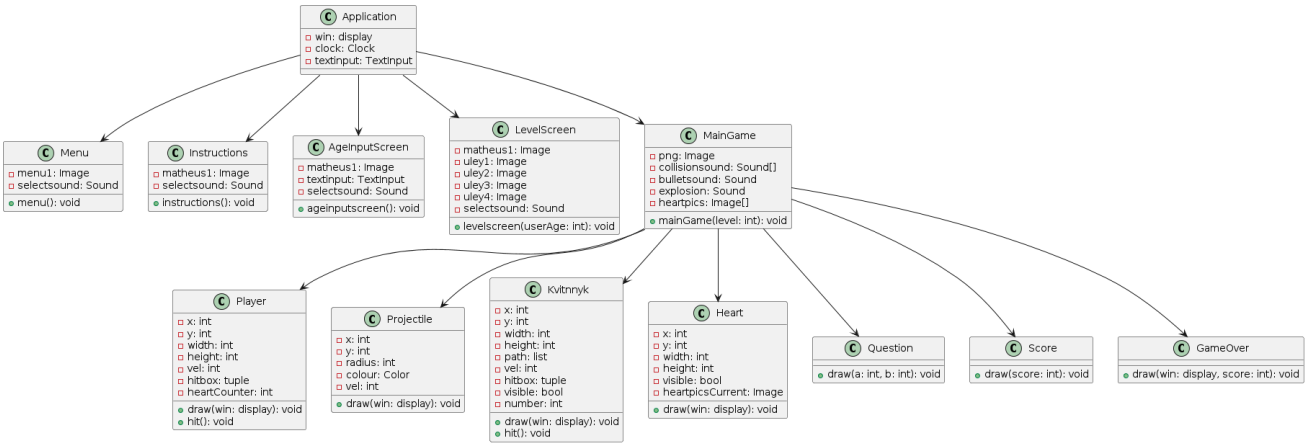

Рисунок 2.2 – Діаграма класів

Ця діаграма являє собою діаграму класів для програми. Діаграма класів показує різні класи програми та їхні відносини один до одного.

Application – це головний клас програми. Він містить інші класи, які відповідають за різні частини програми.

Menu – відповідає за відображення меню програми.

Instructions – відповідає за відображення інструкцій до гри.

AgeInputScreen – відповідає за отримання віку користувача.

LevelScreen – відповідає за відображення рівня гри.

MainGame – відповідає за саму гру.

Kvitnnyk – відповідає за квіти в грі.

Heart – відповідає за серця в грі.

Player – відповідає за гравця.

Projectile – відповідає за снаряди.

Question – відповідає за питання в грі.

Score – відповідає за рахунок в грі.

GameOver – відповідає за екран завершення гри.

На діаграмі показано, які класи взаємодіють один з одним. Клас Application використовує класи Menu, Instructions, AgeInputScreen, LevelScreen, MainGame. Клас MainGame використовує класи Player, Projectile, Kvitnnyk, Heart, Question, Score, GameOver.

2.7.2 Створення меню

Для створення першого етапу (першого екрану) за допомогою функції menu(). На екрані відображається фон menu1 та текст із заголовком гри та кнопкою «Play». Обрано колір, а також шрифт.

Функція menu()

redrawGameWindow(): Ця функція відповідає за оновлення екрану гри.

Використовує win.blit(menu1, (0, 0)) для відображення зображення фону (menu1) на вікні (win).

Викликає функцію screenText() для відображення тексту на екрані.

Оновлює дисплей, щоб показати зміни за допомогою pygame.display.update().

screenText(): Встановлює шрифт для тексту за допомогою pygame.font.Font('ARCADECLASSIC.TTF', 150).

Створює текст з параметрами font.render('Sweetmath', 1, (150, 0, 24)), де 'Sweetmath' - це сам текст, 1 - антиаліасинг, а (150, 0, 24) – це колір тексту.

Відображає текст на екрані за допомогою win.blit(text, (500 - ((text.get_width() / 2) - 20), 40)), позиціонуючи його по центру.

Основний цикл подій:

run = True: Використовується для контролю основного циклу подій.

clock.tick(27): Встановлює швидкість оновлення екрану.

keys = pygame.key.get_pressed(): Отримує стан всіх клавіш на клавіатурі.

for event in pygame.event.get(): Обробляє всі події, що відбуваються (натискання клавіш, рух миші тощо).

if event.type == pygame.QUIT: Якщо натиснута кнопка виходу, то виходить з програми за допомогою quit().

if event.type == pygame.MOUSEBUTTONDOWN: Перевіряє, чи була натиснута кнопка миші.

if mouse[0] >= 430 and mouse[0] <= 570 and mouse[1] >= 280 and mouse[1] <= 320: Перевіряє, чи координати миші знаходяться в певному діапазоні.

selectsound.play(): Відтворює звук при натисканні кнопки.

instructions(): Викликає наступний екран з інструкціями.

mouse = pygame.mouse.get_pos(): Отримує поточну позицію курсора миші.

redrawGameWindow(): Оновлює екран з новими параметрами.

Загалом створюється основний екран меню гри, який відображає фонове зображення та текст заголовка "Sweetmath". Користувач може взаємодіяти з меню за допомогою миші. Якщо користувач натискає на визначену область екрану (кнопку), відтворюється звук і викликається функція instructions(), яка, імовірно, відображає екран з інструкціями до гри.

Рисунок 2.3 – Перший вітальний екран Play

Другий екран – це екран інструкцій, який створюється за допомогою функції instructions(). На цьому екрані відображається фон matheus1 та текст із інструкціями для гри.

Для створення другого екрану, який є екраном інструкцій у грі на Python за допомогою бібліотеки Pygame, код складається з кількох функцій та основного циклу подій.

Функція instructions()

redrawGameWindow(): Ця функція відповідає за оновлення екрану гри.

Використовує win.blit(matheus1, (0, 0)) для відображення зображення фону (matheus1) на вікні (win).

Викликає функцію screenText() для відображення тексту на екрані.

Оновлює дисплей, щоб показати зміни за допомогою pygame.display.update().

screenText(): Встановлює шрифт для тексту заголовка за допомогою pygame.font.Font('ARCADECLASSIC.TTF', 150).

Створює та відображає текст заголовка "Sweetmath" за допомогою win.blit(text, (500 - ((text.get_width() / 2) - 20), 40)).

Встановлює шрифт для тексту інструкцій за допомогою pygame.font.Font('ARCADECLASSIC.TTF', 60) та відображає текст "Instructions".

Встановлює шрифт для абзаців тексту за допомогою pygame.font.Font('ARCADECLASSIC.TTF', 30).

Створює та відображає текст інструкцій, розбитий на кілька рядків (text3, text4, text5, text6, text7), позиціонуючи їх відповідно.

Малює прямокутник кнопки за допомогою pygame.draw.rect(win, (0, 100, 200), [(500 - (140 / 2)), 440, 130, 40]).

Встановлює шрифт для тексту кнопки та відображає текст "Next".

Основний цикл подій:

run = True: Використовується для контролю основного циклу подій.

clock.tick(27): Встановлює швидкість оновлення екрану.

for event in pygame.event.get(): Обробляє всі події, що відбуваються (наприклад, натискання клавіш, рух миші тощо).

if event.type == pygame.QUIT: Якщо натиснута кнопка виходу, то виходить з програми за допомогою quit().

if event.type == pygame.MOUSEBUTTONDOWN: Перевіряє, чи була натиснута кнопка миші.

if mouse[0] >= 430 and mouse[0] <= 570 and mouse[1] >= 440 and mouse[1] <= 480: Перевіряє, чи координати миші знаходяться в певному діапазоні (передбачається, що це область кнопки).

selectsound.play(): Відтворює звук при натисканні кнопки.

ageinputscreen(): Викликає наступний екран, який, ймовірно, є екраном для введення віку.

mouse = pygame.mouse.get_pos(): Отримує поточну позицію курсора миші.

redrawGameWindow(): Оновлює екран з новими параметрами.

Створено екран інструкцій для гри, який відображає фонове зображення, заголовок гри "Sweetmath", текст інструкцій та кнопку "Next". Користувач може взаємодіяти з екраном за допомогою миші. Якщо користувач натискає на визначену область екрану (кнопку "Next"), відтворюється звук і викликається функція ageinputscreen(), яка, ймовірно, відображає екран для введення віку. Таким чином, другий екран з інструкціями матиме такий вигляд:

Рисунок 2.4 – Другий екран з інструкцією до гри

На третьому екрані відображається фон matheus2 та текст із запитанням про вік користувача. Ця функція викликається, коли користувач натискає кнопку "Next" на другому екрані. На цьому екрані користувач може ввести свій

вік та почати гру. Після цього користувач переходить до наступного екрану, де відображаються рівні гри

Функція ageinputscreen()

redrawGameWindow(): Ця функція відповідає за оновлення екрану гри.

Використовує win.blit(matheus1, (0, 0)) для відображення зображення фону (matheus1) на вікні (win).

Викликає функцію screenText() для відображення тексту на екрані.

Відображає текстове поле для введення віку за допомогою win.blit(textinput.get_surface(), (450, 293)).

Викликає функцію ageChecker(), ймовірно, для перевірки введеного віку.

Оновлює дисплей, щоб показати зміни за допомогою pygame.display.update().

screenText(): Встановлює шрифт для тексту заголовка за допомогою pygame.font.Font('ARCADECLASSIC.TTF', 150).

Створює та відображає текст заголовка "Sweetmath" за допомогою win.blit(text, (500 - ((text.get_width() / 2) - 20), 40)).

Встановлює шрифт для тексту інструкції щодо віку за допомогою pygame.font.Font('ARCADECLASSIC.TTF', 80) та відображає текст "Enter your age".

Малює прямокутник для кнопки "Next" за допомогою pygame.draw.rect(win, (0, 100, 200), [(500 - (140 / 2)), 440, 130, 40]).

Малює прямокутник для введення віку за допомогою pygame.draw.rect(win, (255, 255, 255), [(500 - (200 / 2)), 300, 200, 80]).

Встановлює шрифт для тексту кнопки та відображає текст "Next".

Основний цикл подій:

run = True: Використовується для контролю основного циклу подій.

clock.tick(27): Встановлює швидкість оновлення екрану.

for event in pygame.event.get(): Обробляє всі події, що відбуваються (натискання клавіш, рух миші тощо).

if event.type == pygame.QUIT: Якщо натиснута кнопка виходу, то виходить з програми за допомогою quit().

if event.type == pygame.MOUSEBUTTONDOWN: Перевіряє, чи була натиснута кнопка миші.

if mouse[0] >= 430 and mouse[0] <= 570 and mouse[1] >= 440 and mouse[1] <= 480: Перевіряє, чи координати миші знаходяться в певному діапазоні (передбачається, що це область кнопки).

selectsound.play(): Відтворює звук при натисканні кнопки.

nextScreen(): Викликає наступний екран, який, ймовірно, є наступним етапом гри.

textinput.update(events): Оновлює стан текстового вводу на основі подій.

mouse = pygame.mouse.get_pos(): Отримує поточну позицію курсора миші.

redrawGameWindow(): Оновлює екран з новими параметрами.

Створюється третій екран для введення віку, який відображає фонове зображення, заголовок гри "Sweetmath", текст інструкцій щодо введення віку, поле для введення віку та кнопку "Next". Користувач може взаємодіяти з екраном за допомогою миші та клавіатури. Якщо користувач натискає на визначену область екрану (кнопку "Next"), відтворюється звук і викликається функція nextScreen(), яка, ймовірно, відображає наступний етап гри.

Рисунок 2.5 – Введення віку

Також є обмеження по віку від 5 до 13 років.

Функція ageChecker() перевіряє введений вік і відображає відповідні повідомлення, якщо вік занадто молодий, занадто старий або введене значення не є дійсним числом.

Якщо користувач занадто молодий, відображається «You are too young!».

Якщо користувач занадто старий, відображається «You are too old!».

Якщо користувач вводить неправильний вік, відображається «Будь ласка, введіть вірний вік!».

Функція levelscreen(userAge) викликається, коли вік користувача знаходиться в допустимому діапазоні.

Ця функція відображає екран рівня з фоном matheus2, matheus3 або matheus4, залежно від віку користувача. Користувач може вибрати рівень, натиснувши на відповідну бджілку.

Рисунок 2.5.1 – Вік занадто малий

Рисунок 2.5.2 – Вік занадто великий

Рисунок 2.5.3 – Введено невірні дані

Четвертий екран – це екран вибору рівня гри. Він відображає чотири кнопки з номерами рівнів (1, 2, 3, 4) та текст "Recommended level" (Рекомендований рівень), який вказує на рівень, що відповідає віку користувача.

Цей екран також реагує на кліки миші, дозволяючи користувачеві вибрати рівень гри. Коли користувач клікає на одну з кнопок, програма змінює значення змінної level та викликає функцію mainGame з відповідним рівнем.

Окрім того, на цьому екрані відображається заголовок гри "Sweetmath" та текст "Select Level" (Виберіть рівень).

Функція redrawGameWindow(userAge): Відображає фон, зображення чотирьох вуликів та текстові елементи на екрані.

Викликає функцію screenText(userAge) для відображення тексту.

Функція screenText(userAge): Відображає заголовок гри "Sweetmath".

Відображає текст "Select Level".

Відображає номери рівнів (1, 2, 3, 4) над відповідними вуликами.

Відображає рекомендований рівень для користувача на основі його віку. Рекомендований рівень розміщується під відповідним вуликом з урахуванням вікових груп (5-7 років, 8-10 років, 11-12 років, 13 років).

Основний цикл подій:

Обробляє події, такі як натискання миші та вихід з гри.

Визначає, на який рівень натиснув користувач, і запускає відповідний рівень гри, використовуючи функцію mainGame(level).

Логіка вибору рівня:

В залежності від віку користувача (userAge), текст "Recommended level" відображається під відповідним вуликом, що вказує на рекомендований рівень для даного віку.

При натисканні на відповідний вулик користувачем, викликається функція mainGame(level), передаючи номер вибраного рівня.

Цей екран забезпечує інтуїтивно зрозумілий інтерфейс для вибору рівня гри, де користувачі можуть легко обрати рівень, рекомендований для їх вікової групи.

Основний цикл подій

Основний цикл подій обробляє взаємодії користувача з екраном, такі як натискання клавіш та натискання миші для вибору рівня. В залежності від вибраного рівня, запускається відповідний рівень гри.

Рисунок 2.6 – Рівні у вигляді вуликів

2.7.3 Створення рівня

Після вибору рівня, функція mainGame відповідає за запуск основного ігрового процесу. В цій функції виконується ініціалізація всіх необхідних об'єктів, таких як гравець, проектили (кулі), квіти та інші ігрові елементи. Основні дії, які відбуваються в mainGame:

Ініціалізація об'єктів: Створюються всі необхідні об'єкти, такі як гравець, квіти, серця, запитання та кулі.

Генерація запитань: В залежності від обраного рівня генеруються числа для математичних запитань.

Основний ігровий цикл: У цьому циклі обробляються всі події, відображаються об'єкти на екрані та виконується логіка гри.

Перевірка зіткнень: Перевіряється, чи зіткнувся гравець з ворогами або чи влучили кулі у ворогів.

Оновлення екрану: Відображення всіх ігрових елементів на екрані.

Якщо розглядати докладно функцію mainGame і всі її класи, які визначають поведінку ігрових об'єктів, щоб зрозуміти, що саме виконується і для чого це потрібно, можна побачити, що

Функція mainGame запускає основний ігровий процес, який включає ініціалізацію ігрових об'єктів, обробку подій, оновлення стану гри та відображення всіх елементів на екрані.

Клас player:

Ініціалізація гравця: Цей метод створює об'єкт гравця (бджілку) з початковими координатами, розмірами, швидкістю та кількістю життів.

self.x та self.y визначають позицію гравця на екрані.

self.width та self.height задають розміри гравця.

self.vel визначає швидкість руху гравця.

self.hitbox задає область для зіткнень.

self.heartCounter тримає кількість життів гравця.

Відображення гравця: Цей метод малює гравця на екрані та оновлює його hitbox.

win.blit(png, (self.x, self.y)) малює зображення бджілки на екрані.

self.hitbox оновлюється відповідно до поточної позиції гравця.

Рисунок 2.7 – Головний персонаж – бджілка

Обробка зіткнень: Цей метод зменшує кількість життів гравця при зіткненні з квіткою, відтворює звук зіткнення та оновлює відображення сердець.

random.choice(collisionsound).play() відтворює випадковий звук зіткнення.

bee.heartCounter -= 1 зменшує кількість життів гравця.

В залежності від кількості життів, оновлюється зображення сердець.

Якщо життів більше немає (bee.heartCounter == 0), виводиться "game over".

Клас projectile

Ініціалізація снаряда: Цей метод створює об'єкт снаряда з початковими координатами, радіусом, кольором та швидкістю.

self.x та self.y визначають позицію снаряда на екрані.

self.radius задає розмір снаряда.

self.colour визначає колір снаряда.

self.vel задає швидкість снаряда.

Відображення снаряда: Цей метод малює снаряд на екрані.

pygame.draw.circle(win, self.colour, (self.x, self.y), self.radius) малює коло на екрані, яке представляє снаряд.

Клас kvitnnyk

Ініціалізація квітки: Цей метод створює об'єкт квітки з початковими координатами, розмірами, швидкістю та видимістю.

self.x та self.y визначають початкову позицію квітки.

self.width та self.height задають розмір квітки.

self.path визначає шлях руху квітки.

self.vel задає швидкість квітки.

self.hitbox визначає область для зіткнень.

self.visible визначає, чи квітка видима.

self.number тримає значення, яке відображається на квітці.

Відображення квітки: Цей метод малює квітку на екрані, якщо вона видима.

Рисунок 2.8 – Квітка

В залежності від рівня (level), відображається відповідне зображення квітки.

Self.hitbox оновлюється відповідно до поточної позиції квітки.

Відображається номер квітки (self.number).

Обробка попадання по квітці: Цей метод робить квітку невидимою, якщо по ній влучив снаряд.

Клас heart

Ініціалізація серця: Цей метод створює об'єкт серця з початковими координатами, розмірами та видимістю.

self.x та self.y визначають позицію серця на екрані.

self.width та self.height задають розмір серця.

self.visible визначає, чи серце видиме.

self.heartpicsCurrent задає поточне зображення серця.

Відображення серця: Цей метод малює серце на екрані, якщо воно видиме.

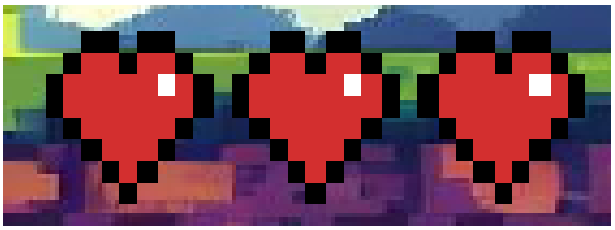

Рисунок 2.9 – Сердечка (життя)

Клас question

Ініціалізація питання: Цей метод задає значення змінних a і b для питання та відображає питання на екрані.

self.a та self.b визначають множники питання.

Питання відображається внизу екрана.

Рисунок 2.10 – Математичне запитання

Клас score

Ініціалізація оцінки: Цей метод задає значення поточного рахунку та відображає його у верхньому правому куті екрану. Також малюється кнопка виходу у верхньому лівому куті.

self.score визначає поточний рахунок гравця.

Рисунок 2.11 – Відображення рахунку

Клас gameover

Ініціалізація завершення гри: Цей метод перевіряє, чи кількість життів гравця дорівнює нулю, і якщо так, відображає текст "GAME OVER" та рахунок гравця.

Текст "GAME OVER" відображається у верхній частині екрана.

Рахунок гравця відображається під текстом "GAME OVER".

Текст з інструкціями для переходу в меню відображається нижче.

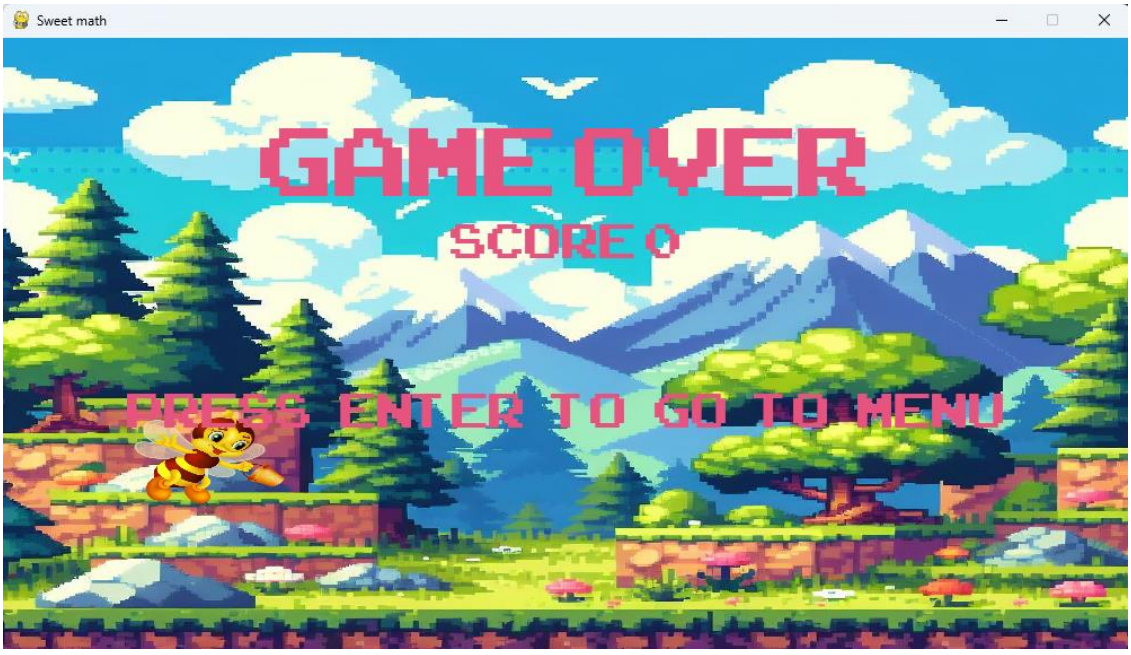

Рисунок 2.12 – Вікно завершення гри

Визначаються основні об'єкти гри, їхню ініціалізацію, поведінку та відображення. Гравець (бджілка) рухається по екрану, стріляє у вірну відповідь,

щоб пролетіти далі (квітку), отримує питання для вирішення, і його життя відображаються за допомогою сердець. Ігровий процес включає обробку зіткнень, оновлення рахунку та завершення гри при втраті всіх життів.

Також відбувається відображення фонового зображення залежно від рівня (level):

Код перевіряє значення змінної level і відображає відповідне фонове зображення за допомогою функції blit об'єкта win.

Наприклад, якщо level == 1, використовується зображення matheus1.

Малювання інших об'єктів:

Після відображення фону, програма малює інші об'єкти на екрані за допомогою методів draw об'єктів bee, kvitkaTop, kvitkaMiddle, kvitkaBottom, heartInGame, questionGame, scoreGame та gameoverGame.

Кожен об'єкт може бути окремим класом, який містить метод draw, що реалізований для малювання відповідного об'єкта на екрані.

Відображення куль (bullets):

Потім код малює кожну кулю зі списку bullets за допомогою методу draw.

Оновлення екрану:

Завершується фрагмент коду з викликом pygame.display.update(), що оновлює вміст екрану, щоб відобразити всі зміни, внесені до цього моменту.

Інші дії:

Є такі дії, як встановлення значень number1 та number2 в залежності від level, вибір випадкового числа whichKvitochka і подальше встановлення числових значень для об'єктів kvitkaTop, kvitkaMiddle і kvitkaBottom.

Змінна scorequestionVisible використовується для визначення того, чи потрібно малювати об'єкти, що відображають рахунок і запитання.

Є основноий цикл, що постійно виконується, щоб оновлювати відображення гри. Кожен об'єкт (гравець, квітка, серце, куля) має свій метод draw, який малює його на екрані. Фон змінюється в залежності від рівня гри, що визначається змінною level, і відповідне фонове зображення вибирається для відображення.

Обробка зіткнень:

У кожному циклі перевіряється, чи сталося зіткнення між бджілкою (bee) і квіткою (kvitkaTop, kvitkaMiddle, kvitkaBottom), якщо вони є видимими (visible == True).

Для кожної квітки перевіряється, чи бджілка зіткнулася з його областю hitbox. Це виконується за допомогою перевірки координат hitbox бджілки і квітки.

Якщо зіткнення відбулося (bee.x == kvitkaTop.x або для інших квіток), викликається метод bee.hit(), який, ймовірно, зменшує кількість життів (heartCounter) у бджілки.

Обробка подій і взаємодія з користувачем:

В цьому розділі обробляються різні події, що виникають під час гри. pygame.event.get() використовується для отримання всіх подій, які відбулися (натискання клавіш, клацання мишею тощо).

Якщо подія pygame.QUIT, гра завершується за допомогою функції quit().

Якщо відбулося клацання мишею (подія pygame.MOUSEBUTTONDOWN), перевіряється, чи миша натиснута на конкретну область (наприклад, кнопку виходу). Якщо так, виконується функція menu(), і гра повертається до головного меню.

Рух та взаємодія куль:

Перевіряється кожна куля (bullet) у списку bullets.

Для кожної кулі перевіряється, чи вона зіткнулася з правильною квіткою (whichKvitochka == 1, 2 або 3). Якщо так, вона знищується (bullets.pop(bullets.index(bullet))), додається до рахунку очок (scorenumber += 1), відтворюється звук вибуху (explosion.play()) і викликається метод hit() для квітки, що відповідає за його знищення або зміну стану.

Взаємодію об'єктів гри, обробку взаємодії з користувачем та реалізацію основних механік гри, таких як рух об'єктів, взаємодія зіткнень і вистрілів описано далі:

Управління бджілкою (bee):

Перевіряється, чи натиснута клавіша pygame.K_SPACE (пробіл). Якщо так, і кількість куль (bullets) менше однієї, то створюється новий об'єкт projectile (куля) з координатами, що відповідають позиції бджілки. Цей об'єкт додається до списку куль (bullets), і відтворюється звук кулі (bulletsound.play()).

Рисунок 2.13 – Випущено кульку

Рух бджілки (bee):

Перевіряється, чи натиснута клавіша pygame.K_UP (стрілка вгору). Якщо так і позиція bee.y більше 50, то зменшується координата y бджілки на її швидкість (bee.vel), що дає відчуття підйому бджілки.

Якщо натиснута клавіша pygame.K_DOWN (стрілка вниз) і позиція bee.y менше 400, то збільшується координата y бджілки на її швидкість, що дає відчуття спуску бджілки.

Перемальовування вікна гри:

В кінці кожного циклу викликається функція redrawGameWindow(), яка оновлює вікно гри з урахуванням останніх змін. В цей момент оновлюється зображення бджілки, квіток, куль, рахунку та інших елементів, які можуть змінюватися з часом.

Вихід з гри:

Після завершення основного циклу гри викликається функція menu(), що відображає головне меню або інші елементи інтерфейсу після завершення гри.

Після цього викликається pygame.quit(), що завершує роботу із бібліотекою Pygame і закриває вікно гри.

Тобто відбувається відображення основного циклу гри, включаючи управління гравцем (bee), стрільбу, рух об'єктів, обробку зіткнень і взаємодію з користувачем через клавіатуру та мишу. Він також включає завершення гри і повернення до головного меню після закінчення гри.

ВИСНОВКИ

Цей ігровий засосунок повинен навчити гравців швидко вирішувати приклади з математики, виконувати множення. Цільовою аудиторією є діти від 5 до 13 років, освітньою користю є те, щоб гра сприяла розвитку математичних навичок у дітей через веселі та інтерактивні завдання.

Для розробки гри були використані такі технології та інструменти як мова програмування Python, яка використовувась для написання логіки гри та взаємодії об'єктів. Використано бібліотеку графіки Pygame, яка забезпечує функції для створення графічних ігрових інтерфейсів, обробки подій та анімації. Також використано звукові ефекти та музику, а також формлено графічний дизайн.

Результати розробки гри мають значний практичний вплив і практичне значення в навчанні через ігри тому, що гра сприяє візуальному та інтерактивному навчанню математики серед дітей. Вона забезпечує не лише засвоєння математичних фактів, а й розвиває логічне мислення та швидкість реакції, сприяючи активному і веселому способу навчання. Звісно, є і розвиток когнітивних вмінь, бо інтерактивний процес гри сприяє розвитку когнітивних вмінь у дітей, включаючи концентрацію, увагу до деталей, пам'ять та розуміння математичних операцій.

Загалом, гра не лише навчає математики, а й стимулює розвиток різних когнітивних навичок, важливих для успішної навчальної та соціальної адаптації дітей.

Для подальшого розвитку гри можна розширювати математичні завдання, додати опції для вибору інших математичних операцій, таких як додавання, віднімання, ділення. Крім того, краще було б розширити ігрові режими, таких як додавання нових режимів гри з різними умовами та завданнями. В подальшому можна розробити версії гри для різних платформ, щоб забезпечити доступність для широкої аудиторії дітей.

Загалом гра дозволяє дітям вчитися через гру, що зроблено доступним і захопливим способом, стимулюючи їхній інтерес до предмета та підвищуючи

мотивацію до навчання. Вона може бути ефективним доповненням до традиційних методів викладання математики, допомагаючи учням розвивати ключові навички, необхідні для їхнього успіху.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Що таке Python. Переваги та недоліки.
[Електронний ресурс]. – Режим доступу: https://acode.com.ua/intro-python/.
2. PyGame: A Primer on Game Programming in Python [Електронний ресурс]. – Режим доступу: https://realpython.com/pygame-a-primer/
3. Ukrainian Parent Guide: Довідник для батьків: Початок роботи з Khan Academy Kids [Електронний ресурс]. – Режим доступу: https://khankids.zendesk.com/hc/en-us/articles/5382924247835
4. STEM-освіта [Електронний ресурс]. – Режим доступу: https://imzo.gov.ua/stem-osvita/.
5. Де використовується Python і чому вам потрібно знати цю мову [Електронний ресурс]. – Режим доступу: https://genius.space/lab/de-vikoristovuyetsya-python-i-chomu-vam-potribno-znati-tsyu-movu/
6. Review: Storyteller Is a Puzzle Game That Leaves You Wanting More [Електронний ресурс]. – Режим доступу:
https://www.gamecrate.com/reviews/review-storyteller
7. Pygame [Електронний ресурс]. – Режим доступу:
https://uk.wikipedia.org/wiki/Pygame
8. Kodable can help any kid learn to code [Електронний ресурс]. – Режим доступу: https://www.kodable.com/
9. Відмінності між гейміфікацією й ігровим навчанням [Електронний ресурс]. – Режим доступу: https://osvita.ua/school/method/80683/
10. Використання ігрових технологій навчання на уроці інформатики у початковій школі як чинник реалізації Концепції нової української школи [Електронний ресурс]. – Режим доступу: https://naurok.com.ua/vikoristannya-igrovih-tehnologiy-na-urokah-informatiki-340312.html

Додаток А

Відомість матеріалів кваліфікаційної роботи

<table>
<tr><td></td><td></td><td colspan="3">Позначення</td><td>Найменування</td><td>Кільк. аркушів</td><td>Примітка</td></tr>
<tr><td>1</td><td></td><td colspan="3"></td><td></td><td></td><td></td></tr>
<tr><td>2</td><td></td><td colspan="3"></td><td>Документація</td><td></td><td></td></tr>
<tr><td>3</td><td></td><td colspan="3"></td><td></td><td></td><td></td></tr>
<tr><td>4</td><td></td><td colspan="3">ІСТ.КР 24.__.ДА.ПЗ</td><td>Пояснювальна записка</td><td>87</td><td></td></tr>
<tr><td>5</td><td></td><td colspan="3"></td><td></td><td></td><td></td></tr>
<tr><td>6</td><td></td><td colspan="3"></td><td>Презентація</td><td></td><td></td></tr>
<tr><td>7</td><td></td><td colspan="3"></td><td></td><td></td><td></td></tr>
<tr><td>8</td><td></td><td colspan="3"></td><td>Диск CD з презентацією</td><td>1</td><td></td></tr>
<tr><td></td><td></td><td colspan="3"></td><td colspan="3" rowspan="2">ІСТ.КР 24.__.ДА.ПЗ</td></tr>
<tr><td></td><td></td><td colspan="3"></td></tr>
<tr><td>Зм.</td><td>Ар-куш</td><td>№ докум</td><td>Підпис</td><td>Дата</td></tr>
<tr><td colspan="2">Розроб.</td><td>К. Д. Чала</td><td></td><td></td><td rowspan="5">Матеріал кваліфікаційної роботи</td><td>Літ</td><td>Аркуш</td><td>Арку-шів</td></tr>
<tr><td colspan="2">Керівник</td><td>Сергєєва К. Л.</td><td></td><td></td><td>Н</td><td>1</td><td>1</td></tr>
<tr><td colspan="2">Рецензент</td><td>Ширін А. Л.</td><td></td><td></td><td colspan="3" rowspan="3">НТУ «ДП»,
12; 126-20-1</td></tr>
<tr><td colspan="2">Н.контр.</td><td>Г. М. Коротенко</td><td></td><td></td></tr>
<tr><td colspan="2">Зав. каф.</td><td>В. В. Гнатушенко</td><td></td><td></td></tr>
</table>

Додаток Б

Реалізація main.py

```
import pygame
import random
import pygame_textinput

pygame.init()

win = pygame.display.set_mode((1000, 550))


pygame.display.set_caption("Sweet math")


pngRaw = pygame.image.load('Images/png.png')
png = pygame.transform.scale(pngRaw, (150, 90))
kvitkaRaw = pygame.image.load('Images/kvitka.png')
kvitkaPic = pygame.transform.scale(kvitkaRaw, (120, 120))
kvitka1Pic = pygame.transform.scale(kvitkaRaw, (170, 120))
heart0 = pygame.transform.scale(pygame.image.load('Images/Heart0.png'), (90,
40))
heart1 = pygame.transform.scale(pygame.image.load('Images/Heart1.png'), (130,
50))
heart2 = pygame.transform.scale(pygame.image.load('Images/Heart2.png'), (130,
50))
heart3 = pygame.transform.scale(pygame.image.load('Images/Heart3.png'), (130,
50))
heartpics = [heart1, heart2, heart3, heart0]
menu1raw = pygame.image.load('Images/menu.jpg')
menu1 = pygame.transform.scale(menu1raw, (1000, 550))
matheus1raw = pygame.image.load('Images/matheus1.jpg')
matheus1 = pygame.transform.scale(matheus1raw, (1000, 550))
matheus2raw = pygame.image.load('Images/matheus2.jpg')
matheus2 = pygame.transform.scale(matheus2raw, (1000, 550))
matheus3raw = pygame.image.load('Images/matheus3.jpg')
matheus3 = pygame.transform.scale(matheus3raw, (1000, 550))
matheus4raw = pygame.image.load('Images/matheus4.jpg')
matheus4 = pygame.transform.scale(matheus4raw, (1000, 550))
uley1raw = pygame.image.load('Images/uley.png')
uley1 = pygame.transform.scale(uley1raw, (150, 150))
uley2raw = pygame.image.load('Images/uley.png')
uley2 = pygame.transform.scale(uley2raw, (150, 150))
uley3raw = pygame.image.load('Images/uley.png')
uley3 = pygame.transform.scale(uley3raw, (150, 150))
uley4raw = pygame.image.load('Images/uley.png')
uley4 = pygame.transform.scale(uley4raw, (150, 150))

# імпортування всіх звуків
collisionsound = [pygame.mixer.Sound('Audio/1.wav'),
pygame.mixer.Sound('Audio/2.wav'), pygame.mixer.Sound('Audio/3.wav'),
```

```
                    pygame.mixer.Sound('Audio/4.wav'),
pygame.mixer.Sound('Audio/5.wav'), pygame.mixer.Sound('Audio/6.wav'),
                    pygame.mixer.Sound('Audio/7.wav'),
pygame.mixer.Sound('Audio/8.wav'), pygame.mixer.Sound('Audio/9.wav')]
bulletsound = pygame.mixer.Sound('Audio/Gunshot.wav')
selectsound = pygame.mixer.Sound('Audio/selectsound.wav')
explosion = pygame.mixer.Sound('Audio/explosion.wav')
music = pygame.mixer.music.load('Audio/BgMusic.mp3')
pygame.mixer.music.play(-1)

clock = pygame.time.Clock()
textinput = pygame_textinput.TextInput()


def menu():
    def redrawGameWindow():
        win.blit(menu1, (0, 0))
        screenText()
        pygame.display.update()

    def screenText():

        font = pygame.font.Font('ARCADECLASSIC.TTF', 150)
        text = font.render('Sweetmath', 1, (150, 0, 24))
        win.blit(text, (500 - ((text.get_width() / 2) - 20), 40))




        font2 = pygame.font.Font('ARCADECLASSIC.TTF', 50)
        text2 = font2.render('Play', 1, (150, 0, 24))
        win.blit(text2, (500 - ((text2.get_width() / 2) - 8), 262))


    run = True
    while run:
        clock.tick(27)

        keys = pygame.key.get_pressed()

        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                quit()

            if event.type == pygame.MOUSEBUTTONDOWN:
                if mouse[0] >= 430 and mouse[0] <= 570 and mouse[1] >= 280 and
mouse[1] <= 320:
                    selectsound.play()
                    instructions()

        mouse = pygame.mouse.get_pos()
```

```
        redrawGameWindow()


def instructions():
    def redrawGameWindow():
        win.blit(matheus1, (0, 0))
        screenText()
        pygame.display.update()

    def screenText():


        font = pygame.font.Font('ARCADECLASSIC.TTF', 150)
        text = font.render('Sweetmath', 1, (150, 0, 24))
        win.blit(text, (500 - ((text.get_width() / 2) - 20), 40))


        font2 = pygame.font.Font('ARCADECLASSIC.TTF', 60)
        text2 = font2.render('Instructions', 1, (150, 0, 24))
        win.blit(text2, (500 - (text2.get_width() / 2), 150))


        font3 = pygame.font.Font('ARCADECLASSIC.TTF', 30)
        text3 = font3.render((
                'Welcome' + '   ' + 'to' + '   ' + 'Sweetmath' + '   ' + 'Your'
+ '   ' + 'mission' + '   ' + 'is' + '   ' + 'to' + '   ' + 'collect'),
            1, (231, 84, 128))
        text4 = font3.render((
                'all' + '   ' + 'the' + '   ' + 'honey' + '   ' + 'by' + '   ' +
'answering' + '   ' + 'math' + '   ' + 'questions'),
            1, (231, 84, 128))
        text5 = font3.render((
                'Use' + '   ' + 'the' + '   ' + 'up' + '   ' + 'and' + '   ' +
'down' + '   ' + 'arrow' + '   ' + 'keys' + '   ' + 'to' + '   ' + 'move' +
'   ' + 'the' + '   ' + 'bee'),
            1, (231, 84, 128))
        text6 = font3.render((
                'and' + '   ' + 'the' + '   ' + 'space' + '   ' + 'bar' + '   '
+ 'to' + '   ' + 'shoot' + '   ' + 'Press' + '   ' + 'space ' + '   ' + 'bar'+
'   ' + 'to ' + '   ' + 'shoot'),
            1, (231, 84, 128))
        text7 = font3.render((
                'Good' + '   ' + 'luck' + '   ' + 'friend'),
            1, (231, 84, 128))
        win.blit(text3, (500 - (text3.get_width() / 2), 220))
        win.blit(text4, (500 - (text4.get_width() / 2), 240))
        win.blit(text5, (500 - (text5.get_width() / 2), 260))
        win.blit(text6, (500 - (text6.get_width() / 2), 280))
        win.blit(text7, (500 - (text7.get_width() / 2), 300))

        pygame.draw.rect(win, (0, 100, 200), [(500 - (140 / 2)), 440, 130, 40])
```

```
        font4 = pygame.font.Font('ARCADECLASSIC.TTF', 50)
        text4 = font4.render('Next', 1, (150, 0, 24))
        win.blit(text4, (500 - (text4.get_width() / 2), 435))

    run = True
    while run:
        clock.tick(27)
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                quit()
            if event.type == pygame.MOUSEBUTTONDOWN:
                if mouse[0] >= 430 and mouse[0] <= 570 and mouse[1] >= 440 and
mouse[1] <= 480:
                    selectsound.play()
                    ageinputscreen()

        mouse = pygame.mouse.get_pos()

        redrawGameWindow()

def ageinputscreen():
    def redrawGameWindow():
        win.blit(matheus1, (0, 0))
        screenText()
        win.blit(textinput.get_surface(), (450, 293))
        ageChecker()
        pygame.display.update()

    def screenText():


        font = pygame.font.Font('ARCADECLASSIC.TTF', 150)
        text = font.render('Sweetmath', 1, (150, 0, 24))
        win.blit(text, (500 - ((text.get_width() / 2) - 20), 40))


        font2 = pygame.font.Font('ARCADECLASSIC.TTF', 80)
        text2 = font2.render('Enter your age', 1, (150, 0, 24))
        win.blit(text2, (500 - (text2.get_width() / 2), 180))

        pygame.draw.rect(win, (0, 100, 200), [(500 - (140 / 2)), 440, 130, 40])
        pygame.draw.rect(win, (255, 255, 255), [(500 - (200 / 2)), 300, 200,
80])


        font4 = pygame.font.Font('ARCADECLASSIC.TTF', 50)
        text4 = font4.render('Next', 1, (150, 0, 24))
        win.blit(text4, (500 - (text4.get_width() / 2), 435))

    def ageChecker():
        if ageVar == 1:
            font4 = pygame.font.Font('ARCADECLASSIC.TTF', 50)
```

```
            text4 = font4.render('You are too young!', 1, (231, 84, 128))
            win.blit(text4, (500 - (text4.get_width() / 2), 380))

        if ageVar == 2:
            font4 = pygame.font.Font('ARCADECLASSIC.TTF', 50)
            text4 = font4.render('You are too old!', 1, (231, 84, 128))
            win.blit(text4, (500 - (text4.get_width() / 2), 380))

        if ageVar == 3:
            font4 = pygame.font.Font('ARCADECLASSIC.TTF', 50)
            text4 = font4.render('Please   enter   a   valid   age!', 1, (255,
255, 255))
            win.blit(text4, (500 - (text4.get_width() / 2), 380))

    ageVar = 0
    run = True
    while run:
        clock.tick(27)
        events = pygame.event.get()
        for event in events:
            if event.type == pygame.QUIT:
                quit()
            if event.type == pygame.MOUSEBUTTONDOWN:
                if mouse[0] >= 430 and mouse[0] <= 570 and mouse[1] >= 440 and
mouse[1] <= 480:
                    selectsound.play()
                    try:
                        userAge = int(textinput.get_text())

                        if userAge >= 5 and userAge <= 13:
                            levelscreen(userAge)
                        else:
                            if userAge < 5:
                                ageVar = 1
                            if userAge > 13:
                                ageVar = 2

                    except ValueError:
                        ageVar = 3

        textinput.update(events)

        mouse = pygame.mouse.get_pos()

        redrawGameWindow()

def levelscreen(userAge):
    def redrawGameWindow(userAge):
        win.blit(matheus1, (0, 0))
        win.blit(uley1, (105, 323))
        win.blit(uley2, (315, 323))
```

```
        win.blit(uley3, (530, 323))
        win.blit(uley4, (740, 323))
        screenText(userAge)
        pygame.display.update()

    def screenText(userAge):


        font = pygame.font.Font('ARCADECLASSIC.TTF', 150)
        text = font.render('Sweetmath', 1, (150, 0, 24))
        win.blit(text, (500 - ((text.get_width() / 2) - 20), 40))


        font2 = pygame.font.Font('ARCADECLASSIC.TTF', 80)
        text2 = font2.render('Select Level', 1, (150, 0, 24))
        win.blit(text2, (500 - (text2.get_width() / 2), 180))


        font3 = pygame.font.Font('ARCADECLASSIC.TTF', 80)
        text3 = font3.render('1', 1, (150, 0, 24))
        text3a = font3.render('2', 1, (150, 0, 24))
        text3b = font3.render('3', 1, (150, 0, 24))
        text3c = font3.render('4', 1, (150, 0, 24))
        win.blit(text3, (180 - (text3.get_width() / 2), 400 -
(text3.get_height() / 2)))
        win.blit(text3a, (393.3 - (text3a.get_width() / 2), 400 -
(text3.get_height() / 2)))
        win.blit(text3b, (606.6 - (text3b.get_width() / 2), 400 -
(text3.get_height() / 2)))
        win.blit(text3c, (819.9 - (text3c.get_width() / 2), 400 -
(text3.get_height() / 2)))

        if userAge >= 5 and userAge <= 7:
            k = 100
        if userAge >= 8 and userAge <= 10:
            k = 307
        if userAge >= 11 and userAge <= 12:
            k = 517
        if userAge == 13:
            k = 730

        font2 = pygame.font.Font('ARCADECLASSIC.TTF', 20)
        text2 = font2.render('Recommended level', 1, (231, 84, 128))
        win.blit(text2, (k, 300))

    run = True
    while run:
        clock.tick(27)
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                quit()
```

```
            if event.type == pygame.MOUSEBUTTONDOWN:
                if mouse[0] >= 137.5 and mouse[0] <= 222.5 and mouse[1] >= 357.5
and mouse[1] <= 442.5:
                    level = 1
                    selectsound.play()
                    mainGame(level)
                if mouse[0] >= 350.8 and mouse[0] <= 435.8 and mouse[1] >= 357.5
and mouse[1] <= 442.5:
                    level = 2
                    selectsound.play()
                    mainGame(level)
                if mouse[0] >= 564.1 and mouse[0] <= 649.1 and mouse[1] >= 357.5
and mouse[1] <= 442.5:
                    level = 3
                    selectsound.play()
                    mainGame(level)
                if mouse[0] >= 777.4 and mouse[0] <= 862.4 and mouse[1] >= 357.5
and mouse[1] <= 442.5:
                    level = 4
                    selectsound.play()
                    mainGame(level)

        mouse = pygame.mouse.get_pos()

        redrawGameWindow(userAge)


def mainGame(level):
    class player(object):
        def __init__(self, x, y, width, height):
            self.x = x
            self.y = y
            self.width = width
            self.height = height
            self.vel = 10
            self.hitbox = (self.x + 10, self.y, 100, 90)
            self.heartCounter = 3

        def draw(self, win):
            win.blit(png, (self.x, self.y))
            self.hitbox = (self.x + 10, self.y, 100, 90)


        def hit(self):
            (random.choice(collisionsound)).play()
            pygame.display.update()
            bee.heartCounter -= 1


            if bee.heartCounter == 2:
                heartInGame.heartpicsCurrent = heartpics[1]
            if bee.heartCounter == 1:
```

```
            heartInGame.heartpicsCurrent = heartpics[0]
        if bee.heartCounter == 0:
            heartInGame.heartpicsCurrent = heartpics[3]
            print("game over")


        i = 0
        while i < 3:
            pygame.time.delay(10)
            i += 1
            for event in pygame.event.get():
                if event.type == pygame.QUIT:
                    i = 301
                    pygame.quit()

class projectile(object):
    def __init__(self, x, y, radius, colour):
        self.x = x
        self.y = y
        self.radius = radius
        self.colour = colour
        self.vel = 70

    def draw(self, win):
        pygame.draw.circle(win, self.colour, (self.x, self.y), self.radius)

class kvitnnyk(object):
    def __init__(self, x, y, width, height, end):
        self.x = x
        self.y = y
        self.width = width
        self.height = height
        self.path = [x, end]
        self.vel = 8
        self.hitbox = (self.x, self.y + 10, 120, 100)
        self.visible = True
        self.number = 0

    def draw(self, win):
        if self.visible:
            if level == 1:
                win.blit(kvitkaPic, (self.x, self.y))
            if level == 2 or level == 3 or level ==4:
                win.blit(kvitka1Pic, (self.x, self.y))
            self.hitbox = (self.x, self.y + 10, 120, 100)

            font2 = pygame.font.Font('ARCADECLASSIC.TTF', 80)
            text = font2.render((str(self.number)), 1, (255, 255, 255))
            win.blit(text, (self.x + 30, self.y + 30))


    def hit(self):
```

```
            self.visible = False

    class heart(object):
        def __init__(self, x, y, width, height):
            self.x = x
            self.y = y
            self.width = width
            self.height = height
            self.visible = True
            self.heartpicsCurrent = heartpics[2]

        def draw(self, win):
            if self.visible:
                win.blit(self.heartpicsCurrent, (self.x, self.y))

    class question(object):
        def draw(self, a, b):
            self.a = a
            self.b = b
            font2 = pygame.font.Font('ARCADECLASSIC.TTF', 80)
            text = font2.render((str(self.a) + " x " + str(self.b)), 1, (255,
255, 255))
            win.blit(text, (500 - (text.get_width() / 2), 450))

    class score(object):
        def draw(self, score):
            self.score = score
            font2 = pygame.font.Font('ARCADECLASSIC.TTF', 40)
            text = font2.render(("Score " + str(score)), 1, (255, 255, 255))
            win.blit(text, (830, 5))
            text2 = font2.render(('Exit to menu'), 1, (255, 255, 255))
            win.blit(text2, (10, 5))

    class gameover(object):
        def draw(self, win, score):
            if bee.heartCounter == 0:
                font3 = pygame.font.Font('ARCADECLASSIC.TTF', 120)
                text = font3.render(('GAME OVER'), 1, (231, 84, 128))
                win.blit(text, ((500 - text.get_width() / 2), 50))
                font4 = pygame.font.Font('ARCADECLASSIC.TTF', 60)
                text2 = font4.render(('SCORE ' + str(score)), 1, (231, 84, 128))
                win.blit(text2, ((500 - text2.get_width() / 2), 150))
                text3 = font4.render(
                    ('Press' + '   ' + 'enter' + '   ' + 'to' + '   ' + 'go' +
'   ' + 'to' + '   ' + 'menu'), 1,
                    (231, 84, 128))
                win.blit(text3, ((500 - text3.get_width() / 2), 300))



    def redrawGameWindow(a, b, score, scorequestionVisible, level):
```

```
    if level == 1:
        win.blit(matheus1, (0, 0))
    if level == 2:
        win.blit(matheus2, (0, 0))
    if level == 3:
        win.blit(matheus3, (0, 0))
    if level == 4:
        win.blit(matheus4, (0, 0))

    bee.draw(win)
    kvitkaTop.draw(win)
    kvitkaMiddle.draw(win)
    kvitkaBottom.draw(win)
    heartInGame.draw(win)


    if scorequestionVisible == True:
        questionGame.draw(a, b)
        scoreGame.draw(score)

    gameoverGame.draw(win, score)

    for bullet in bullets:
        bullet.draw(win)

    pygame.display.update()


bee = player(104, 235, 100, 100)
kvitkaTop = kvitnnyk(800, 40, 100, 100, 300)
kvitkaMiddle = kvitnnyk(800, 190, 100, 100, 300)
kvitkaBottom = kvitnnyk(800, 340, 100, 100, 300)
heartInGame = heart(30, 490, 30, 30)
questionGame = question()
scoreGame = score()
gameoverGame = gameover()
bullets = []
run = True


if level == 1:
    number1 = random.randint(1, 3)
    number2 = random.randint(1, 3)
if level == 2:
    number1 = random.randint(1, 6)
    number2 = random.randint(1, 6)
if level == 3:
    number1 = random.randint(1, 9)
    number2 = random.randint(1, 9)
if level == 4:
    number1 = random.randint(1, 19)
    number2 = random.randint(1, 19)
```

```
scorenumber = 0
whichKvitochka = random.randint(1, 3)


if whichKvitochka == 1:
    kvitkaTop.number = number1 * number2
    kvitkaMiddle.number = random.randint(1, 30)
    kvitkaBottom.number = random.randint(1, 30)
if whichKvitochka == 2:
    kvitkaTop.number = random.randint(1, 30)
    kvitkaMiddle.number = number1 * number2
    kvitkaBottom.number = random.randint(1, 30)
if whichKvitochka == 3:
    kvitkaTop.number = random.randint(1, 30)
    kvitkaMiddle.number = random.randint(1, 30)
    kvitkaBottom.number = number1 * number2
w = 1
scorequestionVisible = True

while run:
    clock.tick(27)

    if bee.heartCounter == 0:
        if kvitkaTop.x == 1200:
            kvitkaTop.vel = 0
            kvitkaMiddle.vel = 0
            kvitkaBottom.vel = 0
        bee.vel = 0
        if keys[pygame.K_KP_ENTER] or keys[pygame.K_RETURN]:
            menu()
        scorequestionVisible = False

    kvitkaTop.x -= kvitkaTop.vel
    kvitkaMiddle.x -= kvitkaMiddle.vel
    kvitkaBottom.x -= kvitkaBottom.vel

    if kvitkaTop.x == -176:
        kvitkaTop.x = 1200
        kvitkaTop.visible = True

        if level == 1:
            number1 = random.randint(1, 3)
            number2 = random.randint(1, 3)
        if level == 2:
            number1 = random.randint(1, 6)
            number2 = random.randint(1, 6)
        if level == 3:
            number1 = random.randint(1, 9)
            number2 = random.randint(1, 9)
        if level == 4:
            number1 = random.randint(1, 19)
```

```
            number2 = random.randint(1, 19)

        whichKvitochka = random.randint(1, 3)


    if kvitkaMiddle.x == -176:
        kvitkaMiddle.x = 1200
        kvitkaMiddle.visible = True
    if kvitkaBottom.x == -176:
        kvitkaBottom.x = 1200
        kvitkaBottom.visible = True

        if whichKvitochka == 1:
            kvitkaTop.number = number1 * number2

            if level == 1:
                kvitkaMiddle.number = random.randint(1, 9)
                kvitkaBottom.number = random.randint(1, 9)
            if level == 2:
                kvitkaMiddle.number = random.randint(1, 40)
                kvitkaBottom.number = random.randint(1, 40)
            if level == 3:
                kvitkaMiddle.number = random.randint(1, 90)
                kvitkaBottom.number = random.randint(1, 90)
            if level == 4:
                kvitkaMiddle.number = random.randint(1, 400)
                kvitkaBottom.number = random.randint(1, 400)

        if whichKvitochka == 2:
            kvitkaMiddle.number = number1 * number2
            if level == 1:
                kvitkaTop.number = random.randint(1, 9)
                kvitkaBottom.number = random.randint(1, 9)
            if level == 2:
                kvitkaTop.number = random.randint(1, 40)
                kvitkaBottom.number = random.randint(1, 40)
            if level == 3:
                kvitkaTop.number = random.randint(1, 90)
                kvitkaBottom.number = random.randint(1, 90)
            if level == 4:
                kvitkaTop.number = random.randint(1, 400)
                kvitkaBottom.number = random.randint(1, 400)

        if whichKvitochka == 3:
            kvitkaBottom.number = number1 * number2
            if level == 1:
                kvitkaMiddle.number = random.randint(1, 9)
                kvitkaTop.number = random.randint(1, 9)
            if level == 2:
                kvitkaMiddle.number = random.randint(1, 40)
                kvitkaTop.number = random.randint(1, 40)
            if level == 3:
```

```
                    kvitkaMiddle.number = random.randint(1, 90)
                    kvitkaTop.number = random.randint(1, 90)
                if level == 4:
                    kvitkaMiddle.number = random.randint(1, 400)
                    kvitkaTop.number = random.randint(1, 400)

        if kvitkaTop.visible == True:
            if bee.hitbox[1] < kvitkaTop.hitbox[1] + kvitkaTop.hitbox[3] and
bee.hitbox[1] + \
                    bee.hitbox[
                        3] > kvitkaTop.hitbox[1]:
                if bee.x == kvitkaTop.x:
                    bee.hit()

        if kvitkaMiddle.visible == True:
            if bee.hitbox[1] < kvitkaMiddle.hitbox[1] + kvitkaMiddle.hitbox[3]
and bee.hitbox[1] + \
                    bee.hitbox[3] > kvitkaMiddle.hitbox[1]:
                if bee.x == kvitkaMiddle.x:
                    bee.hit()

        if kvitkaBottom.visible == True:
            if bee.hitbox[1] < kvitkaBottom.hitbox[1] + kvitkaBottom.hitbox[3]
and bee.hitbox[1] + \
                    bee.hitbox[3] > kvitkaBottom.hitbox[1]:
                if bee.x == kvitkaBottom.x:
                    bee.hit()

        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                quit()

            if event.type == pygame.MOUSEBUTTONDOWN:
                if mouse[0] >= 10 and mouse[0] <= 240 and mouse[1] >= 10 and
mouse[1] <= 50:
                    selectsound.play()
                    menu()
        mouse = pygame.mouse.get_pos()

        for bullet in bullets:

            if whichKvitochka == 1:
                if kvitkaTop.visible == True:
                    if bullet.y - bullet.radius < kvitkaTop.hitbox[1] +
kvitkaTop.hitbox[
                        3] and bullet.y + bullet.radius > \
                            kvitkaTop.hitbox[1]:
                        if bullet.x + bullet.radius > kvitkaTop.hitbox[0] and
bullet.x - bullet.radius < \
                                kvitkaTop.hitbox[0] + \
                                kvitkaTop.hitbox[2]:
```

```
                            bullets.pop(bullets.index(bullet))
                            scorenumber += 1
                            explosion.play()
                            kvitkaTop.hit()
            if whichKvitochka == 2:
                if kvitkaMiddle.visible == True:
                    if bullet.y - bullet.radius < kvitkaMiddle.hitbox[1] +
kvitkaMiddle.hitbox[
                        3] and bullet.y + bullet.radius > \
                            kvitkaMiddle.hitbox[1]:
                        if bullet.x + bullet.radius > kvitkaMiddle.hitbox[0] and
bullet.x - bullet.radius < \
                                kvitkaMiddle.hitbox[
                                    0] + kvitkaMiddle.hitbox[2]:
                            bullets.pop(bullets.index(bullet))
                            scorenumber += 1
                            explosion.play()
                            kvitkaMiddle.hit()

            if whichKvitochka == 3:
                if kvitkaBottom.visible == True:
                    if bullet.y - bullet.radius < kvitkaBottom.hitbox[1] +
kvitkaBottom.hitbox[
                        3] and bullet.y + bullet.radius > \
                            kvitkaBottom.hitbox[1]:
                        if bullet.x + bullet.radius > kvitkaBottom.hitbox[0] and
bullet.x - bullet.radius < \
                                kvitkaBottom.hitbox[
                                    0] + kvitkaBottom.hitbox[2]:
                            bullets.pop(bullets.index(bullet))
                            scorenumber += 1
                            explosion.play()
                            kvitkaBottom.hit()

            if bullet.x < 1000 and bullet.x > 0:
                bullet.x += bullet.vel
            else:
                bullets.pop(bullets.index(bullet))


        keys = pygame.key.get_pressed()

        if keys[pygame.K_SPACE]:
            if len(bullets) < 1:
                bullets.append(
                    projectile(round(bee.x + bee.width + 10 // 2),
                               round(bee.y + bee.height // 2), 7, (117, 237,
255)))  # creates bullet
                bulletsound.play()

        if keys[pygame.K_UP] and bee.y > 50:
            bee.y -= bee.vel
```

```
        elif keys[pygame.K_DOWN] and bee.y < 400:
            bee.y += bee.vel

        redrawGameWindow(number1, number2, scorenumber, scorequestionVisible,
level)  # викликає функцію перемальовування


menu()
pygame.quit()
```

Додаток В

pygame_textinput.py

```
import os.path

import pygame
import pygame.locals as pl

pygame.font.init()


class TextInput:

    def __init__(
            self,
            initial_string="",
            font_family="ARCADECLASSIC.TTF",
            font_size=100,
            antialias=True,
            text_color=(0, 0, 0),
            cursor_color=(0, 0, 1),
            repeat_keys_initial_ms=400,
            repeat_keys_interval_ms=35,
            max_string_length=-1,
            password=False):



        self.antialias = antialias
        self.text_color = text_color
        self.font_size = font_size
        self.max_string_length = max_string_length
        self.password = password
        self.input_string = initial_string

        if not os.path.isfile(font_family):
            font_family = pygame.font.match_font(font_family)

        self.font_object = pygame.font.Font(font_family, font_size)


        self.surface = pygame.Surface((1, 1))
        self.surface.set_alpha(0)


        self.keyrepeat_counters = {}
        self.keyrepeat_intial_interval_ms = repeat_keys_initial_ms
        self.keyrepeat_interval_ms = repeat_keys_interval_ms


        self.cursor_surface = pygame.Surface((int(self.font_size / 20 + 1),
self.font_size))
```

```
        self.cursor_surface.fill(cursor_color)
        self.cursor_position = len(initial_string)
        self.cursor_visible = True
        self.cursor_switch_ms = 500  # /|\
        self.cursor_ms_counter = 0

        self.clock = pygame.time.Clock()

    def update(self, events):
        for event in events:
            if event.type == pygame.KEYDOWN:
                self.cursor_visible = True


                if event.key not in self.keyrepeat_counters:
                    if not event.key == pl.K_RETURN:
                        self.keyrepeat_counters[event.key] = [0, event.unicode]

                if event.key == pl.K_BACKSPACE:
                    self.input_string = (
                        self.input_string[:max(self.cursor_position - 1, 0)]
                        + self.input_string[self.cursor_position:]
                    )


                    self.cursor_position = max(self.cursor_position - 1, 0)
                elif event.key == pl.K_DELETE:
                    self.input_string = (
                        self.input_string[:self.cursor_position]
                        + self.input_string[self.cursor_position + 1:]
                    )

                elif event.key == pl.K_RETURN:
                    return True

                elif event.key == pl.K_RIGHT:

                    self.cursor_position = min(self.cursor_position + 1,
len(self.input_string))

                elif event.key == pl.K_LEFT:

                    self.cursor_position = max(self.cursor_position - 1, 0)

                elif event.key == pl.K_END:
                    self.cursor_position = len(self.input_string)

                elif event.key == pl.K_HOME:
                    self.cursor_position = 0

                elif len(self.input_string) < self.max_string_length or
self.max_string_length == -1:
```

```
                    self.input_string = (
                        self.input_string[:self.cursor_position]
                        + event.unicode
                        + self.input_string[self.cursor_position:]
                    )
                    self.cursor_position += len(event.unicode)

            elif event.type == pl.KEYUP:

                if event.key in self.keyrepeat_counters:
                    del self.keyrepeat_counters[event.key]


        for key in self.keyrepeat_counters:
            self.keyrepeat_counters[key][0] += self.clock.get_time()


            if self.keyrepeat_counters[key][0] >=
self.keyrepeat_intial_interval_ms:
                self.keyrepeat_counters[key][0] = (
                    self.keyrepeat_intial_interval_ms
                    - self.keyrepeat_interval_ms
                )

                event_key, event_unicode = key, self.keyrepeat_counters[key][1]
                pygame.event.post(pygame.event.Event(pl.KEYDOWN, key=event_key,
unicode=event_unicode))


        string = self.input_string
        if self.password:
            string = "*" * len(self.input_string)
        self.surface = self.font_object.render(string, self.antialias,
self.text_color)


        self.cursor_ms_counter += self.clock.get_time()
        if self.cursor_ms_counter >= self.cursor_switch_ms:
            self.cursor_ms_counter %= self.cursor_switch_ms
            self.cursor_visible = not self.cursor_visible

        if self.cursor_visible:
            cursor_y_pos =
self.font_object.size(self.input_string[:self.cursor_position])[0]

            if self.cursor_position > 0:
                cursor_y_pos -= self.cursor_surface.get_width()
            self.surface.blit(self.cursor_surface, (cursor_y_pos, 0))

        self.clock.tick()
        return False
```

```
    def get_surface(self):
        return self.surface

    def get_text(self):
        return self.input_string

    def get_cursor_position(self):
        return self.cursor_position

    def set_text_color(self, color):
        self.text_color = color

    def set_cursor_color(self, color):
        self.cursor_surface.fill(color)

    def clear_text(self):
        self.input_string = ""
        self.cursor_position = 0



if __name__ == "__main__":
    pygame.init()


    textinput = TextInput()

    screen = pygame.display.set_mode((1000, 200))
    clock = pygame.time.Clock()

    while True:
        screen.fill((225, 225, 225))

        events = pygame.event.get()
        for event in events:
            if event.type == pygame.QUIT:
                exit()


        textinput.update(events)

        screen.blit(textinput.get_surface(), (10, 10))

        pygame.display.update()
        clock.tick(30)
```

ДОДАТОК Г

ВІДГУК

на кваліфікаційну роботу бакалавра

"Розробка ігрового застосунку для розвитку дітей мовою Python"

студентки групи 126-20-1 Чалої Карини Дмитрівни

1. Метою кваліфікаційної роботи є створення інтерактивного ігрового застосунку, спрямованого на покращення математичних навичок у дітей шляхом надання їм можливості вивчати та практикувати математичні завдання через ігрове середовище, що стимулює учнів до активного навчання та розвитку.

2. Завдання та зміст кваліфікаційної роботи відповідають основній меті – оцінці знань і ступеня підготовленості здобувачки вищої освіти за спеціальністю 126 "Інформаційні системи та технології".

3. Тема роботи представляється актуальною і обумовлена важливістю використання ігрових застосунків для навчання математики дітей, що сприяє розвитку їх когнітивних і соціальних навичок.

4. Для досягнення мети кваліфікаційної роботи Чала К.Д. підготувала проєктне рішення ігрового застосунку з декількома рівнями складності, призначеного для розвитку дітей, розробила інтерактивне ігрове середовище з використанням Python і відповідних бібліотек (таких як Pygame).

5. Оформлення пояснювальної записки виконано, в основному, відповідно до діючих стандартів і нормативних вимог.

6. У якості зауваження слід відзначити недостатній опис зв’язків між компонентами розробленого застосунку.

Незважаючи на зазначений недолік, кваліфікаційна робота заслуговує оцінки "_____".

Керівник,

доцент кафедри ІТКІ К.Л. Сергєєва

ДОДАТОК Д
РЕЦЕНЗІЯ

на кваліфікаційну роботу бакалавра

"Розробка ігрового застосунку для розвитку дітей мовою Python"

студентки групи 126-20-1 Чалої Карини Дмитрівни

1. Тема кваліфікаційної роботи, присвячена розробці ігрового застосунку для розвитку дітей, є актуальною і спрямована на навчання гравців швидко вирішувати приклади з математики.

2. У рецензованій роботі Чала К.Д. реалізувала логічні принципи та правила взаємодії об'єктів гри з використанням мови програмування Python, використала бібліотеку графіки Pygame для створення графічного ігрового інтерфейсу застосунку, обробки подій та анімації, також використала звукові ефекти та музику, оформила графічний дизайн.

3. Практична значимість результатів кваліфікаційної роботи Чалої К.Д. полягає у можливості впровадження результатів у школах, дитячих садках, позашкільних установах, бібліотеках, на домашніх уроках, в літніх таборах, на освітніх платформах і мобільних додатках. Зроблено змістовні висновки.

4. Робота цілком відповідає вимогам, що пред'являються до кваліфікаційних робіт рівня бакалавра.

Зауваження: у тексті пояснювальної записки відсутній опис вимог до апаратної складової роботи застосунку, а також не наведені числові значення показників ефективності.

Беручи до уваги зазначене зауваження, кваліфікаційна робота в цілому може бути відзначена оцінкою "_____".

Рецензент,

доцент кафедри ПЗКС А.Л. Ширін