МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ КРИВОРІЗЬКИЙ ДЕРЖАВНИЙ ПЕДАГОГІЧНИЙ УНІВЕРСИТЕТ Кафедра інформатики та прикладної математики

Реєстраційний №\_\_\_\_\_\_\_\_\_\_ «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_2016р.

**КУРСОВА РОБОТА З ПРОГРАМУВАННЯ**

**РОЗРОБКА ІГРОВОЇ ПРОГРАМИ «МОРСЬКИЙ БІЙ» В ПРОГРАМІ ІНТЕРПРИТАТОР**

студента І курсу групи І-15

фізико-математичного факультету

напрям підготовки

6.040302 Інформатика

**Пірогов Владислава Миколайовича**

Керівник асис.,д. ф. в г. і. т.

Дубан Роман Миколайович

Оцінка:

Національна шкала\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Шкала ECTS\_\_\_ Кількість балів\_\_\_

Члени комісії \_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

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

Кривий Ріг 2016

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ КРИВОРІЗЬКИЙ ДЕРЖАВНИЙ ПЕДАГОГІЧНИЙ УНІВЕРСИТЕТ**

Кафедра інформатики та прикладної математики

Завдання

на курсову роботу

студенту\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ група\_\_\_\_\_\_\_\_

*(прізвище, ім’я, по-батькові)*

|  |
| --- |
| Термін здачі роботи \_\_\_ \_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_ |

|  |
| --- |
| **Тема: РОЗРОБКА ІГРОВОЇ ПРОГРАМИ «МОРСЬКИЙ БІЙ» В ПРОГРАМІ ІНТЕРПРИТАТОР** |
| 1. Аналіз предметної області й постановка задачі. |
| 1.1 Мотивація |
| 1.2 Аналіз аналогів та предметної області. |
| 1.3 Особливості реалізації |
| 2. Проектування задачі. |
| 2.1 User Story. |
| 2.2 Use Case. |
| 2.3 Алгоритми задачі. |
| 2.4 Інтерфейси. |
| 3. Архітектура задачі. |
| 3.1 Дані. |
| 3.2 Алгоритми реалізації. |
| 3.3 Функції модулі. |
| 4. Тестування. |
| 4.1 Unit Test. |
| 4.2 User Test |
| 4.3 Інструкція, Презентація. |
|  |

Зміст

[Вступ 16](#_Toc451775978)

[1. Аналіз предметної області й постановка задачі. 16](#_Toc451775979)

[1.1 Мотивація 16](#_Toc451775980)

[1.2 Аналіз аналогів та предметної області. 16](#_Toc451775981)

[1.3 Особливості реалізації 16](#_Toc451775982)

[2. Проектування задачі. 16](#_Toc451775983)

[2.1 User Story. 16](#_Toc451775984)

[2.2 Use Case. 16](#_Toc451775985)

[2.3 Алгоритми задачі. 16](#_Toc451775986)

[2.4 Інтерфейси. 16](#_Toc451775987)

[3. Архітектура задачі. 16](#_Toc451775988)

[3.1 Дані. 16](#_Toc451775989)

[3.2 Алгоритми реалізації. 16](#_Toc451775990)

[3.3 Функції модулі. 16](#_Toc451775991)

[4. Тестування. 16](#_Toc451775992)

[4.1 Unit Test. 16](#_Toc451775993)

[4.2 User Test 16](#_Toc451775994)

[4.3 Інструкція. 16](#_Toc451775995)

[Висновки 16](#_Toc451775996)

[Список використаних джерел 16](#_Toc451775997)

**Вступ**

Дана програма слугує для розвитку реакцій та розваги користувача . Та гарного проведення часу .

Оскільки в даній програмі було використано об'єктно-орієнтоване програмування , через те що даний метод програмування використовує об’єкти - це дозволяє використовувати інкапсуляцію та наслідування класів. Як наслідок ми здатні використати структури даних та інші характеристики створені в батьківському класі об’єктів в інших класах шо утворені від батьківського. В такий спосіб ми зменшуємо кількість рядків , та покращимо сам код програми та його загальний об’єм.

Сама програма представляє з себе гру «Пожежники» . Програма створює меню в якому користувач запускає гру. За умовою гравець повинен встигати ловити усіх падаючих людей . Інакше, якщо він не впіймає хоч одного чоловіка то ,користувач програє. Також користувач не повинен ловити жодного камінця , у випадку якщо гравець впіймає хоч один камінь то він втрачає бали . Так у випадку втрати всіх балів гравець програє .Гравець виграє тільки у випадку набору 100 балів .В графічному інтерфейсі гри є індикатор шо показує кількість балів.

1. **Аналіз предметної області й постановка задачі.**
   1. Мотивація. Короткий опис.
   2. Аналіз аналогів та предметної області.

1) MICO MACO.

Офіційний сайт : <http://www.igames.com.ua/FlashGame.aspx?name=mico_maco>

Користувач повинен грати проти комп’ютера. В самій грі присутня велика кількість ігрових об’єктів та їх ефектів , як , наприклад , при падінні на землю так і у випадку якщо гравець зловить об’єкт. Наприклад об’єкт банан: в випадку якщо користувач піймає банан він отримує бали , але якщо користувач не зловить об’єкт банан , то банан впаде на землю и у випадку коли гравець наступить на банан гравця перемістить за ігрове поле і він втратить одну з життів . Так в грі присутня велика кількість анімацій пов’язаних з взаємодією гравця та ігрових об’єктів . Управління в самій грі на відміну від проекту є окрема кнопка шо виконує функцію переходу до режиму «ловіння». В цьому режими персонаж не здатен рухатись. На відміну від моєї програми. Графічний інтерфейс досить гарно оформлений в ньому присутній індикатор життів та показник балів набрати гравцем . Перед запуском гри , користувач попаде на головне меню , з нього можливо запустити гру , переглянути лист рекордів , та присутня інструкція . Та сама гра не є зрозумілою, адже правила та ефекти після взаємодії з об’єктами , ні де пояснюється . Також в грі є звукові ефекти , так постійно головному меню грає музика та при натисненні на кнопки грає звуковий ефект . В самій грі грає вже інша музика . Але нема можливості її налаштувати , зробити тихіше , або взагалі вимкнути і з часом ці звуки починають відволікати та злити . Управління зв’язано з стрілками ( в ліво , в право ) та з кнопкою ( пробіл ) .

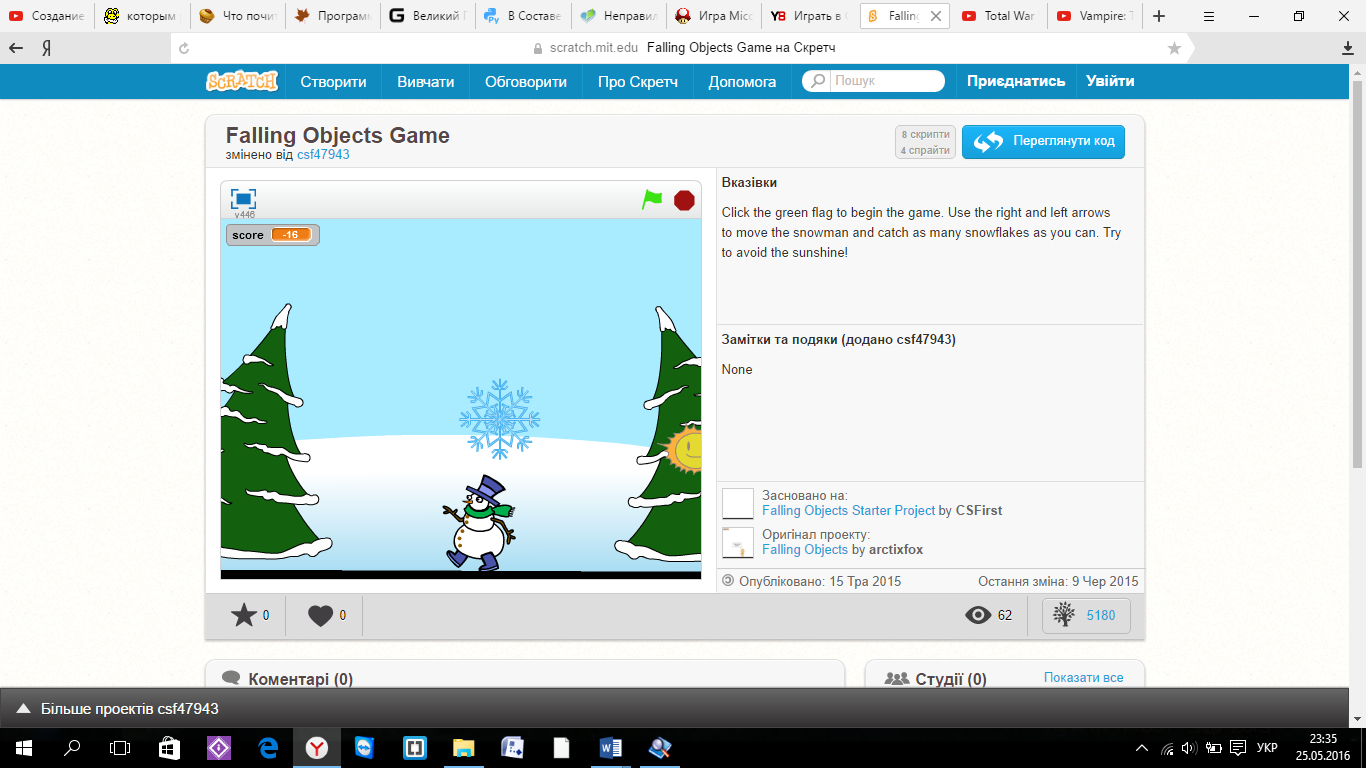


2) СНІГОВИК

Офіційний сайт :

[**https://scratch.mit.edu/projects/62494276/**](https://scratch.mit.edu/projects/62494276/)

Досить проста гра. Присутній досить простий ігровий інтерфейс в якому є відображення кількості набраних балів гравця . Управління пов’язано з стрілками ( в ліво , в право ) . До її недоліків можна віднести те що вона нескінчена, в ній неможливо програти або виграти.

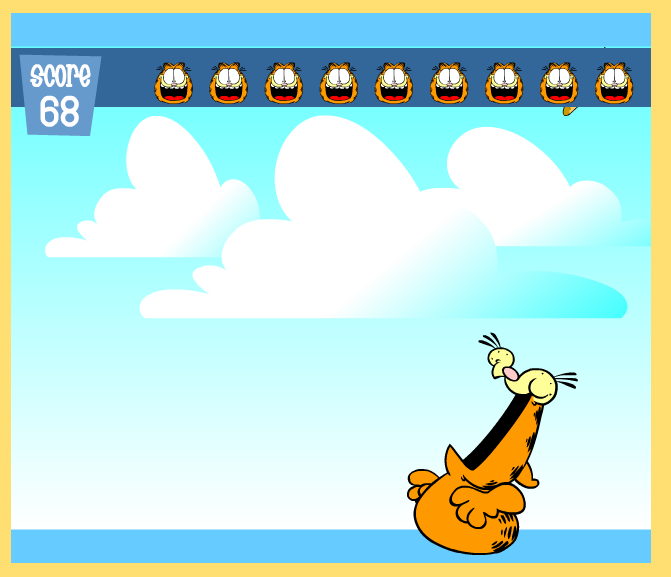


3) GARFIELD : LASAGNA FROM HEAVEN

Офіційний сайт :

<http://ru.y8.com/games/garfield_lasagna_from_heaven>

Ця гра досить гарно оформлена . В самій грі присутня велика кількість ігрових об’єктів , сам інтерфейс представлений дуже просто , в ньому є показник часу до кінця рівня . Та в грі присутня анімація , так кіт одкриває рот при взаємодій з об’єктом . Але управління в грі є незручним , тому що воно пов’язано з мишкою та не завжди реагує на дії користувача .



1.3 **Особливості реалізації.**

**Опис**

Ця курсова робота студента першого курсу , що є результатом самостійного опрацювання матеріалу ООП.   
Данна гра має безліч аналогів , та багато версій гри , але дана робота матиме найпростіший вигляд.

Дана програма служить для розваги користувача.

Якщо користувач запускає гру , йому відкриється меню звідки і буде запушена гра .

Основний опис умови гри представляє з себе наступне . З верхньої частини вікна рухається вниз певна кількість об’єктів Y, гравець повинен "зловити" всі об'єкти. Інакше гравець програє . Також в грі присутні об’єкти пастки , які ловити заборонено , кожен пійманий об’єкт пастка віднімає певну кількість балів , якщо кількість балів менша за нуль або рівне – гравець програє . Управління в грі здійснюється за допомогою . Далі йде обсяг проекту .

**Обсяг проекту:**

* Ігрове меню з трьома кнопками :

“Start”

“Record List”

“Quit”

* Вікно з фоном
* Лічильник часу
* Об'єкт (X): Графічно відображається: батут розміром 1/8 вікна. Рухається по осі.Ох
* Об'єкт (Y): Графічно відображається: падаючий людина розміром дорівнює батуту. Рухається зверху в низ
* Об'єкт пастка: Графічно відображається: камінь розміром 1/8 вікна. Рухається зверху в низ.

**Дії:**

Натиснути на кнопку “Start” – Почнеться гра

Натиснути на кнопку “Record List” – Покаже список набраних балів

Натиснути на кнопку “Quit” – Закриває меню

Спіймати - нижні координати Об’єкта Y або Об'єкта пастка співпадають з верхніми координатами Об’єкта Х.

Слідство: якщо зловити Об’єкта Y він знищитися, а гравець отримує один бал . Якщо координати співпадають з кінцем вікна гра закінчитися. Якщо зловити Об'єкт пастка гравець втрачає 15 балів . У випадку втрати всіх балів гравець програє .Умови перемоги: гравець повинен набрати 100 балів.

1. **Проектування задачі.**
   1. User Story.

Призначені для користувача історії (англ. User Story) - спосіб опису вимог до розроблюваної системі, сформульованих як одне або більше пропозицій на повсякденному або діловою мовою користувача. Призначені для користувача історії використовуються гнучкими методологіями розробки програмного забезпечення для специфікації вимог (разом з приймальною іспитаніяміruen). Кожна призначена для користувача історія обмежена в розмірі та складності. Часто історія пишеться на маленькій паперовій картці. Це гарантує, що вона не стане занадто великий. В екстремальних програмуванні призначені для користувача історії пишуться користувачами (замовниками) системи. У методології SCRUM - пишуться або схвалюються роллю власника продукту (англ. Product Owner). Для замовників (користувачів) призначені для користувача історії є основним інструментом впливу на розробку програмного забезпечення.

Призначені для користувача історії - швидкий спосіб документувати вимоги клієнта, без необхідності розробляти великі формалізовані документи і згодом витрачати ресурси на їх підтримку. Мета користувальницьких історій полягає в тому, щоб бути в змозі оперативної і без накладних витрат реагувати на швидко змінюються вимоги реального світу.

Призначена для користувача історія залишається неофіційними визначенням вимог, поки відсутня процедура приймального тестування. Перш, ніж реалізовувати призначену для користувача історію, клієнт повинен визначити відповідну приймальню процедуру, щоб гарантувати що цілі користувальницької історії були досягнуті.

З даного визначення , для кращого розуміння програми та її більш точної розробки . Було створено наступні User Story :

1. Я як користувач хочу мати меню гри з кнопками .
2. Я як користувач хочу мати кнопку початку гри для того щоб почати гру
3. Я як користувач хочу мати кнопку для перегляду результатів гри .
4. Я як користувач хочу мати кнопку для виходу з гри .
5. Я як користувач хочу мати вікно де гра буде відображатися для того щоб почати гру.
6. Я як користувач хочу мати задній фон .
7. Я як користувач хочу мати об’єкт люди для того щоб мати можливість набирати бали .
8. Я як користувач хочу щоб об’єкт люди рухались того щоб мати можливість зловити їх .
9. Я як користувач хочу мати об’єкт батут для того щоб мати можливість ловити людей та набирати бали .
10. Я як користувач хочу мати можливість рухати батут для того щоб мати можливість ловити людей та набирати бали .
11. Я як користувач хочу мати об’єкти пастка для того щоб в грі можна було програти .
12. Я як користувач хочу мати інформаційну лінію для того щоб бачити кількість балів .
13. Я як користувач хочу мати можливість вийти з гри для того щоб завершити гру.

2.2 Use Case.

Прецедент використання (англ. use case) - в розробці програмного забезпечення та системному проектуванні це опис поведінки системи, коли вона взаємодіє з кимось (або чимось) із зовнішнього середовища. Система може відповідати на зовнішні запити Актора (англ. Actor) (може застосовуватися термін актантом [1]), може сама виступати ініціатором взаємодії. Іншими словами, сценарій використання описує, «хто» і «що» може зробити з розглянутої системою, або що система може зробити з «ким» або «чому». Методика сценаріїв використання застосовується для виявлення вимог до поведінки системи, відомих також як призначені для користувача і функціональні вимоги.

У системному проектуванні сценарії використання застосовуються на більш високому рівні ніж при розробці програмного забезпечення, часто представляючи мети зацікавлених осіб або місії. На стадії аналізу вимог сценарії використання можуть бути перетворені в ряд детальних вимог та задокументовані за допомогою діаграм вимог SysML або інших подібних механізмів.

З даного визначення , для кращого розуміння програми та її більш точної розробки . Було створено наступні Use Case :

№1 :

Назва: Камінь, що падає

Опис: Камінь, що падає з гори екрана

Передумова: гра триває

Тригер : Користувач набрав достатню кількість очок

Основний сценарій : Камінь падає з гори екрана, користувач ухиляється від нього

Результат: Користувач залишається при своїх балах, гра продовжується.

Додатковий сценарій : Камінь падає з гори екрана, користувач не ухиляється від нього

Додатковий результат: Користувач втрачає 15 балів , якщо у користувача було менше 15 балів, гра завершується

№2 :

Назва: Людина, що падає

Опис: Людина, що падає з гори екрана

Передумова: гра почалась

Тригер : гра почалась

Основний сценарій : Людина падає з гори екрана, користувач її ловить

Результат: Користувач отримує один бал, гра продовжується.

Додатковий сценарій : Людина падає з гори екрана, користувач не ловить її

Додатковий результат: гра завершується

№3 :

Назва: Батут, що рухається

Опис: Батут, що рухається вздовж екрану

Передумова: Гра почалась

Тригер : Гра почалась

Основний сценарій : Гравець тисне на клавіши(стрілка ліво, стрілка право), батут рухається

Результат: Користувач отримує можливість ловити об’єкти .

Додатковий сценарій : Гравець не тисне на клавіши

Додатковий результат: Батут не рухається

№4 :

Назва: Меню запуску

Опис: Меню, в якому користувач обирає складність та починає гру

Передумова: Гра була запущена

Основний сценарій : Гравець обирає складність та запускає гру

Результат: Запускається гра з обраним рівнем складності

Додатковий сценарій : Гравець вимикає меню

Додатковий результат: Гра закривається

№5:

Назва: Вивід балів в інформаційному рядку

Опис: Рядок, в якому показуються бали гравця

Передумова: Гра була запущена

Тригер : Гра почалась

Основний сценарій : Користувач отримує бали

Результат: Бали відображаються на інформаційному рядку

2.3Алгоритми Задач

Алгоритм (латінізов. Algorithmi за араб. Ім'ям узб. Математика аль-Хорезмі) - набір інструкцій, Які опісують порядок Дій Виконавця, щоб досягті результату розв'язання задачі за скінченну Кількість Дій; система правил виконан дискретного процесса, яка досягає поставленої мети за скінченній годину. Для візуалізації алгоритмів часто Використовують блок-схеми.

Для комп'ютерних програм алгоритм є списком деталізованіх інструкцій, что реалізують процес обчислення, Який, починаючі з початкових стану, відбувається через послідовність логічніх станів, яка завершується кінцевім станом. Перехід з попередня до следующего стану НЕ обов'язково детермінованій - деякі алгоритми могут містіті елементи віпадковості.

На основі даних Use Case і User Story було розроблені такі алгоритми задач:

№1 Меню гри

Початок : Гра запушена → меню запускається → користувач натиснув на кнопку “Record List” → Відкривається список результатів → користувач натиснув на кнопку “Quit” → Меню закривається → Користувач натиснув на кнопку “Start” → Запускається головний цикл гри.

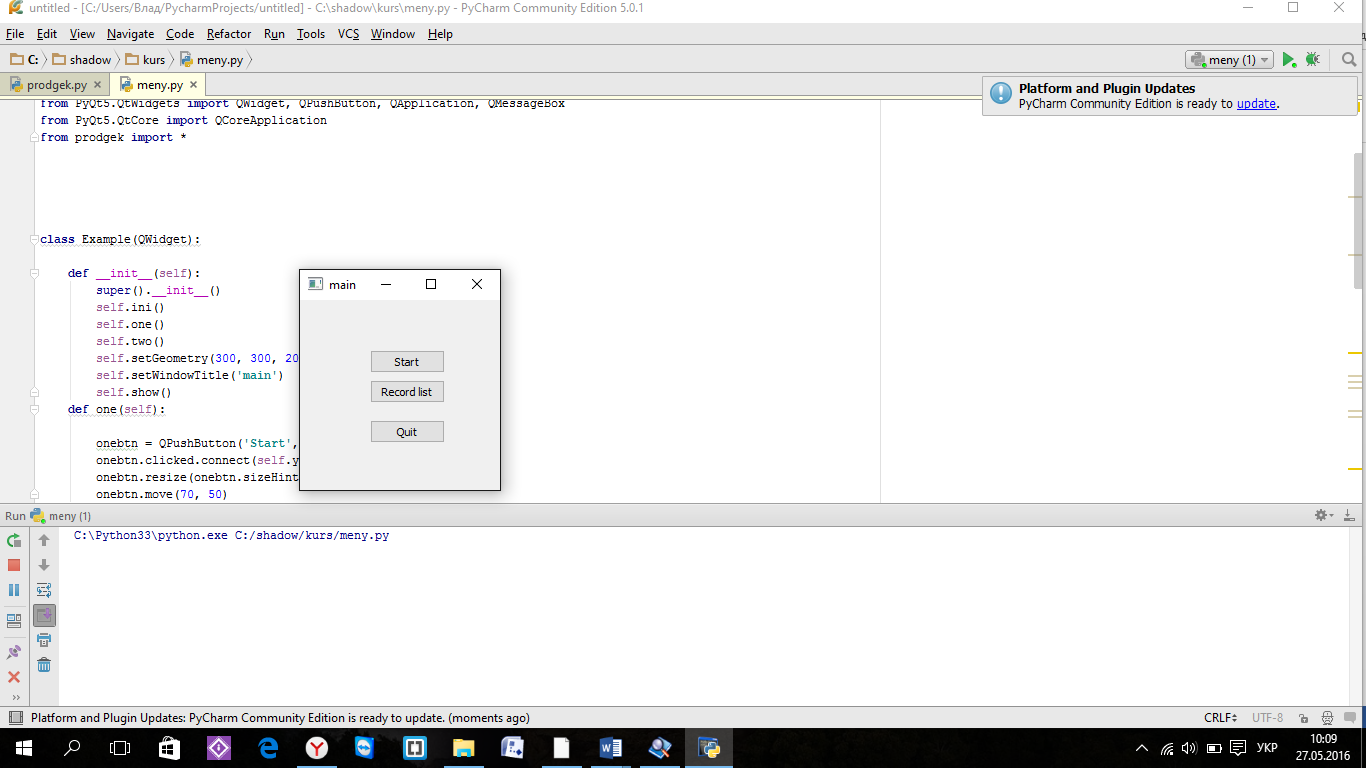
№2 Люди

Початок : Користувач натиснув на кнопку “Start” → За границею екрана випадково генеруються об’єкти люди → На кожній ітерацій циклу до координат по вісі ОY об’єкта додається деяке число → Якщо об’єкт зіткнувся з батутом , він знищується та гравець отримує один бал → Якщо об’єкт зіткнувся з краєм поля гра завершиться.

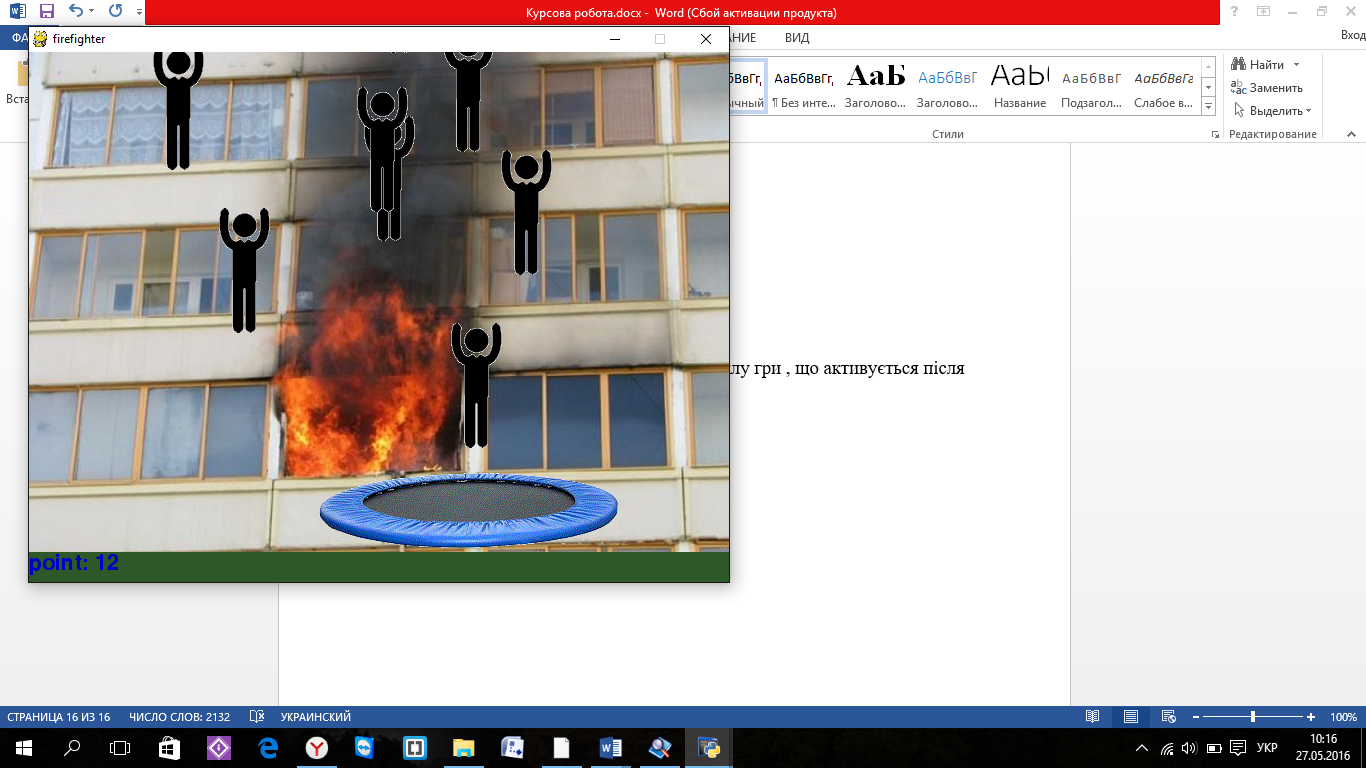
2.4 Інтерфейси

На основі даних Use Case і User Story та розроблених алгоритмів задач було створено такі інтерфейси :

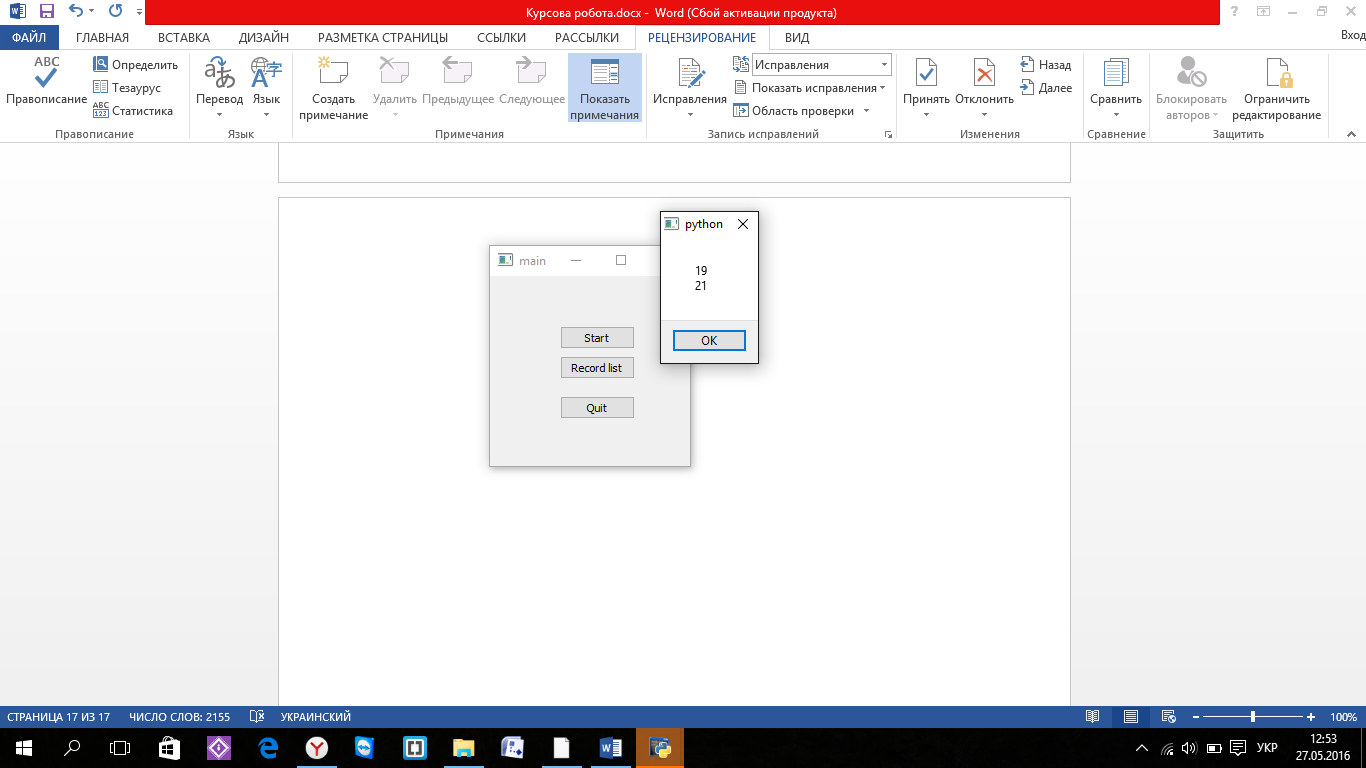
1)Розроблено інтерфейс меню за допомогою бібліотеки PyQt5 , що має три кнопки : “Start”; “Record List”; “Quit”.



2) Розроблено інтерфейс головного циклу гри , код гри було створено за допомогою бібліотеки Pygame та об’єктно-орієнтованого програмування , активується після натиснення кнопки “Start” в головному меню :



3) Розроблено інтерфейс списку результатів з використанням бібліотеки PyQt5:



1. Архітектура задачі
   1. Дані
   2. Алгоритми реалізації
2. Головний цикл гри :



* 1. Функції модулі.