TOWER DEFENCE

Małgorzata Krupanek, Remigiusz Kozicki

1. **ZARYS SYSTEMU**
    - **Technologia**
      Do wykonania projektu planujemy użyć biblioteki PyGame.
    - **Opis**
      Głównym celem gry jest powstrzymanie przeciwników przed dotarciem do końca wyznaczonej ścieżki. Gra jest typu Tower Defence zatem gracz ma do wyboru różne rodzaje wieżyczek, bomb, min i wiele innych środków prewencyjnych pomagających w powstrzymaniu przeciwników. Jeśli jakikolwiek przeciwników przejdzie całą ścieżkę i dotrze do jej końca gracz traci jedno ze swoich trzech żyć. Wraz z utratą ostatniego życia gracz przerywa. Na początku gracz otrzymuje startową pulę pieniędzy umożliwiającą zakup pierwszych podstawowych wieżyczek. Wraz z upływem gry, gracz odblokowuje nowe rodzaje broni, wieżyczek, które może kupić za kredyty otrzymywane za zniszczenie przeciwników.

2. **HARMONOGRAM PRACY**
    - 14.03 - harmonogram prac i zarys struktury programu
    - 14.03 - 29.03
        - Wstępne zaprojektowanie silnika gry
        - Utworzenie podstawowych ekranów aplikacji
    - 29.03 - 25.04
        - Zaprojektowanie mapy, przeciwników i dostępnych broni
        - Zaprogramowanie ruchu przeciwników
    - 25.04 - 09.05
        - Dodawanie broni na mapę
        - Zabijanie przeciwników
        - System zakupów, rund i walki
    - 09.05 - 23.05
        - Odejmowanie żyć
        - Ekran końca gry
    - 23.05 - Oddanie projektu po raz pierwszy
    - 23.05 - 20.06
        - Ostateczne szlify
    - 20.06 - Ostateczne oddanie projektu

3. **Podział pracy**
    - Małgorzata Krupanek:
        - WavesMenager.py: zarządzanie kolejnymi rundami i falami przeciwników
        - WeaponMenager.py: zarządzanie wieżami
        - Bar.py: zarządzanie sklepem i systemem zakupów
        - MainGame.py: refaktoryzacja kodu odpowiedzialengo za główną częśc gry do osobnej klasy, odpowiednie wyszukiwanie wrogów i zabijania przez wieżyczki, oraz system stawiania broni na mapę.
        - MyFirstGame.py: refaktoryzacja oraz dadawanie poszczególnych funkcjonalności
        - Map.py: stworzenie systemu przechowywania informacji dla map i scierzek przeciwników
        - Target.py: tworzenie przeciwników i ich system poruszania sie
        - Tower.py: struktura klasy Tower i system tworzenia wieżyczek
        - Main.py: plik odpalający grę
    - Remigiusz Kozicki:
        - WeaponMenager.py: zarządzanie bombami
        - Bar.py: wizualny aspekt komponentu sklepu
        - EndGame.py: ekran końcowy gry
        - LevelsMenu.py: system wyboru poszczególnych map
        - MainGame.py: obsługa “Game Loop’a” i pygame.events oraz system zabijania oraz strzelania wieży, część funkcjonalności bomb.
        - Meny.py: ekran Menu
        - MyFirstGame.py: zarządzanie ekranami gry, i game loop’y
        - Bomb.py: klasa odpowiedzialna za tworzenie bomb(min)
        - Bullte.py: klasa odpowiedzialna za wystrzeliwane pociski
        - Button.py: klasa odpowiedzialna za tworzenie przycisków
        - Crosshair.py: klasa odpowiedzialna za kursor
4. **Użyte technologie:**
   W projekcje w głównej mierze korzystano z biblioteki “pygame” umożliwiającej sprawne wyświetlanie ekranów gry oraz zarządzanie obiektami które są na nich wyświetlane. Biblioteka jest przeznaczona do tworzenia gier stąd jej użycie.
    - Zalety biblioteki pygame:
        - szybkie i łatwe tworzenie ekranów gry
        - prostota w tworzeniu obiektów i manipulacją ich na ekranie
        - wiele wbudowanych funkcji ułatwiających implementację funkcjonalności gry.
        - duża ilość materiałów referencyjnych

- Wady biblioteki pygame:
    - wiele zależności pomiędzy różnymi typami obiektów biblioteki pygame a wbudowanych funkcji utrudniające niedoświadczonemu użytkownikowi pisanie gry
    - …?

5. **Co można zmienić lub usprawnić?:**
   Projekt jest naszym pierwszym spotkaniem z Biblioteką pygame co prawdopodobnie skutkowało w różne niedociągnięcia w niektórych aspektach projektu, które bardziej doświadczony twórca gier by zauważył. Poprawić można by zarządzanie poszczególnymi ekranami gier.