# Projekt z Programowania Obiektowego - Gra Tanky

Karol Machoś

24 czerwca 2020

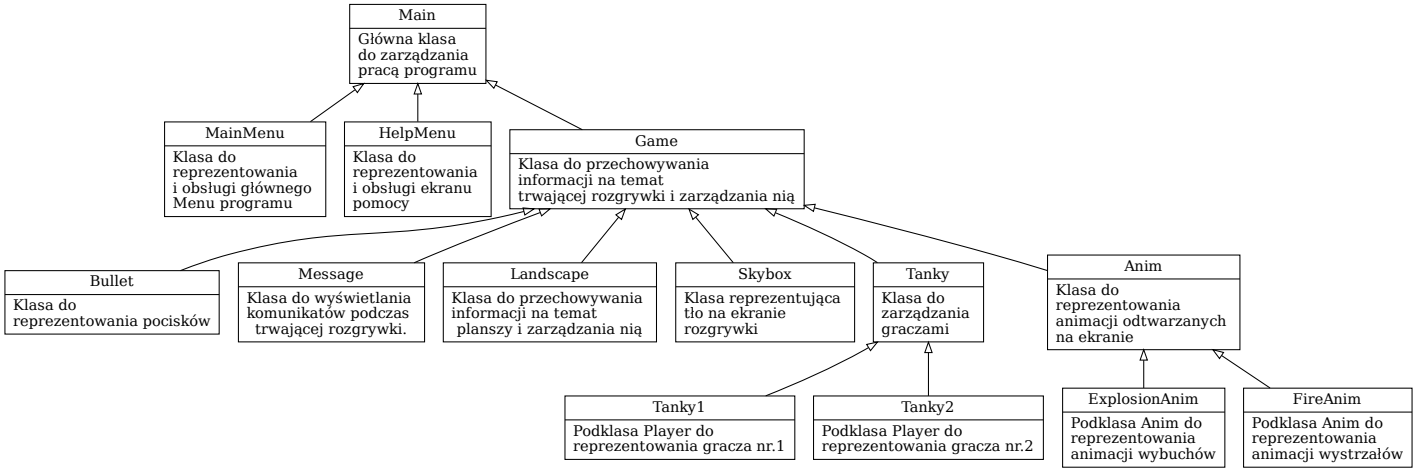
## Opis:

Gra będzie napisanym w pythonie3 (przy uzyciu biblioteki pygame) programem wzorującym się na takich grach jak np. ”Scorched Earth” czy ”Worms”. Rozgrywka będzie się toczyć (turowo) na losowo generowanej planszy 2D, gdzie dwoch graczy będzie mierzyć się w pojedynku czołgów. Każdy z graczy w czasie swojej tury, będzie mógł wystrzelić pocisk, oraz przemiescic się o pewien dystans. Po wystrzale zaczynać się będzie tura drugiego gracza. Zadaniem kazdego z graczy jest zniszczenie czołgu przeciwnika, przez trafienie go odpowiednią ilość razy swoimi pociskami. Dodatkowo pociski uderzając w teren, będą go niszczyć, co umożliwi graczom korzystanie z niego jak z osłon.

## Koniec rozgrywki:

Po zniszczeniu jednego z czołgów wyswietlony zostanie odpowiedni ekran, informujący o tym który gracz wygrał, i umożliwiający rozpoczęcie rozgrywki od nowa.

## Finalny diagram klas:




## Zmiany które zaszły względem początku projektu:

Największa zmiana jakiej dokonałem na etapie programowania mojego projektu, to zrezygnowanie z GameOverScreen na rzecz menu pomocy. Stwierdziłem, że ekran końca rozgrywki nie jest potrzebny (krótki komunikat o zwycięzcy jest moim zdaniem wystarczający). Dodatkowo dodałem podklasę Anim - TraceAnim, celem rysowania na ekranie toru lotu pocisku.

## Dokumentacja:

W swojej finalnej postaci mój projekt zawiera następujące klasy:

## Main

Główna klasa do zarządzania pracą całego programu. Ma ona następujące składowe:

- game_obj: Obiekt klasy Game do stworzenia rozgrywki.
- help_obj: Obiekt klasy HelpMenu do stworzenia ekranu pomocy.
- menu_obj: Obiekt klasy MainMenu do stworzenia ekranu głównegu menu.
- running: Zmienna boolowska do sprawdzenia czy program powinien dalej działać.
- screen: Obiekt reprezentujący ekran do wyświetlania całości programu.
- run(): Funkcja do obsługi działania programu.

## MainMenu

Klasa do reprezentowania głównego menu. Ma ona następujące składowe:

- BtnEffect: Obiekt przechowujący efekt dźwiękowy klikięcia na przycisk.
- HelpBtnTexture: Obiekt przechowujący teksturę przycisku prowadzącego do HelpMenu.
- Helprect: Obiekt przechowujący wymiary i pozycję przycisk HelpBtn.
- StartBtnTexture: Obiekt przechowujący teksturę przycisku prowadzącego do rozgrywki.
- Startrect: Obiekt przechowujący wymiary i pozycję przycisk StartBtn.
- TitleScreenTexture: Obiekt przechowujący teksturę głównego menu.
- running: Zmienna boolowska do sprawdzenia czy MainMenu powinno dalej działać.
- return_val: Zmienna której wartość informuje Main, do jakiego elementu przejść po wyjściu z MainMenu.
- screen: Obiekt reprezentujący ekran do wyświetlania całości programu.
- run(): Funkcja do obsługi działania MainMenu.
- draw(screen): Funkcja do obsługi wyświetlania MainMenu.
- events(): Funkcja do obsługi wejścia przekazywanego do programu w czasie działania MainMenu.

## HelpMenu

Klasa do reprezentowania ekranu pomocy. Ma ona następujące składowe:

- HelpScreenTexture: Obiekt zawierający teksturę ekranu pomocy.
- running: Zmienna boolowska do sprawdzenia czy HelpMenu powinno dalej działać.
- screen: Obiekt reprezentujący ekran do wyświetlania całości programu.
- run(): Funkcja do obsługi działania HelpMenu.
- draw(screen): Funkcja do obsługi wyświetlania HelpMenu.
- events(): Funkcja do obsługi wejścia przekazywanego do programu w czasie działania MainMenu.

## Game

Klasa do zarządzania rozgywką. Ma ona następujące składowe:

- clock : Obiekt do ograniczenia framerate'u gry do 60fps.
- anim: Lista zawierająca wszystkie animacje do wyświetlania.
- bullets: Lista zawierająca wszystkie pociski do obsługi.
- drawable_objects: Lista zawierająca pozostałe obiekty do wyświetlania.
- freeze: Zmienna boolowska do przechowywania informacji czy rozgrywka powinna zostać wstrzymana.
- next_player: Zmienna zawierająca indeks w liście players następnego gracza (który zostanie aktywowany w następnej turze)
- players: Lista zawierająca obiekty graczy.
- player1: Obiekt będący graczem numer1.
- player2: Obiekt będący graczem numer2.
- running: Zmienna boolowska do sprawdzenia czy rozgrywka powinna dalej działać.
- skybox: Obiekt zawierający teksturę nieba.
- terrain: Obiekt zawierający teren.
- timeout: Zmienna do odliczania klatek przed wyjsciem z rozgrywki. Służy do opóźnienia wyjscia z rozgrywki po jej zakończeniu.
- screen: Obiekt reprezentujący ekran do wyświetlania całości programu.
- end(looser): Funkcja do zakończenia rozgrywki w momencie wygranej jednego z graczy.
- run(): Funkcja do obsługi działania MainMenu.
- draw(screen): Funkcja do obsługi wyświetlania MainMenu.
- events(): Funkcja do obsługi wejścia przekazywanego do programu w czasie działania MainMenu.

## Tank

Klasa do reprezentowania graczy w rozgrywce. Ma ona następujące składowe:

- EngineEffect: Obiekt będący efektem dźwiękowym poruszania się czołgu.
- GunEffect: Obiekt będący efektem dźwiękowym podnoszenia i opuszczania lufy czołgu.
- ShotEffect: Obiekt będący efektem dźwiękowym wystrzału.
- active: Zmienna boolowska zawierająca informację na temat tego czy czołg powinien reagować na wejście użytkownika, czy nie.
- game: Obiekt zawierający rozgrywkę w obrębie której działają gracze.
- hp: Zmienna zawierająca poziom zdrowia czołgu.
- range: Zmienna zawierająca zasięg ruchu czołgu.
- muzzle: Obiekt zawierający pozycję wylotu lufy.
- r: Zmienna zawierająca kąt ustawienia lufy.
- r_speed: Zmienna zawierająca prędkość obrotu lufy (W danym momencie).
- x: Zmienna zawierająca pozycję x czołgu na ekranie.
- y: Zmienna zawierająca pozycję y czołgu na ekranie.
- x_speed: Zmienna zawierająca prędkość ruchu czołgu w kierunku równoległym do podłoża (W danym momencie).
- action(event): Funkcja do obsługi akcji czołgu w oparciu o wejście użytkownika.
- shoot(): Funkcja do oddania strzału przez czołg.
- toggle(): Funckja do aktywowania lub dezaktywowania czołgu

## Tank1

Podklasa Tank do reprezentowania gracza numer 1. Ma ona następujące składowe (oprócz tych odziedziczonych z Tank):

- Body: Obiekt zawierający teksturę kadłubu czołgu.
- Gun_original: Obiekt zawierający oryginalną teksturę lufy czołgu.
- Gun: Obiekt zawierający teksturę lufy czołgu po zastosowaniu rotacji.
- EnginePlaying: Zmienna boolowska informująca o tym czy dzwięk silnika powinien być odtwarzany.
- GunPlaying: Zmienna boolowska informująca o tym czy dzwięk ruchu lufy powinien być odtwarzany.
- Track: Zmienna zawierająca indeks wersji gąsienic do wyświetlania.
- Tracks: Lista zawierająca tekstury gąsienic (animacja ruchu czołgu)
- draw(screen): Funkcja do rysowania czołgu na ekranie
- update(terrain): Funkcja do aktualizowania pozycji czołgu.

## Tank2

Podklasa Tank do reprezentowania gracza numer 2. Analogiczna implementacja do Tank1.

## Anim

Klasa do reprezentowania animacji. Ma ona następujące składowe:

- ExpFrames: Lista zawierająca załadowane klatki animacji wybuchu
- ShotFrames: Lista zawierająca załadowane klatki animacji wystrzału
- TraceFrames: Lista zawierająca załadowane klatki animacji śladu pocisku
- frame: Zmienna przechowująca indeks aktualnie wyświetlanej klatki w liście
- texture: Surface będące aktualnie wyświetlaną klatką
- r: Zmienna określająca rotacje klatki na ekranie
- x: Zmienna przechowująca pozycję x animacji na ekranie
- y: Zmienna przechowująca pozycję y animacji na ekranie

## ExplosionAnim

Podklasa Anim do obsługi animacji wybuchu. Ma ona następujące składowe (oprócz tych odziedziczonych z Anim:

- draw(screen): Funkcja służacą do obsługi wyświetlania klatki animacji na ekranie
- update(terrain): Funkcja służąca do zmiany klatki na następną w czasie życia animacji

### ShotAnim

Podklasa Anim do obsługi animacji wystrzału. Ma ona następujące składowe (oprócz tych odziedziczonych z Anim:

- draw(screen): Funkcja służacą do obsługi wyświetlania klatki animacji na ekranie
- update(terrain): Funkcja służąca do zmiany klatki na następną w czasie życia animacji

### TraceAnim

Podklasa Anim do obsługi animacji wystrzału. Ma ona następujące składowe (oprócz tych odziedziczonych z Anim:

- draw(screen): Funkcja służacą do obsługi wyświetlania klatki animacji na ekranie
- update(terrain): Funkcja służąca do zmiany klatki na następną w czasie życia animacji

### Bullet

Klasa do reprezentowania pocisków. Ma ona następujące składowe:

- alive: Zmienna boolowska do przechowywania informacji czy pocisk nadal powinien istnieć
- boom: Zmienna boolowska do przechowywania informacji czy doszło do wybuchu (kolizji pocisku z terenem)
- texture: Surface zawierające oryginalną teksturę pocisku
- draw_texture: Zmienna przechowująca teksturę po zastosowaniu na niej rotacji
- og_r: Zmienna określająca oryginalny kąt wystrzału
- r: Zmienna okręślająca aktualną rotację pocisku
- x: Zmienna przechowująca pozycję x animacji na ekranie
- y: Zmienna przechowująca pozycję y animacji na ekranie
- gravity: Zmienna określająca grawitację działającą na pocisk
- l_speed: Zmienna określająca prędkość początkową pocisku
- y_speed: Zmienna określająca prędkość pocisku w kierunku prostopadłym do podłoża
- draw(screen): Funkcja odpowiedzialna za wyświetlanie obiektu na ekranie
- update(terrain): Funkcja odpowiedzialna za aktualizowanie pocisku (jego prędkości, rotacja, itp)

### Landscape

Klasa do reprezentowania terenu podczas rozgrywki. Ma ona następujące składowe:

- BoomEffect: Obiekt będący efektem dźwiękowym wybuchu.
- grid_size: Zmienna przechowująca ile pikseli reprezentuje jeden punkt heightmapy
- height_map: Lista zawierająca na indeksach, wartości będące wysokościami w odpowiednich punktach.
- min_height: Zmienna przechowująca minimalną możliwą wysokość terenu.
- max_height: Zmienna przechowująca maksymalną możliwą wysokość terenu.
- texture: Obiekt będący teksturą terenu.
- total_points: Zmienna zawierająca ilość punktów heightmapy.
- boom(x,y): Funkcja odpowiadająca za wybuch w punkcie x,y ekranu.
- draw(surface): Funkcja odpowiadająca za rysowanie terenu na ekranie.
- generate(): Funkcja odpowiadająca za pseudolosowe wygenerowanie heightmapy.
- get_height(x): Funkcja do uzyskania wysokości terenu w pozycji x.
- set_height(x,height): Funkcja do ustawienia wysokości terenu w pozycji x.
- update_texture(): Funkcja do zaktualizowania tekstury terenu.
- update_texture_part(x,r,h): Funkcja do zaktualizowania tekstury terenu (utworzenia krateru w teksturze) w pozycji x w promieniu r i na wysokości h.

### Message

Klasa do reprezentowania komunikatów tekstowych w czasie rozgrywki. Ma ona następujące składowe:

- MainFont: Obiekt będący czcionką którą wykorzystują wyświetlane teksty.
- text: Obiekt będący tekstem wyświetlanym.
- textRect: Obiekt zawierający w sobie wymiary tekstu oraz jego pozycję na ekranie.
- draw(screen): Funkcja do wyświetlania tekstu na ekranie.

### Skybox

- texture: Obiekt będący teksturą nieba.
- draw(screen): Funkcja do wyświetlania nieba na ekranie.