RAPORT Z POSTĘPU PRAC

przedmiot: projekt MO3D

temat projektu: „Symulator wyścigów powietrznych w dowolnej scenerii wygenerowanej z mapy wysokościowej terenu.”

skład sekcji:

1) Dudek Piotr
2) Smoll Mateusz
3) Stachyra Krzysztof

Treść raportu:

Krzysztof Stachyra:

1) HUD: wyświetlanie informacji o końcu gry (GameOverText) oraz o aktualnej liczbie punktów (ScoreText). Skrypty aktualizaujące pola tekstowe (GameOverTextController oraz ScoreTextController).
2) Skrypt GameController: w momencie końca gry i wciśnięcia klwisza 'R' restart gry (mapy); obsługa końca gry (zatrzymanie)
3) Skrypt z klasą zawierającą wszystkie stałe w programie (takie jak nazwa samolotu): Constants
4) Skrypt obsługujący kolizje
5) Kamery podążające za modelem samolotu (z tyłu, z góry, z perspektywy pierwszej osoby, z lewej strony samolotu); skrypt przełączający widoki kamery: CameraSwitcher
6) Skrypt CollsionDetector (nałożony na terrain): wykrywanie kolizji samolotu z terenem lub drzewami. W momencie kolizji zatrzymanie gry.
7) Skrypt AirplaneMovement: obracanie samolotu (metoda Rotate) oraz poruszanie (metoda Move). Poruszanie zrealizowane wstępnie jako przesunięcie (w momencie wciśnięcia klawiszy strzałek)
8) Losowe generowanie bramek (wyznaczenie im współrzędnych 3D na mapie) – zabezpieczenie by nie znajdowały się pod terenem.
9) Trzymanie informacji o konkretnym torze przelotu (aby gracz przelatywał przez bramki w określonej kolejności) – po wygenerowaniu bramek ustalana jest ich kolejność. Wyświetlanie na mapie tylko 2 najbliższych bramek (według kolejności przelotu), przez jakie użytkownik musi przelecieć – w różnych kolorach.
10) Wykrywanie czy użytkownik przeleciał przez aktualną bramkę i reakcja na to (wyświetlenie kolejnej, zmiana koloru następnej bramki). Wykrycie przelotu przez ostatnią bramkę (koniec gry – wyświetlenie czasu)
11) Wyświetlanie aktualnego czasu przelotu.
12) Wyświetlanie czasu jaki pozostał graczowi (fajnie coś takiego wprowadzić – przy przelocie przez bramkę by dostawał dodatkowy czas).

Mateusz Smoll:

1) Załadowanie modelu samolotu i podpięcie pod skrypt AirplaneMovement.
2) Dodanie trzecio-osobowej kamery i niektórych ustawień kamer
3) Poprawki w skrypcie poruszania samolotu.
4) Dodanie opcji zmiany prędkości samolotu za pomocą przycisków LCTRL oraz LALT.
5) Dodanie dźwięku silników samolotu podczas lotu.
6) Usprawnienie lotu samolotu – nadawanie siły za pomocą Rigidbody.
7) Stworzenie modelu i przygotowanie obiektów prefab do generowania bramek.
8) Poprawienie skryptu obsługi kamer.

Piotr Dudek:

1) Utworzenie pierwszej prezentacji.
2) Implementacja w C# w Visual Studio menu startowego z możliwością wyboru z dysku pliku z mapą wysokościową terenu.
3) Implementacja odczytu danych wysokościowych (poziomów szarości poszczególnych pikseli) z wybranego pliku z obrazem mapy wysokościowej (dostępne formaty: *raw, *.jpg, *.jpeg, *.bmp, *.png, *.tiff, *.gif).
4) Implementacja zabezpieczeń przed wprowadzeniem niepoprawnych danych wejściowych (wybór nieobsługiwanego formatu pliku, błędne rozmiary mapy).
5) Scalenie utworzonego menu startowego z projektem Unity (skrypty StartMenu.cs i Form1.cs).
6) Dodanie generacji płaskiego terenu z poziomu kodu (skrypt MyTerrain.cs).
7) Dodanie tekstury do generowanego terenu.
8) Dodanie ustawiania wysokości poszczególnych punktów terenu na podstawie odczytanych z mapy poziomów szarości (na razie szwankuje).
9) Utworzenie drugiej prezentacji.
10) Poprawa generacji terenu tak, aby wysokości były poprawnie ustawiane według odczytanych z mapy danych.
11) Dodanie do terenu losowo wstawianych drzew według algorytmu „Perlin Noise” na odpowiednich wysokościach.
12) Podział terenu na warstwy, z odpowiednio ponakładanymi różnymi teksturami, na różnych poziomach (bądź też wymieszanymi).