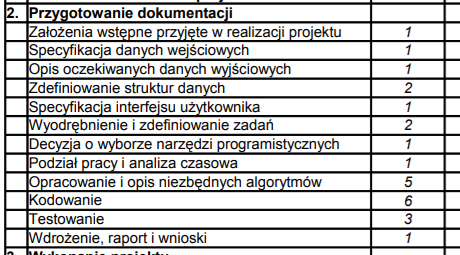
|  |
| --- |
| Projekt nr 25: Generowanie i animacja fraktali |
| Jan Paluch, Maciej Mikołajek, Wiktoria Szewczyk |



# Opis projektu

// skrócony opis czego projekt dotyczy

Głównym celem projektu jest napisanie programu, który wyświetla dwuwymiarowe fraktale. Dodatkowo tworzy animację przekształcając („morfując”) różne fraktale między sobą. Program umożliwia wyświetlanie dowolnych fraktali, jeśli dostarczymy mu dane na temat jego przekształceń afinicznych.

# Założenia wstępne przyjęte w realizacji projektu

// doprecyzowanie założeń podanych w zleceniu + dodatkowe założenia w zleceniu nie ujęte

Jednym z podstawowych założeń projektu jest wczytywanie z pliku parametrów animacji, między innymi rozmiar animacji, liczba klatek oraz liczba fraktali oraz ich parametry (odpowiednie współczynniki przekształceń afinicznych).

Nasz program powinien sam zadbać o odpowiednie skalowanie rozmiaru fraktali, tak aby mieściły się na ekranie i nie były za małe.

Kolor pikseli wyświetlanego fraktala powinien zależeć od tego, które przekształcenie zostało użyte do obliczenia jego pozycji.

Dodatkowo program będzie dysponował interfejsem graficznym, żeby korzystanie z pliku wejściowego nie było konieczne. Umożliwia on określenie dwóch fraktali o maksymalnie 5 przekształceniach, ilości klatek animacji, oraz rozmiaru animacji.

# Analiza projektu

**Specyfikacja danych wejściowych**

// - specyfikacja danych wejściowych rodzaje, postać i formaty danych wejściowych

Rozmiar animacji: rozmiar x i y bitmap.

Ilość iteracji: ilość punktów, z których złożony jest fraktal.

Ilość fraktali.

Ilość przekształceń afinicznych dla konkretnego fraktala.

Przekształcenie: współczynniki A, B, C, D, E, F przekształcenia afinicznego.

**Oczekiwane dane wyjściowe**

- opis oczekiwanych danych wyjściowych rodzaj, postać i formaty danych wyjściowych

Na ekranie wyświetla się wybrany fraktal, zbudowany z odpowiednio obliczonych punktów. Dodatkowo po kliknięciu w przycisku „Morfuj” widzimy animacje przejścia między kolejnymi fraktalami (jeśli na wejściu podaliśmy dane dla więcej niż jednego fraktala).

**Struktury danych**

- zdefiniowanie struktur danych w jakich strukturach będą przechowywane danych

Zdecydowaliśmy się na stworzenie 4 klas: *Point*, *Trans*, *TransSet* oraz *Instruction*.

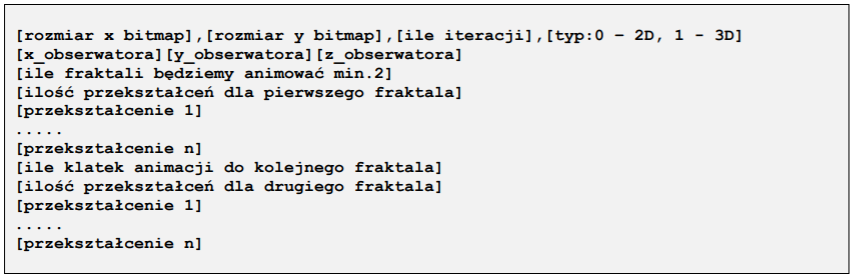
// dokładnie co robią to trzeba napisać w pkt 6 Kodowanie

**Specyfikacja interfejsu użytkownika**

- specyfikacja interfejsu użytkownika w miarę możliwości szczegółowy opis komunikacji z użytkownikiem

Użytkownik może wprowadzić dane za pomocą wcześniej przygotowanego pliku lub wprowadzić dane w odpowiednie pola w programie.

Struktura wczytywanego pliku jest następująca:



Zakładamy że tworzymy tylko fraktale 2D więc dostarczone dane dotyczące pozycji obserwatora są pomijane przez program. Przykładowy plik został dołączony do katalogu BIN.

Wprowadzanie danych za pomocą pliku daje większe możliwości, ponieważ możemy wprowadzić więcej niż dwa fraktale o więcej niż 5 przekształceniach.

**Wyodrębnienie i zdefiniowanie zadań**

- wyodrębnienie i zdefiniowanie zadań podział całego projektu na mniejsze, jednoznacznie zdefiniowane moduły

// nie wiem o co tu chodzi do końca, podział na pliki, podział na to co kolejno robiliśmy typu, zbudowanie interfejsu, wczytywanie z pliku, wyświetlanie pojedynczego fraktala, animacja,

**Wybór narzędzi programistycznych**

- decyzja o wyborze narzędzi programistycznych czego i dlaczego będziemy używać (kompilator, środowisko, biblioteki)

Wybraliśmy środowisko Microsoft Visual Studio ze względu na, że jesteśmy z nim już zaznajomieni oraz …generalnie git jest.

Dodatkowo korzystaliśmy z biblioteki wxWidgets, ponieważ umożliwiła nam ona łatwe zaimplementowanie interfejsu oraz wyświetlanie obrazów.

# Podział pracy i analiza czasowa

// kto co będzie robił i jak to będzie rozłożone w czasie

Regularnie spotykaliśmy się, żeby pracować nad projektem razem. Na początku zapoznaliśmy się z celami projektu i opracowaliśmy plan na struktury danych w projekcie. Praca nad projektem została rozłożona równomiernie. Aspekty wczytywania danych z pliku, tworzenia struktur danych lub proces wyświetlania fraktali na ekranie mogliśmy rozdzielić między siebie, jednak trudniejsze zagadnienia jak obliczanie i skalowanie kolejnych punktów lub sposób animacji omawialiśmy i implementowaliśmy wspólnie.

# Opracowanie i opis niezbędnych algorytmów

// opis użytych algorytmów, opracowanie własnych jeśli potrzebne

Przekształcenie afiniczne ma postać:

Pierwszy punkt zawsze jest punktem , a następne obliczane są na podstawie poprzedniego.

… itd.

# Kodowanie

// opis programu, schemat blokowy xDDDDDDDDD, wykresy przepływu danych, grafy ważniejszych algorytmów, szczegółowy opis klas, funkcji i zmiennych

# Testowanie

// opis procedury testowej, przygotowanie testowych zestawów danych

– testy niezależnych bloków

- testy powiązań bloków

- testy całościowe

- określenie niezmienników (jeśli to możliwe)

// co jeśli dostarczymy dane tylko jednego fraktala

# Wdrożenie, raport i wnioski

// uruchomienie z niezależnymi danymi, raport dotyczący wykonania (co się udało, co nie i dlaczego), co można poprawić lub zmienić w przyszłości