Mrówka Langtona

Jakub Przygodzki Adam Suski Grupa 1

Kwiecień 2018

Spis treści

1 Wprowadzenie

Napisany przez nas program symuluje mrówka Langtona. Jest to prosty automat komórkowy wymyślony i opisany przez Chrisa Langtona w 1986 roku. W każdym kroku wyróżniona jest jedna komórka nazywana "mrówką", która oprócz koloru ma określony także kierunek, w którym się porusza. Mrówka zachowuje się według następujących zasad:
1. jeśli znajduje się na polu białym to obraca się w lewo (o kąt prosty), zmienia kolor pola na czarny i przechodzi na następną komórkę;
2. jeśli znajduje się na polu czarnym to obraca się w prawo (o kąt prosty), zmienia kolor pola na biały i przechodzi na następną komórkę;
3. porusza się na nieskończonej planszy podzielonej na kwadratowe komórki (pola) w dwóch możliwych kolorach: czarnym i białym.

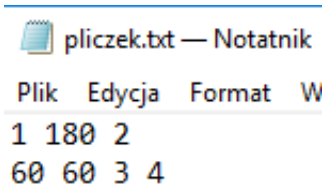
2 Założenia początkowe

Program został napisany jako model mrówki Langtona. Zostały nałożone pewne ograniczenia.
Liczba mrówek nie może byc większa od 5.
Rozmiar planszy nie może byc większy niż 181 pól.
Dla ułatwienia kazda mrówka ma swój kolor zadany przez twórców kodu.
Maksymalny odstęp między wyswietlanymi ruchami mrówek wynosi 4 sekundy (za długo trzeba czekac na poszczególne ruchy mrówek).

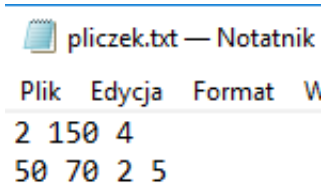
2.1 Plik konfiguracyjny

Wymagane jest stworzenie pliku konfiguracyjnego. Jest on dostarczany do pragramu jako argument linii poleceń. Poszczególne dane w pliku konfiguracyjnym są rozdzielone spacjami.
Pierwszy wiersz zawiera informacje o ilości mrówek, rozmiarze planszy po jakiej te mrówki się poruszają (rzeczywista liczba pól to liczba podana - 1) oraz czas odsepu pomiędzy wykonaniem się poszczególnych przesunięc mrówek.
Każdy i-ty wiersz zawiera informacje o i-1 mrówce (2 wiersz pliku zawiera informacje o 1 mrówce, itd.). Pierwsza liczba określa współrzędną x-ową mrówki na planszy, druga jej współrzędną y-kową. Trzecia liczba mówi o tym czy mrówka porusza się sposobem RL (2) czy LR (3). Ostatnia i czwarta liczba charakteryzuje kierunek, w którym skierowana jest mrówka (4 oznacza góra, 5 prawo, 6 dół, 7 lewo).

Zakładamy, że podane przez użytkownika cechy mrówki tzn. liczby z odpowiedniego zakresu (np. użytkownik nie ustawi współrzędnej mrówki poza rozmiarem planszy).



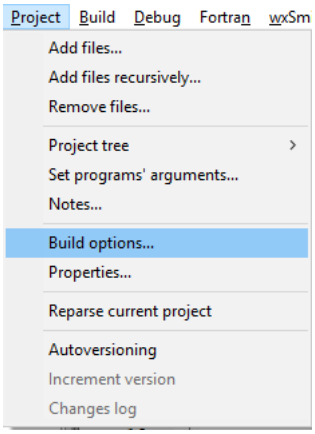
Rysunek 1: Przykład poprawnego pliku konfiguracyjnego



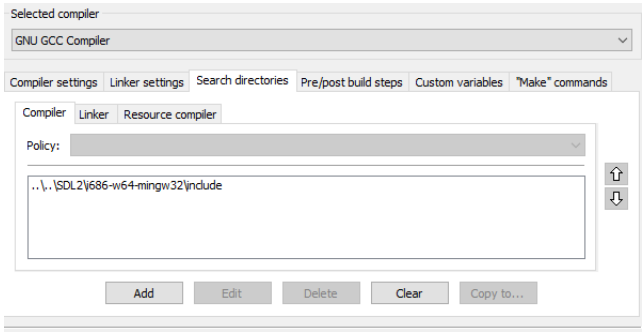
Rysunek 2: Przykład niepoprawnego pliku konfiguracyjnego

2.2 Biblioteka graficzna

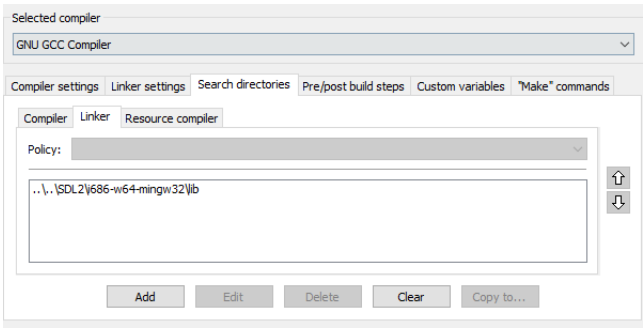
Do poprawnego działania programu wymagane jest zainstalowanie dodatkowej biblioteki graficznej SDL2. Przykład poprawnie skonfigurowanego IDE (w tym przypadku Code::Blocks 17.12).
W folderze z projektem należy zamieścić plik SDL2.dll. Dodatkowo w opcjach linkowania nalezy wpisać -lmingw32 -lSDL2main -SDL2.



Rysunek 3: Otwarte podmenu Project z zaznaczoną opcją, którą należy wybrać w celu skonfigurowania Build Options



Rysunek 4: Compiler



Rysunek 5: Linker

3 Opis programu

Program został stworzony z 4 modułów:
1) main - główny moduł,
2) board - moduł odpowiedzialny za tworzenie planszy po której poruszają się mrówki,
3) antt - moduł odpowiedzialny za poruszanie mrówek,
4) displaySDL - moduł odpowiedzialny za oprawę graficzną.

3.1 main

Główna funkcja programu ma za zadanie zczytać wszytskie dane z pliku konfiguracyjnego (antAmount - liczba mrówek, boardSize - rozmiar planszy, milisec - czas pomiędzy kolejneymi krokami algorytmu mrówki).
makeAnts() oraz antsAlgorithm() funkcje z modułu antt. boardGenrate() funkcja z modułu board.

3.2 board

Zawiera tylko jedną funkcję, która zwraca tablicę dwuwymiarową (a właściwie wskaźnik na wskaźniki). Jako parametr przyjmuje integer czyli rozmiar planszy po, któej poruszają się mrówki (może ona mieć tylko wymiar kwadratowy).
int** boardGenerate(int x)
Funkcja przyjmuje jako parametr zmienną x, która oznacza długość boku planszy.
Zwracana jest macierz, która symbolizuje planszę, po której poruszają się mrówki, z wartością każdej komórki równej 0.

3.3 antt

Zdefiniowane pewne stałe w celu zwiększenia przejzystości i lepszemu zrozumieniu algorytmu.
#define ACTIVEA 1
#define NONACTIVEA 0
#define righth 2
#define lefth 3
#define N 4
#define E 5
#define S 6
#define W 7
W stałych programowych ACTIVEA i NONACTIVEA literka A na końcu obydwu wyrazów oznacza ant.

Struktura, która zawiera informacje o mrówkach.
x współrzędna x mrówki.
y współrzędna y mrówki.
handling zawiera informacje o aktualnym sterowaniu mrówki.
handlingOriginal zawiera informacje o pierwotnym sterowaniu mrówki(czy zaczyna od R czy L).
handlingDerivate jest przeczywieństwem handlingOriginal.
direction określa kierunek mrówki(góra, dół, prawo, lewo).

typedef struct ant{
int x, y, handling, handlingOriginal, handlingDerivate, direction;
};
Został stworzony wskaźnik na tę strukturę, żeby łatwiej przechowywało informacje o kilku mrówkach.

ants_t makeAnts(int);
Tworzy tablicę mrówek o rozmiarze zadanym przez użytkownika.

int checkActivity(ants_t, int i, int);
Sprawdza aktywność (czy znajduje się na planszy i -tej mrówki). Jako trzeci parametr podany jest rozmair planszy. Zwraca 1 jeśli mrówka jest aktywna (na planszy), 0 w przeciwnym wypadku.

void changePosition(ants_t, int i);
Zmienia pozycję i-tej mrówki. Zasada działania jest zgodna z algorytmem mrówki dal 2 możliwości poruszani (RL i LR).

void antsAlgorithm(int**, int, ants t, int, int);
Zasadniczy algorytm poruszania mrówkami. Zmienia sterowanie zależnie od pozycji w jakiej znajduje się mróka, następnie wywołuje funkcję changePosition() i checkActivity() dla każdej mrówki. W między czasie zlicza ilość aktywnych mrówek. Po wykonaniu skoku każdej z mrówek, wyświetla aktualny stan planszy (antShow()). Jeśli, któa kolwiek mrówka jest już nieaktywna, bądź użytkownik nacisnął przycisk zamknięcia okna praca z programem się kończy.

3.4 displaySDL

#include "SDL2/SDL.h"
Określony rozmiar mrówki, który pozwala we względnie wygodny sposób obserwować ruch mrówki.
#define ANTSIZE 4
Window* window;
Renderer* renderer;
SDL Rect ant;

void drawBlack(int xm, int ym);
Zamalowywuje komórke napotkaną przez mrówkę z powrotem na czarno.
xm współrzędna x mrówki.
ym współrzędna y mrówki.

void drawColor(int xm, int ym , int i);
Zamalowywuje komórke napotkaną przez i-tą mrówkę na odpowiadający jej kolor.
xm współrzędna x mrówki.
ym współrzędna y mrówki.

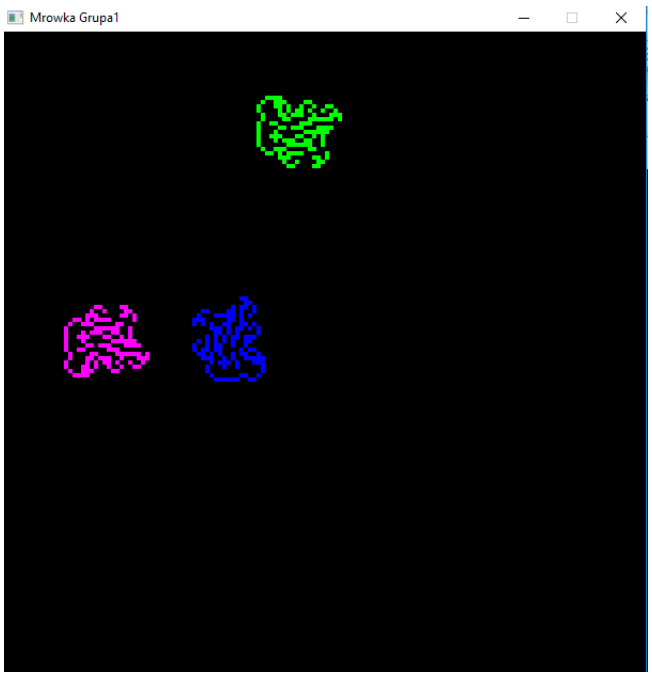
int initEverything(int boardSize);
Inicjalizuje pracę SDL
Zwraca 1 jeśli uda się zacząć pracę z SDL, w przeciwnym wypadku zwraca 0.

void destroymySDL();
Kończy pracę z SDL.

void antShow(int delay);
Odpowiada za wywołanie rendera oraz opóźnienia (delay) między wykonanymi krokami algorytmu mróki

4 Dane wyjściowe

Po porawnym uruchomieniu programu ukazuje się ekran przedstawiający planszę po której poruszają się mrówki. Użytkownik zobaczy każde przejście po kolei, aż do osiągnięcia przez mrówkę granicy planszy lub naciśnięcia przez użytkownika przycisku zamknięcia okna.



Rysunek 6: Przykład uruchomienia programu

W przypadku nieprawidłowego uruchomienia programu (nie podanie lub błędny plik konfiguracyjny, nieodpowiednie dane wejściowe, itp.) zostanie wyświetlony odpoweidni komunikat o wystąpionym błędzie i program sie zakończy.