|  |  |  |  |
| --- | --- | --- | --- |
|  | Języki i Metody Programowania  **Sprawozdanie do projektu pierwszego** | **Data** **utworzenia**:  17.03.2020  **Data** **ostatniej** **zmiany**:  17.03.2020  **Numer** **zespołu**: 5  **Autorzy**:  Matłacz Jakub 307371  Kuś Szymon Twój\_Indeks | |
| **Automat Komórkowy Gra w życie Johna Conwaya.** | | |  |

­­­1. **Wstęp**:

Zagadnieniem, nad którym miano się pochylić był automat komórkowy, Gra w życie (Game of Life) Johna Conwaya. To świetny przykład tego jak z kilku prostych zasad może wynikać stosunkowo ogromna złożoność struktur powstałych na ich podstawie. Widzi się analogie między tym niepozornym automatem, a rzeczywistością, w której według Kwantowej Teorii Pola (Quantum Field Theory), wszelkie cuda otaczającego nas świata wynikają z superpozycji prostych pól.

2. **Zasady** **gry**:

Proszę wyobrazić sobie pole złożone z prostych komórek, których jedyną właściwością jest ich stan: zapalony lub zgaszony. Komórka zgaszona – martwa, zapalona – żywa. Komórki mogą zmieniać swój stan według następujących zasad:

1. Martwa komórka o 3 żywych sąsiadach zapali się w następnym cyklu.
2. Żywa komórka o 2 lub 3 sąsiadach pozostanie żywa w następnym cyklu.
3. Żywa komórka o liczbie sąsiadów różnej od 2 lub 3 zgasi się w następnym cyklu.

Przez stan automatu rozumiemy informację o stanie każdej symulowanej komórki w danym skwantowanym czasie.

3. **Opis** **implementacji**:

3.1 **Teoretycznie**:

Stworzono tablicę dwuwymiarową o wymiarach r+2 – rzędów i c+2 – kolumn, gdzie r oraz c określają symulowany obszar, natomiast powiększenie tych wartości o 2 służy zrobieniu miejsca dla „ramki”, która otacza ten obszar – jest to udogodnienie ułatwiające wdrożenie pomysłu.

Każda komórka jest strukturą zawierającą trzy elementy: stan obecny, stan w przyszłym cyklu oraz informację czy dana komórka jest czy nie jest „ramką”.

Dla oszczędności pamięci użyto typu char. Przy dużych rozmiarach symulowanej przestrzeni ma to niemały efekt.

Na podstawie stanu obecnego określamy stan następny automatu. Przechodzimy przez każdą komórkę i sprawdzamy kolejno warunki na narodziny, śmierć i przetrwanie. Kolejność ma tutaj znaczenie i powinna zostać zachowana.

1. Narodziny - do stanu w przyszłym cyklu zapisana jest informacja o zapaleniu komórki.
2. Śmierć – do stanu w przyszłym cyklu zapisana jest informacja o zgaszeniu komórki.
3. Przetrwanie - do stanu w przyszłym cyklu zapisana jest informacja o zapaleniu komórki (lub jeśli ktoś woli skopiowaniu stanu obecnego – niezmienianiu go).

Kluczowe jest w tym momencie efektywne zliczanie żywych sąsiadów, ponieważ jest wykonywane dla każdej komórki w każdym cyklu. Stworzono więc funkcję, która na podstawie pozycji danej komórki (opisanej z pomocą dwu indeksów) wykonuje to zadanie z wykorzystaniem wskaźników dla większej szybkości.

Pod koniec każdego cyklu „stanowi obecnemu” przypisywany jest „stan w przyszłym cyklu”.

3.2 **Funkcjonalności** **programu**:

Wywołanie programu przyjmuje dwa argumenty: 1 – nazwa pliku do wczytania stanu początkowego automatu, 2 – ilość cykli do symulacji. W przypadku niepodania argumentów we właściwej postaci lub ich zupełnym braku program przyjmuje odpowiednie wartości domyślne.

Program udostępnia szereg funkcjonalności:

1. Zapisywanie kolejnych stanów automatu do plików .png.
2. Zapisywanie kolejnych stanów automatu do plików .life, czyli zwykłego pliku tekstowego z opisem wymiarów oraz zerojedynkowym opisem stanu. Możliwe jest dzięki temu łatwe ponowne uruchomienie programu z danym stanem początkowym opisanym w tymże pliku.
3. Przedstawienie kolejnych stanów automatu w formie animacji w terminalu.

4. **Podsumowanie**:

Udało nam się udostępnić wszystkie oczekiwane funkcjonalności zaimplementowane w języku C z wygodnym podziałem na moduły.