Specyfikacja Funkcjonalna
Gra w życie

Michał Sut

1 OPIS OGÓLNY

1.1 Nazwa programu

Automat komórkowy *Life*

1.2 Wstęp teoretyczny

Program realizuje *Grę w życie* - rodzaj automatu komórkowego autorstwa Johna Conwaya.
Ogólna zasada działania automatów komórkowych jest następująca: Dana jest siatka - dla uproszczenia dwuwymiarowa w postaci "kratownicy". Siatka ta składa się z przylegających do siebie komórek, należących do pewnego uporządkowanego zbioru. Komórki mogą znajdować się w jednym z kilku możliwych stanów.
Komórki zmieniają swój stan na podstawie swojego obecnego stanu oraz komórek sąsiadujących, według pewnych określonych reguł. Stan wszystkich komórek w danej chwili nazywamy generacją. Niniejszy program jest takim właśnie automatem, opartym o zbiór zasad stworzonych przez Johna Conwaya, zwanym *Grą w życie*.

1.3 Zasady "Gry w życie" Johna Conwaya

- Martwa komórka, która ma dokładnie 3 żywych sąsiadów, staje się żywa w następnej jednostce czasu (rodzi się)
- Żywa komórka z 2 albo 3 żywymi sąsiadami pozostaje nadal żywa; przy innej liczbie sąsiadów umiera (z "samotności" albo "zatłoczenia").

W Grze w życie stosuje się sąsiedztwo Moore'a.

2 OPIS FUNKCJONALNOŚCI

2.1 Jak korzystać z programu

Program uruchamiany jest z linii komend, a dane wejściowe oraz opcje wykonania programu wprowadzane są jako lista argumentów uruchomienia programu. Dostępne opcje:

-n liczba_generacji - domyślnie: 10
-p przedrostek - domyślnie: "picture"
-q przyrostek - domyślnie: brak
-s sąsiedztwo {-s Moore, -s Neumann} - domyślnie: "Moore"
-c kolor_żywej_komórki - domyślnie: "black"

2.1.1 Przykładowe uruchomienie programu

```
./life input.txt output.txt -p outpng -n 30
```

Powyższe polecenie uruchamia program *Life*. Wejściowa generacja znajduje się w pliku **input.txt**. Generacja wyjściowa zapisywana będzie do pliku **output.txt**, a graficzne przedstawienie każdej z 30 generacji zapisane zostanie do pliku rozpoczynającego się od "outpng" (np.: outpng001, outpng002, ..., outpng030) .

2.2 Możliwości programu

Podstawowe:

- Wczytanie dowolnej generacji początkowej z pliku o ustalonym formacie;
- Przeprowadzenie zadanej liczby **n** generacji;
- Wygenerowanie **n** obrazów z poszczególnymi generacjami (w formacie **png**);
- Zapisanie bieżącej generacji do pliku, który będzie mógł posłużyć jako plik wejściowy.

Dodatkowe:

- Wybór sąsiedztwa poprzez opcje (np -s Moore/ -s Neumann);
- Wybór koloru żywej komórki;
- Wybór przyrostku lub przedrostku nazw plików z generacjami.

3 FORMAT DANYCH I STRUKTURA PLIKÓW

3.1 Struktura katalogów

- *Life* - Nadrzędny katalog projektu
  - *src* - Katalog przechowujący pliki źródłowe.
  - *pictures* - Miejsce zapisu obrazów zawierające osobne podkatalogi dla każdego uruchomienia programu.
  - *data* - Katalog z danymi wejściowymi i wyjściowymi.
  - ***life*** - Plik wykonywalny

3.2 Dane wejściowe

Do uruchomienia programu niezbędne będzie użycie danych wejściowych. Będą one miały postać pliku tekstowego zawierającego początkową generację komórek. Aby wczytywanie danych się powiodło, muszą być one zapisane w określonym formacie:
W pierwszej linii - liczba naturalna określająca szerokość siatki.
W drugiej linii - liczba naturalna określająca wysokość siatki.
Następnie, w kolejnych liniach, pary liczb naturalnych będące współrzędnymi żywych komórek.
Przykład:

```
100
120
20 40
22 40
21 41
22 41
21 42
```

3.3 Dane wyjściowe

Program będzie tworzył dwa rodzaje danych wyjściowych. Po pierwsze, plik tekstowy zawierający **n-tą** produkcję siatki, w takim samym formacie jak dane wejściowe, aby umożliwić wznawianie programu od ostatniej generacji. Po drugie, **n** obrazów w formacie **png**, przedstawiających kolejne etapy siatki. Martwe komórki oznaczone będą kolorem białym, a żywe domyślnie czarnym (możliwość wyboru przy uruchomieniu). Pliki graficzne będą gromadzone w osobnym katalogu, w różnych podkatalogach dla każdego pliku wejściowego.

4 SCENARIUSZ DZIAŁANIA PROGRAMU

4.1 Scenariusz ogólny

1. Uruchomienie programu;
2. Sprawdzenie poprawności argumentów i opcji;
   (a) W przypadku błędu - podjęcie działań opisanych w podrozdziale 4.2.
   (b) W przypadku powodzenia - kontynuowanie działania programu.
3. Wczytanie danych wejściowych;
4. Wykonanie zadanej liczby generacji z jednoczesnym tworzeniem obrazów;
5. Zapisanie ostatniej generacji do wyjściowego pliku tekstowego;
6. Zakończenie działania programu.

4.2 Obsługa błędów, nieprawidłowych danych oraz sytuacji wyjątkowych

4.2.1 Brak nazw plików z danymi wejściowymi oraz wyjściowymi

Jeżeli nie podano żadnych plików, lub gdy przynajmniej jeden z plików jest błędny program zwróci błąd i przerwie działanie.

4.2.2 Brak opcji uruchomienia programu lub ich błędne wartości

W przypadku, gdy użytkownik nie poda danej opcji lub jej wartość jest inna niż oczekiwana, zostanie jej przypisana wartość domyślna, oraz wyświetlone zostanie powiadomienie informujące o uruchomieniu programu z wartością domyślną danej opcji.

4.2.3 Błędny format danych wejściowych

Błędny format danych obejmuje: liczby ujemne lub ciągi znaków, zbyt mało lub zbyt dużo liczb, a także współrzędne wykraczające poza określony wcześniej rozmiar siatki. Przy pierwszym napotkaniu błędu program przerwie wczytywanie i zakończy się wyświetlając odpowiedni komunikat.

4.2.4 Błąd w trakcie wykonywania programu

Jeżeli program napotka dowolny inny błąd w trakcie wykonywania zapisze ostatnią poprawną generację siatki zarówno do pliku tekstowego jak i graficznego, po czym zakończy działanie.

Dodatkowo, każde błędne uruchomienie programu, oprócz podjęcia odpowiedniej akcji (podrozdziały 4.2.1 do 4.2.3), spowoduje wyświetlenie instrukcji poprawnego uruchomienia wraz z przykładem.