Specyfikacja funkcjonalności

Elżbieta Karasińska, Paweł Malinowski 04.03.2024r.

1 Wstęp

W dokumencie omówiona zostanie specyfikacja funkcjonalności programu pozwalającego na znalezienie drogi łączącej dwa określone punkty w dwuwymiarowym labiryncie. Dokument omawia działanie programu, sposób jego wywołania oraz obsługę błędów. Przedstawia także wynik przykładowego wywołania programu.

2 Szczegółowe omówienie programu

W tej części omówiono definicje pojęć używanych w specyfikacji, założenia, jakie musi spełniać labirynt oraz wymagania spełniane przez program.

2.1 Wymagania spełniane przez program

Program znajduje drogę łączącą dwa określone pola w labiryncie. Podczas działania program alokuje nie więcej niż **512kB pamięci.** Znaleziona droga wypisywana jest w postaci indeksów kolejnych skrzyżowań po których należy się poruszać.

2.2 Definicje terminów używanych w specyfikacji

Definiujemy:

- ścieżkę, jako pola na których może znajdować się droga,
- ściany, jako pola niebędące ścieżką,
- sąsiedztwo, jako 4 pola znajdujące nad, pod, z prawej i z lewej od danego pola,
- skrzyżowanie, jako pole, w którego sąsiedztwie przynajmniej jedno pole w pionie i jedno pole w poziomie są ścieżkami.

2.3 Założenia dotyczące labiryntu

Labirynt musi spełniać następujące warunki:

- maksymalny rozmiar labiryntu wynosi 1024×1024 pola, licząc po ścieżkach,
- istnieje przynajmniej jedna droga łącząca dwa dowolne pola będące ścieżką,
- \bullet nie istnieją kwadraty pól 2×2 lub większe, takie, że wszystkie pola są ścieżkami.

3 Sposób wywołania

W celu uruchomienia programu należy wykonać następujące kroki:

- 1. skompilować, wywołując w oknie terminalu komendę make
- 2. uruchomić, wywołując komendę ./out. z odpowiednimi argumentami:
 - (a) -h, przełącznik wyświetlający wiadomości dotyczące prawidłowego uruchamiania programu,
 - (b) -s (int)(int), przełącznik pozwalający na wprowadzenie pola startowego. Brak wyboru tego przełącznika ustawi pole startowe na lew górny róg labiryntu,
 - (c) -k (int)(int), przełącznik pozwalający na wprowadzenie pola końcowego. Brak wyboru tego przełącznika ustawi pole końcowe na prawy dolny róg labiryntu,
 - (d) -m (string), przełącznik pozwalający na wprowadzenie ścieżki pliku testowego z labiryntem do rozwiązania. Brak wybrania tego przełącznika spowoduje próbę załadowania pliku o nazwie maze.txt znajdącego się w folderze programu.

4 Obsługa błędów

Podczas działania programu mogą wystąpić niespodziewane błędy. W tej części omówiono zachowanie programu w przypadku ich wystąpienia.

- 1. Podanie nieprawidłowych argumentów dla któregoś z przełączników. Zostanie wyświetlony komunikat o błędzie, w którym podane zostanie, który argument jest niepoprawny i jakie może przyjmować on wartości.
- Nieprawidłowy format lub adres pliku. Użytkownikowi zostanie przekazana informacja o nieznalezionym pliku testowym o podanej przez niego ścieżce.
- 3. Labirynt nie spełnia warunków. Jeśli podczas próby odczytywania plików okaże się, że kolumny i wiersze podanego pliku nie są stałej długości lub labirynt jest zapisany inaczej niż za pomocą znaków X i " ", program przekaże użytkownikowi informację o błednym formacie zapisu labiryntu.

5 Przykładowe wywołanie programu

W tej części zaprezentowano przykładowe wywołanie programu i jego rezultat. W celu uzyskania pliku wejściowego wygenerowano labirynt o wymiarach 8 × 4 korzystając ze strony http://tob.iem.pw.edu.pl/maze/, a następnie pobrano i zapisano dane o nim w pliku labirynt.txt.

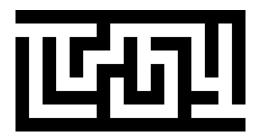


Figure 1: Przykładowy wygenerowany labirynt



Figure 2: Zawartość pliku tekstowego labirynt.txt

Następnie należy skompilować i uruchomić program wywołując w terminalu komendy:

\$ make

\$./out -m labirynt.txt

W rezultacie program wypisze znalezioną drogę:

```
Odnaleziona droga (wiersz, kolumna): (0,0), (0,1), (2,1), (2,2), (1,2), (1,3), (0,3), (0,6), (2,6), (2,7), (3,7)
```