1 Opis ogólny

1.1 Nazwa programu

SUGP -The Shortest Undirected Graph Path (Poszukiwanie najkrótszej ścieżki w grafie nieskierowanych.)

1.2 Cel programu

Celem programu jest znalezienie najkrótszej ścieżki w grafie nieskierowanym. Graf będzie wczytywany z pliku o ustalonej strukturze lub generowany na podstawie zadanych parametrów wejściowych. Generowany graf będzie zapisywany w postaci pliku.

2 Format danych i struktura plików

2.1 Przechowywanie danych w programie - teoria

Dane programu będą wczytywane do list sąsiedztwa. Lista sąsiedztwa umożliwia efektywne (szybkie) wyszukiwanie sąsiadów. Z punktu działania programu jest to bardzo ważna kwestia, gdyż algorytm poszukiwania ścieżki często odwołuje się do wierzchołków sąsiadujących z wierzchołkiem przetwarzanym (połączonym z nim krawędzią). Dodatkowo lista sąsiedztwa jest najwydajniejszym sposobem reprezentacji grafu ze względu na pamięć komputera. Prezentuje się ona znacznie lepiej niż macierz sąsiedztwa (pamięć rzędu O (n^2), gdzie n – liczba wierzchołków) czy macierz incydencji (pamięć rozmiaru O (m x n)), ponieważ zajmuje ona pamięć rzędu O(m), gdzie m to liczba krawędzi grafu.

2.2 Dane wejściowe

Istnieją dwa możliwe warianty danych wejściowych:

1. Nazwa pliku zawierającego strukturę grafu, dwa parametry określające wierzchołek startowy i końcowy.

   Wywołanie:

   ```
   ./out -nazwa_pliku -nr_wierzcholka_start -nr_wierczholka_koniec
   ```

2. Zestaw parametrów wejściowych na podstawie których zostanie wygenerowany graf.

   Wywołanie:

   ```
   ./out -liczba_wierszy -liczba_kolumn -lewa_granica_zakresu
           - prawa_granica_zakresu.
   ```

Lub 2 wariant wywołania:

`./out -liczba_wierszy -liczba_kolumn.`

Ad. 1 W przypadku wczytywania grafu dane wejściowe zawarte będą w pliku tekstowym (*.txt), którego nazwę będzie należało podać jako argument wywołania.

Struktura pliku:

- Pierwszy wiersz pliku zawierać będzie kolejno: liczbę kolumn i liczbę wierszy grafu.
- Kolejne wiersze pliku dotyczyły będą kolejno wierzchołków od 0 do n, gdzie n to liczba wierzchołków pomniejszona o 1 (numeracja wierzchołków zaczyna się od 0). Każdy z wierszy zawierać będzie listę par (numer wierzchołka :wagę krawędzi), gdzie separatorem jest „;”, a liczby zmiennoprzecinkowe wyrażające wagi zapisywane są przy pomocy znaku kropki (zamiast przecinka).

Przykładowo:

```
7 4
1 :0.8864916775696521  4 :0.2187532451857941
5 :0.2637754478952221  2 :0.6445273453144537  0 :0.4630166785185348
6 :0.8650384424149676  3 :0.42932761976709255  1 :0.6024952385895536
7 :0.5702072705027322  2 :0.86456124269257
8 :0.9452864187437506  0 :0.8961825862332892  5 :0.9299058855442358
1 :0.5956443807073741  9 :0.31509645530519625  6 :0.40326574227480094
10 :0.7910000224849713  7 :0.7017066711437372  2 :0.20056970253149548
6 :0.9338390704123928  3 :0.797053444490967  11 :0.7191822139832875
4 :0.7500681437013168  12 :0.5486221194511974  9 :0.25413610146892474
13 :0.8647843756083231  5 :0.8896910556803207  8 :0.4952122733888106
14 :0.5997502519024634  6 :0.5800735782304424  9 :0.7796297161425758
15 :0.3166804339669712  10 :0.14817882621967496  7 :0.8363991936747263
13 :0.5380334165340379  16 :0.8450927265651617  8 :0.5238810833905587
17 :0.5983997022381085  9 :0.7870744571266874  12 :0.738310558943156
10 :0.8801737147065481  15 :0.6153113201667844  18 :0.2663754517229303
19 :0.9069409600272764  11 :0.7381164412958352  14 :0.5723418590602954
20 :0.1541384547533948  17 :0.3985282545552262  12 :0.29468967639003735
21 :0.7576872377752496  13 :0.4858285745038984  16 :0.28762266137392745
17 :0.6628790185051667  22 :0.9203623808816617  14 :0.8394013782615275
18 :0.6976948178131532  15 :0.4893608558927002  23 :0.5604145612239925
24 :0.8901867253885717  21 :0.561967244435089  16 :0.35835658210649646
17 :0.8438726714274797  20 :0.3311114339467634  25 :0.7968809594947989
21 :0.6354858042070723  23 :0.33441278736675584  18 :0.43027465583738667
27 :0.8914256412658524  22 :0.8708278171237049  19 :0.4478162295166256
```

```
20 :0.35178269705930043  25 :0.2054048551310126
21 :0.6830700124292063  24 :0.3148089827888376  26 :0.5449034876557145
27 :0.2104213229517653  22 :0.8159939689806697  25 :0.4989269533310492
26 :0.44272335750313074  23 :0.4353604625664018
```

Ad. 2 W przypadku generowania grafu na podstawie parametrów wejściowych należy podać kolejno:

- Liczbę wierszy grafu
- Liczbę kolumn grafu
- Zakres zmienności losowanych wag – 2 liczby rzeczywiste nieujemne w porządku rosnącym (wartości będą losowane w zakresie od pierwszej liczby do drugiej)

Jeżeli nie zostanie podany zakres zmienności losowanych wag przyjęty zostanie przedział domyślny - <0,1>.

Przykładowo:

- 7 4 0 2 – podanie takich argumentów wejściowych oznacza wygenerowanie grafu o 7 wierszach, 4 kolumnach i wylosowanych wagach w przedziale <0,2>
- 2 3 – podanie takich argumentów wejściowych oznacza wygenerowanie grafu o 2 wierszach, 3 kolumnach oraz domyślnym przedziale wartości <0,1>

Dane powinne zostać podane w odpowiednim formacie. Żeby program był w stanie znaleźć rozwiązanie należy podać liczby w zakresie:

- Liczbę wierzchołków labiryntu-liczba całkowita od 2 do 10000
- Liczbę krawędzi - liczba całkowita od 1 do 10000
- Lewa granica zakresu (w przypadku losowania wag krawędzi) - od 0 do 10000
- Prawa granica zakresu (w przypadku losowania wag krawędzi) - od 0 do 10000; Przy tym lewa granica >= prawa granica

2.3 Dane wyjściowe

1. Zapis grafu w przypadku jego generacji

   Dane wyjściowe zapisywane będą w takim samym formacie jak dane wejściowe w przypadku wczytywania grafu – plik tekstowy (Graph.txt).

   Struktura pliku:

   - Pierwszy wiersz pliku zawierać będzie kolejno: liczbę kolumn i liczbę wierszy grafu.
   - Kolejne wiersze pliku dotyczyły będą kolejno wierzchołków od 0 do n, gdzie n to liczba wierzchołków pomniejszona o 1 (numeracja wierzchołków zaczyna się od 0). Każdy z wierszy zawierać będzie listę par (numer wierzchołka :wagę krawędzi), gdzie separatorem jest „:", a liczby zmiennoprzecinkowe wyrażające wagi zapisywane są przy pomocy znaku kropki (zamiast przecinka).

   Przykładowo:

```
2 3
0 :0.2637754478952221  1 :0.6445273453144537  3 :0.4630166785185348
0 :0.8650384424149676  2 :0.42932761976709255  4 :0.6024952385895536
1 :0.9452864187437506  5 :0.8961825862332892
0 :0.5956443807073741  4 :0.31509645530519625
3 :0.7910000224849713  5 :0.7017066711437372  1 :0.20056970253149548
4 :0.9338390704123928  1 :0.797053444490967  2 :0.7191822139832875
```

2. Zapis wyniku działania programu - znalezienie najkrótszej ścieżki.

   Dane wyjściowe z najkrótszą ścieżką do wyjścia zapisywane będą w pliku tekstowym (Result.txt).

   Plik wyjściowy zawierać będzie:

   - sumę wag krawędzi, czyli całkowitą "długość" ścieżki
   - listę przejść między komórkami reprezentowanymi w postaci par połączeń wierzchołekPoczątkowy wierzchołekDocelowy wagaKrawędziMiędzyWierzchołakmi.

   Przykładowo:

```
1.3375
0 1 0.3911
1 4 0.3212
4 5 0.6252
```

3 Scenariusz działania programu

3.1 Scenariusz ogólny

Program na podstawie podanych argumentów wejściowych ustali czy należy generować graf czy wczytać go z pliku tekstowego.

- Jeżeli dostaje 2 lub 4 liczby, to
  1. Generuje graf ważony o wagach w zakresie podanym przez użytkownika, lub, jeżeli zakres nie zostanie podany, program generuje wagi w domyślnym zakresie <0,1>.
  2. Zapisuje wygenerowany graf do pliku Graph.txt.
- Jeżeli plik tekstowy - przechodzi do następnego kroku.
  1. Program sprawdza spójność grafu, w przypadku grafu spójnego kontynuje działanie, a w przypadku grafu niespójnego wyświetla komunikat o błędzie.
  2. Program wyszukuje najkrótsze ścieżki pomiędzy wybranymi punktami.

W przypadku, gdy użytkownik chce wygenerować graf oraz znaleźć w nim ścieżkę pomiędzy poszczególnymi punktami, należy uruchomić program dwa razy. Pierwszy raz, żeby wygenerować graf na podstawie podanych parametrów i zapisać go do pliku, drugi - żeby znaleźć najkrótszą ścieżkę we wcześniej wygenerowanym grafie pomiędzy punktami podanymy jako argumenty wywołania.

3.2 Scenariusz szczegółowy

Poniżej opisane są 2 podstawowe przypadki użycia programu, pierwszy z zadanym grafem, natomiast w drugim graf jest generowany.
Aby uruchomić program z zadanym grafem użytkownik powinien :

- Uruchomić terminal.
- Przejść do folderu z plikiem wykonywalnym “out”.
- Użyć polecenie : ./out -nazwapliku.Tym poleceniem przekazujemy programowi informację o pliku, który zawiera graf oraz ilości wierzchołków i krawędzi).
- Otrzymać:
  – W pliku Result.txt najkrótsze możliwe wyjście z punktu start do punktu koniec.
  W przypadku niepodania lub podania nieprawidłowej liczby argumentów wywołania program wyświetli komunikat z błędem.

Aby uruchomić program i wygenerować graf użytkownik powinien :

- Uruchomić terminal.
- Przejść do folderu z plikiem wykonywalnym “out”.
- Użyć polecenie : ./out x1 x2 x3 x4 lub ./out x1 x2, gdzie x1,x2,x3,x4 - parametry wejściowe opisane w p.2.2.
- Otrzymać:
  - W pliku Graph.txt wygenerowany graf. W przypadku niepodania lub podania nieprawidłowej liczby argumentów wywołania program wyświetli komunikat z błędem.

4 Komunikaty błędów

- Can't open file – Program nie jest w stanie otworzyć pliku prawdopodobnie program nie może znaleźć pliku o podanej nazwie albo plik nie istnieje.
- Wrong data format in file - Program nie jest w stanie poprawnie odczytać danych z podanego pliku, prawdopodobnie zostały one zapisane w złym formacie lub podane są nie poprawne wartości. Dane wejściowe powinne być zgodne z informacją zawartą w punkcie 2.2.
- Invalid program invocation arguments - Program nie jest w stanie poprawnie zinterpretować argumentów wejściowych, prawdopobdobnie zostały podane złe argumenty lub ich liczba nie jest odpowiednia. Argumenty wejściowe należy podawać w sposób zaprezentowany w punkcie 2.2.
- inconsistent graph - Podany albo wygenerowany graf jest niespójny, program nie jest w stanie znaleźć najkrótszą ścieżkę.