

Praca domowa 1 – 15path

Termin zwrotu : 10 marca godz. 23.00

Zadanie uznaje się za zaliczone, gdy praca oceniona zostanie na co najmniej 6 pkt.

Wykaz krawędzi pewnego grafu zorientowanego przechowywany jest w SQL-owym repozytorium danych (w bazie danych) w tabeli o nazwie GData. Struktura tabeli utworzona została z wykorzystaniem instrukcji

```
CREATE TABLE [dbo].[GData] (  
    [x] [int] NOT NULL,  
    [y] [int] NOT NULL,  
    [p] [float] NOT NULL,  
    CONSTRAINT [PK_GData] PRIMARY KEY CLUSTERED  
    (  
        [x] ASC,  
        [y] ASC  
    ) ON [PRIMARY]  
    ) ON [PRIMARY]
```

Połączenie do SQL-owej bazy danych (dostęp do bazy) realizowany jest z wykorzystaniem driverów JDBC poprzez wykonanie metody

```
String database = <connection_string>; // gdzie <connection_string> parametr linii komend  
Connection conn = DriverManager.getConnection(database);
```

Wierzchołki grafu numerowane są od 1 do n, przy czym $n = \max(\max(x), \max(y))$ – określone jest największą wartością etykiety wierzchołka występującą w tabeli GData.

Każdy wiersz tabeli interpretowany jest jako opis krawędzi łączącej wierzchołki x oraz y. Atrybut p krawędzi oznacza maksymalną przepustowość na odcinku pomiędzy x oraz y.

Należy wyznaczyć ścieżkę o najmniejszym koszcie transportu łączącą wierzchołek o numerze 1 z wierzchołkiem k którego indeks określony jest parametrem linii komend <indeks> oraz wyznaczyć wartość tego kosztu. Koszt transportu wzdłuż każdej krawędzi ścieżki równy jest odwrotności przepustowości tej krawędzi, czyli dla krawędzi (x,y) o przepustowości p_{xy} koszt transportu wyniesie $1/p_{xy}$. Koszt transportu dla każdego z węzłów ścieżki (dla węzła początkowego oraz końcowego z definicji przyjmujemy 0) definiowany jest jako bezwzględna wartość różnicy przepustowości krawędzi ścieżki o końcach w tym wierzchołku, czyli np. w przypadku węzła y przez który przechodzi ścieżka z wykorzystaniem krawędzi (x,y) oraz (y,z) o przepustowości odpowiednio p_{xy} i p_{yz} koszt transportu wyniesie $|p_{xy} - p_{yz}|$. Łączny koszt transportu wzdłuż ścieżki definiowany jest jako suma kosztów transportu wszystkich krawędzi wchodzących w skład ścieżki oraz wierzchołków leżących na ścieżce.

Program ma być zapisany wyłącznie w dwóch plikach : GPath.java zawierającym implementację mechanizmu poszukiwania rozwiązania (kosztu wzdłuż ścieżki), oraz Main.java – zawierającym programem główny. Program nie może korzystać z jakichkolwiek bibliotek zewnętrznych oraz nie może być zależny od jakiegokolwiek dialektu SQL.

Proces kompilacji musi być możliwy z użyciem komendy

```
javac -Xlint GPath.java Main.java
```

Uruchomienie programu winno być możliwe z użyciem komendy

```
java Main <connection_string> <indeks>
```

Wynik końcowy (w strumieniu wyjściowym nie powinny pojawiać się jakiegokolwiek inne elementy – np. wydruki kontrolne) działania programu musi zawierać pojedynczą liczbę określającą koszt transportu ścieżki łączącej wierzchołki 1 oraz k (poprawnie zaokrągloną z dokładnością do 3 miejsc dziesiętnych), a więc np.

Koszt : 76.75200

Wymagania :

- Klasa implementująca problem winna zostać zdefiniowana w pliku GPath.java
- Klasa implementująca mechanizm program główny (metoda main) winny być zdefiniowane w pliku Main.java
- W pliku README.pdf winien być zawarty szczegółowy opis organizacji struktur danych oraz szczegółowy opis zastosowanego mechanizmu (metody) poszukiwania ścieżki o minimalnym koszcie transportu.
- Proces poszukiwania rozwiązania dla zestawu danych o rozmiarze n rzędu 10^4 winien się kończyć w czasie nie przekraczającym 1 min (orientacyjnie dla typowego notebooka). Po przekroczeniu limitu czasu zadanie będzie przerywane, i traktowane podobnie jak w sytuacji błędów wykonania (czyli nie podlega dalszej ocenie).

Sposób oceny :

- 1 pkt – **Weryfikacja** : czy program jest skompletowany i spakowany zgodnie z ogólnymi zasadami przesyłania zadań.
- 1 pkt – **Kompilacja** : każdy z plików winien być kompilowany bez jakichkolwiek błędów lub ostrzeżeń (w sposób omówiony wyżej)
- 1 pkt – **Wykonanie** : program powinien wykonywać się bez jakichkolwiek błędów i ostrzeżeń (dla pliku danych wejściowych zgodnych z wyżej zamieszczoną specyfikacją) z wykorzystaniem omówionych wyżej parametrów linii komend
- 2 pkt – **README** : plik README.pdf dokumentuje w sposób kompletny i właściwy struktury danych, oraz opis przyjętej koncepcji algorytmu

- 1 pkt – **Styl kodowania** : czy funkcji i zmienne posiadają samo-wyjaśniające nazwy ? Czy podział na funkcje ułatwia czytelność i zrozumiałość kodu ? Czy funkcje eliminują (redukuja) powtarzające się bloki kodu ? Czy wcięcia, odstępy, wykorzystanie nawiasów itp. (formatowanie kodu) są spójne i sensowne ?
- 4 pkt – **Poprawność algorytmu** : czy algorytm został zaimplementowany poprawnie a wynik odpowiada prawidłowej (określonej zbiorem danych testowej) wartości.