DOKUMENTACJA PROJEKT ZALICZENIOWY PYTHON – DANIEL KUŁAGA

TEMAT PROJEKTU:

Algorytm Prima - minimalne drzewo rozpinające dla grafu nieskierowanego ważonego spójnego.

Opis projektu:

W skład drzewa rozpinającego grafu wchodzą wszystkie wierzchołki, ale nie wszystkie krawędzie. Minimalne drzewo rozpinające jest drzewem rozpinającym, którego suma krawędzi jest najmniejsza z możliwych. Zaimplementowany przeze mnie algorytm Prima wyznacza wspomniane minimalne drzewo, oprócz tego algorytmu, możliwe jest wyznaczenie go za pomocą np.: algorytmu Kruskala.

Opis algorytmu:

Algorytm Prima, ma następujące kroki:

1. Tworzę drzewo zawierające jeden dowolnie wybrany wierzchołek.
2. Tworzę kolejkę priorytetową, zwierające wierzchołki osiągalne z MDR, o najmniejszym priorytecie (czyli najmniejszej wadze krawędzi).
3. Wśród nieodwiedzonych wierzchołków (spoza MDR) wybieram ten, dla którego koszt dojścia z obecnego MDR jest najmniejszy.
4. Dodaje do obecnego MDR wierzchołek i krawędź realizującą najmniejszy koszt.
5. Aktualizuję kolejkę priorytetową, uwzględniając nowe krawędzie wychodzące z dodanego wierzchołka.
6. Kroki od pkt. 3 do pkt.5 powtarzam dopóki drzewo nie obejmuje wszystkich wierzchołków grafu.
7. Zwracam minimalne drzewo rozpinające, co kończy algorytm.

Szczegóły implementacji:

W skład kodu zawartego w pliku main.py , wchodzą 4 klasy:

`class QueueItem:` klasa opisująca cechy obiektów dołączanych do kolejki priorytetowej.

`class PriorityQueue:` klasa opisująca kolejkę protytetową (metody dołączania kolejnych elementów, usuwania, sposobu wyświetlania itd.).

`class Edge:` klasa reprezentująca krawędzi w grafie.

`class Graph``:` klasa reprezentująca graf, występują w niej następujące metody:

`add_node(self, node)` wstawianie wierzchołku do grafu

`add_edge_undirected(self, Edge)` Dodaje krawedź do grafu nieskierowanego

`list_nodes(self)` Zwraca listę wierzchołków grafu

`list_node_edges(self)` Zwraca listę krawędzi konkretnego wierzchołka

`print_graph(self)` wyświetla graf

PrimAlgorithm(self, selected_node): metoda implementująca algorytm Prima

Instrukcja korzystania z programu:

Po uruchomieniu pliku main.py pojawia się interfejs konsolowy:

```
Witaj w programie generującym minimalne drzewo rozpinające algorytmem Prima
Wybierz: A - gotowy graf, B- chce podać swoj graf :
```

Możemy wybrać A , wtedy zostanie wygenerowany graf spójny, ważony zainstalowany domyślnie.

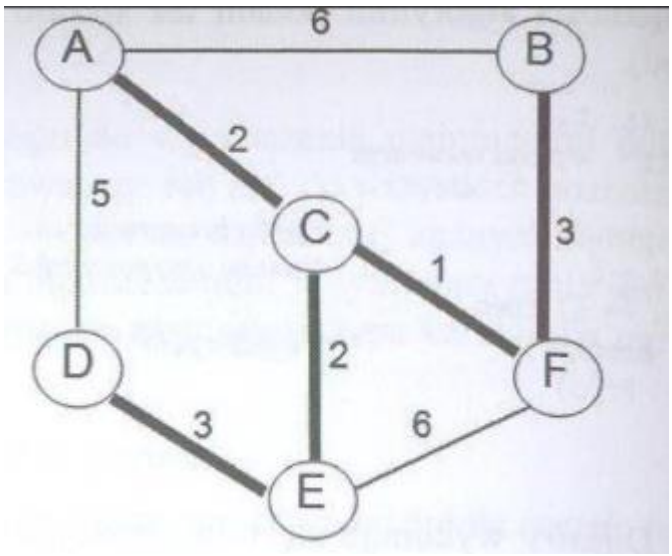
Przy wyborze B, użytkownik zostanie poproszony o podanie następujących danych:

```
Witaj w programie generującym minimalne drzewo rozpinające algorytmem Prima
Wybierz: A - gotowy graf, B- chce podać swoj graf : B
Podaj ilość krawedzi twojego grafu spójnego i ważonego: 8
Podaj dane krawędzi: wierzcholek_startowy(spacja)wierzcholek_koncowy(spacja)waga_krawedzi A C 2
Podaj dane krawędzi: wierzcholek_startowy(spacja)wierzcholek_koncowy(spacja)waga_krawedzi A D 5
Podaj dane krawędzi: wierzcholek_startowy(spacja)wierzcholek_koncowy(spacja)waga_krawedzi B A 6
Podaj dane krawędzi: wierzcholek_startowy(spacja)wierzcholek_koncowy(spacja)waga_krawedzi
```

Następnie użytkownik podaje, od którego wierzchołka algorytm ma zacząć pracę i poniżej otrzymuje wygenerowane MDR:

```
Podaj od którego z powyższych wierzchołków ma zacząc algorytm: A
A : C(2)
C : A(2) F(1) E(2)
F : C(1) B(3)
E : C(2) D(3)
D : E(3)
B : F(3)
```

Powyższy graf: (pogrubione linie to minimalne drzewo rozpinające)



Źródła:

https://pl.wikipedia.org/wiki/Algorytm_Prima

https://docs.python.org/3/library/heapq.html

P.Wróblewski – Minimalne drzewa rozpinające