Dawid Mazurkiewicz 130547

Projekt – Wyszukiwarka połączeń komunikacji miejskiej dla miasta Kraków

Link do projektu: https://github.com/Dydek123/Jakdojade

Założenia projektowe kodu:

Głównym zadaniem projektu jest wyszukiwanie bezpośrednich oraz najkrótszych połączeń komunikacji miejskiej oraz znalezienie rozkładów jazdy danej linii. Całość projektu opiera się na bazie danych MPK Kraków. Wyszukiwanie najkrótszych tras będzie opierało się na zliczaniu przystanków pomiędzy dwoma punktami i wybraniu najmniejszej liczby.

Ogólny opis kodu:

1. Połączenia bezpośrednie:
   Algorytm wyszukiwania połączeń bezpośrednich opiera się na tworzeniu kolejnych zapytań do bazy danych. Użytkownik wprowadza przystanek startowy i końcowy jako dane. Po wykonaniu odpowiednich funkcji, na ekran zostaje wypisana liczba znalezionych tras bezpośrednich oraz odpowiadające tym trasom numery linii.

2. Najkrótsze trasy:
   Na początku programu tworzony jest graf dostępnych połączeń pomiędzy przystankami autobusowymi. Jako krawędzie grafu zapisuję połączenia pomiędzy dwoma najbliższymi przystankami (krawędzie mogą zostać wpisane podwójnie tylko w odwróconej kolejności, ponieważ niektóre linie mają różne trasy w zależności od kierunku). Następnie, po wprowadzeniu danych przez użytkownika, za pomocą algorytmu BFS zostają znalezione najkrótsze trasy pomiędzy wprowadzonymi przystankami.

3. Rozkład linii
   Jako dane użytkownik wprowadza numer linii, dla której ma zostać wyświetlona trasa. Za pomocą zapytań do bazy danych znajduję wszystkie przystanki pośrednie dla danej linii i wypisuję je na ekran

Co udało się zrobić, z czym były problemy

Udało się zrealizować wszystkie główne założenia programu, jednak zostały napotkane pewne problemy:

- W bazie danych brak jest danych odnośnie czasów odjazdu autobusów/tramwajów z przystanków, dlatego wybieranie najkrótszej trasy, odbywa się na podstawie ilości przystanków pomiędzy przystankami, zamiast czasu jazdy.
- Przy wyszukiwaniu najkrótszych tras, zdarza się, że ilość znalezionych kombinacji połączeń jest ogromna, dlatego na ekran wypisywane zostają jedynie połączenia z najmniejszą liczbą przesiadek.
- Okazało się, porównywanie znalezionych wyników na różnych trasach z uwzględnieniem kary za przesiadkę trwała zbyt długo, dlatego postanowiłem ograniczyć wyszukiwanie tras do jednej najkrótszej.

Brak zauważonych problemów z testami.

Linki do istotnych fragmentów kodu:

1. List comprehensions :
    - https://github.com/Dydek123/Jakdojade/blob/74faa7520c9da30e26d6eb2e651ae9ba19ac4ad5/szukaj.py#L290
    - https://github.com/Dydek123/Jakdojade/blob/74faa7520c9da30e26d6eb2e651ae9ba19ac4ad5/szukaj.py#L328-L330
    - https://github.com/Dydek123/Jakdojade/blob/74faa7520c9da30e26d6eb2e651ae9ba19ac4ad5/szukaj.py#L374-L375

2. Klasy:
    - https://github.com/Dydek123/Jakdojade/blob/74faa7520c9da30e26d6eb2e651ae9ba19ac4ad5/szukaj.py#L257-L529
    - https://github.com/Dydek123/Jakdojade/blob/74faa7520c9da30e26d6eb2e651ae9ba19ac4ad5/szukaj.py#L532-L586
    - https://github.com/Dydek123/Jakdojade/blob/74faa7520c9da30e26d6eb2e651ae9ba19ac4ad5/szukaj.py#L589-L699
    - https://github.com/Dydek123/Jakdojade/blob/74faa7520c9da30e26d6eb2e651ae9ba19ac4ad5/szukaj.py#L702-L729
3. Wyjątki:
    - Zdefiniowane własnych klas wyjątku: https://github.com/Dydek123/Jakdojade/blob/74faa7520c9da30e26d6eb2e651ae9ba19ac4ad5/szukaj.py#L45-L65
    - Rzucanie/łapanie wyjątków https://github.com/Dydek123/Jakdojade/blob/74faa7520c9da30e26d6eb2e651ae9ba19ac4ad5/szukaj.py#L223-L239 https://github.com/Dydek123/Jakdojade/blob/74faa7520c9da30e26d6eb2e651ae9ba19ac4ad5/szukaj.py#L242-L254
4. Podział na moduły:
    - https://github.com/Dydek123/Jakdojade/blob/74faa7520c9da30e26d6eb2e651ae9ba19ac4ad5/szukaj.py
    - https://github.com/Dydek123/Jakdojade/blob/74faa7520c9da30e26d6eb2e651ae9ba19ac4ad5/GUI.py
5. Podpowiedzi przy wprowadzaniu przystanków(po naciśnięciu przycisku tab, funkcja może nie działać w przypadku uruchamiania przez programy np. PyCharm)
    - https://github.com/Dydek123/Jakdojade/blob/74faa7520c9da30e26d6eb2e651ae9ba19ac4ad5/szukaj.py#L67-L83
6. Generowanie grafu i algorytm BFS
    - https://github.com/Dydek123/Jakdojade/blob/74faa7520c9da30e26d6eb2e651ae9ba19ac4ad5/szukaj.py#L86-L194