Wstęp do programowania w C

Robert Ferens

Lista 8 - 7 grudnia 2021

Zadanie 1

Napisz w osobnym module kolejkę priorytetową opartą na strukturze kopca. Możesz zrobić to za pomocą tablicy, lub pełnej struktury drzewa, powinna jednak obsługiwać dowolną ilość danych(być allocowana i/lub reallocowana dynamicznie). Struktura ma obsłużyć wstawienie elementu oraz pobranie największej wartości oraz usuwanie największego elementu. Ma ona działać na elementach typu int, jednak funkcja porównująca powinna być pobrana/pobierana od użytkownika(który mógłby np. traktować te liczby jako elementy tablicy).

W głównym pliku pobierz listę imion, nazwisk i wieku:

```
Gustaw Krawczyk 70
Juliusz Zielski 12
Miroslaw Witkowski 56
Klaudiusz Rutkowski 30
Aleksy Pietrzak 23
Kajetan Przybylski 13
Filip Wasilewski 5
Konrad Pawlak 19
Kamil Kubiak 26
Eryk Krupa 67
Anatolia Gorska 87
Bogumila Gajewska 45
Aniela Sikora 21
Florencja Szymczak 3
Aisha Urbanska 6
Oktawia Sikora 15
Aleksandra Sikorska 57
```

Stwórz strukturę do przechowywania tych danych, posortuj je wkładając wszystkie do kolejki priorytetowej, a następnie wyciągając cały czas najmniejszy element(przez odpowiednie ustawienie funkcji porównania).

Na końcu wygeneruj losowo 5000 liczb i podobnie posortuj je i wypisz.

Zadanie 2

Wrocławski Rower Miejski udostępnia dane o wszystkich przejazdach z ostatniego tygodnia `https://www.wroclaw.pl/open-data/dataset/wrmprzejazdy_data`. W danych tych najbardziej interesująca jest nazwa przystanku początkowego, końcowego i czas przejazdu z dokładnością do minut. Ilość danych jest jednak ograniczona - przez wzgląd na zimę mniej osób korzysta z rowerów. Nie ma żadnych informacji o czasie przejazdu z niektórych przystanków do innych np. z Joliot Curie na Stadion Olimpijski. Prowadzący chciałby jednak dowiedzieć się coś o czasach dojazdu z Joliot Curie do każdej innej stacji WRM.

Przyjmijmy zatem uproszczony model zgodnie z którym czas dojazdu z jednej stacji do drugiej trwa tyle co najkrótsza możliwa napotkana w danych przeprawa przez stacje pośrednie bądź bezpośrednia. Np.

```
Trasa: 'Joliot-Curie (Uwr)' --23m-> 'Stadion Olimpijski'
Przez: Joliot-Curie (Uwr) --9m-> Nowowiejska / Górnickiego --7m->
    -> al. Kochanowskiego / Kopernika --7m-> Stadion Olimpijski
```

Zadanie w odpowiedzi powinno wyświetlić posortowane(do sortowania można użyć swojego kopca albo funkcji sort z algorithm.h) od najmniejszego czasu trasy przejazdów do wszystkich stacji wraz z czasami przejazdu. Format może być podobny do podanego wcześniej przykładu. Jaka jest najbliższa i najdalsza stacja wg tych danych?

Zadanie można zrobić za pomocą algorytmu Dijkstry - stworzyć graf przejazdów, poustawiać początkowe czasy bezpośrednio na podstawie danych, a potem obliczyć najkrótsze trasy (algorytm jest dość wyczerpująco opisany w sieci). Przy obliczaniu zadania za pomocą Dijkstry przyda się struktura kolejki priorytetowej z poprzedniego zadania(`https://pl.wikipedia.org/wiki/Algorytm_Dijkstry`). Alternatywnie bez kolejki można użyć algorytmu Floyda-Warshalla.

Prowadzący dla ułatwienia przygotował lekko przetworzone dane z 6 grudnia - w jednym pliku wypisał wszystkie numery stacji wraz z ich nazwami po średniku, w drugim zbiorcze czasy przejazdów na trasie w postaci:

```
P1 P2 MEAN MIN COUNT #komentarz o jakie stacje chodzi
```

```
P1, P2 -int- to odpowiednio numery stacji początkowych i końcowych zgodne z pierwszym plikiem
MEAN - float - średni czas przejazdu z danych - niepotrzebny dla tego zadania
MIN - int - najkrótszy czas przejazdu - przydatny dla tego zadania
COUNT - int - ilość przejazdów na tej trasie
```

Można ale nie trzeba korzystać z tej pomocy(zaletą niekorzystania jest możliwość łatwego zaktualizowania wyników dla bardziej aktualnych danych). Przy przetwarzaniu z danych zostały usunięte informacje o parkowaniu poza stacją.

Zadanie 3

Dostępne ze sprawdzaczką na skosie.