Specyfikacja implementacyjna

Piotr Szumański

Listopad 2020

1 Informacje ogólne

1. Nazwa programu
   Projekt indywidualny 2020

2. Poruszany problem
   Minimalizacja kosztów sprzedaży szczepionek przy jednoczesnym zapewnieniu dostaw do wszystkich aptek.

2 Wstep

W projekcie mamy doczynienia z jednym z rodzaju problemu marszrutyzacji. W zwiazku z tym znalezienie idealnego rozwiazania nie jest takie proste i oczywiste. Dlatego w moim projekcie zastosuje jeden z algorytmów heurystycznych, które jest rozwiazaniem problemu dla niektórych kombinacji danych.

3 Opis działania programu

1. Program prosi użytkownika o podanie ścieżki do pliku z danymi

2. Pobiera dane z podanego pliku za pomoca klasy FileReader i wpisuje je do odpowiednich list w klasie Result w zależności od rodzaju danych (lista producentów, aptek czy połaczeń)

3. Po pobraniu danych do list, sortuje liste połaczeń po cenie (od najtańszego) za pomoca Collection.sort

4. Nastepnie iteruje po tej liście połaczeń i najpierw pobiera dane producenta i apteki, których id jest w rozpatrywanym połaczeniu

5. Wybiera najmniejsza z trzech wartości

   - dziennej produkcji
   - dziennego zapotrzebowania
   - maksymalnej liczby dostarczanych szczepionek

6. Wstawiamy odpowiednie wartości do pliku wynikowego (wynik.txt), a mianowicie:

   > Producent X − > Apteka Y [Koszt = (najm. watość) * (cena) = wynik]

7. Dodajemy wynik do sumy opłat całkowitych
8. Użyta najmniejsza wartość odejmujemy od dziennej produkcji i dziennego zapotrzebowania
9. Bierzemy nastepne połaczenie
10. Po iteracji wstawiamy sume opłat do pliku z wynikami jako:

    > Opłaty całkowite: "suma" zł

    W przypadku błednych danych w dokumencie czy źle napisanej ścieżki do pliku program natychmiast powiadomi użytkownika o błedzie

4 Opis obiektów

W tych obiektach wystepuja odpowiednie konstruktory oraz gettery dla wszystkich podanych zmiennych.

Manufacturer - obiekt producent

- int id - id producenta
- String name - nazwa producenta int dailyProduction - dzienna produkcja (do tej zmiennej używany jest dodatkowo setter)

Pharmacy - obiekt apteka

- int id - id apteki
- String name - nazwa apteki
- int dailyRequire - dzienne zapotrzebowanie (do tej zmiennej używany jest dodatkowo setter)

Connection - obiekt połaczenie, które implementuje interfejs Comparator<Connection>

- int idM - id producenta
- int idP - id apteki
- int maxDaily - dzienna maksymalna liczba dostaczanych szczepionek

- double price - cena
- compareTo() - metoda nadpisana od Comparatora, napisana by porównywała połaczenia po cenie, a w przydadku remisu porównywała po max. liczbie dostarczeń (odwrotnie jak cena)

5 Opis klas i metod

1. Main - klasa z główna metoda inicjalizujaca działanie programu
   - main() - prosi o wprowadzenie ścieżki pliku oraz wywołuje klase Result
2. Result - oblicza najlepsza możliwość na podstawie dostarczonej ścieżki do pliku i wysyła ja do pliku wynik.txt
   - void calculate() - metoda z algorytmem
   - Manufacturer findManufacturer(int idM) - zwraca producenta na podstawie id producenta z danego połaczenia
   - Pharmacy findPharmacy(int idP) - zwraca apteke na podstawie id apteki z listy połaczeń
   - int minValueOfThree(int a, int b, int c) - zwraca najmniejsza z trzech podanych wartości
   - void writeAnwser(String line) - wysyła podany ciag znaków do pliku wynik.txt
3. FileReader - wczytuje odpowiednie dane do danych list:
   - List<Manufacturer> manufacturers - lista producentów
   - List<Pharmacy> pharmacies - lista aptek
   - List<Connection> connections - lista połaczeń

   Dla tych list zostana dodane gettery, aby klasa Result miała do nich dostep.
   - loadData(String filePath) - metoda do wczytywania danych z pliku na podstawie podanej ścieżki, wywoływana w klasie Result
   - void addManufacturer(String line) - dodaje nowo wytworzonego producenta do listy producentów, pobiera dane z danego ciagu znaków
   - void addPharmacy(String line) - dodaje nowo wytworzona apteke do listy aptek, pobiera dane z danego ciagu znaków
   - void addConnection(String line) - dodaje nowo wytworzone połaczenie do listy połaczeń, pobiera dane z danego ciagu znaków

6 Zakończenie

W przydadku powodzenia sie tego algorytmu możliwa jest opcja dodania jeszcze jednego algorytmu dla zwiekszenia prawdopodobienstwa znalezienia poprawnej konfiguracji oraz dodania testów dla każdego z tych algorytmów.