Задание практикума, поиск путей в графе

1 Описание

Предполагается, что программа пишется для некоторого условного пассажира, который собирается путешествовать между городами. Предполагается, что транспортная система представлена несколькими разными видами транспорта, например: **самолёты**, **поезда**, **автобусы**, **паромы** и т.д.. Для каждого транспорта если он представлен в данном городе указано:

- город отправления (*from_city*),
- город прибытия (*to_city*),
- тип транспорта (*transport_type*),
- время проезда (*cruise_time*),
- стоимость проезда (*cruise_fare*).

Пассажиру надо доехать из одного города в другой, если города напрямую не соеденнины пассажир будет покупать несколько билетов. В нашей модели транспортной системы, в отличии от реальной ситуации, будем считать, что если транспорт физически следует транзитом через несколько городов, то пассажиру продаётся отдельный билет на каждую пару городов задействованных в транзите.

Требуется написать программу, которая будет находить следующее:

1. Среди кратчайших по времени путей между двумя городами найти путь минимальной стоимости. Если город достижим из города отправления.
2. Среди путей между двумя городами найти путь минимальной стоимости. Если город достижим из города отправления.
3. Найти путь между 2-мя городами минимальный по числу посещённых городов.
4. Найти множество городов, достижимых из города отправления не более чем за *limit_cost* денег и вывести кратчайшие по деньгам пути к ним.
5. Найти множество городов достижимое из города отправителя не более чем за *limit_time* времени и вывест кратчайшие пути по времени к ним.

Для всех вариантов поиска должно быть предусмотрено ограничение на виды транспорта, которым желает пользоваться пассажир.

Для нахождения путей и областей достижимости предлагается провести модификацию следующих классических алгоритмов:

1. Алгоритм Дейккстры. `https://ru.algorithmica.org/cs/shortest-paths/dijkstra/`.
2. Алгоритм Белмана Форда. `https://ru.frwiki.wiki/wiki/Algorithme_de_Bellman-Ford`, `https://temofeev.ru/info/articles/algoritm-bellmana-forda/`.
3. алгоритм поиска в Ширину `https://ru.algorithmica.org/cs/shortest-paths/bfs/`.

2 Интерфейс

В параметрах программе передаётся имя файла с графом (множеством всех связей между городами). Программа устраивает интерактивный диалог с пользователем (использование модуля ncurses6 возможно). Программа запрашивает у пользователя (пассажира) в каком режиме он будет осуществлять поиск, либо что он вообще хочет выйти из программы. Затем программа запрашивает множество видов транспорта, которым собирается или наоборот не собирается пользоваться пользователь. Можно это организовать например как номера видов транспорта перечисленные через запятую. Далее запрашиваются город отправления, город прибытия, либо город оправления и ограничения. В ответ программа должна выдать путь, либо для других режимов множество городов.

Для каждого города необходимо напечать стоимость, время добирания до него. А для печати пути ещё указать вид транспорта, которым необходимо воспользоваться. После выдачи ответа переходим обратно на шаг, когда ждём от пользователя новый запрос или желания выйти из программы.

Должны быть обработаны ошибочные ситуации: несуществующий вид транспорта, несуществующий город, и.т.п.

3 Формат входного файла

В файле на каждой строчке указывается: город отправления, город прибытия, тип транспорта, время проезда, стоимость проезда. Города и тип транспорта пишутся в двойных кавычках. Стоимость и время проезда 32-х битные беззнаковые числа записанные как текст в десятичной системе счисления. Поля разделяются пробельными символами, могут встречаться пустые строки и коментарии. Коментарии начинаются с символа решётка.

```
# Описание некоторого простейшего графа
# поля: "from_city" "to_city" "transport_type" cruise_time cruise_fare
"Москва" "Санкт-Петербург" "самолёт" 60 7000
"Санкт-Петербург" "Москва" "самолёт" 60 7500

"Москва" "Тверь"  "поезд" 180 200
"Тверь"  "Москва" "поезд" 200 200

"Тверь" "Бологое" "поезд" 120 150
"Бологое" "Тверь" "поезд" 120 150

"Бологое" "Новгород Великий" "поезд" 140 250
"Новгород Великий" "Бологое" "поезд" 140 250

"Бологое" "Санкт-Петербург" "поезд" 240 400
"Санкт-Петербург" "Бологое" "поезд" 240 400

"Москва" "Тверь" "автобус" 240 600
"Тверь" "Москва" "автобус" 240 600
"Тверь" "Новгород Великий" "автобус" 240 600
#
# обратного рейса автобуса нет
#
"Санкт-Петербург" "Тверь" "автобус" 420 3000 # да дороговато будет.

"Новгород Великий" "Санкт-Петербург" "автобус" 180 500
"Санкт-Петербург" "Новгород Великий" "автобус" 180 500

"Санкт-Петербург" "Бологое" "автобус" 180 1000
"Бологое" "Санкт-Петербург" "автобус" 180 1000
```

Каждый уважающий себя студент может дополнить пример по вкусу и довести граф до описания транспорта сразу на всём земном шаре. В приведённом примере стоимость проезда и длительность могут быть далеки от реальной действительности.

4 Требования к реализации

Для рёбер графа (рейсов) требуется создать собственный класс, где вместо имён городов и видов транспорта должны хранится их идентификаторы. считается, что городов и видов транспорта не может быть больше, чем 2^32. Пути реализовать отдельным классом и определить для них *operator+*, добавляющий новый элемент к пути, и *operator []* для доступа к элементу пути. При выполнении задания предолагается по максимому использовать контейнеры STL.

По завершению каждой операции и по завершеню всей программы сообщить максимальное значение потраченной физической памяти (max_rss) и астрономическое время работы программы. В коде не должно быть ошибок обращения в память, и не должно быть утечек памяти.