Домашнее задание 1.

Расписание городского транспорта


# 1 Введение

Технологический прогресс существенно изменил то, как функционируют современные города . Заказ такси через мобильное приложение, каршеринг, точки доступа в Интернет , веб-камеры в режиме реального времени, интеллектуальное освещение - вот лишь несколько примеров. Одно из больших изменений произошло в сфере общественного транспорта.

Домашние задания для курса программирования (весна 2018 г.) будут направлены на создание информационной системы для управления городским транспортом

система. Что ж, как вы увидите, это будет всего лишь очень упрощенная модель реальной модели.

По мере прохождения курса и изучения новых тем вы будете постепенно улучшать свое программное обеспечение, добавляя новые компоненты и изменяя существующие. Однако задания будут оцениваться независимо друг от друга.

В этом первом задании вам нужно будет разработать модель упрощенной системы планирования перевозок и алгоритм для отображения в режиме реального времени отправлений в разных местах. Мы применим следующие ограничения:

- Рассматривается только один вид наземного транспорта (например, автобус, трамвай) - вы можете выбрать любой вид по своему усмотрению.
- Фактическое движение транспорта не отслеживается, система учитывает только фиксированное расписание (никаких задержек, отмен, поломок и подобных проблем).
- Время между любыми двумя остановками на маршруте одинаково в обоих направлениях.
- На всех маршрутах движение начинается и заканчивается одновременно на обоих концах. Интервал между последующими вылетами постоянен и не меняется в течение дня.

## 2 Проверенные навыки

- Базовый ООП и состав / агрегирование классов
- Коллекции
- Файловый ввод-вывод

## 3 Описание задач

Вам необходимо сохранить следующую информацию о расписании общественного транспорта:

- Названия остановок / станций
- Маршруты. Единый маршрут формируется из последовательности остановок/станций. Для каждого маршрута вам необходимо сохранить время первого и последнего отправления с каждой конечной остановки/ вокзала (terminus) и интервал между автобусами / трамваями / поездами в минутах. Вам также необходимо запомнить время, необходимое для того, чтобы добраться до каждой из станций, образующих маршрут, начинающийся с конечной остановки.

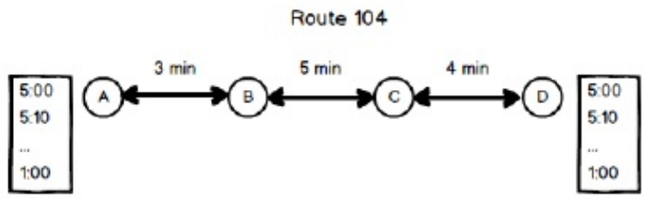
  Например, рассмотрим абстрактный автобусный маршрут 104, который следует через 4 остановки: A,B, C, D (см. Рисунок ниже). Предположим, что движение с начинается с 5:00 утра как от пункта A, так и от пункта D и продолжается с 10-минутным интервалом до 01:00. Время в пути между станциями, указанное на рисунке, означает, что на станцию B будут прибывать автобусы из пункта A в пункт 5:03, 5:13, 5:23, и т.д., последний

автобус прибывает в 01:03 ночи. В те же дни
автобусы прибудут в B из C в 5:09, 5:19, 5:29, и т.д.

Обратите внимание, что одна и та же остановка / станция может быть
частью нескольких маршрутов (будет показано в другом примере ниже).

Ваша главная задача в этом задании - придумать алгоритм,
который, учитывая текущее время и станцию,
печатает расписание ближайших отправлений с нее.

Эффективность алгоритма будет в значительной степени зависеть от структуры данных, которую вы решите использовать. Может даже существовать несколько независимых структур данных.




Информация должна быть сохранена в любом текстовом формате и
загружена из файла при запуске программы. Вы будете работать с файлом в режиме
только для чтения в этом задании информация может быть изначально
сохранена с помощью любого текстового редактора.

## Выполнение задания

Работа в консольном приложении.

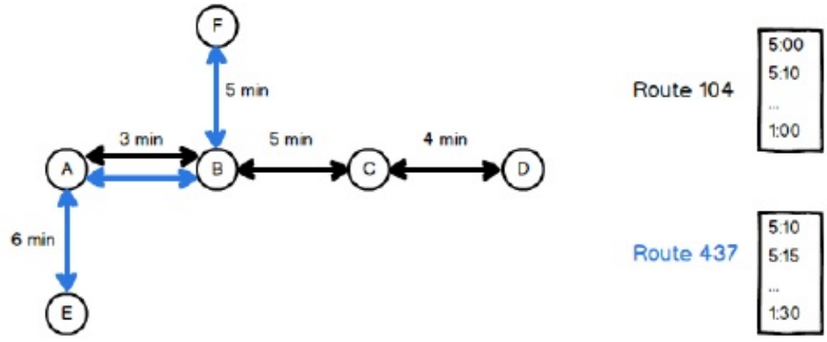
- (3 балла) Разработайте модель данных для данной предметной области и
  закодируйте ее в наборе классов вашей программы на C #. Возможно, вы
  не сразу подберете подходящую модель. Будет идеально, если вы измените
  ее по мере разработки основного алгоритма.

- (4 балла) Реализуйте основной алгоритм, который принимает станцию и
  возвращает список ближайших автобусов / трамваев / поездов всех маршрутов, проходящих
  через станцию. Чтобы протестировать алгоритм, вы можете сначала создать образец
  набора данных в самом программном коде (жестко запрограммируйте свои данные).

- (2 балла) Создайте текстовый файл, в котором хранится вся необходимая
  информация о расписании перевозок. Ваша программа должна
  считывать всю информацию из этого файла при запуске. После
  выполнения этого шага удалите все жестко закодированные данные.

- (1 балл) Внедрите рабочий процесс взаимодействия с пользователем. Это
  должен быть бесконечный цикл, в котором пользователь вводит название
  станции, а программа печатает информацию о следующих отправлениях с нее.
  Программа должна завершиться , когда пользователь введет пустую строку.

Пример потока программы




Здесь приведены два автобусных маршрута. Первый из них уже был описан выше. Второй маршрут (437) состоит из станций E-A-B-F, таким образом, станции A и B и отрезок между ними являются общими для двух маршрутов. Обратите внимание, что у второго маршрута другое расписание (автобусы отправляются с 5:10 с 5-минутным интервалом до 01:30).

В приведенной ниже программе ввод данных пользователем показан синим цветом:

```
Введите станцию: B
Текущее время:
14:50 Расписание:

437, пункт назначения E, 0
мин. 104, пункт назначения D,
3 мин. 437, пункт назначения F,
4 мин. 104, пункт назначения A, 9
мин. Вход на станцию: A
Текущее время: 14:52
Расписание: 437, пункт
назначения E, 1 минута 437,
пункт назначения F, 4 минуты
104, пункт назначения D, 8
минут Вход на станцию: H
Нет такой станции!
```

## 4 Подсказки

- Используйте DateTime.Теперь, чтобы узнать текущее время
- Возможно, вам окажется проще хранить время в виде единственного целого числа, вычисляемого как час * 60+ минута, например, 14:53 равно 893, поэтому любое время в течение дня попадает в диапазон [0..1439]

## 5 Отправка

Отправьте свое решение, выполнив следующие действия:

1. Удалите две временные папки - bin и ob j из папки проекта.
2. Добавьте всю папку с решением в ZIP-файл (не RAR или 7Z) Архив.
3. Загрузите архив в Canvas LMS в соответствующем разделе

## 6 Политика выставления оценок

Окончательная оценка за задание обязательная защита в выставляется после урока.

Как правило, оценка определяется количеством и качеством выполненных заданий. Оценка может быть снижена в следующих случаях:

- Неэффективная реализация алгоритма (-1 балл)
- Информация, многократно дублируемая в файле (-1 балл)
- Плохой стиль программирования (-1 балл) (уточните у своего инструктора определение плохого)