Homework 5: Train Schedule

1. Objective

Between the two largest cities of Russia, St. Petersburg and Moscow make daily trips of N trains. For each train, it is known its departure time from St. Petersburg and the time of arrival in Moscow. You need to find the fastest train and its speed. It is guaranteed that the fastest train is determined uniquely.

In this project you need to use:

1. Objects
2. Arrays
3. File Input
4. User Interface

In this homework, you need to use the user interface client code for working with your supplier classes. See the reference on the page.

1. Input Data Files and storage

The text file INPUT.TXT, your program will read from, follows this format:

1. start with one integer on its own line represents the distance between two cities
2. next line – one integer (the number of trains in the everyday schedule 1 ≤ N ≤ 100)
3. followed by a series of lines that describe every train. This description contains 3 pieces of information:
   * **a train name** - a string with a length of 1 to 50 characters. It can contain letters of the English alphabet, spaces, numbers, and dashes ("-"). Lowercase and uppercase letters in the names of trains vary.
   * **time of departure and arrival** is indicated in a 24-hour format as follows HH MM. The format specifies hours (2 digits) and minutes (2 digits) separated by a space. The journey time for each of the trains is at least one minute and does not exceed 24 hours. Pay attention to this example, if the train departs at 19:50 and arrives at 8:30, it means that it arrives the next day and the time of journey is 12 hours and 40 minutes.

Your program does not have to be responsible for files that do not match this format (in other words, if the end user gives you a filename with bad data and the program crashes, that's ok). You can create any text file you want for testing (use a program like Notepad or any other basic text editor). Here is a file I've created for you to use: trains.txt.

1. Output

Print the name of the fastest train and its speed. Speed output in kilometers per hour and round to the nearest whole by mathematical rules. Follow the output format shown in the examples.

1. Code Specification

Implement the class specifications below. To get full credit, your program's public interface must match these descriptions **exactly**.

**Objects You’ll Create**

Here are UML Class Diagrams for the objects you are to create. Pay attention to the diagram notation indicating whether methods are public (+) or private (-); ask questions if you need clarification. Understanding the model is of *critical* importance here.

|  |  |  |
| --- | --- | --- |
| **Class: Time** |  | **Class: Train** |
| **Properties**  *(you figure out the private data needed)* |  | **Properties**  (*you figure out the private data needed*) |
| **Constructor**  +Time(hour : Integer, minute : Integer)  **Accessors**  + getHour() : Integer  + getMinute() : Integer  **Mutators**  +setHour(Integer)  +setMinute(Integer)  +timeBetween(Time) : Time  +zeroTime(Integer) : String  +toString() : String |  | **Constructor**  +Train(name : String, departure : Time, arrival: Time, distance : Integer)  **Accessors**  +getDeparture () : Time  +getArrival () : Time  +getDistance () : Integer  **Mutators**  +setDeparture(Time)  +setArrival(Time)  +setDistance(Integer)  +averageSpeed() : Integer  +travelTime() : Time  +toString() : String |