# Quiz 2

**Subject:** Classes and Inheritance
**Due Date:** 02.04.2023 23:59

## Introduction

In this quiz, you are expected to write a program that records certain information about train trips. You should to design your program according to the structure given in diagram. The class definitions, access specifiers, and functions in your program must match the diagram shown in Figure 1. Some important variables for loops, flags, numbers, etc. are not specified in the diagram, so you can define them as your own additional variables in the program.
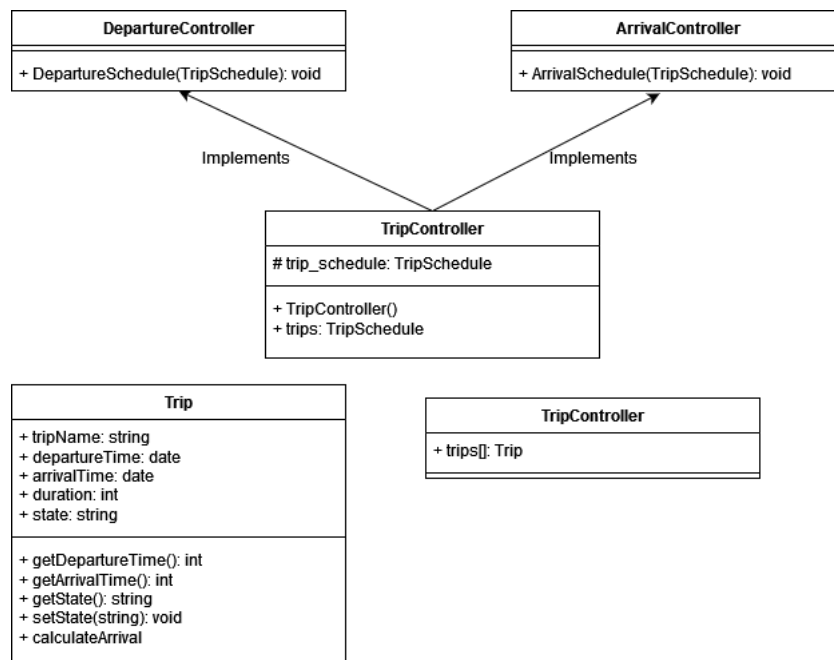


Figure 1: UML Diagram

## Experiment

Your program will have a class called **TripController**, which is implements from two interfaces, **DepartureController** and **ArrivalController**. The **TripController** class uses methods from the **DepartureController** and **ArrivalController** interfaces.

Your **TripController** class must have an object of type **TripSchedule** that will contain an array of objects of type **Trip**. You can define this array with 100 elements. When you define **Trip** objects, the constructor of these objects will read the following entries from the input file: **trip name**, **departure time**, and **trip duration**. The state of the Trip object will be "Idle" as default in the constructor, and you will call the function within the Trip class that

will calculate the arrival time for your Trip object using the duration and departure time. The definition of the Trip class also includes the necessary get and set methods specified in the diagram and needed throughout the program.

In the constructor of your **TripController**, you will pass **TripSchedule** object you created to the departureSchedule and arrivalSchedule functions in your base classes, which perform a sort operation between these trip objects. DepartureController checks the departure times of objects and sorts the array in ascending order. After sorting, the departure order of these trips is output sequentially. A similar approach is used in ArrivalController, and after the sort is complete, the arrival order is output. The outputs consist of trip name, departure time, status of the trip. This applies to both departure and arrival controllers.

Class definitions for both the DepartureController and the ArrivalController must also contain all of the necessary members and variables in correct access specifiers and expected types.

Important Remarks:

- You can use the help of any sorting algorithms your own sorting algorithms, bubble sort etc. To order the array of objects in the departure and arrival controllers.

- If there are two or more train trips within train departure or arrival times, their status is changed from the default value "Idle" to "Delayed", other values are printed as default.

- Some constructors and methods are not given in the uml diagram. Please add the necessary constructor and functions to your classes.

## Input and Output

In the input file, you will be given a different trip information on each line according to the format given below. You should produce your output file in the format specified below according to this input file you have read.

```
<tripName>TAB<departureTime>TAB<durationOfTrip>
```

**Example input file:**

```
S224Y   10:10   70
S22LY   10:20   15
S22IG   11:25   90
```

**Example output file:**

```
Departure order:
S224Y depart at 10:10   Trip State:IDLE
S22LY depart at 10:20   Trip State:IDLE
S22IG depart at 11:25   Trip State:IDLE


Arrival order:
S22LY arrive at 10:35   Trip State:IDLE
S224Y arrive at 11:20   Trip State:IDLE
S22IG arrive at 12:55   Trip State:IDLE
```

## Grading Policy

- Your work will be graded over a maximum of 100 points.

- Your total score will be partial according to the grading policy stated below.

| Correct class structure | 20p |
|---|---|
| Correct output | 80p |

## Execution and Test

– Upload your java files to your server account (dev.cs.hacettepe.edu.tr)

– Compile your code (javac *.java)

– Run your program (java Main input.txt output.txt)

– Control your output

### Notes

∗ Do not miss the submission deadline.

∗ Save all your work until the quiz is graded.

∗ You can ask your questions via Piazza and you are supposed to be aware of everything discussed on Piazza.

∗ You must submit your work with the file hierarchy as stated below:

– `<studentid>.zip`
   – `<src>`
      – `Main.java`
      – `DepartureController.java`
      – `ArrivalController.java`
      – `TripController.java`
      – `Trip.java`
      – `TripSchedule.java`
      – `*.java`