SOFE 2710U
Object Oriented Programming and Design
Fall 2018

Assignment2

Individual Programming Assignment #2

Resources

- Eclipse Java IDE, BlueJ, netbeans, or Grasp.
- Notepad++.

Marking Rubric

1. Program design correctness: [15%].
2. Proper comments and documentation: [10%]
   a) Your code is required to be properly commented and necessary to have a header section that shows the author, date, and the assignment number that it belongs to. It further describes the purpose of the program.
   b) In your code, use comments to explain the purpose of the declared variables and the lines of your code.
3. Compilation: [30%] your code compiles without error.
4. Execution and output: [40%] your code is required to be executed without any crashes and the program must produce the correct output.
5. Code appearance and appropriate indentation: [5%] your code must be properly formatted. Matching pairs of { } are required to be in the same column with proper indentation. Lines in the program are required to be uniformly indented and/or separated to enhance the readability.

Deliverables

- The assignment should be submitted by the due date using the Blackboard dropbox.

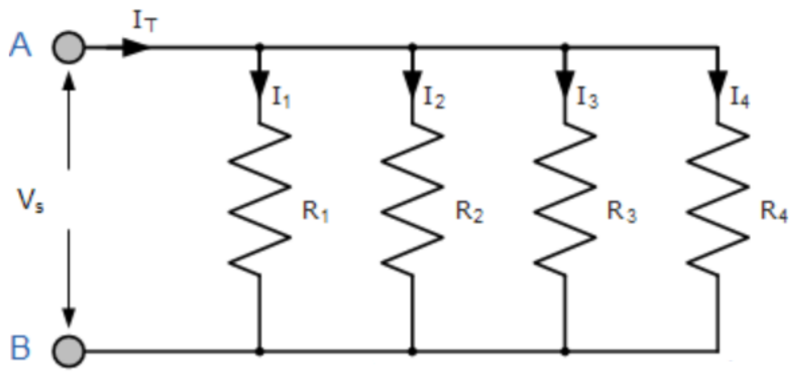
- Copies of the **Java programs**. i.e. (*.java)

Application 1: Calculate the individual branch currents and total current drawn from the power supply for the following set of resistors connected together in a parallel combination. The supply voltage is common to all the resistors in a parallel circuit.

Calculate the individual branch current by using Ohm's Law as follows:

$$I_1 = \frac{V_S}{R_1}$$

$$I_T = I_1 + I_2 + I_3 + I_4$$



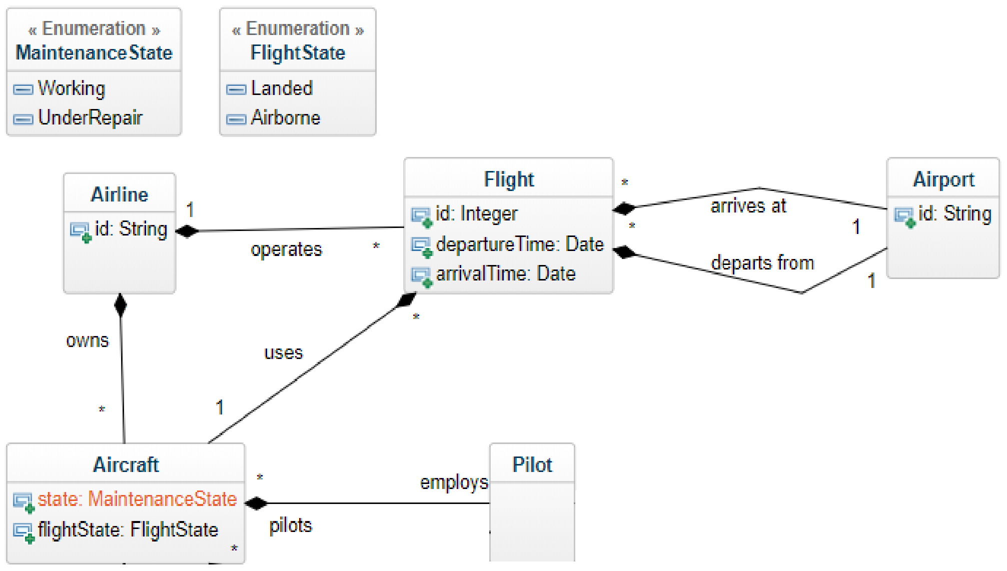
An input file "**data.txt**" (download it from Blackboard) contains several designs for the circuit. Each line of the file provides the source voltage and four resistances listed in order. Some of the designs may generates negative current values. Write a program that reads the input file (the user may use a different file, so don't hard code it in the application, but instead read it as an argument to the application from the command line), copies the designs that regenerate a positive current to a file named ***forwardCurrent.txt***, and displays the list of designs with negative current on the console.

A sample output:

```
C:\Fall2018\Courses\SOFE2710\Code>java -cp . CircuitApplication data.txt
Reading filename: data.txt
Saving positive current to forwardCurrent.txt
The following designs have negative current:
-10.0    10.0    20.0    10.0    10.0
-120.0   100.0   200.0   150.0   30.0
-40.0    34.0    22.0    32.0    12.5
```

Application 2: Airport System:

For this task, you will design and develop an airport system. The application UML diagram is shown below. The application then prints some statistics as described below.



We want to model a system for management of flights and pilots as follows:

- An airline operates flights. Each airline has ID.
- Each flight has an ID and departure airport and an arrival airport.
- Each flight has a pilot and co-pilot, and it uses an aircraft of a certain type. A flight also has a departure time and an arrival time.
- An airline owns a set of aircraft of different types.
- An aircraft can be in a working state or it can be under repair.
- In a particular moment an aircraft can be landed or airborne.
- In this model assume that we have a set of pilots.
- Assume that, an areophane needs minimum a pilot and a co-pilot to operate.
- Skeleton java files will be provided. (Check the assignment folder)

Requirements:

- Write a method in the Aircraft class called `joinPilot(Pilot  pilot)` that adds the given pilot to the Aircraft.
- Write a method in the Aircraft class called `printPilot()` that displays the list of all pilots working the Aircraft in arbitrary order. It should show all pilots information
- Write a method in the Airline class called `owns(Aircraft aircraft)` that add an aircraft to an airline. If the aircraft is already owned by the airline then the method does nothing.

- Write a method in the Airline class called `printFlightByName()` that displays a list of all flights who are owned by the current airline. This method should show all flights information
- Write a method in the Pilots class called `isPilotForFlight(AirCraft aircraft)` that returns a Boolean value indicating whether or not the given pilot is working for any flight in this aircraft.
- Write a method in the Airline class called `pilotsWorkingForAirlines(Airline airline)` that returns an ArrayList containing all pilots who are working in the with the given code.
- Write a method in the Airline class called `getDeparureFlightByAirport(Airport airport)` that return a list of flights that departure from the current airport on a specific date.

- Write a method in the Airline class called `getArrivalFlightByAirport(Airport airport)` that return a list of flights that arrive from the current airport on a specific date.