COP3503 Computer Science 2
Dr. Andrew Steinberg
Spring 2025

Programming Assignment 5
Building Railroads
Due: 4/21/2025 at 11:59pm

Objective: Students will apply concepts of graph algorithms in this programming assignment.

Assignment Description: You have been hired by the Knights Train Company to develop some sort of cost-efficient railroad system to allow students to travel all over the state of Florida. The company wants to spend the least amount of money on developing some sort of railroad system. In order to be super cost efficient, the company decided that the railroad system also cannot have any cycles (**hint** think about how disjoint sets can help; a sample disjoint set class was posted in the assignment page). That means students will have to stop at a couple of stops if they want to get to their respective destination. The company has also provided you with all possible tracks and their cost to design the railroad system. In this assignment, you will apply Kruskal's Algorithm in designing such a cost-efficient railroad system.

For this assignment, you must follow these **requirements**.

1. Create a public class called `Railroad`.
2. The class constructor for `Railroad` takes two parameters.
    a. The first parameter is a primitive int that represents the number of tracks you are given in designing the railroad system. This is the total number of lines in each respective text file.
    b. The second parameter is a String object that contains the name of a text file with all possible tracks the company has provided. NOTE: All text files are in the same directory as your driver file. Do not create any packages. Packages used will result in points being deducted in the respective categories of the rubric with no exception.
3. Inside the driver file, you will see a method `buildRailroad` being called for each test case. This method will perform the main action of designing the cost-efficient railroad system. The method returns a String object that contains message of the tracks used to build such a railroad system along with the total cost to build the railroad system. When each track is displayed, you will need to display the points lexicographically separated by "---". Here is an example from one of the scenarios. *Note: There is an escape tab character between the second vertex and cost of the track.*

```
Orlando---Tampa $100
Jacksonville---Orlando  $120
Miami---Orlando $230
The cost of the railroad is $450.
```

I have also provided a snippet of code that shows the string format to assist with formatting:
`src + "---" + dest + "\t$" + resultingtracks[i].weight + "\n"`

4. Inside the provided text files (each represents a test case), you will find three entries per line separated by a single white space character.
    a. VERTEX VERTEX COST (ORLANDO TAMPA 140)
        i. VERTEX represents the label on a map. Note: Some labels are letters and others are actual city names based on the test case. Your code must be able to handle this type of input labeling.
        ii. COST represents the cost of designing such a track between two vertices.
5. The railroad text files MUST be in the same directory as your java files. Please do not create sub directories. If you utilize sub directories, then points will be deducted.
6. You may use (and it's recommended) helper classes that you can defined in your Railroad.java file.
7. You may create additional helper methods and attributes if needed as long as they are implemented and called in your solution file and NOT called from the driver file. Adding extra methods to call in the driver file will not match to what the graders will use evaluate your code. This will result in a low score with no changes to be applied!
8. Your code must run within 3 seconds on Eustis. If it runs longer than 3 seconds, you will not be eligible for any points in the rubric except for code compilation, program run, code style, code submission/header, and comments.

A driver file (`RailroadDriver.java`) has been provided for you to show you how the methods are called along with 5 test cases. The text files are also provided for you that you will need to use in building the railroads. The text file itself must be in the same directory as the runner file.

What to submit: Submit a file called `Railroad.java` to webcourses. You are not required to submit the driver file as that will be provided for the graders to test your code. Please make sure the driver file provided works for your code. Any name changes may cause your program not to work when graded, which will result in a lower score on the assignment and would not be changed.

What happens if my code doesn't compile or is stuck in an infinite loop? How will my code be graded? If you are code falls under not compiling or is stuck in an infinite loop, you will not be eligible to receive any points in any categories associated in the rubric except for code style, code submission/header, and comments. That means you would receive 0 points in those respective categories. The only full points you would be eligible for are code style, comments, and code submission/header if directions were followed. You will also receive low marks if your code does not compile.

Important Note for running Eustis: Many of you are probably using IDEs like Netbeans and Eclipse to build your Java Solutions. Please note that some of these IDEs will automatically place your Java file in some sort of package. Please make sure your Java file is not defined in

some package as this can result package private errors. Any such error that occurs during the grading will not be fixed and points will be deducted as such in accordance with the respective categories in the rubric. Also, DO NOT create a main method in your solution file!! This will result in your code not running properly with the driver file which will result in points being deducted from the respective categories.