COP3503

Spring 2025

Programming Assignment 5

Connected Campus Network Planner

You have been hired by the National Education Technology Group to plan internet infrastructure across several U.S. states, one at a time. Your job is to connect all public university campuses in a given state with the lowest possible cost, while ensuring no campus is connected by more than one path (i.e., no cycles are allowed).

Each state will have its own input file (e.g., `Florida.txt`, `Texas.txt`, `California.txt`, etc.) containing potential direct cable connections between universities. Each connection includes the names of the two campuses and the cost of installation.

To build a complete network for a state:

- Select the cheapest possible connections
- Avoid creating cycles or redundant connections
- Ensure that all campuses are reachable in the final network

Input Format

Each input file will contain multiple lines, each in the following format:

```
CampusA CampusB Cost
```

Example:

```
UF UCF 120
UCF FSU 300
FSU FIU 200
UF FIU 500
```

- Each line represents a possible cable connection between two universities and the associated cost.
- Names may include abbreviations or full university names.
- No duplicate connections will appear.

Requirements

1. Create a public class named `CampusNetworkPlanner`.
2. The constructor should accept:
   - An `int` representing the number of possible connections (lines in the file)

   - A `String` representing the name of the input file
   - The input file must be placed in the same directory as your Java class and the driver script. Do not use packages in your submission.
3. Implement a method `public String buildNetwork()` that:
   - Selects the connections needed to create a single connected network of campuses
   - Ensures no cycles are formed
   - Returns a formatted `String` showing:
     - Each selected connection (sorted lexicographically by campus names)
     - Total cost of the network
4. Output Format:
   - Each connection: `CampusA---CampusB $Cost`
   - Sort final selection lexicographically by campusA, then campusB (as shown in the sample output)
   - Final line: `Total Cost: $<total>`
   - Your output must be written to the standard output (the terminal window).
5. Runtime Requirement:
   - Your implementation must run in O(E logE) or O(E logV), where E is the number of possible connections and V is the number of vertices in the graph.
   - You must design your code to avoid any operations that would degrade to $O(n^2)$ in worst-case scenarios.
   - You are encouraged to analyze your sorting and cycle-checking logic to ensure compliance.

Sample Output

```
Enter input file name: Florida.txt

Selected Connections:
FIU---FSU   $200
FSU---UCF   $300
UCF---UF    $120

Total Cost: $620
```

Submission Instructions

1. Submit `CampusNetworkPlanner.java` to Webcourses/Canvas (No need to submit the driver file).
2. Ensure all methods match the function prototypes provided.
3. Code should compile successfully in Eustis (Failure to compile will result in a 0).
4. Follow proper coding style and documentation.
5. Ensure your file
   - Compiles and runs correctly with the provided driver

   - Does not contain a `main()` method, your code will be called from the driver script.
   - Matches the required output format

Testing your code:

Your program will be tested using at least 5 different input files, each representing a different U.S. state. For example:

- Florida.txt
- Texas.txt
- Georgia.txt
- California.txt
- Illinois.txt

Each file will contain between 5 and 50 campus connections, with varying university names and connection costs.

Your program will be tested with each file one at a time, and it must produce the correct output for every test case without requiring any modifications to your code. with varying university names and connection costs. Your code should work on all of them with no modifications.