Assignment 4 –Binary Tree

TA:Louis(isuchang@ccu.edu.tw)
Deadline: 11:59 p.m., December 5, 2024

1. Camera Placement in the Campus Tree

You are responsible for the campus security system and need to design a plan to ensure the safety of all important buildings on campus. To do this, you will install surveillance cameras at specific buildings to maximize coverage across the campus.

Your goal is to **minimize** the number of cameras needed to monitor all the buildings. The distance between the campus security center (root node) and each building is used as the key for each node in a binary search tree. After constructing the binary search tree, you will determine the minimum number of cameras required and identify the specific buildings where the cameras should be placed.

Note: The distance of the security center is 0, and it serves as the root node of the binary search tree.

Your first input is the number of buildings n, the next input is the location of the campus security center, formatted as longitude and latitude information separated by a space, and the next n sets of inputs, each of which includes the name of the building and its latitude and longitude. As the example is shown in TABLE I , your program will first calculate the Haversine distance(km) between the security center and each building. Next, insert each node with its Haversine distance into the tree to create a binary search tree. **Please note that the nodes need to be inserted "in the order of the inputs"; otherwise, the final result will be affected**. Fig. 1 shows the binary search tree.

In this binary search tree, you must place cameras at certain nodes (buildings) to ensure that all nodes, including the root node (campus security center), are monitored with the fewest cameras possible. Each camera can monitor the building where it is placed, its parent node, and its immediate child nodes. Your task is to determine the minimum number of cameras needed and identify the specific buildings where these cameras should be installed. Fig.2 and Fig.3 show the camera installation example

Your program will read the input. The format of the input is shown below:

Input:

All geographic locations are represented by latitude and longitude as floating point numbers, **rounded to three decimal places**. The latitude and longitude should be separated by spaces:

- The first line contains an integer n, representing the number of buildings that need monitoring.
- The second line describes the geographic location (latitude and longitude) of the campus security center.
- The next n lines describe n buildings. Each line consists of the following three pieces of information separated by spaces:
  - The name of the building (a one-word string without spaces)
  - Latitude
  - Longitude

Output:

The outputs should be separated by lines:

- Output the minimum number of cameras needed to monitor all buildings.
- List the names of the buildings where the cameras should be installed.
  - Ensure that the buildings are listed in the order they appear in the constructed binary search tree, from top to bottom and from left to right.
  - This problem guarantees that there is only one unique solution.

-SAMPLE TESTS-

Input

```
8
25.29 121.67
Library 25.123 121.456
StudentCenter 25.234 121.567
ScienceBuilding 25.345 121.678
Cafeteria 25.210 121.430
Gym 25.280 121.570
Stadium 25.200 121.480
Auditorium 25.260 121.550
Dorm 25.265 121.560
```

Output

```
3
Library
Gym
Stadium
```

Building	Geographic Location	Haversine distance(km)
Library	(25.123, 121.456)	28.43
StudentCenter	(25.234, 121.567)	12.085
ScienceBuilding	(25.345 121.678)	6.168
Cafeteria	(25.210 121.430)	25.724
Gym	(25.280,121.570)	10.115
Stadium	(25.200, 121.480)	21.570
Auditorium	(25.260, 121.550)	12.518
Dorm	(25.265, 121.560)	11.400

TABLE I. Geographic locations of all buildings and their Haversine distances to the player in sample tests

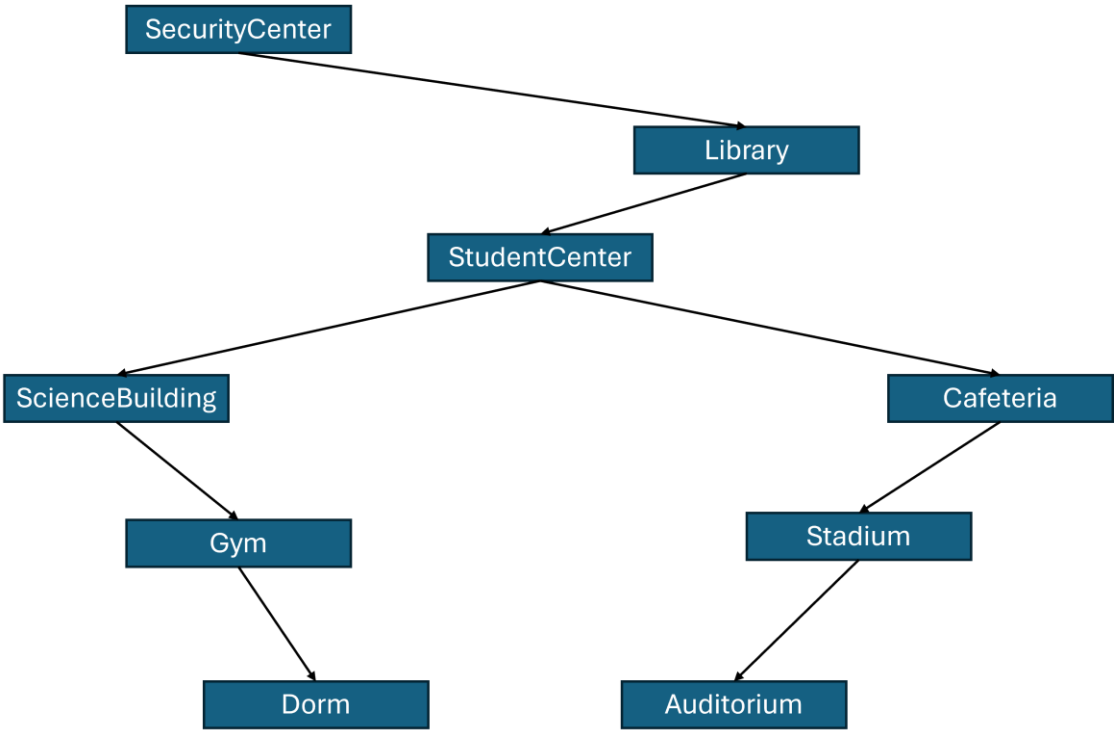

Fig. 1. The binary search tree

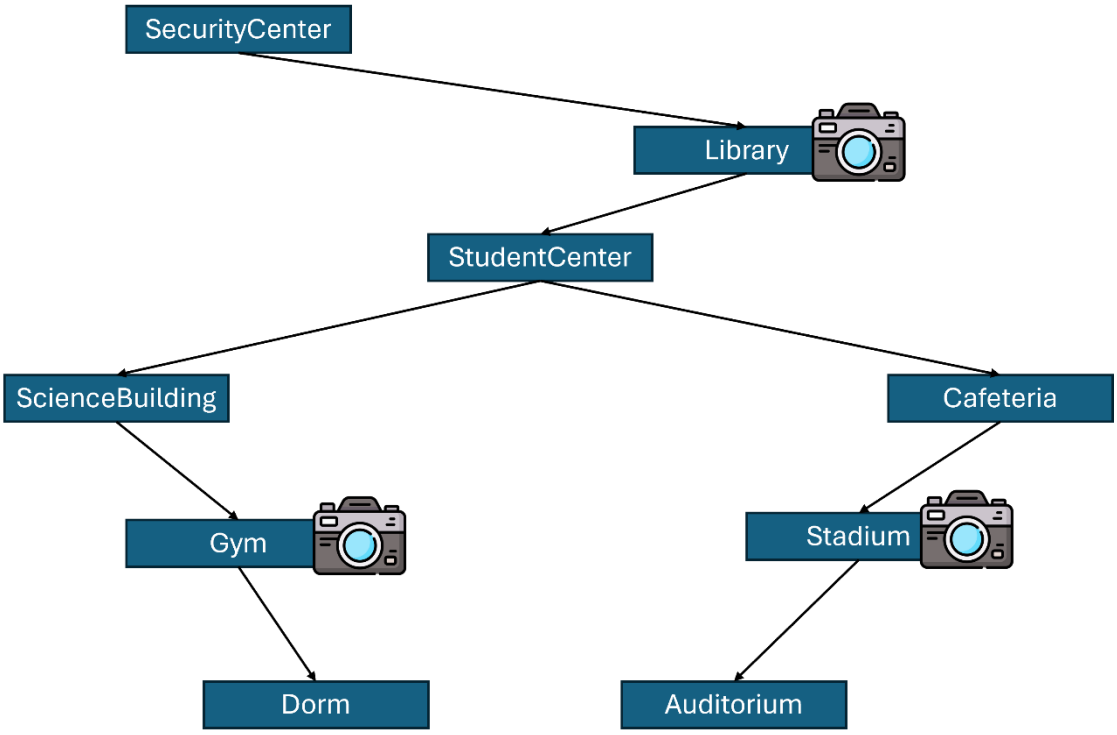

Fig. 2. The camera installation example I

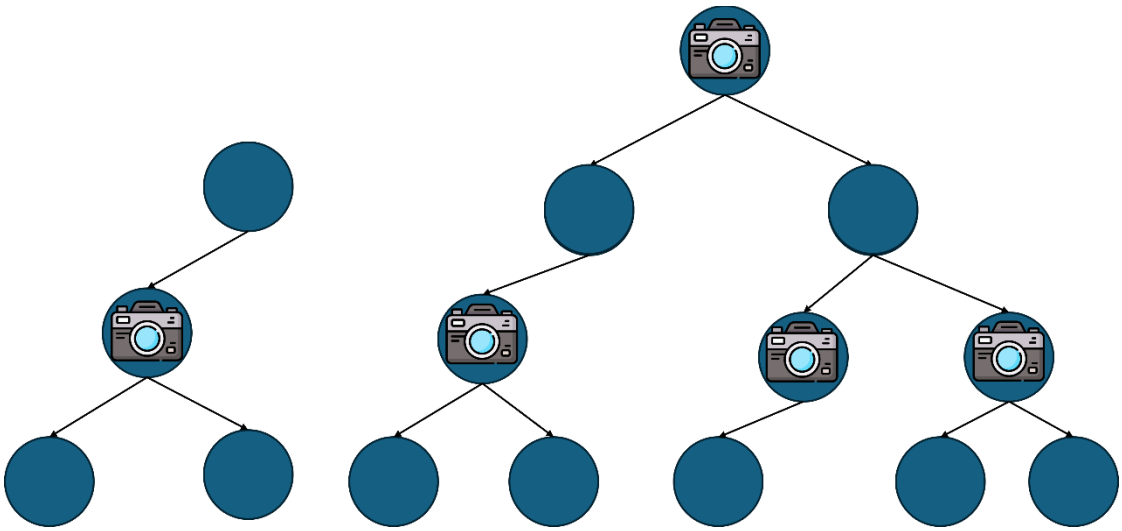

Fig. 3. The camera example installation examples II and III

2. Readme, comments, and coding style

An indicator of good source code is readability. To keep source code maintainable and readable, you should add comments to your source code where reasonable. A consistent coding style also helps a lot when tracing the source code. For this assignment, please also compose a readme file in *.txt format and name it “README.txt”. This file should contain a brief explanation of how to use your program. Please remember to have your source code comments and readme file in English.

3. Submission

To submit your files electronically, log in to the DomJudge through the following URL: http://domjudge.csie.io:54321

Press the submit button and choose the homework questions you want to submit. After submitting your code, DomJudge will give you a result to tell you whether your code is correct or not. Please note that your code will be evaluated by different sets of test cases. Please make sure your code can work correctly based on the description above. Additionally, you must compress your code and the README file into a zip file and upload it to Ecourse2. Otherwise, you will get zero points.

ATTENTION: Do NOT copy others’ work or you will get a zero.

4. Grading policies

The TA(s) will mark and give points according to the following grading policies:

- 95 % Binary tree
- 5% Readme, comments, and coding style.

The Readme file should include your name, class ID, a brief description of the code, and other issues students think will be helpful for the TAs to understand their homework.