BINUS University

Academic Career: *Undergraduate / ~~Master~~ / ~~Doctoral~~ *)*	**Class Program:** *~~International~~/Regular/~~Smart Program~~/Global Class*)*
□ **Mid Exam** ☑ **Final Exam** □ **Short Term Exam** □ **Others Exam :** ____________	**Term : ~~Odd~~/Even/~~Short~~ *)**
☑ **Kemanggisan** ☑ **Alam Sutera** □ **Bekasi** □ **Senayan** □ **Bandung** □ **Malang**	**Academic Year :** **2019 / 2020**
Faculty / Dept. : School of Computer Science	Deadline: Day / Date : Monday / Jun 29th, 2020 Time : 13:00 - 16:20
Code - Course : COMP6048 – Data Structures	Class : All Classes
Lecturer : Team	Exam Type : Online
*) *Strikethrough the unnecessary items*	
The penalty for CHEATING is DROP OUT!!!	

The total duration of this exam is 200 minutes, including downloading the questions and uploading the answers. Please use the time provided wisely.

THIS EXAM SHOULD BE SOLVED BY THOSE WHO HAS ODD NIM (NIM GANJIL)

NOTE:

1. There are **2 parts** in this exam, Essay and Case.
2. For essay problem:
   a. You are required to solve it using by ***handwritten on a paper***
   b. Subsequently, your essay answers ***should be converted in 1 pdf file*** using this format: ***nim.pdf***
   c. The lecturers won't accept any answers using word processing application in order to prevent copy-paste answers in a last minute
3. For case problem:
   a. ***The submission code is in .cpp file*** and using this format: ***nim.cpp***
4. All your answers ***either essay (nim.pdf) or case (nim.cpp) should be zipped and submitted through*** **https://exam.apps.binus.ac.id/**. Other than that, the submission won't be accepted for any reasons. ***(Note : Please zip both files using this format: nim.zip)***
5. **The exam will be *marked as 0* if any *plagiarism is found or if you solve A WRONG PROBLEM SET (We have a problem set both for NIM GENAP and NIM GANJIL)*.**

Verified by,

Hanry Ham (D5872) and sent to Department on May 29, 2020

I. Essay (60%)

1. **[15%]** Red Black Tree

   a. **[7.5%]** By using the existing tree in Figure 1, please insert nodes **Z, U, X, S, T** subsequently!

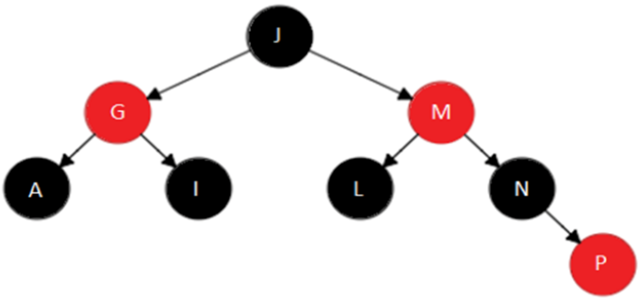

Figure 1 Red Black Tree 1

   b. **[7.5%]** By using the existing tree in Figure 2, please delete nodes **S, G, J, I, M** subsequently!

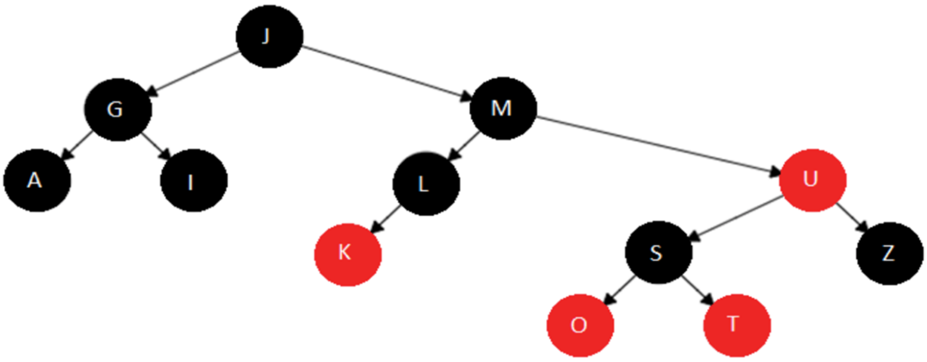

Figure 2 Red Black Tree 2

2. **[15%]** Transform the undirected graph below into **Minimum Spanning Tree using Prim's Algorithm starts from vertex A**. Please answer this question by **using the provided simulation table**!

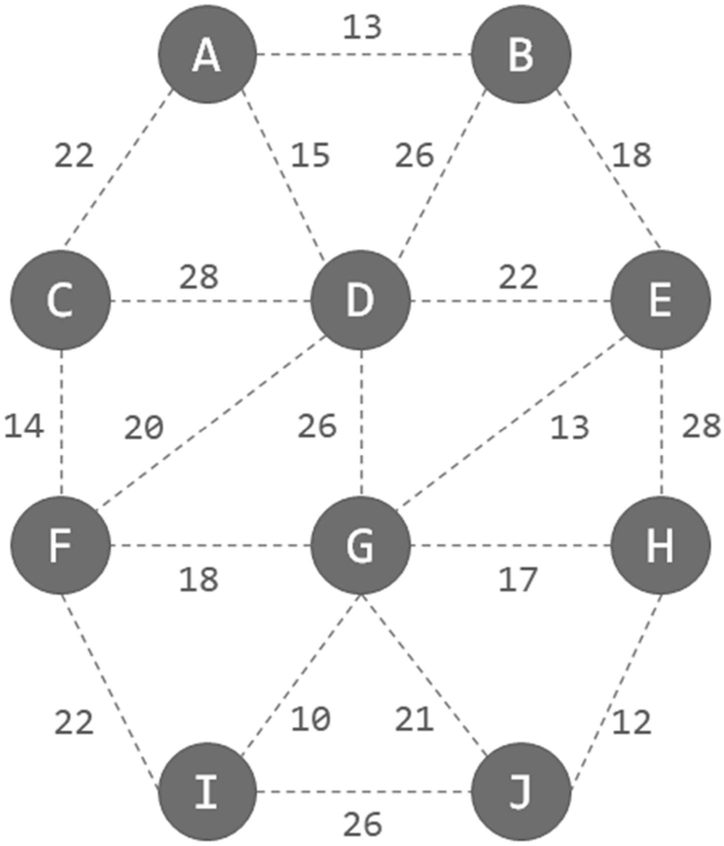

Figure 3 Undirected Graph

Table 1 Simulation Table

Adjacency List	Priority Queue	Track	Visited

Verified by,

Hanry Ham (D5872) and sent to Department on May 29, 2020

3. **[15%]** Which one of the statements is **true** about **Red Black Tree**? Please **elaborate your answer by giving comments for each statement given.**

a. The smallest keys will always be stored in the left most leaf nodes of the tree.

b. The tree always contains both red and black colored nodes.

c. The height of a tree with n nodes is h, where $h \leq 2\log_2(n+1)$.

d. Insert process in the tree is $O(n\log(n))$

4. **[15%]** Pay attention to these statements. Which one of the following is **false**? Please **elaborate your answer by giving comments for each statement given**.

a. AVL and Red Black Tree are self-balancing binary search tree.

b. Heap and Red Black Tree are binary tree.

c. B-Tree and Heap are binary tree.

d. Every node in AVL and Red Black Tree is unique.

II. Case (40%)

BeeAero Database System

BeeAero is a NASA-like agency in BeeCountry. They catalogue many planets and asteroids in this **UNIVERSE**. Now, they decided to develop their own database system using **AVL TREE**. This database does not need any interface, and all the input/output operations are done via **CONSOLE**. There are several requirements in developing this database, and you've been tasked to develop it. The following are the requirements of this prototype database:

1. GENERAL GUIDELINES
   - <u>You must use **AVL Tree**</u>. **No other tree is allowed**.
   - This database must support 4 major queries:
     i. INSERT NEW PLANET/ASTEROID
     ii. DELETE PLANET/ASTEROID
     iii. FIND SIMILAR PLANET/ASTEROID
     iv. SHOW PLANETS/ASTEROIDS
   - For **simplicity**, the database will record only **thermal temperature** of the planet/asteroid and the **planet/asteroid name**.
   - In this BeeCountry universe, it's confirmed that **no planet/asteroid has the same thermal temperature**. Therefore, you can safely assume that each planet/asteroid is unique in terms of **thermal temperature**.
   - The database will store the information in **ASCENDING ORDER** based on the **thermal temperature**.
   - If you **can't output anything**, then **don't output anything**!

2. INSERT NEW PLANET/ASTEROID Queries
   - This command will be used to insert a new planet/asteroid into the database.
   - The thermal temperature of a planet/asteroid is in the range of **-2^{31}** to **$2^{31}-1$**.
   - The planet/asteroid name is in lowercase, has no whitespaces, and it will be between 1 to 30 characters.
   - The command form will be as follow:

```
1 <<thermal-temperature>> <<planet-name>>
```

For example, if there’s a planet named “kepler” with thermal temperature of -215 (Celsius), then the insert command will be:

```
1 -215 kepler
```

- After this command, please output the number of planet/asteroid that **has larger thermal temperature than the current added**.
- If the temperature **already exists** in the database, you can **ignore the command,** but still output the number of planet/asteroid that **has larger thermal temperature than the current added**.

3. DELETE PLANET/ASTEROID
   - Sometimes, planet / asteroid is destroyed via natural process. Thus, we need to update the database.
   - This command will be used to delete a planet/asteroid from the database.
   - This command will delete a planet/asteroid based on its thermal-temperature.
   - The command form will be as follow:

     asdas
   - If you can find the planet/asteroid with the given thermal temperature, then the planet/asteroid will be deleted from database. Otherwise, nothing happened.
   - After this command, please output **the number of planet/asteroid exists in the database**.

4. FIND SIMILAR PLANET/ASTEROID
   - This command will be used by BeeAero to search for other planet/asteroid that has **“similar” thermal temperature**.
   - Here, “similar” means the **absolute difference** between the asked thermal-temperature with all planets is the **smallest**. If there are more than 1 such planets, you need to output the planet/asteroid that has **the largest temperature** among such planets/asteroids (you can see the simulation below for this case).
   - The “asked thermal temperature” is in the range of **-2^{31}** to **$2^{31}-1$**.

- The command form will be as follow:

```
3 ≪asked-thermal-temperature≫
```

For example, if all planets/asteroids data in the database shown using in-order traversal are as follow:

1000 proxima	**1500** mars	**2500** venus	**3500** earth	**3900** jupiter	**6400** uranus	**79850** saturn	**150350** neptune

Then below are the output for each input command given:

INPUT	OUTPUT	EXPLANATION
3 12000	6400 uranus	There are two planets/asteroids thermal temperature “similar” to 12000: 6400 uranus and 79850 saturn. The smallest difference with thermal temperature 12000 is “6400 uranus”.
3 1500	1500 mars	mars thermal temperature is the same with the given thermal temperature.
3 3000	3500 earth	There are two planets/asteroids that has the smallest difference – both with 500 degree Celsius differences -- (venus, and earth). We will choose the planet with **the larger temperature**.
3 100000	79850 saturn	There are two planets/asteroids that has “similar” thermal temperature to 100000: 79850 saturn and 150350 neptune. The smallest difference with 100000 is 79850 saturn.
3 1000000	150350 neptune	There is only one planet/asteroid that has “similar” thermal temperature to 1000000: 150350 neptune.

- This command output the thermal-temperature and the name of the planet/asteroid in a single line separated by spaces like the example above.

5. SHOW PLANETS/ASTEROIDS
   - This command will be used to **show up to 10 planets/asteroids between certain input range**.

- The command form will be as follow:

```
4 ≪from≫ ≪to≫
```

For example, if all planets/asteroids data in the database shown using in-order traversal are as follow:

1000 proxima	**1500** mars	**2500** venus	**…..**	**3900** jupiter	**6400** uranus	**79850** saturn	**150350** neptune
1	**2**	**3**	**…..**	**97**	**98**	**99**	**100**

Then below are the output for each input command given:

INPUT	OUTPUT	EXPLANATION
4 1 2	1: 1000 proxima 2: 1500 mars	The data in the rank between 1 and 2 will be shown.
4 97 106	97: 3900 jupiter 98: 6400 uranus 99: 79850 saturn 100: neptune	Because the last data is in the position 100, therefore only data in rank 97 up to 100 will be shown

- The "from" and "to" is in the range of **1** to **2^{31}-1**.

6. Application usage

- The 1st line in the input file will be an integer N (1 <= N <= 250000), the number of input command to be executed.
- Then, it will be followed by N command that consists of the command number and its parameter. The sample of the input are as follow:

INPUT	OUTPUT	EXPLANATION
10 1 1500 mars 1 1000 proxima 1 2500 venus 3 1000 1 3900 jupiter 2 1500 4 1 10 3 5000 2 1200 1 1000 centaur	0 1 0 1000 proxima 0 3 1: 1000 proxima 2: 2500 venus 3: 3900 jupiter 3900 jupiter 3 2	

Explanation

The first line of output is “0”, because after you insert to the database the thermal-temperature 1500, there’s no planet that has thermal-temperature larger than 1500.

The second line of output is “1”, because after you insert to the database the thermal-temperature 1000, there’s 1 planet that has thermal-temperature larger than 1000.

The third line of output is a “0”, because after you insert to the database the thermal-temperature 2500, there’s no planet that has thermal-temperature larger than 2500.

The fourth line of output is “1000 proxima”, because when you “FIND SIMILAR” the thermal-temperature of 1000, the planet “proxima” temperature is **the same** with the given thermal-temperature. Thus, the difference is 0 (which is the smallest).

The fifth line of output is “0”, because after you insert to the database the thermal-temperature of 3900, there’s no planets that has thermal-temperature larger than 3900.

The sixth line of output is “3”, because after you delete the thermal-temperature 1500 (which belong to planet “mars”), there’re 3 planets/asteroids exists in the database.

The seventh line until the ninth line of output are the planet in rank 1 to 3.

The tenth line of output is “3900 jupiter”, because when you “FIND SIMILAR” the thermal-temperature of 5000, the planet “jupiter” temperature is the smallest differences with the given thermal-temperature.

The eleventh line of output is 3, because after you try to delete the thermal-temperature 1200 (which belong to **no planet**), there still 3 planets/asteroids exists in the database. Note that **no planets/asteroids** are deleted. It’s because **no planet has the thermal-temperature of 1200**.

The last line of output is “2”, because there’re 2 planet that has thermal-temperature larger than 1000. Also, this command get “ignored” because in the database, there already exists “1000 proxima”.

-- Good Luck --

Verified by,

Hanry Ham (D5872) and sent to Department on May 29, 2020