1. Red black tree

* Simulate the process of Red Black Tree insertion using the following data sequentially: 195, 570, 588, 341, 825, 143, 618, 432, 145, 523.
* After simulating the insert process of Red Black Tree in point 1A, simulate the deletion of the following data sequentially: 523, 570, 588, 432, 618.

2. Given the Graph below, simulate Dijkstra shortest path from **A to K.**

Diagram, schematic

Description automatically generated

3. Min Max Heap

* Simulate the process of min max heap insertion using the following data sequentially: 10, 20, 30, 4, 5, 6, 50.
* After simulating the insert process of min max heap in point 1A, simulate the deletion of the following sequentially: min, max, min.

4. Hoven Ring is a massively multiplayer online role-playing game (MMORPG) that just got released recently. This game gains its popularity because of its amazing player versus player (PVP) mechanic. However, this game does not have leaderboard system yet. As their junior intern programmer, you are tasked to implement this leaderboard system. You need to first create the system prototype using C Programming Language and **AVL tree concept**. The following is the requirement of the system.

* Player Data
* Each player contains: **Username, Job, Score, Match Count, Win Count, Lose Count, and Draw Count.**
* There are six jobs represented by numbers **between 0 and 5: Novice, Knight, Sniper, Wizard, Druid, and Assassin.**
* You will be asked to calculate player’s win rate with the following formula:

**Win rate = (win count / match count) \* 100**

* Leaderboard Calculation
* The main data structure for the leaderboard system is **AVL Tree**. The **score** value will be used as the **key**. The score can **range between 0 and 999**.
* It is possible to have several **players having the same score**. In that scenario, you must be able to store those player data **ordered by win rate in ascending**. In case there are **multiple players with the same score and win rate**, order the data by **their data entry time**.
* The Application
* You only need to create the prototype application; hence the application **does not have any user interface**. **All commands will be given via console**.
* The program will insert dummy player data as follows (see player data):

1. Gregor, Sniper, 113, 10, 9, 1, 0

2. Heimdall, Druid, 300, 100, 50, 50, 0

3. Travy, Warrior, 300, 200, 100, 100, 0

4. Tetron, Novice, 300, 50, 25, 25, 0

5. Homer, Druid, 113, 10, 8, 2, 0

6. Garen, Warrior, 113, 100, 70, 30, 0

* First user needs to input an integer **N**, which is the number of commands they want to issue. After that **N** number of **commands** will follow. Some commands have **mandatory parameters**.
* There are **four different** command available as shown in the following table:

|  |  |
| --- | --- |
| Command | Description |
| INSERT | * This command will be used to **entry new player data** into the leaderboard system. * First you need to put string **“INSERT”**, followed by **INT N**, followed by **N player data** in the following order: **Username, Job, Score, Match Count, Win Count, Lose Count, and Draw Count**. Each data is separated by **‘#’** symbol. * You don’t need to make validation for the input. * This command does not produce any output.   Sample Input  1  INSERT  3  DaleLarsen2#3#213#804#756#47#3  HelloWorld#1#50#500#100#400#0  Tetrioc#5#213#100#90#5#5  Sample Output  (This command does not produce any output) |
| FIND | * This command will be used to **search leaderboard data based on the given score**. The input is string **“FIND”** followed by **the score** **to search**. * Show **error message if the data does not exist**. * If the data exists, then **shows all players sharing the same score**. **(Username, job, win rate)**   Sample Input 1  1  FIND  113  Sample Output 1 (if data exists)  Score 113  1. Gregor [Sniper] (90%)  2. Homer [Druid] (80%)  3. Garen [Warrior] (70%)  Sample Input 2  1  FIND  997  Sample Output 2 (if data does not exists)  Score 997  No data found for 997. |
| DELETE | * This command will be used to delete score data from the leaderboard system based on the **given score**. * The input is string “**DELETE**” followed by the **score to delete**. * Make sure to **first remove player data** for that score to save memories. * If the score data is not found, **then display error message**. * This command does not produce any output.   Sample Input 1  3  FIND  113  DELETE  113  FIND  113  Sample Output 1  Score 113  1. Gregor [Sniper] (90%)  Score 113  No data found for 113. |
| SHOWALL | * This command will be used to show all leaderboard data in ascending order. **(Username, job, win rate)** * The input is string “**SHOWALL**”. * This command does not have any parameters. * If there are **multiple** **players having the same score**, **display them in group**.   Sample Input 1  1  SHOWALL  Sample Output 1  Score 113  1. Gregor [Sniper] (90%)  2. Homer [Druid] (80%)  3. Garen [Warrior] (70%)  Score 300  1. Heimdall [Druid] (50%)  2. Travy [Warrior] (50%)  3. Tetron [Novice] (50%) |