ISE315 - Analysis of Algorithms Fall 2021
Homework 3 - Red-Black Tree

Due Date: 12 January, 2022

- Please write your own codes, copying code parts from books, websites or any other source including your friends is considered as plagiarism and results in penalty.
- Do not upload your codes to any public platform (e.g. Github) until the deadline of homework passes.
- Your code should be able to be compiled and run under Ubuntu OS.
- Do not forget to comment your code as you will be graded by comment quality.
- Submit your homeworks before the deadline, late submissions and submissions via e-mail will not be accepted.
- You are **not** allowed to use STL.
- If you have any questions, please use the message board or send an e-mail to akti15@itu.edu.tr

1 Implementation (70 pts)

In this part, you are going to build a basketball player database with **Red-Black Tree**.

- You are given **euroleague.csv** file which contains the information about players. Your code should take filename as an argument.
- The key for each of the nodes should be the corresponding player's name. Point, rebound and assist values should be kept as extra attributes within your nodes.
- You need to read lines from input file and insert the players into the tree. Your insertion operation should insert a new node into the relevant position in the Red–Black Tree only if the player does not exist within the tree. For the players that are inserted before, update the point, rebound and assist.
- The file includes data from different seasons. At the end of the **each** season, you should print **all-time** point, rebound, and assist leaders.
- You should print the tree **only** at the end of the first season. When printing tree, you should represent the height and color of nodes as it is given in the sample output.

Sample Input File

```
Season ,Name,Team,Rebound , Assist , Point
2016-2017,Ali Muhammed,FEN,93,106,386
2016-2017,Ekpe Udoh,FEN,241,68,376
2016-2017,Jan Vesely,FEN,154,49,328
2016-2017,Bogdan Bogdanovic,FEN,84,80,321
2016-2017,Gigi Datome,FEN,122,35,303
2016-2017,Kostas Sloukas,FEN,62,130,268
2016-2017,Nikola Kalinic,FEN,101,51,249
2016-2017,James Nunnally,FEN,67,58,192
2016-2017,Pero Antic,FEN,75,19,130
2016-2017,Melih Mahmutoglu,FEN,10,11,79
2016-2017,Ahmet Duverioglu,FEN,12,1,30
2016-2017,Anthony Bennett,FEN,9,2,12
2016-2017,Baris Hersek,FEN,0,0,4
2016-2017,Berk Ugurlu,FEN,1,2,2
2016-2017,Egehan Arna,FEN,0,0,0
2016-2017,Yordan Minchev,FEN,2,0,0
2017-2018,Jan Vesely,FEN,174,53,424
2017-2018,Brad Wanamaker,FEN,97,138,408
2017-2018,Kostas Sloukas,FEN,87,188,351
2017-2018,Gigi Datome,FEN,117,38,336
2017-2018,Nicolo Melli,FEN,179,62,320
2017-2018,James Nunnally,FEN,59,39,269
2017-2018,Marko Guduric,FEN,56,69,241
2017-2018,Jason Thompson,FEN,140,30,180
2017-2018,Ali Muhammed,FEN,23,25,146
2017-2018,Nikola Kalinic,FEN,30,23,104
2017-2018,Ahmet Duverioglu,FEN,48,14,90
2017-2018,Melih Mahmutoglu,FEN,10,5,35
2017-2018,Sinan Guler,FEN,9,7,23
2017-2018,Egehan Arna,FEN,0,1,2
2017-2018,Baris Hersek,FEN,0,0,0
```

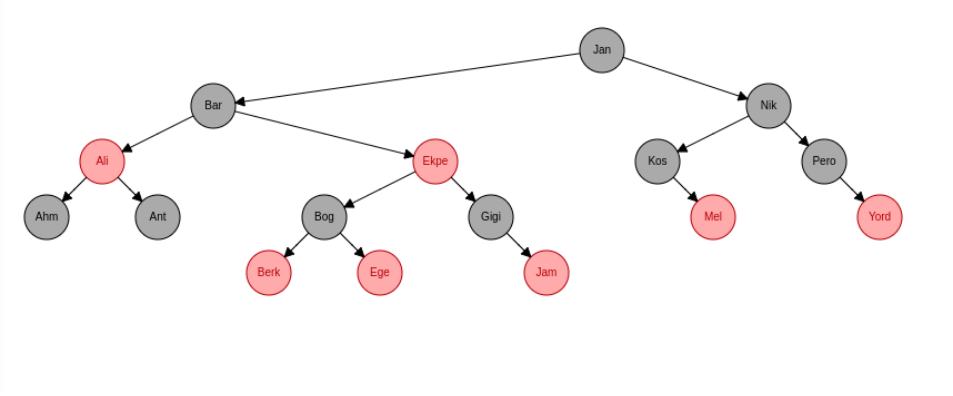

Figure 1: Red-Black Tree at the end of the first season

Sample Run

```
g++ studentID.cpp -o studentID
./studentID sample.csv
(or python3 studentID.py sample.csv)
End of the 2016-2017 Season
Max Points: 386 - Player Name: Ali Muhammed
Max Assists: 130 - Player Name: Kostas Sloukas
Max Rebs: 241 - Player Name: Ekpe Udoh
(BLACK) Jan Vesely
-(BLACK) Baris Hersek
--(RED) Ali Muhammed
----(BLACK) Ahmet Duverioglu
----(BLACK) Anthony Bennett
--(RED) Ekpe Udoh
----(BLACK) Bogdan Bogdanovic
------(RED) Berk Ugurlu
------(RED) Egehan Arna
----(BLACK) Gigi Datome
------(RED) James Nunnally
-(BLACK) Nikola Kalinic
--(BLACK) Kostas Sloukas
----(RED) Melih Mahmutoglu
--(BLACK) Pero Antic
----(RED) Yordan Minchev
End of the 2017-2018 Season
Max Points: 752 - Player Name: Jan Vesely
Max Assists: 318 - Player Name: Kostas Sloukas
Max Rebs: 328 - Player Name: Jan Vesely
```

2 Report (30 pts)

Complexity [10 pts]
Write down the asymptotic upper bound for the insertion and search operations of Red-Black Tree for worst case and average case with detailed explanations.

RBT vs BST [5 pts]
Compare Red-Black Tree with Standard Binary Search Tree in your **own** words.

Augmenting Data Structures [15 pts]
Suppose that you are given the position (Point Guard **PG**, Shooting Guard **SG**, Small Forward **SF**, Power Forward **PF** or Center **C**) of the players. If you were to augment your Red-Black Tree with 5 new methods, i^{th} **PG**, i^{th} **SG**, i^{th} **SF**, i^{th} **PF** and i^{th} **C**, that return the name of the i^{th} Point Guard, i^{th} Shooting Guard, i^{th} Small Forward, i^{th} Power Forward and i^{th} Center, respectively, what will be your strategy? Provide a pseudocode with explanations to implement these methods but **do not implement** them.