Bonus Homework

Eskisehir Technical University Computer Center needs to implement a system to track information about the students in university. Each student is uniquely identified by his/her id Number (idNr). Other information, e.g. student's name, will also be kept in the system. It is a top priority that searching for information about student is 'fast'. Because many people register/erase his/her records it is necessary that updates of information in the system are also fast. According to the current records the student population is 12345. The Computer Center engineers decided that the AVL Tree data structure would be the most appropriate data structure to implement such large problem. Now they are hiring software developers to implement the system and you apply for the job. To test your abilities of programming, you have to implement a simpler version of the problem. You have to implement the AVL Tree data structure. You are required to use the following AVLNode structure and AVLTree class in your implementation:

```
public class AVLNode {

Integer idNr;
String Name

String Surname;
AVLNode left;
AVLNode right;

public void AVLNode(Integer idNr, string name, string
surname);

void setName(String n, String s);
};
```

AVLNode is used to implement a node of the AVL tree. It is a simplified version of the full-implementation node. It uses data member idNr as the search key and Name and Surname to store the name of student with the given ID number. Data members left and right are pointers to the left subtree and the right subtree, respectively. If a pointer is NULL, the corresponding subtree is empty. The constructor initializes the new node, sets ID number to idNr, name to Name, surname to Surname and left and right to NULL. The member function *setName* is used to set new values into the data members Name and Surname.

```
public class AVLTree {

AVLTree();

public void insertStudent(Integer idNr, String name, String
surname);
public AVLNode searchStudent(Integer idNr)
public void deleteStudent(Integer idNr);
public void printSearchPath(Integer idNr);
public void processFile(String filename);

AVLNode root;
};
```

AVL Trees, as you already know, are balanced trees. Therefore, you have to correctly implement member functions *insertStudent* and *deleteStudent* to keep the AVL tree balanced. For that purpose you can add one more data member into the AVLNode structure.

The AVL tree constructor creates the empty tree. The *insertStudent* method inserts a new node into the tree (duplicates are not allowed). The *searchStudent* returns a node of the person with ID number idNr, or the NULL pointer if the student is not in the tree. Function *deleteStudent* removes the specified student from the tree.
Function *printSearchPath* is used for testing purposes. It is used to search for a student with the given ID number. While searching for that node, it prints on the screen the path in the tree that has been traversed. For example, in the complete tree of elements 1, ., 15, if *printSearchPath* is invoked with parameter 11, it will print the following path: 8 -(Right) -> 12 -(Left) -> 10 –(Right) -> 11(Found).

Your test driver has to be implemented as follows: The *processFile* method will take one parameter - name of a text file. That file will contain information about students of Eskisehir Technical University that should be inserted to/deleted from the tree. On the first line, there are two integer numbers: the first one is the number of students that should be inserted to the tree; the second one is the number of students that should be deleted from the tree. Starting with the second line, on each line is stored information about one student in this format: ID Number Name and Surname for insertions, or ID Number for subsequent deletions. Below you can find the contents of a sample input file:

```
5 2
555 Lütfullah Matrakgeçen
11 Hayrünisa Neyapalım
22 Ali Kandemiroğlu
444 Tarkan Çöpdemir
77 Zebercet Mutfaktaçalışır
11
22
```

Your test program will insert five records into the empty tree. Then it will remove two nodes with ID Number 11 and 22 from the tree. Thereafter, your program will interactively ask for an ID Number. If the entered number is 0, it will terminate. Otherwise, the program will use function *printSearchPath* and print its result (the searched path) on the screen. This will allow us to test if your implementation (the final tree structure) is correct.

- Design your own AVLTree
- You have to write your own HW. Collaborition is not allowed. The similar programs will get zero mark.
- You have to send all the necessary files in order to test your program without any trouble. (send the whole project directory. If the program can not be compiled, you will loose 50% of your grade.
- You have to compress all the necessary files to hwBonus*_your_surname.*zip and send me via e-mail
- The subject part of your e-mail has to be HWBonus_480 *"Surname Name"*.
- No delay. Delayed homeworks will not be graded.