2E3 – Lab 5

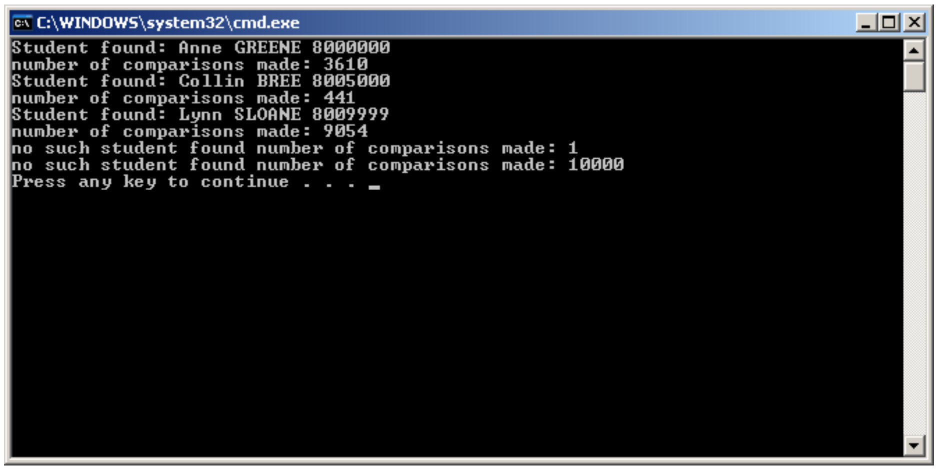
In this Lab, we will compare performance of searching for an item in a linked list and in a binary search tree. This lab consists of 2 parts.

Part 1

You can modify your solution to Lab 4 as a basis for the following:

1. Add method Student* findStudent (string firstname, string lastname) to StudentLinkedList class. Method should return a pointer to Student with matching lastname and firstname, or NULL if such student is not found.
2. Declare a data member numberOfComparisons in StudentLinkedList. At the start of the find method reset this number to 0, and then increase it by 1 each time a comparison is made.
3. Modify your main function to use file student10000.txt as input (available on blackboard). ***(Note: If you like, you can add the students as is, or you can add them alphabetically and see the difference when searching for missing students…)***
4. Modify your main function to search for the following students. If the student searched for was found, print its full details, if student was not found print appropriate message to the screen. Also, after each time find method is called print to the screen number of comparisons made to find the said student. Make a note of these numbers in the comment at the top of your main.

   "Abraham", "CRONIN"
   "Collin", "BREE”
   "Etienne", "HOLOHAN”
   "", "AAAA"
   "", "ZZZZ"



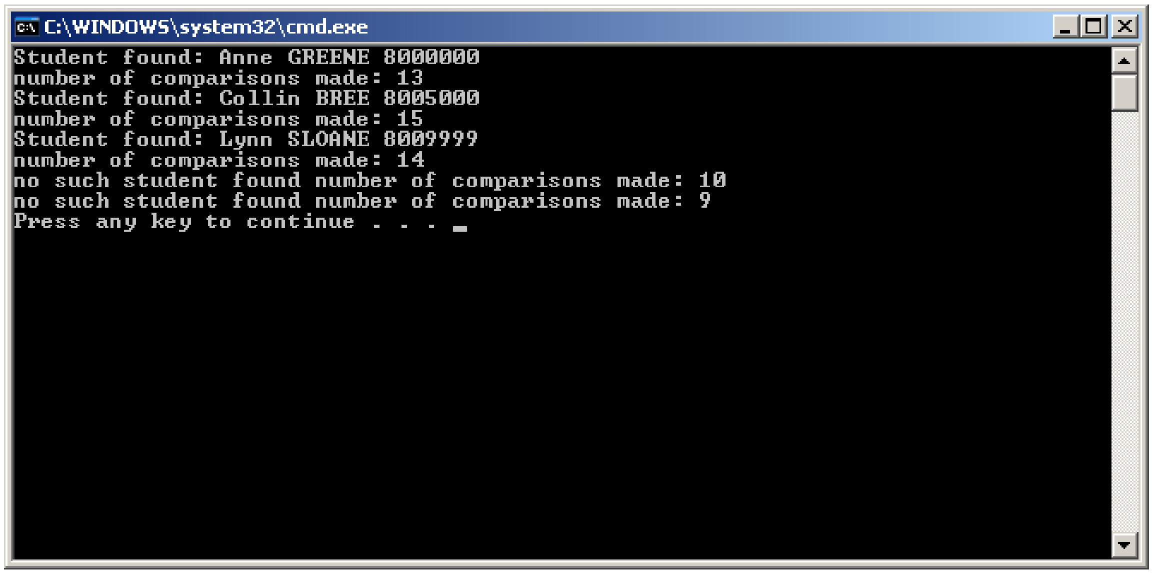
```
Student found: Anne GREENE 8000000
number of comparisons made: 3610
Student found: Collin BREE 8005000
number of comparisons made: 441
Student found: Lynn SLOANE 8009999
number of comparisons made: 9054
no such student found number of comparisons made: 1
no such student found number of comparisons made: 10000
Press any key to continue . . .
```

Part 2

Create a new project containing copies of the code from Part 1.

Make sure to have code from part 1 saved separately as you will need to demo both part 1 and part 2!

Modify the copy of the code to use a binary search tree instead of a linked list. Note that you will need to modify Student class and StudentList class, however your main function won't need to be changed. Repeat the task 4 from part 1, and compare the number of comparisons required now to the number of comparisons required when we were using a linked list. Make a note of these in the comment at the top of your main function as well. Which data structure is more efficient in searching for an item?



```
Student found: Anne GREENE 8000000
number of comparisons made: 13
Student found: Collin BREE 8005000
number of comparisons made: 15
Student found: Lynn SLOANE 8009999
number of comparisons made: 14
no such student found number of comparisons made: 10
no such student found number of comparisons made: 9
Press any key to continue . . .
```

To get you started: Methods to find a student in a binary search tree in a file will be recursive, you will need one public method, and one private helper method for as follows:

```
public:
Student* findStudent (string fn, string ln); //search for a student

private:
Student* findStudent (Student *s, string fn, string ln);
```

You have <u>one week</u> to complete this program, and the marks will be assigned as follows:

This week – Week 9:
0 = did not attend lab
1 = attended the lab session

Max = 1 point

Next week – Week 10:

0 = did not attend lab
1 = attended the second (marking) lab session but have very little working code or the code does not compile. **Your code MUST compile in order to get a mark higher than 1**
2= part 1
1 = modifications in studentList.h (methods + data members)
2= add to tree method
2= find in a tree method
1= number of comparisons counted correctly

Max = 9 points

<u>Overall maximum for two labs together= 10 points</u>

The program needs to be DEMONSTRATED for marks to one of the demonstrators before the end of your lab session next week (Week 10) AS WELL AS SUBMITTED ONLINE through Blackboard.