Data Structures Project

Question 1:

You are supposed to implement a library class that consists of two main components.

1- An array of linked Lists called libraryBooks. The size of this array is 26 (Alphabetic letters count) as each cell will hold books that starts with a specific letter. For example: index 0 contains a linked list of books starts with letter A. (assume first letter of any book name is capital)
2- An array-based list called Borrows which is of class borrowAction that consists of the borrowed book name and the borrower name.

Book class contains book name and author name and count in stock. When a user borrow a book, count of this book is decreased and anew borrowAction is added to the arrayList to be sorted alphabetically based on borrowed book name.

A menu should be displayed that contain these options

1- Add book (book name, author name, count) : add this book object in the correct place but if the book is found, just update the count
2- Borrow book (book name): search the libraryBooks (only the linked list of the first letter of the book) and if found with count greater than zero, make a borrow action and decrease the count.
3- Restore Book (book name, user name): search the Borrows list and remove this borrow action then find the book in library and update the count
4- Display library Status: print all Books in a good view
5- Display borrows: print all borrow actions in a good view (you should print how many books are in stock of this borrowed one)
6- Search (book name): print status of this book in the library books, also its status in the borrows (it could be borrowed more than one time from many users)

Question 2:

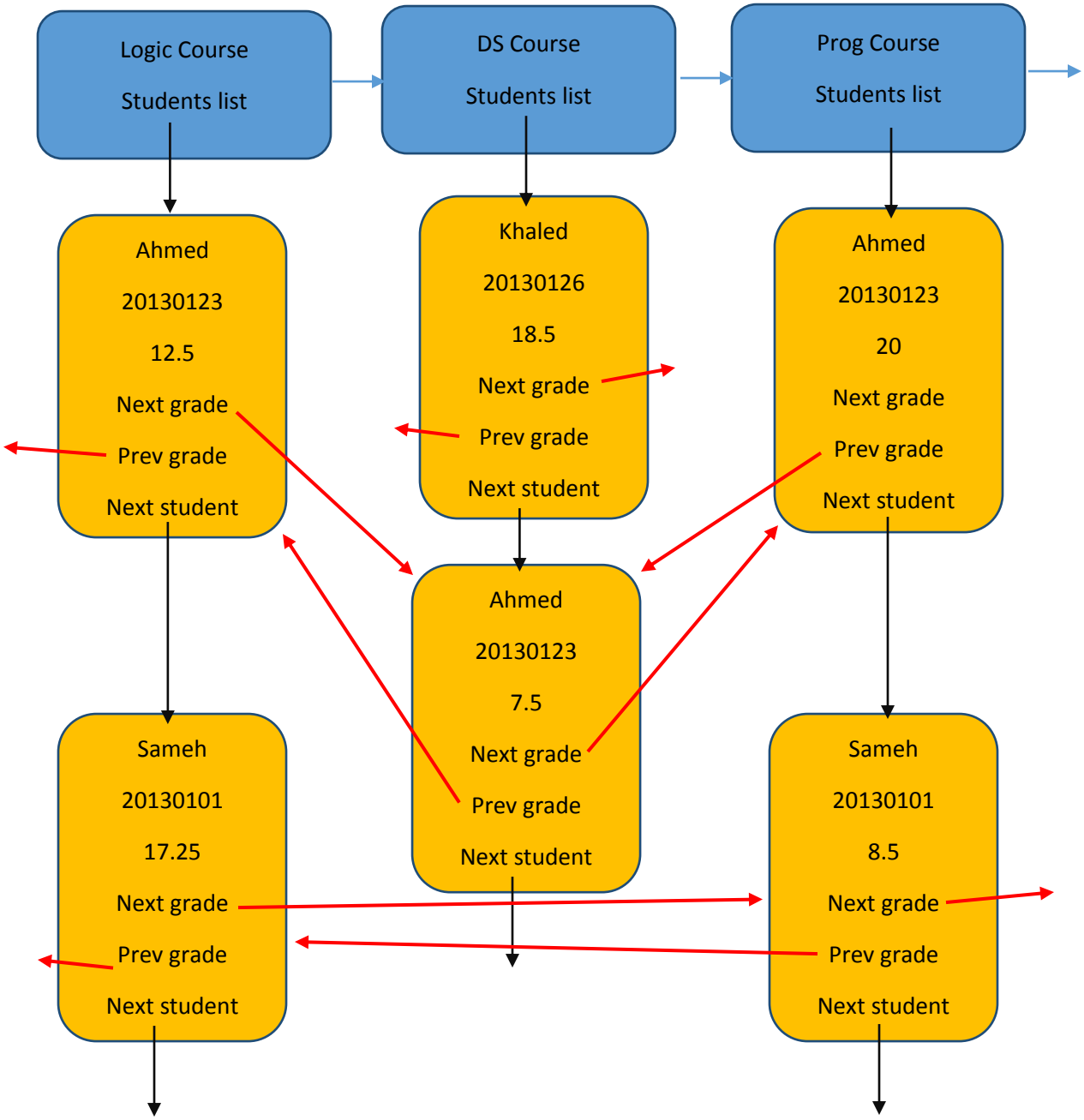
You are supposed to implement a linked list of course nodes where each course node has a name and a linked list of student nodes. Each student has a name, ID and a grade. A student can found in many courses lists with the same name and ID but with different grade.

Implement a main function to test your implementation so that you can add course and add student with his ID and grades of some or all courses found in your list.

Question 3:

Make an updated version of Question 2 so that each student node has also two pointers to other student nodes called previous grade and next grade so that if you want to know the

other grades of the current student, you do not need to go to next course and loop on its nodes but only get next or previous grade pointer.



Question 4:

Write a program that takes as input an arithmetic expression. The program will outputs whether the expression contains matching grouping symbols, and if it's

balanced, it will calculate the expression result. For example, the arithmetic expressions

{25 + (3 – 6) * 8} is balanced and its result equals 1

7 + 8 * 2 is balanced and its result equals 23

However, the expression 5 + {(13 + 7) / 8 - 2 * 9 does not contain matching grouping symbols, so we will not compute its result

Question 5:

Given array of 8 restaurants , Sort them by rating (high to low) then distance (close to far) using in-place sorting algorithm

```
class Restaurant {
Private:
        string name;
        int rating;
        double distance;
public:
        Restaurant(String n, int r, double d){
                name = n;
                rating = r;
                distance = d;
        }
        Static void sortRestaurants ( Restaurant ** arr, int size);
}
```

Sample Input (Array of 8 restaurants)	Sample Output(Array of 8 restaurants)
➔ Rest1 (“A”, 2, 30.1)	Rest (“F”, 5, 50.8)
➔ Rest2 (“B”, 4, 30.8)	Rest (“C”, 5, 60.2)
➔ Rest 3 (“C”, 5, 60.2)	Rest (“B”, 4, 30.8)
➔ Rest 4 (“D”, 2, 50.1)	Rest (“G”, 4, 70.1)
➔ Rest 5 (“E”, 3, 60.6)	Rest (“E”, 3, 60.6)
➔ Rest6 (“F”, 5, 50.8)	Rest (“H”, 2, 10.1)
➔ Rest7 (“G”, 4, 70.1)	Rest (“A”, 2, 30.1)
➔ Rest 8 (“H”, 2, 10.1)	Rest (“D”, 2, 50.1)

Question 6:

In this problem, we will develop a class to use for testing and comparing sorting algorithms. The class will have methods to support the following functions.

1. Generating a given number of random integer data from a certain range. For example, one can generate a vector/array of 10000 integer numbers that fall in the range from 1 to 100000.
2. Generate a given number of nearly sorted data from a certain range (Check this website http://www.sorting-algorithms.com/)

3. Generate a given number of reverse ordered integer data from a certain range.
4. Run a given sorting algorithm on a given set of data and calculate the time taken to sort the data.
5. Run a given sorting algorithm on several sets of data of the same length and same attributes and calculate the average time. Note it is not sufficient to calculate the performance of the algorithm on a given set of data once but instead, we develop several data sets, e.g., 3 or 5 data sets, of the same characteristics and run the algorithm on them and then get the average.
6. Develop an experiment to run a given sorting algorithm and calculate its performance by deciding on the input values of n and the type of data (random, nearly sorted, reversed) and then running the algorithm and collecting the average processing time for each input value n. For example I should be able to design an experiment to:

- Run Insertion Sort
- Run Shell Sort (Gaps will be N/2 , N/4, … 1 The first row in the table in this link http://en.wikipedia.org/wiki/Shellsort#Gap_sequences)
- Run Shell Sort (Gaps will be 1 , 2 , 3, 4, 6, 8, 9 , 12 , … The fifth row in the table in this link http://en.wikipedia.org/wiki/Shellsort#Gap_sequences , We will use the reverse order of these gaps , [for example , if the array's size is 10 , we will use (9, 8, 6, 4, 3, 2, 1)])
- Run merge Sort
- Run Count sort

on randomly sorted integers data taken from the range (1 to 1,000,000) and with input value (data size) from 0 to 100000, with step 5000. This means we will run the algorithms on data sets of 5000, 10000, 15000, …, 100000 randomly sorted integers. Note that for each value of n, you will develop, say, 5 different sets and take the average of their runs.

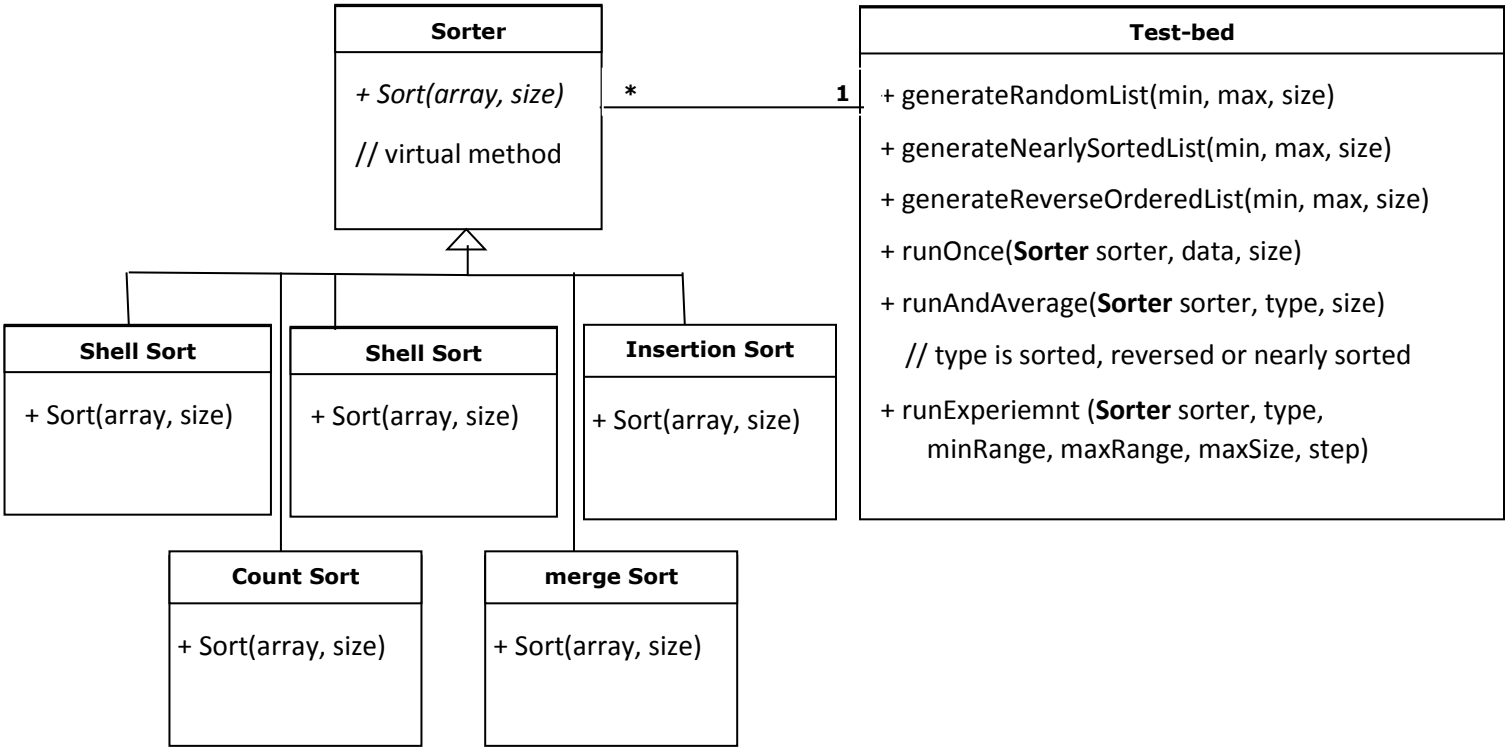
7. The output of the experiment goes to CSV file has the following format , each cell in this table will be the average processing time

Data size	Insertion Sort	Original Shell Sort	Shell Sort (Pratt)	Merge sort	Count sort
5000					
10000					
….					
1000000					

8. Develop a sample demo to demonstrate the functionality of your class.

What to deliver: (1) Source code files in standard C++ (no project files or other files). (2) The csv file of data resulting from running the demo

Proposed Class Diagram



Question 7:

Implement a class which will hold the words in a spelling dictionary.

For the first part of the assignment, you should write a class that implements an Open Hash Table of strings. That is, the table is stored as an array of simple binary search trees of strings (or, more precisely, an array of pointers to such trees).

The class must include(In first three functions you will use your hah function and binary search trees functions to accomplish hash table functions)

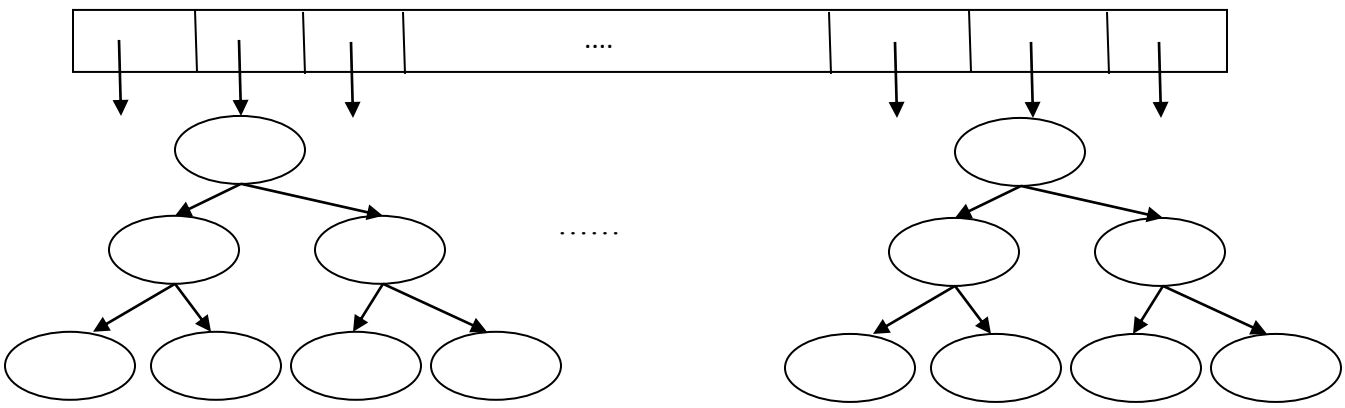
- A function *add(s)* that adds the string s to the table, if it is not already there.
- A function *remove(s)* that removes the string s from the table, if it's there.
- A function *contains(s)* that returns a boolean value that checks whether the string s is in the table.
- A function *size()* that returns the number of strings in the table.
- A hash function to compute the hash code of a string. Design your own hash function to minimize collisions as much as possible.
- A constructor that accepts the size of the table as a parameter.

The second part of the assignment is to write a main program that uses your hash table class. The main program will check individual words. You will be prompted to enter words. When you type a word, there are two possible responses: An output of "ok" means that the word is in the dictionary. An output of "not found" means that the word is not in the dictionary .

The file EnglishWords.csv has more than 30,000 words, read all words from the file and build your hash table of trees. Then it should prompt the user to enter words. For each word, it should check whether the word is in the dictionary. If so, it should say "ok". If not, it should say "not found".

One issue that you will have to settle is what to do with upper-case letters. Note that both the words in the dictionary and the user's input can contain upper case letters. I suggest that you simply convert all letters to lower case.

Your hash table design will be like the following diagram, array of pointers to binary search trees, and each node in any tree will be a word



Assignment Rules (Read them carefully)

- Due date: **16/5/2015, 11:00 pm**
- In Groups of max 3 from the same group
- Deliver your source code and required files on acadox
- The account who will send the assignment should be named with name and ID
- Write Team members names and IDs in a separate file
- Cheating will be graded with negative (Cheating means same code)