CS216: Introduction to Software Engineering Techniques (Spring, 2022)
Lab Assignment 9
(20 points)

Today's Date: Monday, March 28

Due date: Friday, April 1

The purpose of this lab assignment is

- To work on the definition of class named `Autocomplete`
- To review binary search algorithm and a few sorting algorithms
- To get to know generic programming using templates
- To create your own makefile for building executable program for Lab9
- To work on the second part of your Project Assignment 2

Part 1:

1. In the terminal window, make the `CS216` directory, which you created in `Lab1`, your current working directory.

2. Create a directory underneath the `CS216` directory named `Lab9`, and make the `Lab9` directory, your current working directory.

3. Use command curl to download a zip file named `Lab9Source_ac.zip` from the link (https://www.cs.uky.edu/~yipike/CS216/Lab9Source_ac.zip) and save the file into your current working directory **`~/CS216/Lab9`**:

```
$ curl –O http://www.cs.uky.edu/~yipike/CS216/Lab9Source_ac.zip
```

4. Unzip the file you downloaded from step 3 using the command:

```
$ unzip Lab9Source_ac.zip
```

The zip file contains **SIX** files: actors.txt, autocomplete.h, autocomplete.cpp, SortingList.h, SortingList.cpp, and Lab9.cpp. Please note that the definition of the class, named `SortingList`, is exactly the same as that of `Project 2`; and the declaration of the class, named `Autocomplete`, is exactly the same as that of `Project 2.`

5. Copy the definition of the class named `Term`, which you have finished in Lab8, to your current working directory:

```
$ cp ~/CS216/Lab8/term.h ./
```

```
$ cp ~/CS216/Lab8/term.cpp ./
```

6. Open `SortingList.h` and `SortingList.cpp with` your preferred text editor and take a look at the definition of the class, named **`SortingList`** to help you better understand the private data member of the `Autocomplete` class. (Please note that the **`SortingList`**

class is quite similar to the **`TermSortingList`** class in Lab8, however the **`SortingList`** class is a template class). Get familiar with the syntax requirement of a template class definition, although you do not need to provide your own definition for Project 2, you will define a template class in Project 3. Hence make yourself ready for it☺ Try to compile the source code for the single class definition without linking procedure by typing:

```
$ g++ -c SortingList.cpp
```

7. Open `autocomplete.cpp with` your preferred text editor and take a look at the description of each function and provide the implementation of each member function (Please note that you are given the complete implementation of the helper function, named **`binary_searchHelper`**, in **`autocomplete.cpp`**). **(Please do not modify Lab9.cpp!)**

After you complete the implementation of member functions in `autocomplete.cpp`, compile the source files using the command:

```
$ g++ term.cpp autocomplete.cpp SortingList.cpp Lab9.cpp -o
Lab9
```

8. Write your `makefile` for `Lab 9` to help you efficiently generate the executable program. Each time after you modify some code, save the file and run `make`, you may need to fix some errors and run make again. make will help you efficiently rebuild the program every time you make a change.

The following are some examples of running your program named `Lab9`:

```
$ ./Lab9

Usage: ./Lab9 <filename>

$ ./Lab9 actors.txt

Time for sorting all terms: 0.216166 seconds.
Please input the search query (type "exit" to quit):
Tom Ha↵
Time for searching the maximum three of matched terms: 0.139997
seconds.
Data itmes in the list:
4386200484      Tom Hanks
1047104583      Tom Hardy (I)
103774650       Tom Hamilton (II)
103075791       Tom Hallick
97104620        Tom Hauser (V)
96917897        Tom Harvey (XIII)
```

```
72173796        Tom Hatten (I)
66208183        Tom Hartig
63578523        Tom Harkin
58270437        Tom Harper (II)
35990505        Tom Hanna (II)
29938435        Tom Hammond (IV)
28190801        Tom Harvey (I)

Please input the search query (type "exit" to quit):
Tom Han↵
Time for searching the maximum three of matched terms: 0.192501
seconds.
Data itmes in the list:
4386200484      Tom Hanks
35990505        Tom Hanna (II)

Please input the search query (type "exit" to quit):
Zv↵
Time for searching the maximum three of matched terms: 0.115524
seconds.
Data itmes in the list:
79711678        Zviad Sokhadze
54617761        Zvonimir Hace
38445704        Zvee Scooler

Please input the search query (type "exit" to quit):
Emma D↵
Time for searching the maximum three of matched terms: 0.120272
seconds.
Data itmes in the list:
148775460       Emma Dukes
127509329       Emma Degerstedt
30363732        Emma Dewhurst (I)

Please input the search query (type "exit" to quit):
Woody A↵
Time for searching the maximum three of matched terms: 0.169891
seconds.
Data itmes in the list:
517696291       Woody Allen
37421338        Woody Andrews

Please input the search query (type "exit" to quit):
Charles P↵
Time for searching the maximum three of matched terms: 0.145847
seconds.
Data itmes in the list:
```

```
1088884483      Charles Pendelton
301959200       Charles Pestel
244626424       Charles Papasoff
195892912       Charles Payne (II)
178473667       Charles Parks (I)
110327213       Charles Parnell
104245553       Charles Parshley
95595162        Charles Paraventi
80253513        Charles Page (II)
47474114        Charles Perez (I)
31598309        Charles Parish (I)
26008756        Charles Pellegrino (I)

Please input the search query (type "exit" to quit):
Zvo↵
Time for searching the maximum three of matched terms: 0.168694
seconds.
Data itmes in the list:
54617761        Zvonimir Hace

Please input the search query (type "exit" to quit):
Yi Pike↵
Time for searching the maximum three of matched terms: 0.179709
seconds.
No matched query!
Please input the search query (type "exit" to quit):
exit↵
```

Note that the blue part is what you type from the keyboard, ↵ represents the “return” key. (Please note that the time measurement in the above sample output may not exactly match your output.)

8. Then zip together: **makefile**, **term.h**, **term.cpp**, **autocomplete.h**, **autocomplete.cpp**, **SortingList.h**, **SortingList.cpp**, **actors.txt**, and **Lab9.cpp** into one file named **Lab9.zip**. (Note your TA will use your makefile to build your program)

Submission

Open the link to course Canvas page (https://uky.edu/canvas), and log in to your account using your linkblue user id and password. Please submit your file (Lab9.zip) through the submission link for “Lab 9”.

Grading (20 points + Bonus 3 points)

1. Attend the lab session or have a documented excused absence. (5 points)
2. You create a correct makefile. (2 points)
3. Your program correctly solves the problem.
   - The implementation of SIX member functions of Autocomplete class are correct.

- `insert()` member function is correct in `autocomplete.cpp` (1 point)
- `sort()` member function is correct in `autocomplete.cpp` (1 point)
- `binary_search()` function is correct in `autocomplete.cpp` (2 points)
- `search()` member function is correct in `autocomplete.cpp` (4 points)
- `allMatches()` member function is correct in `autocomplete.cpp` (4 points)
- `print()` member function is correct in `autocomplete.cpp` (1 points)

Bonus: Demonstrate your program (including to build your executable program using your own makefile) to your TA and answer TA's questions. (3 points)

(Late assignment will be reduced 10% for each day that is late. The assignment will not be graded (you will receive zero) if it is more than 3 days late. Note that a weekend counts just as regular days. For example, if an assignment is due Friday and is turned in Monday, it is 3 days late.)

Reference

If you have not finished the definition of the class, named Term, in Lab8, you may download the source code from the following link: (Please note that this link will be available at 8:00am on Wednesday, March 30, 2022)
https://www.cs.uky.edu/~yipike/CS216/term.cpp