Software Engineering II: Algorithms and Data Structures - Assignment 1

Aim:
The aim of the assignment is to test students' ability to operate with file I/O, dynamically allocate memory, use a **list structure** to store the data, apply a sorting algorithm on this structure, insert a new element and finally to delete an element from this structure.

The objective of the assignment is for the student to understand the advantages and disadvantages of using a list structure compared to the use of an array structure for manipulation of the data.

Please note that you should build up your data structure from scratch (do not use the existing list container of C++).

Details:
You have to write a program that operates on a list. Each element of the list is a structure that contains two data fields (apart from the pointer to the next element). Field 1 is an integer, and Field 2 is a string.

The program should read in two files. The first file (*data_x.txt, where x is an integer*) contains the data, characters (names) and integers (in that order), and it is in ASCII format. The second file, whose filename should be passed as a command line argument, contains a list of operations that you have to perform on the data from the first file. Details of the two files are given below.

Data file (*data_x.txt*)
This file contains the data that you have to manipulate. It is an ASCII file, the data values are characters and integers and are stored one per line. Instances of this file are *data_1.txt*, *data_2.txt*, …

Command file
This file contains the operations that your program should perform on the data from the data file. The commands are shown in Table 1. Please note that the commands are case-sensitive.

Command	Description
r	This command read the data file *data_x.txt*. The command 'r' is followed by a number x indicating the exact file that has to be read. After the reading operation, no ordering is imposed to the resulting structure.
s	This command applies the Bubble Sort algorithm to sort the elements of the structure in ascending order. The ordering is based on the arithmetic values (**Field 1**)
w	This command writes all the data in the structure to a file *output_x.txt,* where x is determined in the command file. The data should be stored as one data per line.
i	This command inserts a new element in the structure **without** destroying the ordering.
d	This command deletes an element from the structure **without** destroying the ordering. In the case where the structure has multiple elements of the same value, this command deletes **only** one. In the case where this value is not in the structure, the command should not perform any operations. Note that the **Field 2** of the entry to be deleted follows this command.

Table 1. The list of possible commands

An instance of the command file is given below:

```
r
1
s
i
john
45
i
Bill
50
d
Bill
w
1
```

The commands should be interpreted as follows:

1. Read file *data_1.txt* in the internal structure
2. Sort the elements in ascending order
3. Insert entry <John, 45> (at the right place keeping the order)
4. Insert entry <Bill, 50> (at the right place keeping the order)
5. Delete entry <Bill> (keeping the order)
6. Write the resulting data to file *output_1.txt*

The *data_x.txt* and *output_x.txt* files should have one entry per line as it is shown below:

```
john
12
Bill
23
Alex
43
```

Valid Assumptions:

- You can assume that the input and output files are in the same directory as you executable.
- You can assume that the reading command will be performed only once in each execution of your program, and it will be always the first command.
- You can assume the files are properly formed e.g. an integer follows the r command, as in the command file example given above.

Development issues:

You can develop your program in a C++ environment of your choice. However, for the submitted program you should ensure the following:

1. Your program should be able to compile using g++ (using Linux or Windows/cygwin). If you do not know how to do that, the computing lab GTA's will be happy to show you.
2. Your program should be executed as follows:
   assignment1.exe commandFile.txt
   where assignment1.exe is the name of your executable file and commandFile.txt is the name of the command file.
3. You can download a test harness that can be used for testing your program with different input sets. You will find a README file which describes how to use the test harness.

Marking scheme:

Your program will be marked using the following criteria:

1. **Correct** and **efficient** operation for
    1. reading data from the file (10%)
    2. sorting the data (20%)
    3. inserting new data (20%)
    4. deleting an entry (20%)
    5. writing the data to the file (10%)
2. Clear and understandable code, use of functions (e.g comments, proper name selection) (20%)
3. Assignment 1 has a 50% weight, where Assignment 2 has a 50% weight of the overall mark.

Deliverables:

- You should submit via the "Assessment | C++ coursework part 1 of 2" section of the Blackboard course named "E2.18 Algorithms and Data Structures".
- The source code should including your real, email and login name in a comment at the start of the file.