Objected-oriented Programming Report

Assignment #2

|  |
| --- |
| **Professor**  김영민 교수님 |
| **Department**  Computer engineering |
| **Student ID** 2013722004 |
| **Name** 최민기 |
| **Class** 설계 : 1(Mon) , 2(Wed) / 실습 : 3, 4 (Thu) |
| **Date**  2017. 4. 28 (Fri) |

1. To save data efficiently and to avoid wasting resources, you must declare variables to fit the size of the data. In this program, **you must write your own functions(*my\_len, my\_cpy, my\_tok*)**to save data efficiently. The data is stored in ‘**student.txt**’ without line breaks. The proto types of your own functions are as follows:

int my\_len(char \*str1)

char\* my\_cpy(char \*str2, char \*str1)

char\* my\_tok(char \*str1, char \*delimiters)

The input file contains the name of university, major, and ID of student**.** Use only as *my\_len*, *my\_cpy*, and *my\_tok* order. When you use *my\_cpy*, you should use **new** and **delete**. Do not print original string when you print the result of *my\_cpy*.

**Result Screen)**

|  |
| --- |
| [Student.txt]  Kwangwoon\_Univ. computer\_engineering 2010720041 |
|  |
| You can see the result of outputiing a string of data length, copied string, and delimeter. |

**Explanation)**

This Program does not use the string functions that we used easily but implements the String function in our own style. It reads the notepad one line and shows the process of copying data and calculating string length and separating it into tokens.

**Consideration)**

Implementing String functions using file pointers was not that difficult. I think about that “Why not use the String function?”. If use a string function when you are writing code, a warning occurs. implement your own string function. But you use own string function, you can get the same result and no warning.

1. Write a program that counts the number of characters. First, the program reads each line from the ‘**input.txt**’(the number of characters in one line is under 128)and saves the data by using dynamic memory allocation method in C++(e.g., **new** and **delete**). Then the program counts **the number of characters including spaces** in each line, calculates **the average length** of the number of characters in each line, and finds out **the longest line**. Calculate the result rounded off to the first digit after the decimal point. If you find the line with the same number of characters when you print the longest line, print the smallest line number.

**Result Screen)**

|  |
| --- |
| **[input.txt]**  It's been a long day without you, my firend  And I'll tell you all about it when I see you again  We've come a long way from where we began  Oh, I'll tell you all about it when I see you again  When I see you again |
|  |
| You can see that the line with the longest data length, average length, length is output. |

**Explanation)**

This program loads notepad and saves data using dynamic allocation when saving data. Then calculate the longest line for each line length and average length.

**Consideration)**

When load and save data, I have to get a dynamic allocation. At this point, it was most important to have + 1 of the data length. Because when load the data, there is a “\n” value at the end of the data. Program does not work properly when dynamic allocated incorrectly. It was an extension of question 1 and there was no major difficulty in implementing it.

1. Write a program that manages music information. A rank, title, singer, and release year of the music are stored in 'MusicList.txt' divided by ‘/’. You must implement **'Insert', 'Update', 'Delete', and 'Sort' functions** that work as follows.

− **Insert function**: Insert the rank, title, singer, and release year of music in 'MusicList.txt'.

− **Update function**: Find the music with the rank of the music and update the music information in 'MusicList.txt'.

− **Delete function**: Find the music with the rank of the music and delete the music information in 'MusicList.txt'.

− **Sort function**: Sort the music information in 'MusicList.txt' by the rank in ascending order, title in ascending order, singer in ascending order, or release year in descending order. If the singers or release years are the same, sort by the rank. It is assumed that there is no case where the ranks are the same.

**Whenever each function is executed, the result is reflected in 'MusicList.txt'.** The output should be as follows.

**Result Screen)**

|  |
| --- |
| [MusicList.txt] |
| 9/에라 모르겠다/BigBang/2002  2/노래/Zion.T/2006  6/좋다고 말해/볼빨간사춘기/2017  1/밤편지/아이유/2017  44/피 땀 눈물/방탄소년단/2008 |
|  |
| [MusicList.txt] with Insert |
|  |
|  |
| [MusicList.txt] with Update |
|  |
|  |
| [MusicList.txt] with Delete |
|  |
|  |
| [MusicList.txt] with Sort and Quit |
|  |
| Simple Result screen that Performing Insert, Update, Delete, Sort, and Quit functions |

**Explanation)**

It is a program that corrects notepad that contains Music info (Rank/Title/Singer/release year) according to each function. Insert function is inserts new music information in notepad. The update function modifies the information of the music you want to modify. And the delete function delete line you want. And Sort function is sort saved music information by the rank/title/singer/release year.

**Consideration)**

I first saw lecture materials and implemented them using the functions in the lecture materials. But it was difficult for me to implement it with functions in the lecture material. So I wrote the code with a structure that carries music information. It was not difficult to modify the music list file with the structure that carries music information. I have tried many times to implement the functions in the lecture materials, but I spent a lot of time here. But this problem has helped me to implement the rest.

1. Write a program that receive information of student from ‘**information.txt**’ and insert the token information in class. A number on the first line of the input file means the number of students’ information following. You have to use dynamic allocation class array. Then, print who is the oldest and print the oldest student’s information. If there are same age in the list, you have to choose the student who has the smallest student’s ID.(Ex.2010722001<2017722001) When implementing, please observe the following code.

**Result Screen)**

|  |
| --- |
| [information.txt] |
| 5  Han\_Mangi example@email.com 01023153221 2010720041 27  Par\_hyoju example2@email.com 01023514654 2021354855 25  Ho\_Bbangman example3@eamil.com 02135125462 2017215462 23  Inom\_shakeit example4@email.com 03215132153 2014231521 23  Gray\_shakeit example5@email.com 01021315484 2015854152 22 |
|  |
| The result of retrieving the oldest student by bringing up student inform in the notepad. |

**Explanation)**

This program loads the student’s information from Notepad and inserts it into the class. The important point here is to create a dynamic allocation class array. It also outputs the information of the oldest of the information in the class array.

**Consideration)**

It’s a little strange because I’ve never used code to implement a class. I was able to learn more about the class by listening to the design class. After clarifying concept of the class, I can see that this problem is an extension of problem 3. I had a problem with the convenience of class by changing from struct to class.

1. Write a program that manages ***KW store***’s items. A name, price, and quantity of each item are stored in 'ItemList.txt' divided by ‘/’. The program can load, insert, update, delete, print, and save information of items. You must implement **‘Item’ class** which holds the name, price, quantity, and sequence of each item. **The sequence of the item is newly created when it is stored in the class.** You must implement **'Load', 'Insert', 'Update', 'Delete', 'Print', and 'Save' functions** that work as follows.

− **Load function**: Read item's information from 'ItemList.txt' and store them in an array of class type.

− **Insert function**: Insert the name, price, and quantity of an item to the array of class type.

− **Update function**: Find the item with item’s sequence which you want to update and update the item’s information.

− **Delete function**: Find the item with item’s sequence which you want to delete and delete the item’s information.

− **Print function**: Print all the information of items stored in class sorted by sequence, name, price, or quantity. If prices or quantities are same, items are sorted by their sequences.

− **Save function**: Save the information of items in class to 'ItemList.txt' file.

**Result Screen)**

|  |
| --- |
|  |
|  |
| The result screen shows the item list information that you have loaded and that you did not load more than 10 |
| <Print function> |
|  |
| The result screen show when outputting about the sequence, it outputs 10 pieces of information in the item list in sequence order. |
| <Delete function> |
|  |
| The result screen show the delete function to delete information corresponding to sequence 1 |
| <Update function> |
|  |
| You can see the result of modifying the corresponding sequence item information by using the update function. |
| <Save & Quit > |
|  |
|  |
| You can see that the modified item list is saved in Notepad by the saved function. |

**Explanation)**

This program performs item information(sequence/name/price/quantity) according to each function described above.

**Consideration)** It was an extension of Problems 3 and 4. It was not difficult to import Notepad into a class array, and implement functions such as insert. The problem here is that when outputting sorted information, it was a little difficult to output sorting result instead of sorting the information in the class array.

1. Write a program that prints a score of each student in object-oriented programming classes. The day of lectures, name, and score of each student are stored in 'ScoreList.txt' . Object-oriented programming classes consist of 4 different days of lectures(i.e., Wednesdays, Thursdays, Fridays, and non-attendance), and **the text file must store 40 students’ data(10 students for each day)**.The program can print the data of students **sorted by name in ascending order or score in descending order**. Assume that there is no case the number of the students’ data is not 40 in the text file.

In order to store student's score data, you must implement **'Wed', 'Thu', 'Fri', and 'None' class**, and these four classes inherit **'OOP' class** .'OOP' class has **an array of 'Student' class type** as a member variable. **'Student' class** has student names and scores as member variables. When the program starts, it reads the student's score data in the text file and **stores them according to the day of lecture**. Then, the user inputs the sorting condition(the name or score) and day of lectures through the keyboard. The program prints student’s score data sorted. If the names are the same, sort by scores. If the scores are the same, sort by names. Assume that there is no case in which both the name sand the scores are same. **When implementing, you must use polymorphism which is a feature of object-oriented programming and you must observe the following code.**

**Result Screen)**

|  |
| --- |
| <Wed class sorted by name> |
|  |
| <Wed class sorted by Score> |
|  |
| <Thu class sorted by Name> |
|  |
| <Thu class sorted by Score> |
|  |
| <Fri class sorted by Name> |
|  |
| <Fri class sorted by Score> |
|  |
| <None class Sorted by Name> |
|  |
| <None class Sorted by Score> |
|  |
| <Quit> |
|  |

**Explanation)**

This Program is that OOP class are inherited by day of week, and OOP class have a Student class array as member variables. Class for each day of week inherited from the OOP class have the information of the students separated by the day of the week. The overall configuration is as above, input by keyboard, Sorting information of the day of the desired day, and output the result.

**Consideration)**

This problem was implemented using inheritance polymorphism and virtual functions. This was the second most difficult Second assignment. It is difficult because of the unfamiliar and precise concept of inheritance polymorphism. When inheriting from an OOP class, the OOP class carries the Student class array as a member variable, so each class of each OOP class inherits the class with the Student class array. As I learned this process, I wrote the code. In the 6 Question, I was able to know the most important concept of inheritance and access to variables in the class.

1. Debugging your program. Follow the code and report of your data values and order of function. Capture your screen and report how to debug as detail as you can. In this assignment you do not have to make your program. You should use the assignment 1-6 sources. Debug the sources when input C is 1 and input n is 6. Report as detail as you can. Focus on break point.

|  |
| --- |
| <Step 1 : Debug Start > |
|  |
| You can start debugging by pressing ‘**F10**’ |
| <Step 2 : one-step debugging |
|  |
| You can perform one-step debugging by pressing **‘F10’** |
| <Step 3 : Function Access |
|  |
| This is where the current cursor outputs the Fibonacci function. If you press ‘F11’ at this time, you access Fibonacci function and you able to debug |
| <step 4 : Automatic, Local, watch window > |
|  |
| Automatic is fill in the watch window automatically |
|  |
| Watch window can check the information of the variable by specifying the variables that the user wants to see. Also you can substitute the expression you want in the fomula. |
|  |
| Local is a place to replicate local variables. |
| <step 4 : Break Point > |
|  |
|  |
| Since I have composed Loop 1-6 problem, I took breakpoints in the for statement. Since the result of the Fibonacci function is **‘current’** while performing the loop, I give a breakpoint by attaching a conditional expression to ‘**i’**. If you want to know the data change flow when test case is 1 and n is 6, You can set a breakpoint at the beginning of the for loop I and at the end of the for loop I, 2 and 7. |
|  |
| When you press F10 key, you can see current which shows the result when the first breakpoint I value is 2. |
|  |
|  |
|  |
|  |
| Therefore, by continuously pressing the control F10 key, the i value sequentially increases, and the data flow of the current value is known as the result value. |
|  |
| Finally, we can see that the current value is 13. |

**Reference**)

Assignment\_lecture\_1 ( 김영민 교수님, 한만기 조교님, 고은정 조교님 )

Assignment\_lecture\_2 ( 김영민 교수님, 한만기 조교님, 고은정 조교님 )

Assignment\_lecture\_3 ( 김영민 교수님, 한만기 조교님, 고은정 조교님 )