Sabancı University

Faculty of Engineering and Natural Sciences
CS204 Advanced Programming
Summer 2021-2022

Take-Home Exam 6 (BONUS) – Outside The Wall
Due: 6 September 2022 11.55pm (SHARP - REALLY)

DISCLAIMER:

Only checking the sample run cases might not be sufficient as your solution will be checked against a variety of samples different from the provided samples; however checking these cases is highly encouraged and recommended.

You can NOT collaborate with your friends and discuss your solutions with each other. You have to write down the code on your own. Plagiarism will not be tolerated AND cooperation is not an excuse!

Introduction

The aim of this assignment is to practice inheritance and polymorphism. The goal is to implement a base abstract class with three subclasses. The base class will be a container class which is an abstract class. Then you will implement a **sorted LinkedList, Stack** and **Queue** classes that inherit from the **Container** class. The main function is given and you are only supposed to implement the classes.

This assignment will **not** be included in the curve, so you will not lose anything if you do not submit it. If you do the assignment and if it works as desired, you will have 3 points added to your overall grade! Thus, it is completely your choice :)

Please note that we will **not** do any manual grading except for checking the obligations. Your grade can be at most what your program gets from CodeRunner if you obey the rules. Please note that this is an "ALL OR NOTHING"-grading assignment: there will **not** be any partial credits.

Program Flow

The program will iteratively ask the user to select a data structure and then an operation.

The data structures are: **0- LinkedList, 1- Stack, 2- Queue**

The operations are:

0- Insert: Insert an integer to the corresponding data structure. You can assume that no duplicates will be inserted.

1- Delete: Delete an integer from the corresponding data structure.

2- Print: Print the corresponding data structure.

The program will terminate *if the user enters 3 as the first of the coupled inputs*, which semantically stands for "Exit" (i.e. **3- Exit**).

The base class will define these functions as pure virtual functions. Then all the subclasses will implement these functions according to their definitions, as given below. Also, every data structure should display a message explaining the operation being carried out. For further information, please refer to the sample runs.

We will inspect your class implementations thoroughly. This means, disobeying the Object Oriented Design or Inheritance philosophies will hurt your grade badly.

	Sorted Linked List	**Stack**	**Queue**
Insert	Insert in order	Insert to the top	Insert to the rear
Delete	Delete number	Delete number if it is at top	Delete number if it is at front
Print	Print the contents of the respective container		

For the sorted linked list class, we will never use a case where a non-existing element will be supplied for deletion.

We will run your class codes with **another** *main.cpp* for grading purposes. Hence, you **must** implement the required classes in another file. You may have different files for each class or you may implement all in a single file. However, if you modify the main, then your class implementations will be lost during grading.

Sample Runs

Below, we provide some sample runs of the program that you will develop. The *italic* and **bold** phrases are the standard input (cin) taken from the user (i.e., like ***this***). You have to display the required information in the same order and with the same words/spaces as here; in other words, there must be an exact match!

Sample Run 1

```
0
0
5
5 is inserted into the linked list.

0
2
Printing the linked list:
5

0
0
8
8 is inserted into the linked list.

0
2
Printing the linked list:
5 8

0
0
2
2 is inserted into the linked list.

0
2
Printing the linked list:
2 5 8

0
1
5
5 is deleted from the linked list.

0
1
2
2 is deleted from the linked list.
```

```
0
2
Printing the linked list:
8

1
1
5
5 cannot be deleted from the stack.

1
0
1
1 is inserted into the stack.

1
1
7
7 cannot be deleted from the stack.

1
1
1
1 is deleted from the stack.

1
2
Printing the stack:


2
2
Printing the queue:


2
0
7
7 is inserted into the queue.

2
0
1
1 is inserted into the queue.

2
2
Printing the queue:
7 1
```

```
2
1
1
1 cannot be deleted from the queue.

2
1
7
7 is deleted from the queue.

2
2
Printing the queue:
1

3

Destructing the linked list.
Destructing the stack.
Destructing the queue.
Exiting..
```

Some Important Rules

In order to get full credit, your program must be efficient, modular (with the use of functions), well commented and properly indented. Besides, you also have to use understandable identifier names. Presence of any redundant computation, bad indentation, meaningless identifiers or missing/irrelevant comments may decrease your grade in case that we detect them. When we grade your THEs, we pay attention to these issues. Moreover, **we may test your programs with very large test cases**. Hence, take into consideration the efficiency of your algorithms other than correctness.

Sample runs give a good estimate of how correct your implementation is, however, we will test your programs with different test cases and **your final grade may conflict with what you have seen on CodeRunner**. We will also **manually** check your code, indentations and so on, hence do not object to your grade based on the **CodeRunner** results, but rather, consider every detail on this documentation. **So please make sure that you have read this documentation carefully and covered all possible cases, even some other cases you may not have seen on CodeRunner or the sample runs**. The cases that you *do not need* to consider are also given throughout this documentation.

Submit via SUCourse ONLY! Paper, e-mail or any other methods are not acceptable.

The internal clock of SUCourse might be a couple of minutes skewed, so make sure you do not leave the submission to the last minute. In the case of failing to submit your THE on time:

"No successful submission on SUCourse on time = A grade of zero (0) directly."

What and where to submit (PLEASE READ, IMPORTANT)

It'd be a good idea to write your name and last name in the program (as a comment line of course). Do not use any Turkish characters anywhere in your code (not even in comment parts). If your full name is "Duygu Karaoğlan Altop", and if you want to write it as comment; then you must type it as follows:

// Duygu Karaoglan Altop

You should copy the full content of the .cpp file and paste it into the specified “Answer” area in the relevant assignment submission page on SUCourse. ***Please note that the warnings are also considered as errors on CodeRunner, which means that you should have a compiling and warning-free program***.

Since the grading process will be automatic, you are expected to strictly follow these guidelines. If you do not follow these guidelines, your grade will be zero (0). Any tiny change in the output format will result in your grade being zero (0), so please test your programs yourself, and against the sample runs that are available at the relevant assignment submission page on SUCourse.

In the CodeRunner, there are some visible and invisible (hidden) test cases. You will see your final grade (including hidden test cases) before submitting your code. There is no re-submission. You don't have to complete your task in one time, you can continue from where you left last time but you should not press submit before finalizing it. Therefore, you should make sure that it’s your final solution version before you submit it. Also, we still do not suggest that you develop your solution on CodeRunner but rather on your IDE on your computer.

You may visit the office hours if you have any questions regarding submissions.

How to get help?

You may ask your questions to TAs or to the instructor. Information regarding the office hours of the TAs and the instructor are available at SUCourse.

Plagiarism

Plagiarism is checked by automated tools, and we are very capable of detecting such cases. Be careful with that. Exchange of abstract ideas are totally okay but once you start sharing the code with each other, it is very probable to get caught by plagiarism. So, do NOT send any part of your code to your friends by any means or you might be charged as well, although you have done your THE by yourself. THEs are to be done personally and you have to submit your own work. **Cooperation will NOT be counted as an excuse.**

In case of plagiarism, the rules on the Syllabus apply.

Good Luck!
Ahmed Salem, Duygu Karaoğlan Altop