**Sabanci University**

Faculty of Engineering and Natural Sciences

CS204 Advanced Programming

Summer 2018-2019

Homework 5 (BONUS) – Outside The Wall

Due: 20 August 2019 11.55pm (SHARP - REALLY)

|  |
| --- |
| **DISCLAIMER:**  **Your program should be a robust one such that you have to consider all relevant user mistakes and extreme cases; you are expected to take actions accordingly!**    **Only checking the sample run cases might not be sufficient as your solution will be checked against a variety of samples different than the provided samples; however checking these cases are highly encouraged and recommended.**    **You must** 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 homework 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 homework will **not** be included in the curve, so you will not lose anything if you do not submit it. If you do the homework and if works as desired, you will have 2 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 will be exactly what your program gets from GradeChecker if you obey the rules. Please note that this is an "ALL OR NOTHING"-grading homework: 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 operations are:

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

**Delete:** Delete an integer from the corresponding data structure.

**Print:** Print the corresponding data structure.

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!

We will be automatically grading your homework using GradeChecker, so it is very important to satisfy the exact same output given in the sample runs. You can utilize GradeChecker [(http://sky.sabanciuniv.edu:8080/GradeChecker/)](http://sky.sabanciuniv.edu:8080/GradeChecker/) to check whether your code is working in the expected way. To be able to use GradeChecker, you should upload all of your files used in the homework **without zipping them**. Just a reminder, you will see a character ¶ which refers to a newline in your expected output.

**Sample Run 1**

Enter your selection(0-LinkedList, 1-Stack, 2-Queue, 3-Exit): ***0***

Enter the operation(0-insert, 1-delete, 2-print): ***0***

Enter the number to be inserted: ***5***

5 is inserted into the linked list.

Enter your selection(0-LinkedList, 1-Stack, 2-Queue, 3-Exit): ***0***

Enter the operation(0-insert, 1-delete, 2-print): ***2***

Printing the linked list:

5

Enter your selection(0-LinkedList, 1-Stack, 2-Queue, 3-Exit): ***0***

Enter the operation(0-insert, 1-delete, 2-print): ***0***

Enter the number to be inserted: ***8***

8 is inserted into the linked list.

Enter your selection(0-LinkedList, 1-Stack, 2-Queue, 3-Exit): ***0***

Enter the operation(0-insert, 1-delete, 2-print): ***2***

Printing the linked list:

5 8

Enter your selection(0-LinkedList, 1-Stack, 2-Queue, 3-Exit): ***0***

Enter the operation(0-insert, 1-delete, 2-print): ***0***

Enter the number to be inserted: ***2***

2 is inserted into the linked list.

Enter your selection(0-LinkedList, 1-Stack, 2-Queue, 3-Exit): ***0***

Enter the operation(0-insert, 1-delete, 2-print): ***2***

Printing the linked list:

2 5 8

Enter your selection(0-LinkedList, 1-Stack, 2-Queue, 3-Exit): ***0***

Enter the operation(0-insert, 1-delete, 2-print): ***1***

Enter the number to be deleted: ***5***

5 is deleted from the linked list.

Enter your selection(0-LinkedList, 1-Stack, 2-Queue, 3-Exit): ***0***

Enter the operation(0-insert, 1-delete, 2-print): ***1***

Enter the number to be deleted: ***2***

2 is deleted from the linked list.

Enter your selection(0-LinkedList, 1-Stack, 2-Queue, 3-Exit): ***0***

Enter the operation(0-insert, 1-delete, 2-print): ***2***

Printing the linked list:

8

Enter your selection(0-LinkedList, 1-Stack, 2-Queue, 3-Exit): ***1***

Enter the operation(0-insert, 1-delete, 2-print): ***1***

Enter the number to be deleted: ***5***

5 cannot be deleted from the stack.

Enter your selection(0-LinkedList, 1-Stack, 2-Queue, 3-Exit): ***1***

Enter the operation(0-insert, 1-delete, 2-print): ***0***

Enter the number to be inserted: ***1***

1 is inserted into the stack.

Enter your selection(0-LinkedList, 1-Stack, 2-Queue, 3-Exit): ***1***

Enter the operation(0-insert, 1-delete, 2-print): ***1***

Enter the number to be deleted: ***7***

7 cannot be deleted from the stack.

Enter your selection(0-LinkedList, 1-Stack, 2-Queue, 3-Exit): ***1***

Enter the operation(0-insert, 1-delete, 2-print): ***1***

Enter the number to be deleted: ***1***

1 is deleted from the stack.

Enter your selection(0-LinkedList, 1-Stack, 2-Queue, 3-Exit): ***1***

Enter the operation(0-insert, 1-delete, 2-print): ***2***

Printing the stack:

Enter your selection(0-LinkedList, 1-Stack, 2-Queue, 3-Exit): ***2***

Enter the operation(0-insert, 1-delete, 2-print): ***2***

Printing the queue:

Enter your selection(0-LinkedList, 1-Stack, 2-Queue, 3-Exit): ***2***

Enter the operation(0-insert, 1-delete, 2-print): ***0***

Enter the number to be inserted: ***7***

7 is inserted into the queue.

Enter your selection(0-LinkedList, 1-Stack, 2-Queue, 3-Exit): ***2***

Enter the operation(0-insert, 1-delete, 2-print): ***0***

Enter the number to be inserted: ***1***

1 is inserted into the queue.

Enter your selection(0-LinkedList, 1-Stack, 2-Queue, 3-Exit): ***2***

Enter the operation(0-insert, 1-delete, 2-print): ***2***

Printing the queue:

7 1

Enter your selection(0-LinkedList, 1-Stack, 2-Queue, 3-Exit): ***2***

Enter the operation(0-insert, 1-delete, 2-print): ***1***

Enter the number to be deleted: ***1***

1 cannot be deleted from the queue.

Enter your selection(0-LinkedList, 1-Stack, 2-Queue, 3-Exit): ***2***

Enter the operation(0-insert, 1-delete, 2-print): ***1***

Enter the number to be deleted: ***7***

7 is deleted from the queue.

Enter your selection(0-LinkedList, 1-Stack, 2-Queue, 3-Exit): ***2***

Enter the operation(0-insert, 1-delete, 2-print): ***2***

Printing the queue:

1

Enter your selection(0-LinkedList, 1-Stack, 2-Queue, 3-Exit): ***3***

Destructing the linked list.

Destructing the stack.

Destructing the queue.

Exiting..

**Some Important Rules**

Although some of the information is given below, please also read the homework submission and grading policies from the lecture notes of the first week. In order to get a full credit, your program must be efficient, modular (with the use of functions), well commented and 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 homeworks, we pay attention to these issues. Moreover, in order to observe the real performance of your code, we are going to run your programs in Release mode and **we may test your programs with very large test cases**. Hence, take into consideration the efficiency of your algorithms other than correctness.

**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 [Course Google Drive Folder](https://drive.google.com/drive/folders/1EozjiM4Ogau08Zd2gemSbNjzQxBj-Rma?usp=sharing).

**YOU SHOULD USE GRADE CHECKER FOR THIS HOMEWORK!**

You should use Grade Checker (<http://sky.sabanciuniv.edu:8080/GradeChecker/>) to check your expected grade. Just a reminder, you will see a character ¶ which refers to a newline in your expected output.

Make sure you upload the .txt files, too.

Grade Checker and the automated grading system use a different compiler than MS Visual Studio does. Hence, you should check the "***Common Errors***" page to see some extra situations to consider while doing your homework. If you do not consider these situations, you may get a lower score (even zero) even your program works correctly with Visual Studio.

***Common Errors Page***: <http://sky.sabanciuniv.edu:8080/GradeChecker/commonerrors.jsp>

Grade Checker can be pretty busy and unresponsive during the last day of the submission. Due to this fact, leaving the homework for the last day generally is not a good idea. You may wait for hours to test your homework or make an untested submission, sorrily..

Grade Checker and Sample Runs together give a good estimate of how correct your implementation is, however we may test your programs with different test cases and **your final grade may conflict with what you have seen on Grade Checker.** We will also **manually** check your code (comments, indentations and so on), hence do not object to your grade based on the Grade Checker 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 Grade Checker or Sample Runs**. The cases that you *do not need* to consider are also given in this documentation.

Submit via SUCourse ONLY! **Grade Checker is not considered as a submission**. Paper, e-mail or any other methods are not acceptable, either.

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 homework on time:

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

**What and where to submit (PLEASE READ, IMPORTANT)**

You should test your program using Grade Checker. We will use the same UNIX based C++ compiler that Grade Checker uses for grading your homework.

It'd be a good idea to write your name and lastname 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*

Submission guidelines are below. 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*. The lack of even one space character in the output will result in your grade being zero, so please test your programs yourself and with the Grade Checker tool explained above.

* Name your cpp file that contains your program as follows:

***"SUCourseUserName\_hw5.cpp"***

Your SUCourse user name is actually your SUNet username which is used for checking sabanciuniv e-mails. Do NOT use any spaces, non-ASCII and Turkish characters in the file name. For example, if your SU e-mail address is **atam@sabanciuniv.edu**, then the file name must be: **"atam\_hw5.cpp"**

* Please make sure that this file is the latest version of your homework program.
* You should upload all the .txt files and class filesto SUCourse as well.
* Do not zip any of the documents but upload them as separate files only.
* Submit your work **through SUCourse only**! You can use the Grade Checker only to see if your program can produce the correct outputs both in the correct order and in the correct format. It will not be considered as the official submission. You must submit your work to SUCourse.

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

**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 homework by yourself. Homeworks 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!

Tolga Atam, Duygu K. Altop