CS 211 Data Structures and Algorithms Lab
Autumn, 2022

Assignment no.	4
Objective	To implement Stack,Queue and Doubly Linked List
Total marks	10
Due date (without penalty)	12th September (Monday) 11:59 pm
Cut-off date (with penalty - 5%)	15th September(Thursday) 11:59 pm
Penalty for violating naming convention(s)	5%

The objective of this assignment is to implement a Stack using Linked List [Task 1], a Queue using Linked List [Task 2], and a Doubly Linked List [Task 3].

Please read this document carefully before starting coding.

Command-line argument:
Your program should receive two command line arguments: a file name followed by a number.

For example, ***'./a.out input.txt 25'*** will be a typical execution of your program.

Input file:
The input file will be a text file where each line will start with either of the four symbols: '+', '-', '?', '!', where:

- '+' stands for PUSH/ENQUEUE/INSERT
- '-' stands for POP/DEQUEUE/DELETE
- '?' stands for SEARCH
- '!' stands for DISPLAY

Notes:
(i) '+', '?', and '-' are followed by a positive integer, which is the argument for that particular operation
(ii) the number followed by '-' should be ignored for Task 1 (Stack) and Task 2 (Queue), as it is not relevant

Task 1: Implement a Stack using Linked List

- *The size of the Stack* the number of elements that can be stored in the Stack is given by the second command-line argument.

  Output:
  The output of this task should be in a file named '*stack.txt*'. For every line of the input file there should be a corresponding line in the output file.

  - If there is a line with ***'+ y'*** then your program should ***PUSH*** (if possible) the number 'y' into the Stack and then print 'pushed y' or 'overflow' (whichever is applicable) to the output file.
  - If there is a line with ***'-'*** in the input file then your program should ***POP*** (if possible) an element from the Stack and print 'popped y' or 'underflow' (whichever is applicable) into the output file.
  - A line with ***'? y'*** should cause your program to check if y is there in the Stack and print ***"found y" or "not found y"*** accordingly into the output file.
  - A line with ***'!'*** should cause your program to ***print*** (last-in first) the content of the Stack into the output file.

  For example, the following outline shows the output file *stack.txt* corresponding to the input.txt file provided through CLI. The stack-size is set 5.

Input.txt	*stack.txt*
+ 10	pushed 10
+ 24	pushed 24
+ 1	pushed 1
!	1 24 10
- 24	popped 1
- 1	popped 24
? 10	found 10
? 9	not found 9
!	10

Task 2 : Implement a Circular Queue using singly Linked List

- *The size of the Queue* (the number of elements that can be stored in the Queue) is given by the second command-line argument.

  Output:
  The output of this task should be in a file named *'queue.txt'*. For every line of the input file there should be a corresponding line in the output file.

  - If there is a line with ***'+ y***' then your program should ***ENQUEUE*** (if possible) the number 'y' into the Queue and then print 'enqueued y' or 'overflow' (whichever is applicable) to the output file.

  - If there is a line with ***'-'*** in the input file then your program should ***DEQUEUE*** (if possible) an element from the Queue and print 'dequeued y' or 'underflow' (whichever is applicable) into the output file.

  - A line with ***'? y'*** should cause your program to check if y is there in the Queue and print ***"found y" or "not found y"*** accordingly into the output file.

  - A line with ***'!'*** should cause your program to ***display*** (first-in first) the content of the Queue.

For example, the queue.txt should contain the following lines when invoked with the given input and second command-line argument 5:

Input.txt	*queue.txt*
+ 10	enqueued 10
+ 24	enqueued 24
+ 1	enqueued 1
!	10 24 1
- 10	dequeued 10
- 24	dequeued 24
? 10	not found 10
? 9	not found 9
!	1

Task 3 : Implement a Doubly Linked List (DLL)

- *There is no size limit for the DLL*. So, the second *command-line argument is ignored for this task.*

Output:

The output of this task should be in a file named 'dll.txt'. For every line of the input file there should be a corresponding line in the output file.

● If there is a line with ***'+ y'*** then your program should ***INSERT*** the number 'y' into the DLL and then print 'inserted y' to the output file.

● If there is a line with ***'- y'*** in the input file then your program should ***DELETE*** (if possible) the first occurrence (while searching from list head) of y from the DLL and print 'deleted y' or 'cannot delete y' (whichever is applicable) into the output file.

● A line with ***'? y'*** should cause your program to check if y is there in the DLL and print ***"found y" or "not found y"*** accordingly.

● A line with ***'!'*** should cause your program to ***display*** (last-in first) the content of the DLL.

For example, the dll.txt should contain the following lines when invoked with the given input:

Input.txt	*dll.txt*
+ 10	inserted 10
+ 24	inserted 24
+ 1	inserted 1
!	1 24 10
- 24	deleted 24
- 1	deleted 1
? 10	found 10
? 9	not found 9
!	10

Submission:

- The program you submit should output ***statck.txt, queue.txt*** and ***dll.txt*** when run **(There should be only one main program to handle all the three tasks - of course, the source code can contain multiple files).**
- The main file of your program should be named as <roll no>.<extension>, where roll no. specifies your roll no. and the extension depends on the language you choose (Usage of C is mandatory for this assignment). Ex: 210010001.c.
- Do the stress test of your program well before submission.
  (i) You may use the attached sample input files for testing; the corresponding output files are also attached;
  (ii) We have some hidden inputs with us to test your program. *The mark you obtain is purely based on whether your program correctly gives outputs for the hidden inputs*.
- If your program has only a single source file, please submit the file as it is. If your program has multiple source files, please submit your code as a zip file where the name of the zip file should be your roll number. It is important that you follow the ***input/output conventions*** exactly (including the naming scheme) as we may be doing an automated evaluation. ***There will be a penalty of 5% (on the mark you deserve otherwise) if you do not follow the naming conventions exactly.***
- Follow some coding style uniformly. Provide proper comments in your code.
- *Submit only through moodle*. ***Submit well in advance***. Any hiccups in the moodle at the last minute is never acceptable as an excuse for late submission. Submissions through email or any other means will be ignored.
- Acknowledge the people (other than the instructor and TA) who helped you to solve this assignment. The details of the help you received and the names of the people who helped you (including internet sources, if applicable) should come in the beginning of the main file as a comment. ***Copying others' programs is a serious offense and a deserving penalty will be imposed if found.***

Evaluation:

- To consider for the first evaluation without penalty, you have to submit your program by the due date. If you submit after the ***due date*** but on or before the ***cut-off date***, ***there will be a penalty of 5% on the marks you deserve otherwise***.
- If you do not submit by the ***cut-off date,*** your program will not be considered for the first evaluation.
- We will do the first evaluation after the cut-off date. *The marks you obtain will be proportional to the number of correct lines in the output files*. We will use the ***'diff'*** program to check the differences between the correct output file and the output file generated by your program. So, you may verify the correctness of the output file by using the diff program with a sample output file before submission. (See the man page of diff for more info).
- We will do the second evaluation later. This is for those who want to improve their marks obtained in the first evaluation or who do not submit for the first evaluation. There will be ***a penalty of 20%*** for those who are submitting for the second evaluation. The details of the second evaluation will be shared later.