Lab Exercise 6

Task 1: Suppose you are building a program to manage a to-do list for a busy professional. You decide to use a stack to keep track of the items on the list. Write a code snippet that demonstrates how you would add a new item to the top of the stack, check if the stack is empty, and remove the top item from the stack.

Task 2: Suppose you are implementing a program to keep track of a user's web browsing history using a linked list and a stack. Whenever the user visits a new website, the website URL is added to the front of the linked list to represent the most recent page visited. Additionally, the website URL is pushed onto a stack to allow for the user to easily navigate back to the previous page.

Now suppose the user has visited 5 websites in the following order: Google, Facebook, Twitter, LinkedIn, and Instagram. The user decides to click the "back" button twice to return to the Facebook page. Write a function to implement this behavior using the linked list and stack.

Hint: You will need to pop two elements from the stack and remove the first two nodes from the linked list.

Task 3: You are developing a calculator application in C++ that needs to handle arithmetic expressions entered by the user. The expressions can include the basic arithmetic operators (+, -, *, /) as well as parentheses to group operations. How would you convert the infix expression a+b*(c^d-e)^(f+g*h)-i to postfix notation.

Use the class of stack created in Task 01 (Lab Tasks).

Task 4: Consider you have an expression x=12+13-5(0.5+0.5) +1 which results to 20. Implement a stack-based implementation to solve this question via linked lists (linked lists can be single or double) and the resulted output must be at the top of the stack. Note that the x and the equal sign must be present in the stack and when inserting the top value (20 result) all the values must be present in the stack (You can pop and push them accordingly)

Task 5: Implement a Queue based approach where assume you are the cashier in a supermarket and you need to make checkouts. Customer ID's Are 13,7,4,1,6,8,10. (Note: Use Arrays to accomplish this task with enqueue and dequeue)

Task 6: Consider a messaging application where users can send and receive messages. The application can only handle one message at a time, and each message can take a variable amount of time to send or receive. As messages arrive, they are added to a queue for processing. Once a message is completed, the next message in the queue is processed. What type of queue data structure would be most suitable for the given scenario?

Task 7: Consider a library where books are borrowed and returned by multiple patrons. The library has a limited number of staff members to handle the book transactions. As patrons arrive with books to borrow or return, they are added to a linear queue. Each book transaction can take a variable amount of time to complete, depending on factors such as the number of books being borrowed or returned, and the availability of the staff members. Once a book transaction is completed, the next patron in the queue is serviced.

Write a C++ program that simulates this library scenario using a linear queue data structure. The program should allow patrons to add themselves to the queue, remove themselves from the queue when their book transaction is completed, and display the current queue of patrons waiting for book transactions to be serviced.