Tutorial No. 9 : Linked Lists, Queues, and Stacks

Exercise 1 :

Let L1 and L2 be two singly linked linear lists. Write a procedure that constructs the list L containing all elements of L1 that are not in L2.

Exercise 2 :

A doubly linked list is a list in which each element is linked to both the next element and the previous element in the list.

1. Create such a list.
2. Write a procedure that removes the element located at the i-th position.
3. Write a procedure that inserts a value val after a given value U.

Exercise 3 :

Write a procedure that creates:
The reversed stack of a queue.
The reversed queue of a stack.

Exercise 4 :

Write an algorithm that checks whether a word (expression) is properly parenthesized.
Example :

- Well-parenthesized words : (), ()(()) ou (((())())()).
- Not well-parenthesized words :)(,)(() ou ((()())))).

To do this, we use a stack of characters. We store the opening parentheses that have not yet been closed in the stack. The program reads the input word character by character:

- If it is an opening parenthesis, it is pushed onto the stack.
- If it is a closing parenthesis, the corresponding opening parenthesis is popped.

The word is accepted if: :

- The stack is never empty when reading a closing parenthesis;
- The stack is empty once the entire word has been read.

printf("It is by trying again and again that one finally succeeds. ");