**Aim:**

Write a program to implement a Stack using Linked List.

**Algorithm:**

Step 1 - Start.

Step 2 - Include all the header files used in the program and declare the user-defined functions, push(), pop() and display().

Step 3 - Define a structure 'Node' with two members, 'data' and 'next'.

Step 4 - Define a Node pointer 'head'.

Step 5 - Display a list of operations, get a choice from the user and call functions accordingly.

Step 6 – In push() function, declare variable 'data' and dynamically allocate Node pointer 'ptr'.

Step 7 – Get element to be added in stack and store in variable data.

Step 8 - Check whether the stack is Empty (head == NULL)

Step 9 - If it is Empty, then set ptr -> data = data, ptr → next = NULL and head  = ptr .

Step 10 - If it is Not Empty, set ptr -> data = data, ptr → next = head and head  = ptr.

Step 11 – In pop() function, declare variable 'temp' and node pointer 'ptr'.

Step 12 - check whether stack is Empty (head  == NULL).

Step 13 - If it is Empty, then display "UNDERFLOW!" and terminate the function

Step 14 - If it is Not Empty, then set temp = head -> data and ptr = head.

Step 15 - Set 'head = head → next' and free ptr.

Step 16 – In display() function, Declare Node pointer 'ptr' and set it to 'head'.

Step 17 - Check whether the stack is Empty (head == NULL).

Step 18 - If it is Empty, display ' Empty Stack!' and terminate the function.

Step 19 - If it is Not Empty, Display 'ptr → data' and move to the next node. Repeat the same until ptr reaches the first node in the stack. (ptr → next != NULL).

Step 20 – Stop.

**Result:**

The program was successfully executed and achieved the aim of the program.

**Output:**

  