ALGORITHM:

Singly Linked Stack

Step1: Start

Step2: Declare the node and the required variables

Step3: Declare the function for push, pop, display and search an element.

Step4: Read the choice from the user

Step5: If the user choose to push an element, then read the element to be pushed & call the function to push the element by passing the value to the function

Step5.1: Declare the newnode & allocate memory for newnode

Step5.2: Set newNode → data = value

Step5.3: check if top == null then set newNode → next = NULL

Step5.4: Set newnode → next = top

Step5.5: set top = newNode & then print insertion is successful.

Step6: If user choose to pop an element from the stack then call the function to pop the element

Step6.1: check if top == null then print stack is Empty

Step6.2: Else declare a pointer variable temp & Initialize it to top

Step6.3: print the element that being deleted

Step6.4 : set temp = temp → next

Step6.5 : free the temp.

Step 7 : If the user choose the display then call the function to display the element in the Stack

Step7.1 : check if top = Null then print stack is empty.

Step7.2 : Else declare a pointer variable temp & initialize it to top

Step7.3 : Repeat steps below while temp → next != Null

Step7.4 : print temp → data.

Step7.5 : Set temp = temp → next

Step 8 : If the user choose to search an element from the stack then call the function to search an element.

Step8.1 : Declare a pointer variable ptr and other neccessary variable

Step 8.2 : initialize ptr = top

Step 8.3 : check if ptr = null then print Stack empty

Step 8.4 : Else read the element to be Searched

Step 8.5 : Repeat step 8.6 to 8.8 while ptr != null

Step 8.6 : check if ptr → data == item then print element founded and to be located and Set flag = 1

Step 8.7 : Else Set flag = 0

Step 8.8 : increment i by 1 and Set ptr = ptr → next

Step 8.9 : check if flag = 0 then print the element not found.

Step 9 : End.