Date
Page

Lab-5

* WAP to implement stack using single linked List

=>

```
#include <stdio.h>
#include <stdlib.h>

struct Node {
        int data;
        struct Node *next;
}

void display (struct Node *top){
        if (top != NULL){
            printf("Stack elements are: \n")
            while (top != NULL){
                printf("%d", top -> data);
                top = top -> next;
            }
            printf("\n");
        } else {
            printf("Stack is empty\n");
        }
}
```

CamScanner

```
struct Node *push (struct Node *top,
                       int data) {
    struct Node *newNode = (struct Node*)
    malloc (sizeof (struct Node));
    if (newNode == NULL){
        printf ("Stack Overflow \n");
        return top;
    }

    newNode -> data = data;
    newNode -> next = top;
    top = newNode;

    return top;
}

struct Node *pop (struct Node *top,
  int *poppedData) {

    if (top == NULL) {
        printf ("Stack Underflow \n");
        *poppedData = -1;
        return NULL;
    }
    struct Node *temp = top;
    *poppedData = temp -> data;
    top = top -> next;
    free (temp);
    return (top);
}
```

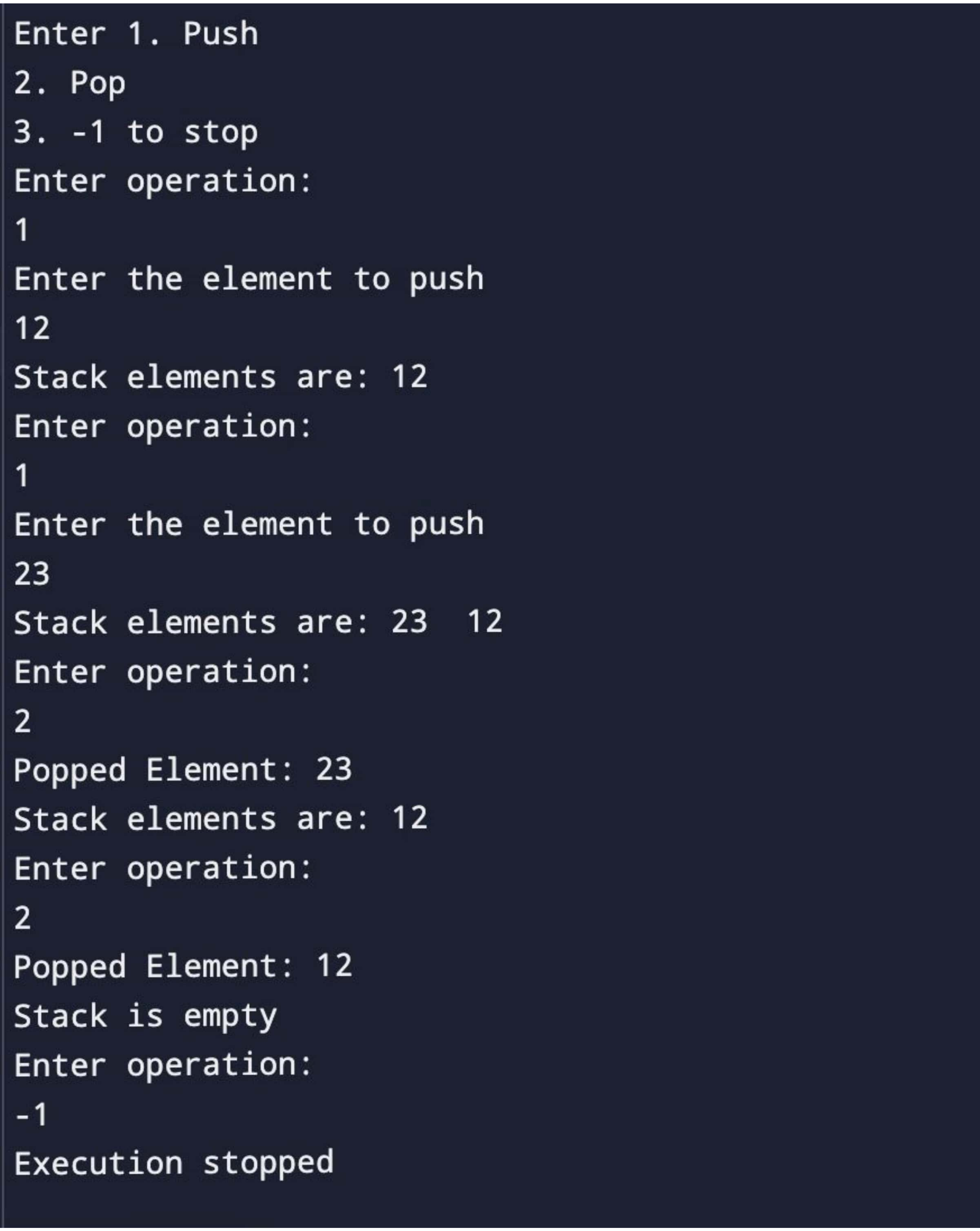
```
int main (){
    int op, n, poppedElement;
    Struct Node *top = NULL;
    printf("Enter 1. Push \n 2. PoP\n
            3. -1 to stop \n");
    while(1){
        printf ("Enter operation:\
        scanf ("%d", &op);

        if (op == -1){
            printf ("Execution Stopped
            break;
        }

        switch (op) {
        case 1;
            printf ("Enter the element
             to push \n");
            scanf ("%d", &n);
            top = push(top, n);
            break;

         case 2 :
            top = pop(top, &poppedElement);
            if(poppedElement != -1){
              printf("Popped Element: %
                      poppedElement)
            }
            display (top);
    }
    return 0;
}
```

Output;

SLL - stacks, Queues.

[illegible] 29/11/24.

```
Enter 1. Push
2. Pop
3. -1 to stop
Enter operation.
1
Enter the element to push
21
Stack elements are: 21
Enter operation.
1
~~Enter~~ elements to push
34
Stack elements are 34 21
Enter operation
2
Popped element: 34.
```



```
Enter 1. Push
2. Pop
3. -1 to stop
Enter operation:
1
Enter the element to push
12
Stack elements are: 12
Enter operation:
1
Enter the element to push
23
Stack elements are: 23  12
Enter operation:
2
Popped Element: 23
Stack elements are: 12
Enter operation:
2
Popped Element: 12
Stack is empty
Enter operation:
-1
Execution stopped
```