Question

Design and implement a stack (Array implementation/ Linked list implementation) and demonstrate its working with necessary inputs. Display the appropriate messages in case of exceptions

Aim: To Implement Stacks using Linked List

Algorithm for Push

1. Allocate memory for a pointer ptr
2. If memory is not allocated to ptr and its value is null, then throw an exception saying that there isn't enough memory
3. If memory is allocated then set the value and set its node pointer to head

Algorithm for Pop

1. If head is null, then throw an exception saying that stack underflows.
2. Else, make a new pointer which points to the node next to head.
3. Now make this pointer the head.

Algorithm for display

1. Make a new node pointer which points to head.
2. If the pointer points to null then display a message saying that the stack is empty
3. Or else print the value of the current and then make the pointer point towards the next node
4. Repeat step 3 till the pointer points to null.

Program

```
#include<stdio.h>
#include<stdlib.h>
 struct node        //creating a structure type for nodes
 {
    int val;
    struct node *next;


 };
 struct node* head=NULL;        //initializing head as null


 void push()            //push function
 {
    int x;
```

```
    struct node *ptr=(struct node*)malloc(sizeof(struct node));
//allocates memory for the push function
     if(ptr==NULL)
    {
        printf("Not able to push the element");
        exit(1);
    }

        printf("Enter the value");
        scanf("%d",&x);

            ptr->val=x;
            ptr->next=head;
            head=ptr;

        printf("Item pushed");

}
void pop()      //pop function
{
    int item;
    struct node *ptr;
    if(head==NULL)
    {
        printf("stack underflow");
    }
    else
    {
        item=head->val;
        ptr=head;
        head=head->next;
        free(ptr);
        printf("Item popped %d\n",item);
    }
}

void display()      //displays elements in the stack
{
    int i;
    struct node *ptr;
```

```
    ptr=head;
    if(ptr==NULL)
    {
            printf("Stack is empty");
    }
    else{
        printf("printing stack");
        while(ptr!=NULL)
        {
            printf("%d\t",ptr->val);
            printf("\n");
            ptr=ptr->next;
        }
    }
 }
 int main()    //driver code
{
    int ch;
    while(1)
    {
    printf("\n1.push,2.pop,3.display,4.exit");
    scanf("%d",&ch);
    switch(ch)
    {
        case 1: push();
                  break;
        case 2: pop();
                  break;
        case 3: display();
                  break;
        case 4: exit(0);
                  break;
        default: printf("INVALID\n");
            break;

    }
}
}
```

Output

```
1.push,2.pop,3.display,4.exit 1
Enter the value3
Item pushed
1.push,2.pop,3.display,4.exit 1
Enter the value3
Item pushed
1.push,2.pop,3.display,4.exit 3
printing stack
3
3

1.push,2.pop,3.display,4.exit 2
Item popped 3

1.push,2.pop,3.display,4.exit 3
printing stack
3

1.push,2.pop,3.display,4.exit
```