THADOMAL SHAHANI
TSEC
ENGINEERING COLLEGE

Assignment 2

Title - Write a program for array implementation of Stack (with structure pointer)

Aim - Perform functions such as push, pop and display on a array using structure pointers in C. ~~Use~~ Implement a Stack.

Theory
A stack follows LIFO principle which means the last element ~~ins~~entering ~~it~~ is the first one to leave the stack.
We use structs and functions to ~~emula~~ use stacks in C

Push - It adds an element to the top of the stack which will be the first one out

a[5] =

0	1	2	3	4
1	4	5		

Push the number 7

top variable has one added to it

a[5] =

0	1	2	3	4
1	4	5	7	

is the result after the push function is performed.
It ~~removes~~ adds the last element to the array, it increases the number of ~~filled~~ used spaces in the array by one.

Pop - It removes the last element from the array and marks that space as unused. It takes the last number added out of the array It reduces the top varible by one.

a[5] :

0	1	2	3	4
1	4	5	7	

Pop remove the last number (Number 7)

a[5] =

0	1	2	3	4
1	4	5		

Result after the pop function is performed

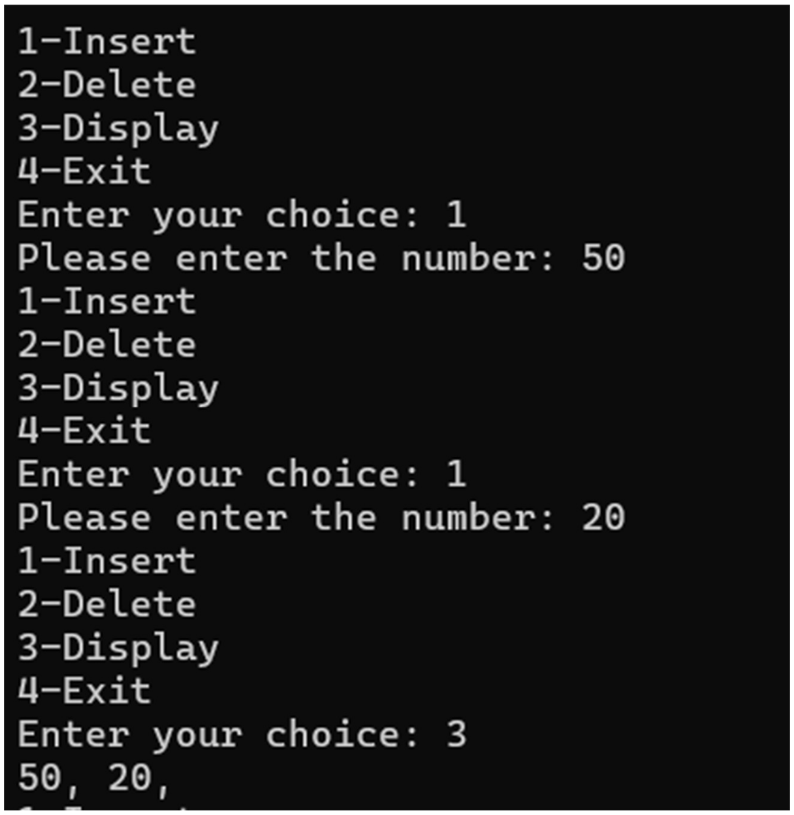
~~Conclusion~~ The display function iterates over the array from the oldest to newest entry. It uses ~~with~~ structure pointers in all the functions.

Conclusion
All fundamental stack operations. push, pop and display were ~~dis~~ performed by passing the structure address to functions.
The stack structure has to be correctly implemented using structure pointers in C.

```c
#include <stdio.h>

# define SIZE 25
struct Stack {
  int top;
  int arr[SIZE];
};

void push(struct Stack *s);
void pop(struct Stack *s);
void display(struct Stack *s);

void main()
{
  struct Stack s;
  s.top = 0;
  int a = 1;
  while (a > 0) {
    int i;
    printf("1-Insert\n2-Delete\n3-Display\n4-Exit\n");
    printf("Enter your choice: ");
    scanf("%d", &i);
    switch (i) {
      case 1:
        push(&s);
        break;
      case 2:
        pop(&s);
        break;
      case 3:
        display(&s);
        break;
      case 4:
        return;
      default:
        printf("Invalid Choice\n");
        break;
    }
  }
}

/***
Adds elements to the top of the stack
***/
void push(struct Stack *s)
{
  int num;
  // Checks if the stack is full
  if (s->top < SIZE)
  {
    printf("Please enter the number: ");
    scanf("%d", &num);
```

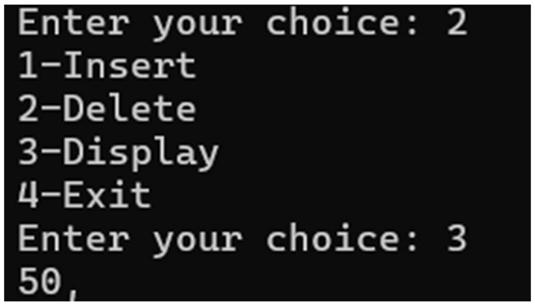
```
        s->arr[s->top] = num;
        s->top++;
    }
    else
    {
        printf("STACK IS FULL\n");
    }
}

/***
Remove the last element from the stack
***/
void pop(struct Stack *s)
{
    //Checks if the stack is not empty
    if (s->top > 0)
    {
        s->top--;
        s->arr[s->top] = 0;
    }
    else
    {
        printf("NO ELEMENTS IN STACK\n");
    }
}
/***
Prints all the elements in the stack
***/
void display(struct Stack *s)
{
    // Checks if the stack is empty
    if (s-> top == 0)
    {
        printf("NO ELEMENTS IN STACK\n");
        return;
    }
    for (int j = 0; j<s->top; j++)
    {
        printf("%d, " , s->arr[j]);
    }
    printf("\n");
}
```

Insert Function



```
1-Insert
2-Delete
3-Display
4-Exit
Enter your choice: 1
Please enter the number: 50
1-Insert
2-Delete
3-Display
4-Exit
Enter your choice: 1
Please enter the number: 20
1-Insert
2-Delete
3-Display
4-Exit
Enter your choice: 3
50, 20,
```

Delete Function



```
Enter your choice: 2
1-Insert
2-Delete
3-Display
4-Exit
Enter your choice: 3
50,
```