

Name: Anujkumar Yadav

Roll No.: 64

Experiment No : 1 To implement stack ADT using arrays

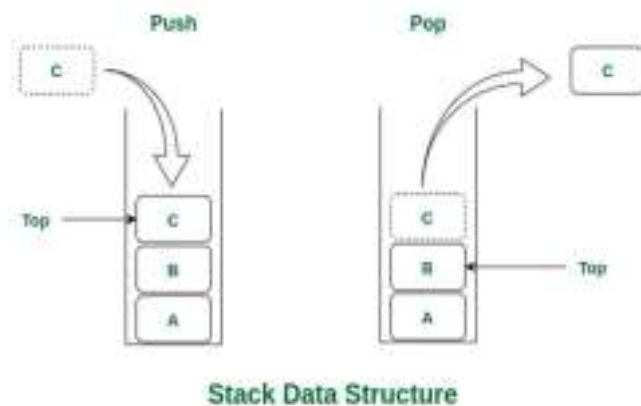
Aim: To implement Stack ADT using arrays

Objective:

- 1) To understand the stack Data Structure & its basic operation
- 2) Understand the method of defining stack ADT & implement the basic operation
- 3) Learn how to create object from an ADT & invoke member function

Theory:

A stack is a linear data structure which uses the LIFO (Last-In-First-Out) principle. The elements in the stack are added and removed only from one end called the top. In the computer's memory, stacks can be represented as a linear array. Every stack has a variable called TOP associated with it, which is used to store the address of the topmost element of the stack. It is in this position where the element will be added to or deleted from. There is another variable called MAX, which is used to store the maximum number of elements that the stack can hold. If $TOP = NULL$, then it indicates that the stack is empty and



If $TOP = MAX-1$, then the stack is full.

A stack supports a few basic operations : push, pop and display.

Push - The push operation is used to insert an element into the stack. The new element is added at the topmost position of the stack. However, before inserting the value, we must first check if $TOP = MAX-1$, because if that is the case, then the stack is full and no more insertions can be done. If an attempt is made to insert a value in a stack that is already full, an OVERFLOW message is printed.

Pop - The pop operation is used to delete the topmost element from the stack. However, before deleting the value, we must first check if TOP=NULL because if that is the case, then it means the stack is

empty and no more deletions can be done. If an attempt is made to delete a value from a stack that is already empty, an UNDERFLOW message is printed.

Display – The display operation is used to display all the elements of the stack. However, the display operation first checks if the stack is empty, i.e., if TOP = -1, then an appropriate message is printed, else the values are returned.

Peek - It is an operation that returns the value of the topmost element of the stack without deleting it from the stack. However, the Peek operation first checks if the stack is empty, i.e., if TOP = -1, then an appropriate message is printed, else the value is returned.

Algorithm:

PUSH(item)

1. If (stack is full)
print "overflow"
 2. top=top+1
 3. stack[top]=item
- Return

POP()

1. if (stack is empty)
print "underflow"
2. Item=stack[top]
3. top=top-1
4. Return item

PEEK()

1. If (stack is empty)
Print "underflow"
2. item=stack[top]
3. top=1
4. Return item

Code:

```
#include<stdio.h>
#include<conio.h>
int stack[100],choice,n,top,x,i;
```

```

void push(void);
void pop(void);
void display(void);
int main()
{
    top=-1;
    printf("\nEnter the size of STACK[MAX=100]:");
    scanf("%d",&n);
    printf("\n\t STACK OPERATIONS USING ARRAY");
    printf("\n\t-----");
    printf("\n\t 1.PUSH\n\t 2.POP\n\t 3.DISPLAY\n\t 4.EXIT");
    do
    {
        printf("\nEnter the choice:");
        scanf("%d",&choice);
        switch(choice)
        {
            case 1:
            {
                push();
                break;
            }
            case 2:
            {
                pop();
                break;
            }
            case 3:
            {
                display();
                break;
            }
            case 4:
            {
                printf("\n\t EXIT POINT ");
                break;
            }
            default:
            {
                printf("Enter valid input, i.e from 1 to 4");
            }
        }
    }
}

```

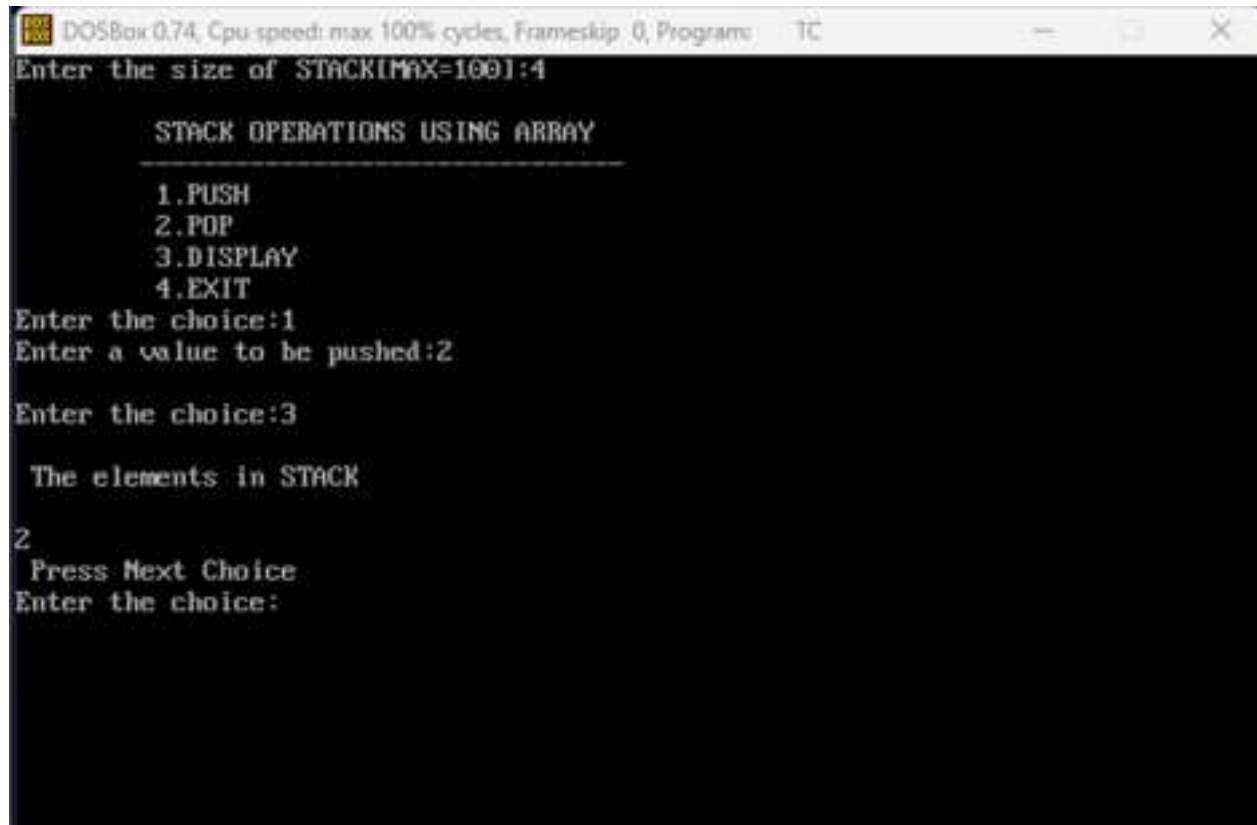
```

while(choice!=4);
{
return 0;
}
}
void push()
{
if(top>=n-1)
{
printf("\n\tSTACK is over flow");
}
else
{
printf("Enter a value to be pushed:");
scanf("%d",&x);
top++;
stack[top]=x;
}
}
void pop()
{
if(top<=-1)
{
printf("\n\tSTACK is under flow");
}
else
{
printf("\n\t The popped elements is %d",stack[top]);
top--;
}
}
void display()
{
if(top>=0)
{
printf("\n The elements in STACK \n");
for(i=top;i>=0;i--)
{
printf("\n%d",stack[i]);
printf("\n Press Next Choice");
}
}
}

```

```
else
{
printf("\n The STACK is empty");
}
}
```

Output:



```
DOSBox 0.74, Cpu speed: max 100% cycles, Frameskip: 0, Program: TC
Enter the size of STACK(MAX=100):4

      STACK OPERATIONS USING ARRAY
      -----
      1.PUSH
      2.POP
      3.DISPLAY
      4.EXIT
Enter the choice:1
Enter a value to be pushed:2

Enter the choice:3

The elements in STACK

2
Press Next Choice
Enter the choice:
```

Conclusion: Hence we've successfully run the stack ADT using array program