Vivekanand Education Society's
Institute of Technology
Approved by AICTE & Affiliated to University of Mumbai

Artificial Intelligence and Data Science Department.

DS / Odd Sem 2021-22 / Experiment 1.

YASH SARANG.
47 / D6AD.

EXPERIMENT - 1.

Aim: Implementation of Stack Data Structure using Array.

Theory:

Stack is a linear data structure that follows a particular order in which the operations are performed. The order may be LIFO (last in first out) or FILO (first in last out).

Mainly the following four operations are performed in the stack :

1. Push: Adds an item in the stack. If the stack is full then it is set to be an overflow condition.

Syntax :

```
void push(int data)
{
        if (top>=n-1)
        {
                printf("Stack is Overfull! overflow");
        }
        else
        {
                top ++;
                stack[top]=data;
        }
}
```

2. Pop: Removes an item from the stack the items are popped in the reverse order in which they are Pushed if the stack is empty then it is set to be an underflow condition.

Syntax:

```
void pop(int data)
{
        if (top<=n-1)
        {
                printf("Stack is empty! underflow");
        }
        else
        {
                printf("The popped elements is %d", Stack[top]);
                top --;
        }
}
```

3. Peek or Top: Return top element of a stack.

Syntax:

```
Int stack_a(int stack[])
{
        int data;
        if (top== -1)
        {
                printf("Stack is Empty");
        }
        else
        {
                return stack [Top];
        }
}
```

4. Is Empty: Returns true if the stack is empty, else false.

Syntax :

```
void isempty(int stack[])
{
        If (top == -1)
        {
                printf ("Stack is empty);
        }
        else
        {
                printf ("Stack is not empty");
        }
}
```

5. Stack full: To check whether the stack is full, Returns true if the stack is full, else false

Syntax :

```
void full(int stake[])
{
        if (top == n-1)
        {
                printf (“Stack is full”);
        }
        else
        {
                printf (“stack is not full”);
        }
}
```

Algorithm:

Step 1: Start.

Step 2: Define Size = 10, top = -1.

Step 3: Define push function.

Step 1: Start

Step 2: Declare Stack[MAX]
//Maximum size of Stack

Step 3: Check if the stack is full or not by comparing top with (MAX-1) If the stack is full, Then print "Stack Overflow" i.e, the stack is full and cannot be pushed with another element

Step 4: Else, the stack is not full Increment top by 1 and Set, a[top] = x which pushes the element x into the address pointed by top.
// The element x is stored in a[top]

Step 5: Stop

Step 4: Define pop function.

Step 1: Start.

Step 2: Declare Stack[MAX].

Step 3: Push the elements into the stack.

Step 4: Check if the stack is empty or not by comparing the top with the base of the array.
i.e 0 If the top is less than 0, then the stack is empty, print "Stack Underflow"

Step 5: Else, If the top is greater than zero the stack is not empty, then store the value pointed by the top in a variable x=a[top] and decrement top by 1.
The popped element is x.

Step 5: Define peek function.

Step 1: Start.
Step 2: Declare Stack[MAX].
Step 3: Push the elements into the stack.
Step 4: Print the value stored in the stack pointed by the top.
Step 5: Stop.

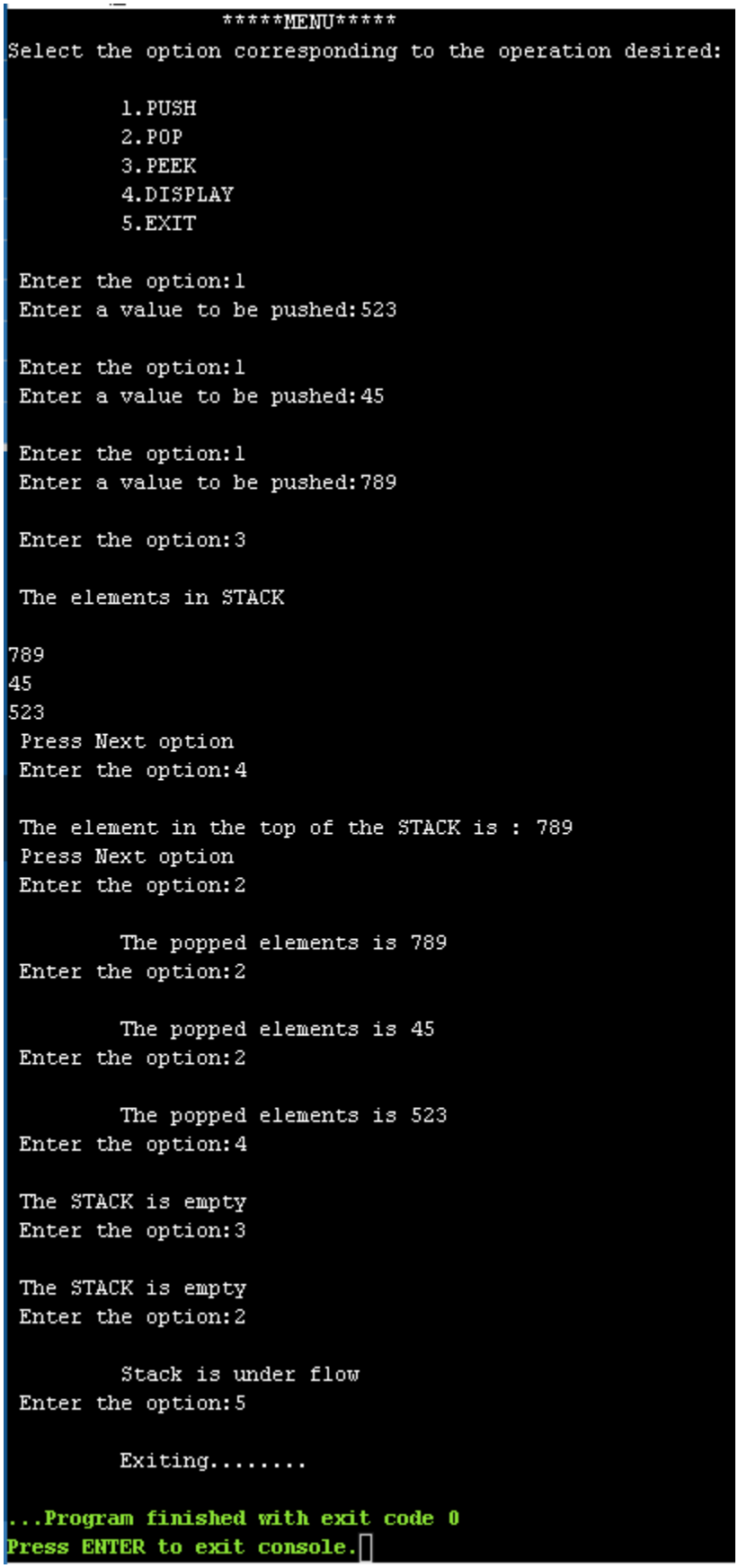
C program:

```
#include<stdio.h>
#define MAX 10

int stack[MAX],option,top=-1,x,i;

void push()
{
   if(top>=MAX-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\t Stack 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 option");
    }
    else
    {
        printf("\n The STACK is empty");
    }
}

void peek()
{
    if(top>=0)
    {
        printf("\n The element in the top of the STACK is : ");
        printf("%d",stack[top]);
        printf("\n Press Next option");
    }
    else
    {
        printf("\n The STACK is empty");
    }
}

int main()
{
    //clrscr();
    printf("\n\t\t *****MENU*****\t\t\nSelect the option corresponding to the operation desired: \n");
    printf("\n\t 1.PUSH \n\t 2.POP\n\t 3.DISPLAY\n\t 4.PEEK\n\t 5.EXIT\n");
    do
    {
        printf("\n Enter the option:");
        scanf("%d",&option);
        switch(option)
        {
```

```
        case 1:
        {
           push();
           break;
        }
        case 2:
        {
           pop();
           break;
        }
        case 3:
        {
           display();
           break;
        }
        case 4:
        {
           peek();
           break;
        }
        case 5:
        {
           printf("\n\t Exiting.....");
           break;
        }
        default:
        {
           printf("\n\t Please Enter a Valid option(1/2/3/4/5)");
        }
     }
  }
  while(option!=5);
  return 0;
}
```

Output:


*****MENU*****
Select the option corresponding to the operation desired:
1.PUSH
2.POP
3.PEEK
4.DISPLAY
5.EXIT
Enter the option:1
Enter a value to be pushed:523
Enter the option:1
Enter a value to be pushed:45
Enter the option:1
Enter a value to be pushed:789
Enter the option:3
The elements in STACK
789
45
523
Press Next option
Enter the option:4
The element in the top of the STACK is : 789
Press Next option
Enter the option:2
The popped elements is 789
Enter the option:2
The popped elements is 45
Enter the option:2
The popped elements is 523
Enter the option:4
The STACK is empty
Enter the option:3
The STACK is empty
Enter the option:2
Stack is under flow
Enter the option:5
Exiting.....
...Program finished with exit code 0
Press ENTER to exit console.

****