Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

# DSA - Experiment 2

**Name:** Divyesh Khunt **Sapid:**60009210116 **Batch: A/3**

**Aim:** To create a stack in c programming and learn its abstract data Types.

**Theory:**

Stack is a linear data structure which follows a order (LIFO Last In First Out) or FILO (First In Last Out) in which operations are performed.

There are 4 operations in stacks push, pop, peek, display

1. PUSH
   When we insert an element in a stack then the operation is known as a push. The next element added is added above the previous one.
2. POP
   When we delete an element from the stack, the operation is known as a pop. Here the topmost element is deleted.
3. PEEK
   It returns the element at the topmost position.
4. DISPLAY
   It prints all the elements available in the stack.

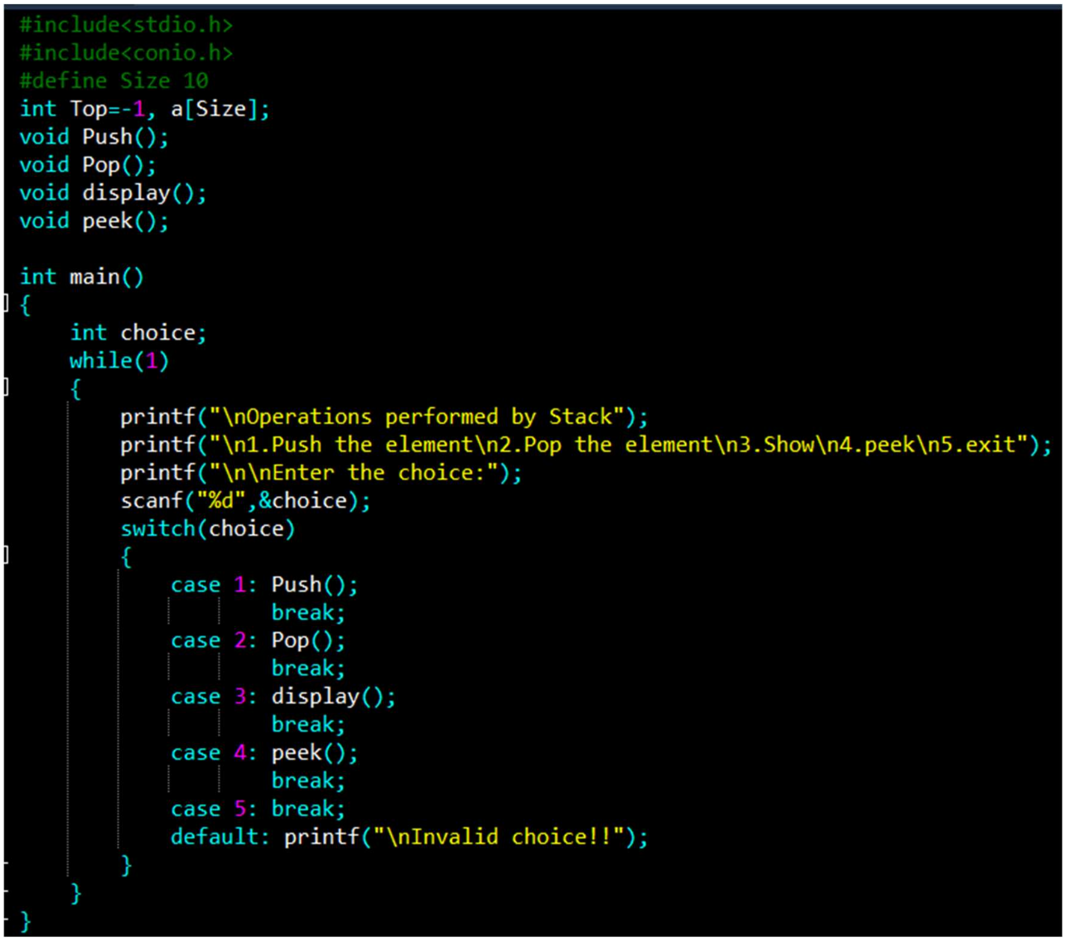
**OVERFLOW:** If the stack is full then the overflow condition occurs.
**UNDERFLOW:**If the stack is empty means that no element exists in the stack, this state is known as an underflow state.

## Time Complexity of Stack Operations

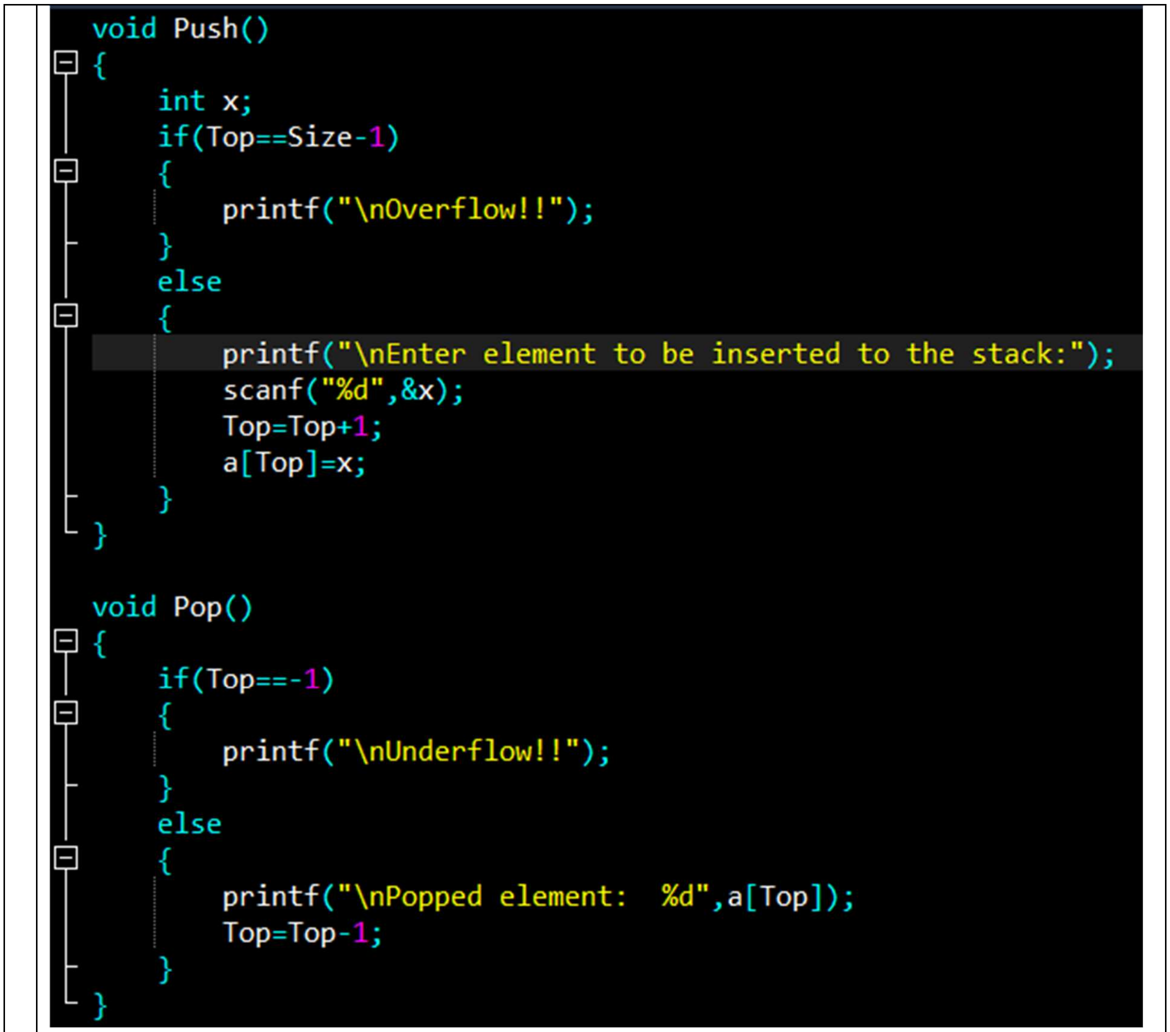
As mentioned above, only a single element can be accessed at a time in Stacks.

While performing push () and pop() operations on the stack, it takes **O(1)** time.

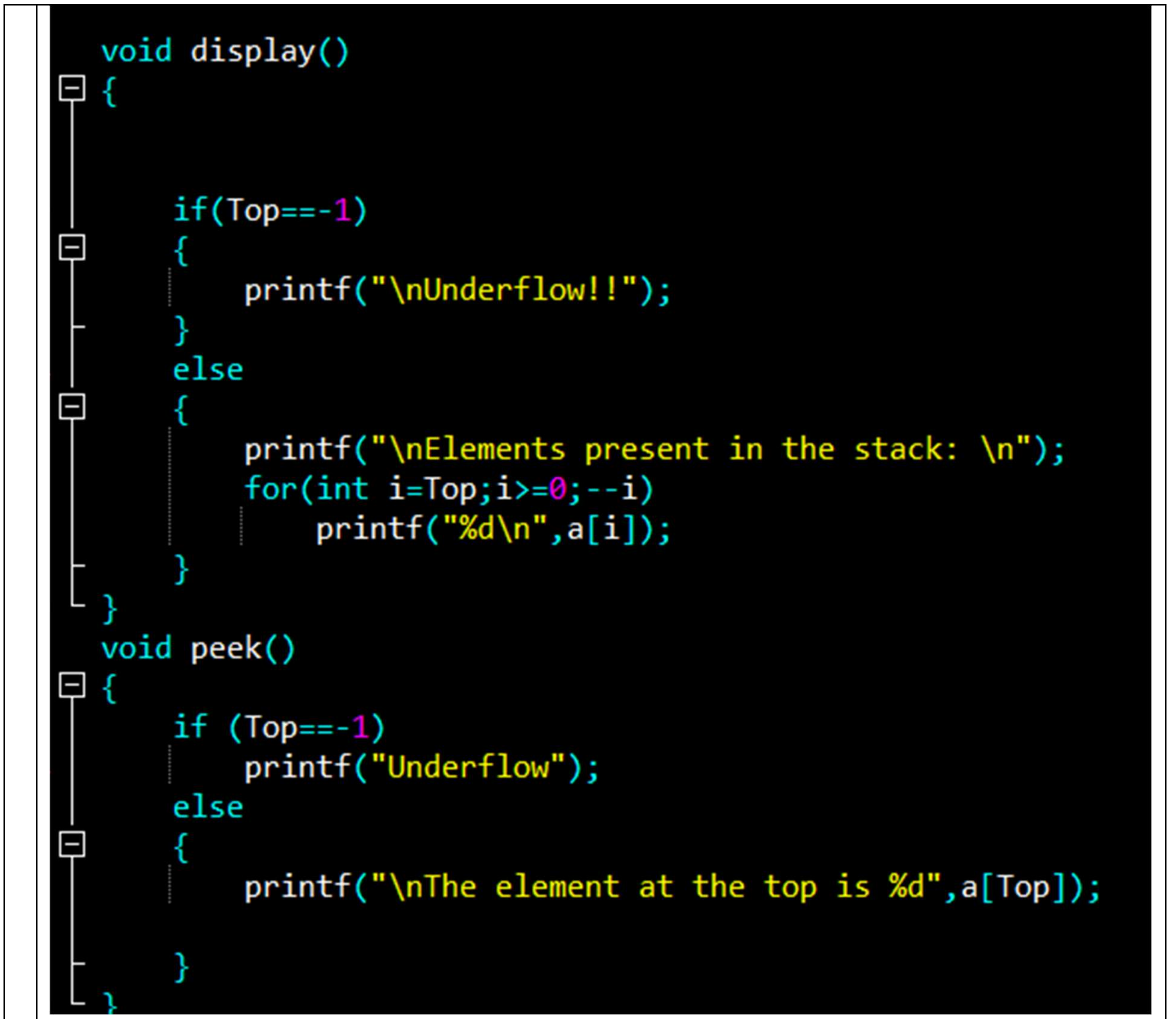
## CODE:



```
#include<stdio.h>
#include<conio.h>
#define Size 10
int Top=-1, a[Size];
void Push();
void Pop();
void display();
void peek();

int main()
{
    int choice;
    while(1)
    {
        printf("\nOperations performed by Stack");
        printf("\n1.Push the element\n2.Pop the element\n3.Show\n4.peek\n5.exit");
        printf("\n\nEnter the choice:");
        scanf("%d",&choice);
        switch(choice)
        {
            case 1: Push();
                    break;
            case 2: Pop();
                    break;
            case 3: display();
                    break;
            case 4: peek();
                    break;
            case 5: break;
            default: printf("\nInvalid choice!!");
        }
    }
}
```



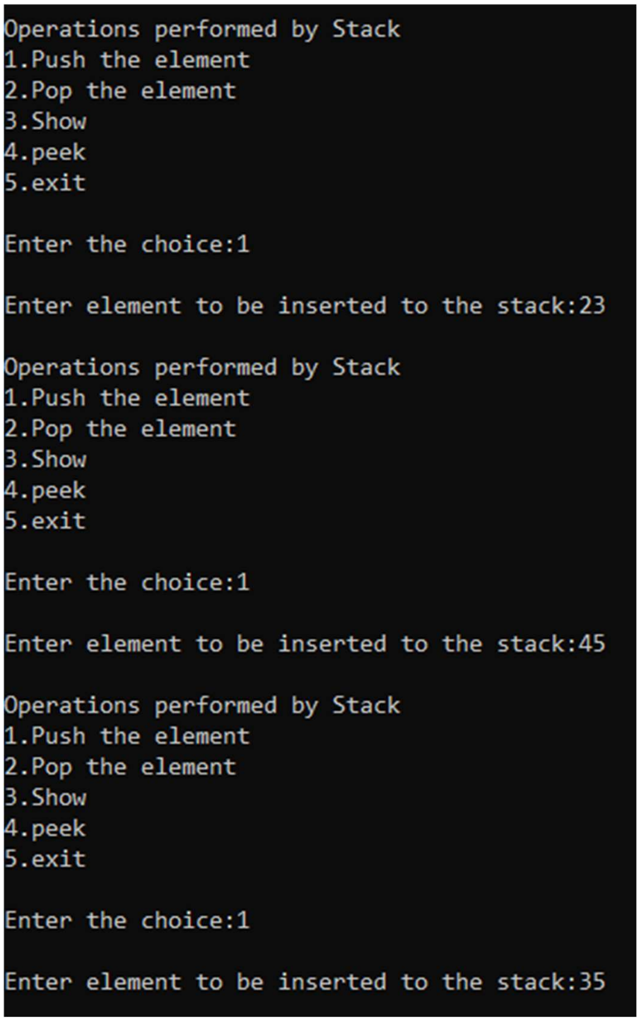
```
void Push()
{
    int x;
    if(Top==Size-1)
    {
        printf("\nOverflow!!");
    }
    else
    {
        printf("\nEnter element to be inserted to the stack:");
        scanf("%d",&x);
        Top=Top+1;
        a[Top]=x;
    }
}

void Pop()
{
    if(Top==-1)
    {
        printf("\nUnderflow!!");
    }
    else
    {
        printf("\nPopped element:  %d",a[Top]);
        Top=Top-1;
    }
}
```



```
void display()
{


    if(Top==-1)
    {
        printf("\nUnderflow!!");
    }
    else
    {
        printf("\nElements present in the stack: \n");
        for(int i=Top;i>=0;--i)
            printf("%d\n",a[i]);
    }
}
void peek()
{
    if (Top==-1)
        printf("Underflow");
    else
    {
        printf("\nThe element at the top is %d",a[Top]);

    }
}
```

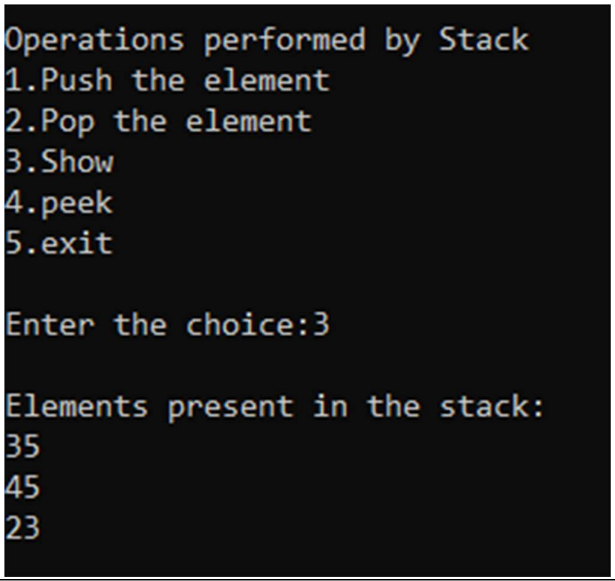
## OUTPUTS:

### PUSH



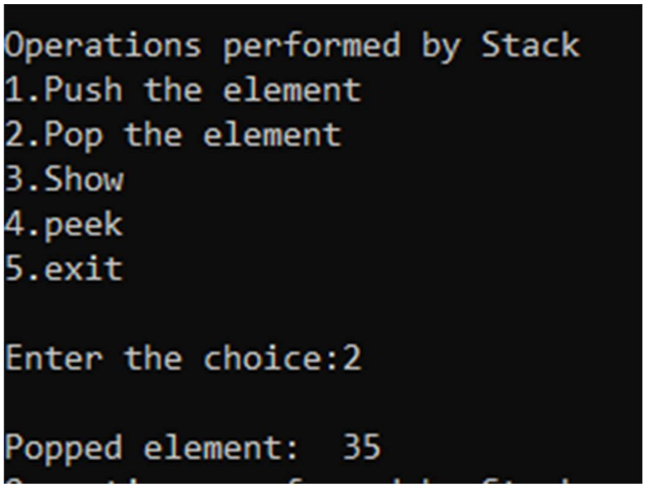
```
Operations performed by Stack
1.Push the element
2.Pop the element
3.Show
4.peek
5.exit

Enter the choice:1

Enter element to be inserted to the stack:23

Operations performed by Stack
1.Push the element
2.Pop the element
3.Show
4.peek
5.exit

Enter the choice:1

Enter element to be inserted to the stack:45

Operations performed by Stack
1.Push the element
2.Pop the element
3.Show
4.peek
5.exit

Enter the choice:1

Enter element to be inserted to the stack:35
```

### DISPLAY



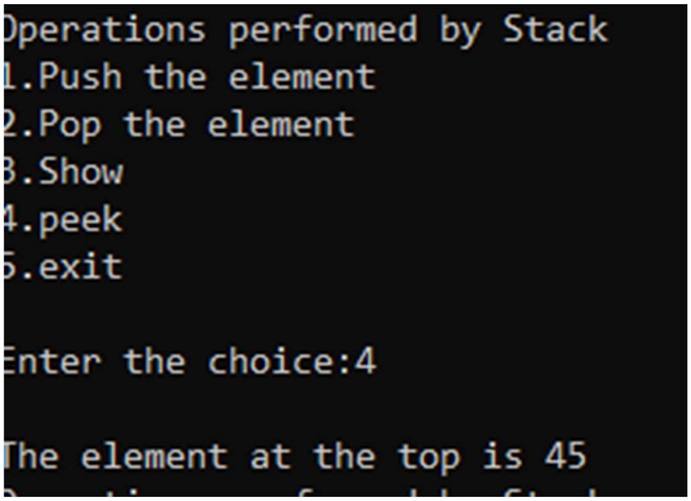
```
Operations performed by Stack
1.Push the element
2.Pop the element
3.Show
4.peek
5.exit

Enter the choice:3

Elements present in the stack:
35
45
23
```

**POP**



```
Operations performed by Stack
1.Push the element
2.Pop the element
3.Show
4.peek
5.exit

Enter the choice:2

Popped element:  35
```

**PEEK**



```
Operations performed by Stack
1.Push the element
2.Pop the element
3.Show
4.peek
5.exit

Enter the choice:4

The element at the top is 45
```

**APPLICATIONS OF STACK:**

1. REVERSING
2. PARENTHESIS CHECKING
3. PREFIX/INFIX CONVERSIONS
4. BACKTRACKING

**CONCLUSION:**

Thus, in this article, we have understood the concept of Stack data structure and its implementation using Arrays in C.