E-Content Module

Subject	: Data Structure Using 'C'
Topic	: Stack operation
Target Learners	: BCA/ MCA
Duration	: 25-30 min

About Author:

Name	: Devi Dayal Sinha
Email-id	: devidayal_ap@impactcollege.in

Stack

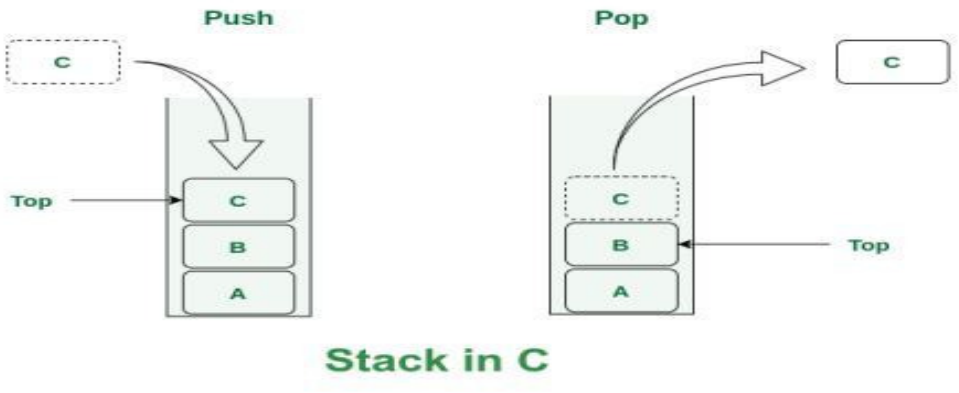
Definition

A **stack in C** is a linear data structure that follows the **LAST IN FIRST OUT (LIFO)** principle, much like a stack of BOOKS. Elements are added and removed only from one end, called the 'TOS' or Top of Stack.

In C, a stack is a manually implemented conceptual data type, basically uses an array having a fixed size list or stack and or a **linked list** for a dynamic allocation using pointers.

Core Operations

Stack operations generally have a constant time complexity, O (1). Key operations include push() to add an element, pop() to remove one, and peek() to view the top element without removing it. Other operations check if the stack is empty (isEmpty()) or full (isFull()).



Stack in C

Implementation in C Using an Array

An array can implement a fixed-size stack using an array and a tos index.
Majorly, push (), pop () are the two operations implemented and these operations are based on accessing the topmost only element in the list. Additionally, peek (), peep () and update () can also be implemented using a little variation in the code segment. Some other

Common Applications

Stacks are used in various areas of computer science, such as managing function calls, implementing undo/redo features, evaluating expressions, and in browser history navigation.

- Stack Data Structure and Implementation in Python, Java and ...
- Stack Push and Pop Program in 'C'

⇨ PROGRAM FOR STACK USING ARRAY WITH ALL POSSIBLE OPTIONS LIKE INSERTION/ DELETION / DISPLAY ETC.

```
#include<stdio.h>
#include<conio.h>
#include<ctype.h>
#include<stdlib.h>
int size=10;
int tos=-1;
int S[10];
void push()
{
        if (tos==-1)
        {
                printf("\n Stack doesn't exist");
                tos=0;
                printf("\n Enter First Value");
                scanf("%d",&S[tos]);
        }
        else if (tos>=0 && tos<size-1)
        {
                ++tos;
                printf("\n Enter next value");
                scanf("%d",&S[tos]);
        }
        else
```

```
                printf("\n Stack Overflow");
}
void pop()
{
        if(tos==-1)
                printf("\n Stack underflow");
        else
        {
                printf("\n Deleted value is %d",S[tos]);
                tos--;
        }
}
void peep()
{
        int p,val;
        printf("|n Enter position to peep");
        scanf("%d",&p);
        if(p<0 ||p>tos)
                printf("\n Invalid position: Try again()");
        else
        {
                val=S[tos-p+1];
                printf("\n Peeped value is %d",val);
        }
}
void update()
{
        int p;
        printf("\n Enter position to update");
        scanf("%d",&p);
        if(p<0 ||p>tos)
                printf("\n Invalid position");
        else
        {
                printf("\n Current Position Value is %d",S[tos-p+1]);
                printf("\n Enter new value");
```

```
                scanf("%d",&S[tos-p+1]);
                printf("\n Updated list:");
        }
}
void display()
{
        int i;
        if(tos==-1)
                printf("\n Nothing to display: No such STACK exists");
        else
        {
                for(i=tos;i>=0;i--)
                        printf("\n %d",S[i]);
        }
}

void main()
{
        int ch;
        char choice='Y';
        clrscr();
        printf("1. Push");
        printf("\n2. Pop");
        printf("\n3. Peep");
        printf("\n4. Update");
        printf("\n5. Display");
        printf("\n6. Exit");
        while(choice=='Y')
        {
                printf("\n Enter your choice(1/2/3/4/5/6)");
                scanf("%d",&ch);
                switch(ch)
                {
                        case 1: push();
                                display();
                                break;
```

```
                case 2: pop();
                        display();
                        break;
                case 3: peep();
                        break;
                case 4: update();
                        display();
                        break;
                case 5: display();
                        break;
                case 6: exit(1);
                default: printf("\n Invalid choice: Try Again");
            }
            printf("\n Enter your choice to continue(Y/N)");
            choice=toupper(getche());
        }
        getch();
}
```