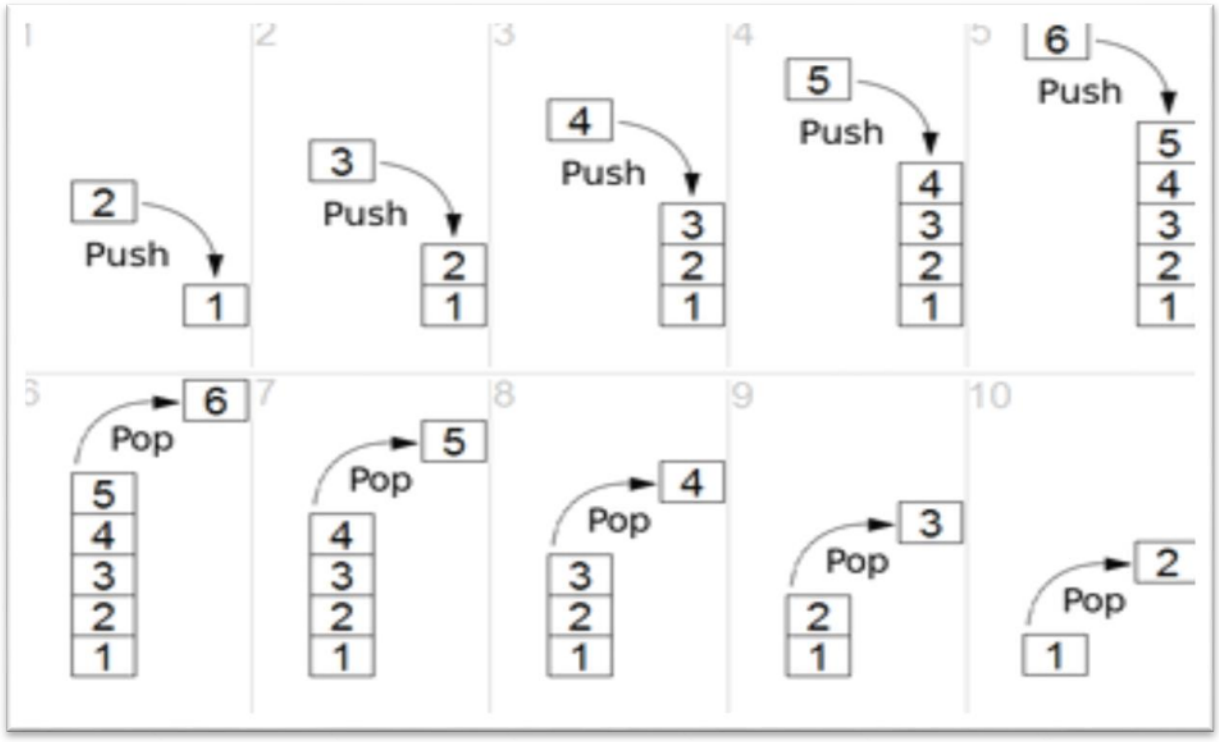
Stack

What is a Stack?

→ A stack is a linear data structure that works on the principle of Last In First out (LIFO).

Ex: - A stack of books, the last book which you kept on the top would be the first book which you pick up.

- To add an element on top is called Push.
- Removing an element from top is called Pop.
- To check if the stack is empty or not IsEmpty.
- To check if the stack is full or not IsFull.



How does it work?

- We use a pointer called Top which keeps track of the topmost element.
- On Pushing an new element we increment the value of Top by 1.

- On Popping out an element we decrease the value by 1.
- Top = -1 indicates that the stack is empty.

Implementation:

```c
#include<stdio.h>
#include<stdlib.h>
//Creating The stack
struct Stack
{
    int *a;
    int top;
    int size;
};
typedef struct Stack STACK;
STACK createStack(int n)
{
    STACK stk;
    stk.a = (int*) malloc(sizeof(int)*n);
    stk.top = -1;
    stk.size = n;
    return stk;
}
int isEmpty(STACK *stk)//to check if stack is empty by comparing Top =-1
{
    return stk->top == -1;
}
int isFull(STACK *stk)
{
    return stk->size -1 == stk->top;
}
//pushing data into stack
void push(STACK *stk,int val)
{
    if(isFull(stk))
    {
        printf("\nStack Overflow!");
        return;
    }
    stk->a[++(stk->top)] = val;
}
//popping data out
void pop(STACK *stk)
{
    if(isEmpty(stk))
    {
        printf("\nStack Underflow!");
        return;
```

```
    }
    printf("\nPopped element : %d ",stk->a[(stk->top)--]);
}
void display(STACK *stk)
{
    if(isEmpty(stk))
    {
        printf("\nStack is empty.");
        return;
    }
    int i;
    printf("\nContents of stack :\n");
    for(i = stk->top; i >= 0; i--)
    {
        printf("\n%d",stk->a[i]);
    }
    printf("\n");
}
int main()
{

    int d, ch,n ;
    STACK stk;
    printf("\nEnter your required size : ");
    scanf("%d", &n);
    stk = createStack(n);
    while(1)
    {
        printf("\n1. Push");
        printf("\n2. Pop");
        printf("\n3. Display");
        printf("\n0. Exit");
        printf("\nEnter your choice : ");
        scanf("%d", &ch);
        switch(ch)
        {
        case 1:
            printf("\nEnter your data to push into stack : ");
            scanf("%d", &d);
            push(&stk, d);
            break;
        case 2:
            pop(&stk);
            break;
        case 3:
            display(&stk);
            break;
        case 0:
```

```
            return 0;
        default:
            printf("\nWrong option selected!");
        }
    }
}
```

Application:

- We can reverse a word by simply pushing it in a stack and popping it out.
- The back button in a browser stores all the URLs you have previously visited in a stack. Every time you visit a new page, it is added to the top of the stack. Pressing the Back button removes the current URL from the stack and accesses the previous URL.