Stack

एक स्टैक आइटम की प्रविष्टि और विलोपन के ढेर के शीर्ष नामक एक अंत में जगह लेता है, जहां एक बुनियादी डेटा संरचना, है. यह संरचना प्रोग्रामिंग के दौरान प्रयोग किया जाता है. बुनियादी अवधारणा प्लेटों या आप इसे से चीजों को दूर करने के क्रम में शीर्ष मद ढेर से दूर ही ले जा सकते हैं जहां किताबों के ढेर के रूप में अपने डेटा की सोच से यह साफ हो सकता है.

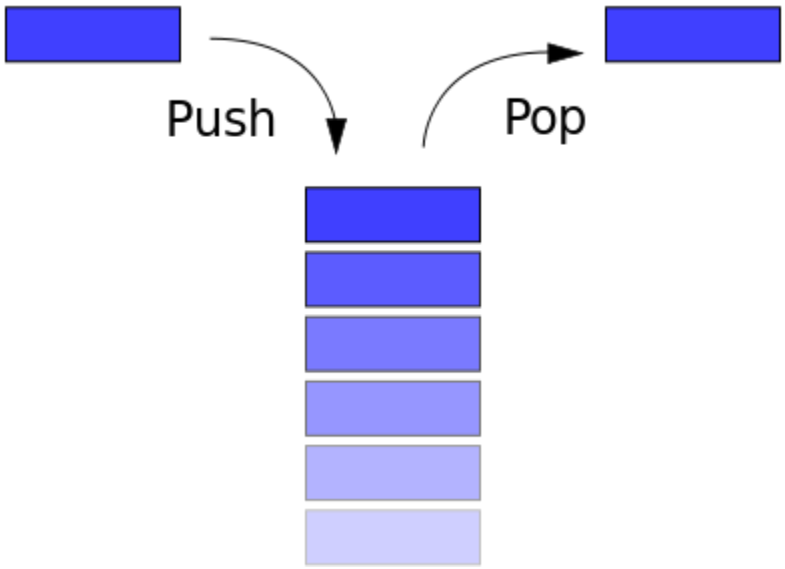
एक स्टैक भी यह डेटा तक जिस तरह का प्रदर्शन करने के लिए एक LIFO (प्रथम आउट में अंतिम) कहा जाता है. ढेर पर प्रदर्शन किया जा सकता है कि तीन आपरेशनों मूल रूप से कर रहे हैं. वे कर रहे हैं

1) एक स्टैक (PUSH) में एक आइटम डालने.

2) स्टैक (POP) से किसी आइटम को हटाने.

3) स्टैक (PEEK) की सामग्री प्रदर्शित.

\



CODES (C)

```
#include<stdio.h>
#include<conio.h>
#include<alloc.h>
#include<string.h>
typedef struct stack
{
        int *a;
        int tos,ms;
}st;

void init(st *s,int n)
{
        s->ms=n;
        s->a=(int *)malloc(s->ms*sizeof(int));
        s->tos=-1;
}

int isfull(st *s)
{
        if(s->tos==s->ms-1)
                return 1;
        else
                return 0;
}

int isempty(st *s)
{
        if(s->tos==-1)
                return 1;
        else
                return 0;
}

void push(st *s,int z)
{
        if(isfull(s)==1)
                printf("Stack is full");
        else
        {
                s->a[++s->tos]=z;
        }
}
```

```
int pop(st *s)
{
	int a=0;
	if(isempty(s)==1)
		printf("Stack is empty");
	else
	{
		a=s->a[s->tos];
		s->tos--;
	}
	return a;
}
char peek(st *s)
{
	char p;
	if(isempty(s)==1)
		printf("Stack is empty");
	else
	{
		p=s->a[s->tos];
	}
	return p;
}
```

CODES (JAVA)

```
public class Stack
{
   private int max;
   private long[] stackArray;
   private int top;
   public MyStack(int s)
   {
      max = s;
      stackArray = new long[max];
      top = -1;
   }
   public void push(long j)
   {
        if(isFull())
                System.out.println(“Stack overflow”);
        else
                stackArray[++top] = j;
   }
   public long pop() {
      if(isEmpty())
        {
                System.out.println(“Stack underflow”);
                return -999;
        }
        else
                return stackArray[top--] = j;

   }
   public long peek() {
      return stackArray[top];
   }
   public boolean isEmpty() {
      return (top == -1);
   }
   public boolean isFull() {
      return (top == max - 1);
   }
}
```