**Stack Operations:**

#include <iostream>

#include <stack>

using namespace std;

int main()

{

stack<int> s;

s.push(10);

s.push(20);

s.push(30);

cout<<s.size()<<endl;

cout<<s.top()<<endl;

s.pop();

cout<<s.top()<<endl;

s.push(5);

cout<<s.top()<<endl;

while(s.empty()==false){

cout<<s.top()<<endl;

s.pop();

}

return 0;

}

Stack representation in form of array:

#include <bits/stdc++.h>

using namespace std;

#define MAX 1000

struct MyStack{

int \*arr;

int cap;

int top;

MyStack(int c){

cap=c;

arr=new int [MAX];

top=-1;

}

void push(int x){

if(top==cap-1){cout<<"Stack is full"<<endl;return;}

top++;

arr[top]=x;

}

int pop(){

if(top==-1){cout<<"Stack is Empty"<<endl;return INT\_MIN;}

int res=arr[top];

top--;

return res;

}

int peek(){

if(top==-1){cout<<"Stack is Empty"<<endl;return INT\_MIN;}

return arr[top];

}

int size(){

return (top+1);

}

bool isEmpty(){

return top==-1;

}

};

int main()

{

MyStack s(MAX);

s.push(5);

s.push(10);

s.push(20);

cout<<s.pop()<<endl;

cout<<s.size()<<endl;

cout<<s.peek()<<endl;

cout<<s.isEmpty()<<endl;

return 0;

}

Stack representation in form of array using class and Public:

/\* C++ program to implement basic stack

operations \*/

#include <bits/stdc++.h>

using namespace std;

#define MAX 1000

class Stack {

int top;

public:

int a[MAX]; // Maximum size of Stack

Stack() { top = -1; }

bool push(int x);

int pop();

int peek();

bool isEmpty();

};

bool Stack::push(int x)

{

if (top >= (MAX - 1)) {

cout << "Stack Overflow";

return false;

}

else {

a[++top] = x;

cout << x << " pushed into stack\n";

return true;

}

}

int Stack::pop()

{

if (top < 0) {

cout << "Stack Underflow";

return 0;

}

else {

int x = a[top--];

return x;

}

}

int Stack::peek()

{

if (top < 0) {

cout << "Stack is Empty";

return 0;

}

else {

int x = a[top];

return x;

}

}

bool Stack::isEmpty()

{

return (top < 0);

}

// Driver program to test above functions

int main()

{

class Stack s;

s.push(10);

s.push(20);

s.push(30);

cout << s.pop() << " Popped from stack\n";

//print all elements in stack :

cout<<"Elements present in stack : ";

while(!s.isEmpty())

{

// print top element in stack

cout<<s.peek()<<" ";

// remove top element in stack

s.pop();

}

return 0;

}

Stack representation in form of Linked List:

#include <bits/stdc++.h>

using namespace std;

struct Node{

int data;

Node \*next;

Node(int x){

data=x;

next=NULL;

}

};

struct MyStack{

Node \*head;

int sz;

MyStack(){

head=NULL;

sz=0;

}

void push(int x){

Node \*temp=new Node(x);

temp->next=head;

head=temp;

sz++;

}

int pop(){

if(head==NULL){cout<<"Stack is Empty"<<endl;return INT\_MAX;}

int res=head->data;

Node \*temp=head;

head=head->next;

delete(temp);

sz--;

return res;

}

int peek(){

if(head==NULL){cout<<"Stack is Empty"<<endl;return INT\_MAX;}

return head->data;

}

int size(){

return sz;

}

bool isEmpty(){

return head==NULL;

}

};

int main()

{

MyStack s;

s.push(5);

s.push(10);

s.push(20);

cout<<s.pop()<<endl;

cout<<s.size()<<endl;

cout<<s.peek()<<endl;

cout<<s.isEmpty()<<endl;

return 0;

}