Stack Implementation in C++

A Stack is a data structure which works on the principle LIFO(Last In, First Out).The basic operations in a Stack are :

1. Push : In which we push the data into the stack.
2. Pop : In which we remove an element from the Stack.

All insertions and removals are done only from one side of the Stack , which is called the 'top' of the Stack.

A stack is generally used in function calls where the local variables are pushed onto the Stack and when the function returns , it pops the variables from the Stack.

This is a simple implementation of Stack in C++.

```
#include <iostream>

using namespace std;

class Stack
{
private:
    int *p;
    int top,length;

public:
    Stack(int = 0);
    ~Stack();

    void push(int);
    int pop();
    void display();
};

Stack::Stack(int size)
{
    top=-1;
    length=size;
    if(size == 0)
        p = 0;
    else
        p=new int[length];
}

Stack::~Stack()
{
    if(p!=0)
        delete [] p;
}
```

```
void Stack::push(int elem)
{
    if(p == 0)               //If the stack size is zero, allow user to mention it at runtime
    {
        cout<<"Stack of zero size"<<endl;
        cout<<"Enter a size for stack : ";
        cin >> length;
        p=new int[length];
    }
    if(top==(length-1))     //If the top reaches to the maximum stack size
    {
        cout<<"\nCannot push "<<elem<<", Stack full"<<endl;
        return;
    }
    else
    {
        top++;
        p[top]=elem;
    }
}
int Stack::pop()
{
    if(p==0 || top==-1)
    {
        cout<<"Stack empty!";
        return -1;
    }
    int ret=p[top];
    top--;
    return ret;
}

void Stack::display()
{
    for(int i = 0; i <= top; i++)
        cout<<p[i]<<" ";
    cout<<endl;
}

int main()
{
    Stack s1;             //We are creating a stack of size 'zero'
    s1.push(1);
    s1.display();
    s1.push(2);
    s1.push(3);
    s1.push(4);
    s1.push(5);
    s1.display();
    s1.pop();
    s1.display();
```

```
    s1.pop();
    s1.display();
    s1.pop();
    s1.display();
    s1.pop();
    s1.display();
}
```

Posted 22nd July 2010 by Varun Gupta

Labels: Data Structures

14 View comments

Vikas Taneja June 25, 2012 at 1:19 PM

Good and simple implementation! :)
You can make use of templates to make the stack class as a generic one. Small improvement though. :)

Reply

zem LoveJC May 28, 2013 at 10:35 PM

i would like to know whether the method display() does conform to the basic format or basic structure of stacks, does it print in reverse of the order of input?

Reply

Varun Gupta May 28, 2013 at 10:43 PM

It prints in the bottom-top order. Its not in the conventional manner.

Reply

Replies

firewolf June 18, 2016 at 2:03 AM

just change for loop set int i = top; i >0; i--

Reply

Allison Martin October 16, 2015 at 11:15 AM

Thank you so so much. This helped me immensely.

Reply

sunil October 16, 2015 at 2:27 PM