ASSIGNMENT NO:7 **DATE:19/9/2015**

PROGRAM TITLE:A class 'Stack' has the normal push and pop functions without provision for dealing with the Overflow and Underflow conditions. Design a class MyStack, which deals with this using Exception Handling.

THEORY: An Exception is an event which occurs during the execution of the program and disrupts the normal execution of the program.

PROGRAM ALGORITHM:

```
//'top' points to the top of the stack. 'max' is the maximum size of the
array 'stk', on which we implement the stack.
Algo_push(item)
{
      if(top=max)
            print "overflow";
      else
      {
            top=top+1;
            stk[top]=item;
      }
}

Algo_pop(top)
{
      if (top=0)
            print "underflow";
      else
      {
            print that the item popped is q[top];
            top=top-1;
      }

}
```

PROGRAM CODE:

```
/*C++ Program to implement classes Stack and MyStack using Excpetion
Handling.*/
#include<iostream>
using namespace std;

/*Class Stack and its associated functions*/
class Stack
{
      int *stk,max,top;
      public:
            void push(int);
            int pop();
};

/*Function to implement pushing an element into the Stack*/
```

```
void Stack::push(int x)
{
      stk[++top]=x;
}

/*Function to implement popping an element from Stack*/
int Stack::pop()
{
      cout<<"\n\tItem popped is "<<stk[top--]<<endl;
      return 0;
}

/*Class MyStack and its associated functions*/
class MyStack:public Stack
{
      int *stk,max,top;
      public:
            MyStack(int);
            void push(int);
            int pop();
};

/*Parameterised constructor of class MyStack*/
MyStack::MyStack(int n)
{
      stk=new int[n];
      max=n;
      top=-1;
}

/*Overriding Function to implement pushing an element into the Stack*/
void MyStack::push(int x)
{
      try
      {
            if(top+1==max)
                  throw "Overflow";
            else
            {
                  stk[++top]=x;
            }
      }
      catch(const char* s)
      {
            cout<<"\n\tOverflow"<<endl;
      }
}

/*Overriding Function to implement popping an element from Stack*/
int MyStack::pop()
{
      try
      {
            if(top==-1)
                  throw "Underflow";
```

```
            else
            {
                  cout<<"\n\tItem popped is "<<stk[top--]<<endl;
            }
      }
      catch(const char *s)
      {
            cout<<"\n\tUnderflow"<<endl;
      }
      return 0;
}

int main()
{
      int ch=0,n;
      cout<<"\n\tEnter size of Stack:";
      cin>>n;

      /*Creating an object of class MyStack*/
      MyStack stk(n);
      do
      {

            /*Menu for implementing Stack functions*/
            cout<<"\n\tMenu for
Stack::\n\t1.Push\n\t2.Pop\n\t3.Exit\n\tYour Choice::";
            cin>>ch;
            switch(ch)
            {
                  case 1: cout<<"\n\tEnter the item to be pushed:";
                        cin>>n;
                        stk.push(n);
                        break;
                  case 2: stk.pop();
                        break;
                  case 3: cout<<"\n\tProgram Terminated.\n";
                        break;
                  default:cout<<"\n\tInvalid Choice.";
            }
      }
      while(ch!=3);
      return 0;
}
```

OUTPUT:

```
      Enter size of Stack:5

      Menu for Stack::
      1.Push
      2.Pop
      3.Exit
      Your Choice::1
```

```
Enter the item to be pushed:10

Menu for Stack::
1.Push
2.Pop
3.Exit
Your Choice::2

Item popped is 10

Menu for Stack::
1.Push
2.Pop
3.Exit
Your Choice::2

Underflow

Menu for Stack::
1.Push
2.Pop
3.Exit
Your Choice::1

Enter the item to be pushed:20

Menu for Stack::
1.Push
2.Pop
3.Exit
Your Choice::1

Enter the item to be pushed:30

Menu for Stack::
1.Push
2.Pop
3.Exit
Your Choice::1

Enter the item to be pushed:40

Menu for Stack::
1.Push
2.Pop
3.Exit
Your Choice::1

Enter the item to be pushed:50

Menu for Stack::
1.Push
2.Pop
3.Exit
Your Choice::1
```

```
Enter the item to be pushed:60

Menu for Stack::
1.Push
2.Pop
3.Exit
Your Choice::1

Enter the item to be pushed:70

Overflow

Menu for Stack::
1.Push
2.Pop
3.Exit
Your Choice::2

Item popped is 60

Menu for Stack::
1.Push
2.Pop
3.Exit
Your Choice::2

Item popped is 50

Menu for Stack::
1.Push
2.Pop
3.Exit
Your Choice::2

Item popped is 40

Menu for Stack::
1.Push
2.Pop
3.Exit
Your Choice::2

Item popped is 30

Menu for Stack::
1.Push
2.Pop
3.Exit
Your Choice::2

Item popped is 20

Menu for Stack::
1.Push
```

```
2.Pop
3.Exit
Your Choice::2

Underflow

Menu for Stack::
1.Push
2.Pop
3.Exit
Your Choice::3

Program Terminated.
```

DISCUSSION:

Stack:

1.The complexity of push is O(1).
2.The complexity of pop is O(1).
3.Overflow occurs only when the top reaches the maximum size of the array.

Here, the whole code is similar to the implementation of a Stack with the exception that we are using the technique of Exception Handling to deal with the Overfolw and Underflow conditions in the class MyStack.