Assignment 1(A)

Title:- Design a base class STACK and Handle runtime anomalies like Overflow when the stack is full and underflow when the stack is empty. Display error codes and messages by using appropriate try and catch blocks to handle the exceptions thrown.

Code:

package javaapplication1;

import java.util.\*;

import java.util.Stack;

/\*\*

\*

\* @author Lenovo

\*/

public class JavaApplication1 {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

// TODO code application logic here

Stack st = new Stack();

try{

System.out.println("Enter a Decimal number : ");

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

int m = n;

int r = 0;

// Binary form of a decimal no. is obtained by dividing the no. by 2 and considering the remainder of operation .

while(n > 0)

{

r = n % 2;

n = n/2;

st.push(r);

}

System.out.print("\nThe Binary form of " + m + " is :- " );

while(!st.empty())

{

System.out.print(st.pop());

}

System.out.println("\n");

} // Handling exception - Only

catch (InputMismatchException e)

{

System.out.println("\nWrong input ! \nEnter valid integer input !..\n");

}

/\*try {

System.out.println("Element removed from stack is : " + st.pop());

System.out.println("Element removed from stack is : " + st.pop());

System.out.println("Element removed from stack is : " + st.pop());

System.out.println("Element removed from stack is : " + st.pop());

}

catch (EmptyStackException e)

{

System.out.println("Stack underflow OR Empty ! " + e);

} \*/

}

}

Output:



