Date:14-10-2020

Day : Wednesday

Name : BUDATI RAMU

Reg .no : 9919004044

Course Name : Java Programming language

Course Code : CSE18R272

Section : A5

Exercise-6:

1. Write a program that creates a user interface to perform integer divisions. The user enters two numbers Num1 and Num2. If Num1 or Num2 is not an integer, the program would throw NumberFormatException. If Num2 is Zero, the program would throw an ArithmeticException. Display the exception.

Code:

Import java.io.\*;

Public class MyClass {

Public static void main(String args[])throws IOException{

String num1,num2;

Int n1,n2,d;

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

Try{

Num1=br.readLine();

Num2=br.readLine();

N1=Integer.parseInt(num1);

N2=Integer.parseInt(num2);

D=n1/n2;

}

Catch(NumberFormatException e)

{

System.out.println(“input are not valid”);

}

Catch(ArithmeticException e)

{

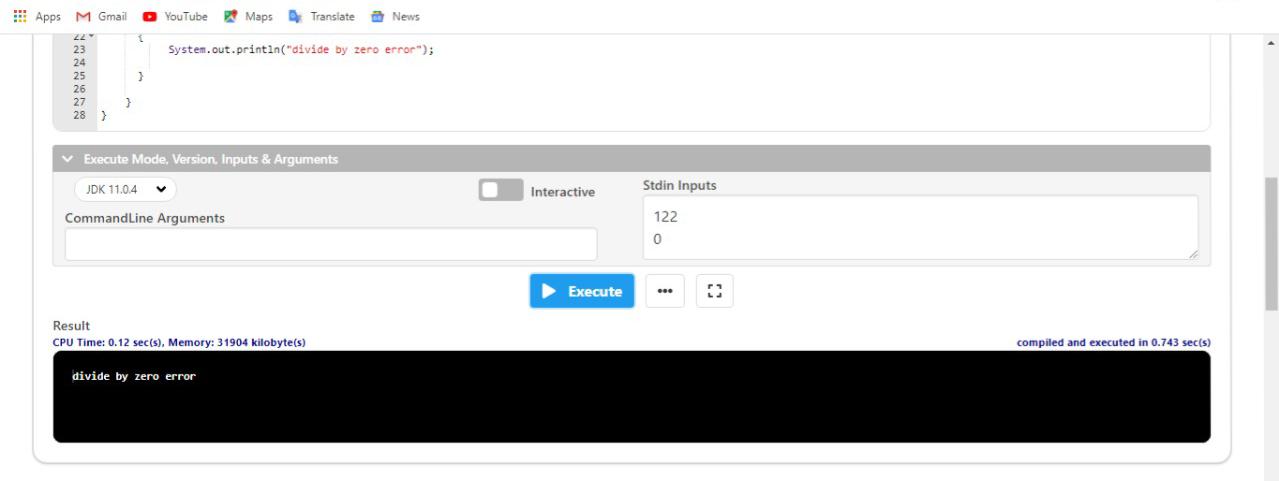
System.out.println(“divide by zero error”);

}

}

}

Output:



1. Java programs to create an bank account with minimum balance, deposit amount, withdraw amount and throws LessBalanceException, create a LessBalanceException class which returns a statement says withdraw amount is not valid, creates 2 accounts and try to withdraw more money than account and see which type of exception occurs.

Code:

Class BalanceCheck extends Exception

{

BalanceCheck()

{

Super(“Transaction failed less balance “);

}

}

Class Account

{

Int accno;

String name;

Double balance;

Static int minimum=500;

Account(int ano,String n,double bal)

{

Accno=ano;name=n;balance=bal;

}

Void withdrawl(int amt)throws BalanceCheck

{

If((balance-amt)>minimum)

{

Balance-=amt;

System.out.println(“transaction was succesfully done”);

}

Else

{

Throw new BalanceCheck();

}

}

Void deposit(int amt)

{

Balance+=amt;

}

Double checkBalance()

{

Return balance;

}

}

Public class MyClass {

Public static void main(String args[]) {

Account a1=new Account(123,”ramu”,15000);

Account a2=new Account(123,”siva”,20000);

Try

{

A1.withdrawl(14800);

A2.withdrawl(10000);

}

Catch(BalanceCheck b)

{

System.out.println(b);

}

System.out.println(“a1 balance =”+a1.checkBalance());

System.out.println(“a2 balance =”+a2.checkBalance());

A1.deposit(10000);a2.deposit(5000);

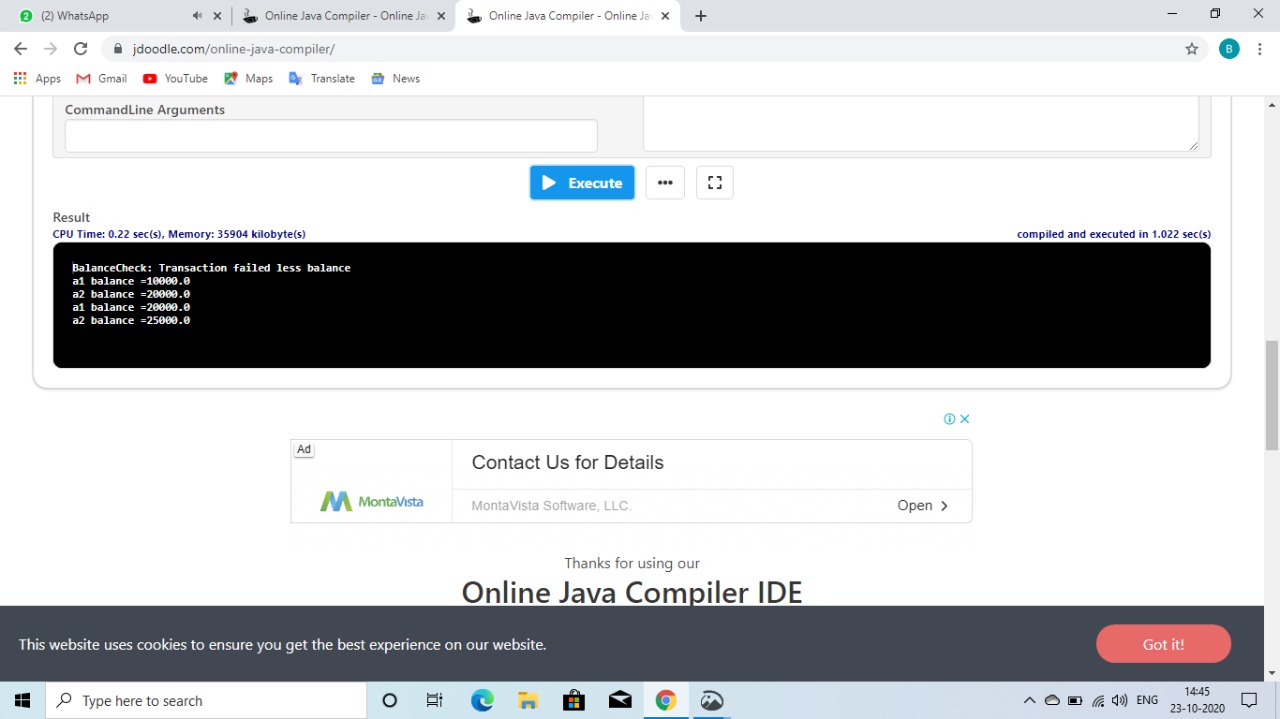
System.out.println(“a1 balance =”+a1.checkBalance());

System.out.println(“a2 balance =”+a2.checkBalance());

}

}

Output:



1. Write a Java program to check whether the age entered is a valid number by creating user defined exception.

Code:

Import java.util.\*;

Class AgeCheck extends Exception

{

AgeCheck()

{

Super(“Exception : invalid age “);

}

}

Public class MyClass {

Public static void main(String args[]){

Int age;

Scanner s=new Scanner (System.in);

Age=s.nextInt();

Boolean b=false;

Try

{

B=checkAge(age);

System.out.println(“valid age”);

}

Catch(AgeCheck ag)

{

System.out.println(ag);

}

}

Static boolean checkAge(int age)throws AgeCheck

{

If(age >0 && age <=120)

Return true;

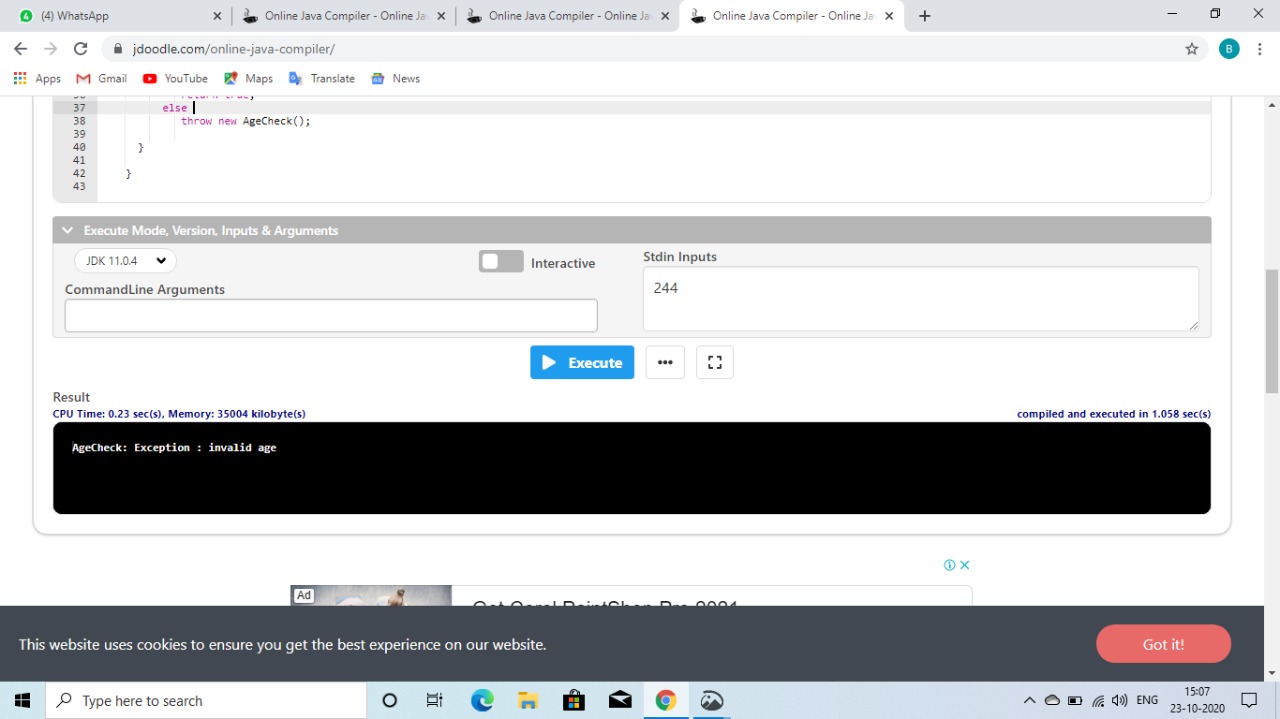
Else

Throw new AgeCheck();

}

}

Output:



1. Design a Java interface for ADT Stack. Implement this interface using array. Provide necessary exception handling in both the implementations.

Code:

Class FullStack extends Exception

{

FullStack()

{

Super(“Stack is Full”);

}

}

Class EmptyStack extends Exception

{

EmptyStack()

{

Super(“Stack is Empty “);

}

}

Class Stack

{

Int top;

Int arr[];

Static int max=10;

Stack()

{

Top=-1;

Arr=new int[max];

}

Void push(int x)throws FullStack

{

If(top==max-1)

{

Throw new FullStack();

}

Else

{

Arr[++top]=x;

}

}

Int pop()throws EmptyStack

{

If(top==-1)

{

Throw new EmptyStack();

}

Else

{

Return(arr[top--]);

}

}

Void print()

{

For(int i=0;i<arr.length;i++)

System.out.print(arr[i]+” “);

System.out.println();

}

}

Public class MyClass {

Public static void main(String args[]) {

Stack s1=new Stack();int x;

For(int i=1;i<=12;i++)

{

Try{

S1.push(i);

S1.print();

}

Catch(FullStack fs)

{

System.out.println(fs);

}

}

For (int i=1;i<=12;i++)

{

Try

{

X=s1.pop();

System.out.print(x +” “);

}

Catch(EmptyStack es)

{

System.out.println(es);

}

}

}

}

Output:

