Kalisalingam Academy Of Research And Education

Java Programming

Name:B.YASHWANTH

Date :23-10-2020

Regno:9919004036

Code :CSE18R272

Exercise No : 6

Programs:

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");

}

}

}

Inputs:

160

0

Output:

divide by zero error.

2. 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.

Program:

import java.io.\*;

class BalanceCheck extends Exception{

BalanceCheck(){

super("Transaction Denied: No min balance found");

}

}

class Bank{

int accountno;

String name;

double balance;

static int min\_amount=500;

Bank(int ano,String nm,double bal,int min){

accountno=ano;

name=nm;

balance=bal;

}

void Withdraw(double cash) throws BalanceCheck{

if((balance-cash)>=min\_amount){

balance=balance-cash;

System.out.println("Transaction Succesful");

System.out.println("the balance after withdrawl is "+ balance);

}

else{

throw new BalanceCheck();

}

}

void Deposit(double cash){

balance = balance+cash;

System.out.println("Transaction Succesful");

System.out.println("the balance after deposit is "+balance);

}

void CheckBal(){

System.out.println("the balance is"+balance);

}

}

public class Main

{

public static void main(String[] args) throws Exception {

Bank b1 = new Bank(4036,"ROY",12000,500);

Bank b2 = new Bank(1436,"YASH",14000,500);

try{

b1.Withdraw(10500);

b2.Withdraw(12000);

}

catch(BalanceCheck b){

System.out.println(b);

}

b1.CheckBal();

b2.CheckBal();

b1.Deposit(100);

b2.Deposit(150);

b1.CheckBal();

b2.CheckBal();

}

}

Output:

Transaction Succesful

the balance after withdrawl is 1500.0

Transaction Succesful

the balance after withdrawl is 2000.0

the balance is1500.0

the balance is2000.0

Transaction Succesful

the balance after deposit is 1600.0

Transaction Succesful

the balance after deposit is 2150.0

the balance is1600.0

the balance is2150.0

3.Write a Java program to check whether the age entered is a valid number bycreating user defined exception.

Program:

import java.util.\*;

class AgeCheck extends Exception

{

AgeCheck()

{

super("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();

}

}

Input: 22

Output: valid age

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

Program:

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:

1 0 0 0 0 0 0 0 0 0

1 2 0 0 0 0 0 0 0 0

1 2 3 0 0 0 0 0 0 0

1 2 3 4 0 0 0 0 0 0

1 2 3 4 5 0 0 0 0 0

1 2 3 4 5 6 0 0 0 0

1 2 3 4 5 6 7 0 0 0

1 2 3 4 5 6 7 8 0 0

1 2 3 4 5 6 7 8 9 0

1 2 3 4 5 6 7 8 9 10

FullStack: Stack is Full

FullStack: Stack is Full

10 9 8 7 6 5 4 3 2 1 EmptyStack: Stack is Empty

EmptyStack: Stack is Empty