Java Programming Lab 2

## Name: Om Ashish Mishra

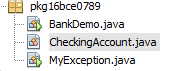
## Registration Number: 16BCE0789

## Slot: G2

**The Quesuions:**

1. Implement a bank application where an alert message is issued when the minimum balance is going below 1000INR. Create an exception class and a Bank class for this application and test it with a Java program.

The Code:



package pkg16bce0789;

import java.util.Scanner;

public class BankDemo {

public static void main(String [] args) {

Scanner sc = new Scanner(System.in);

CheckingAccount c = new CheckingAccount(101);

int d = sc.nextInt();

//System.out.println("Depositing $500...");

c.deposit(d);

try {

System.out.println("Withdrawing $100...");

c.withdraw(100.00);

System.out.println("Withdrawing $600...");

c.withdraw(600.00);

} catch (MyException e) {

System.out.println("Sorry, but you are short $" + e.getAmount());

e.printStackTrace();

}

}

}

package pkg16bce0789;

class CheckingAccount {

private double balance;

private int number;

public CheckingAccount(int number) {

this.number = number;

}

public void deposit(double amount) {

balance += amount;

}

public void withdraw(double amount) throws MyException {

if(amount >1000)

{

if(amount <= balance) {

balance -= amount;

}else {

double needs = amount - balance;

throw new MyException(needs);

}

}

else

{

double needs = amount - balance;

throw new MyException(needs);

}

}

public double getBalance() {

return balance;

}

public int getNumber() {

return number;

}

}

import java.io.\*;

class MyException extends Exception {

private double amount;

public MyException(double amount) {

this.amount = amount;

}

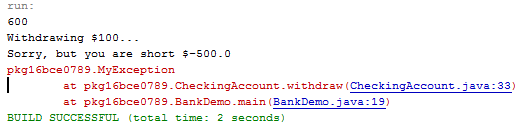
public double getAmount() {

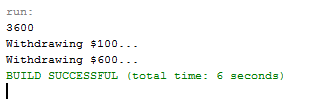
return amount;

}

}

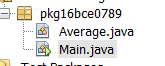
The Output:





1. Develop a Java application for calculating the average mark of ‘n’ students. Read the number of courses they have appeared for the past semester and the marks in all the courses. The number of courses should not be zero and if it is zero handle that case with a standard exception.

The Code:



package pkg16bce0789;

/\*\*

\*

\* @author OM MISHRA

\*/

class Average

{

double avg=0;

void check (double amount) throws ZeroException

{

if (a==0)

throw new ZeroException();

else

continue;

}

Average(int a[])

{

for(int i=0;i<a.length;i++)

{

avg=avg+a[i];

}

}

}

class Main

{

public static void main(String args[])

{

int i;

System.out.println("Enter number of subjects");

Scanner sc=new Scanner(System.in);

int n=sc.nextInt();

int[] a=new int[n];

System.out.println("Enter marks");

for( i=0;i<n;i++)

{

a[i]=sc.nextInt();

}

Average c = new Average(a);

c.check();

System.out.print("Average of (");

for(i=0;i<n-1;i++)

{

System.out.print(a[i]+",");

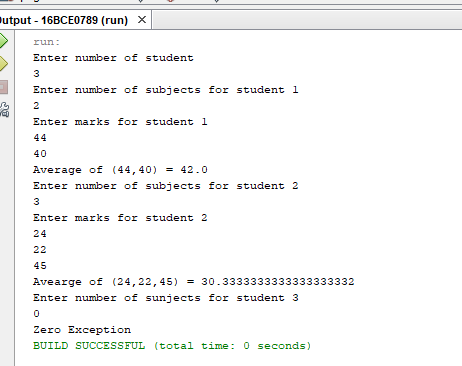
}

System.out.println(a[i]+") ="+c.avg/n);

}

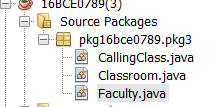
}

The Output:



1. Write a Java program to develop an application where you have a class ‘ClassRoom’ which have to be used by the (thread) objects of class ‘Faculty’ to deliver their course contents. Since the ‘Faculty’ class objects are active simultaneously, synchronize the usage of the object of ‘Classroom’.

The Code:



// A Class used to send a message

class Faculty

{

public void send(String msg)

{

System.out.println("Sending\t" + msg );

try

{

Thread.sleep(1000);

}

catch (Exception e)

{

System.out.println("Thread interrupted.");

}

System.out.println("\n" + msg + "Sent");

}

}

package pkg16bce0789.pkg3;

/\*\*

\*

\* @author OM MISHRA

\*/

class Classroom extends Thread

{

private String msg;

private Thread t;

Faculty faculty;

// Recieves a message object and a string

// message to be sent

Classroom(String m, Faculty obj)

{

msg = m;

faculty = obj;

}

public void run()

{

// Only one thread can send a message

// at a time.

synchronized(faculty)

{

// synchronizing the snd object

faculty.send(msg);

}

}

}

package pkg16bce0789.pkg3;

/\*\*

\*

\* @author OM MISHRA

\*/

class CallingClass

{

public static void main(String args[])

{

Faculty f1 = new Faculty();

Classroom C1 =

new Classroom( " Good Morning " , f1 );

Classroom C2 =

new Classroom( " Thank You " , f1 );

// Start two threads of ThreadedSend type

C1.start();

C2.start();

// wait for threads to end

try

{

C1.join();

C2.join();

}

catch(Exception e)

{

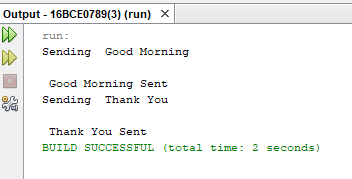
System.out.println("Interrupted");

}

}

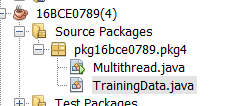
}

The Output:



1. We need to train the patient data to predict whether a new patient may or may not get the disease. Implement a data training application in Java which prepares the data set used in a disease prediction algorithm. The ‘TraingData’ class has an array of records (patient-id, patient-age, diseaseid, and disease-seriousness-index) and two methods ‘writeData’ and ‘readData’. In a multi-threaded environment, if one thread is writing the data then other threads have to wait and if one thread is reading the data then other threads have to wait.

The Code:



package pkg16bce0789.pkg4;

/\*\*

\*

\* @author OM MISHRA

\*/

import java.io.\*;

import java.util.\*;

import java.lang.\*;

public class Multithread

{

public static void main(String[] args)

{

Scanner sc = new Scanner(System.in);

//int n = 8; // Number of threads

int n = sc.nextInt();

for (int i=0; i<n; i++)

{

TrainingData object = new TrainingData();

object.writeData();

object.readData();

object.start();

}

}

}

package pkg16bce0789.pkg4;

/\*\*

\*

\* @author OM MISHRA

\*/

import java.io.\*;

import java.util.\*;

import java.lang.\*;

class TrainingData extends Thread

{

public int Patientid[];

public int age[];

public int disid[];

public int dsi[];

static int i = 0;

public void writeData()

{

Patientid = new int[] {1,2,3,4,5,6,7,8,9};

age = new int[] {12,56,45,89,26,78,24,67,74};

disid = new int[] {1,2,1,1,2,2,1,1,1};

dsi = new int[] {1,2,3,1,1,3,2,1,2};

}

public void readData()

{

while(i!=9)

{

System.out.println("Pateint Id :"+Patientid[i]);

System.out.println("Pateint age :"+age[i]);

System.out.println("Disease Id :"+disid[i]);

System.out.println("disease-seriousness-index :"+dsi[i]);

i++;

}

}

public void run()

{

try

{

// Displaying the thread that is running

int n =(int) Thread.currentThread().getId();

System.out.println ("Thread " + n + " is running");

System.out.println("Pateint id "+Patientid[(n-11)]);

System.out.println("Patient age "+age[(n-11)]);

System.out.println("Disease id "+disid[(n-11)]);

System.out.println("disease-seriousness-index "+dsi[(n-11)]);

}

catch (Exception e)

{

// Throwing an exception

System.out.println ("Exception is caught");

}

}

}

The Output:

