Thota GuruTheja Reddy

19BCD7034

Lab-2

1.

import java.util.Scanner;

class Circle{

double radius;

double pi = 3.14;

Circle(double r){

this.radius = r;

}

public double getArea(){

return (pi\*radius\*radius);

}

public double getCircumference(){

return (2\*pi\*radius);

}

void displayArea()

{

System.out.println("Area of the circle is: "+getArea());

}

void displayCircumference()

{

System.out.println("Circumference of the circle is: "+getCircumference());

}

}

public class Main

{

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter radius of the circle");

double r = sc.nextDouble();

Circle obj = new Circle(r);

obj.getArea();

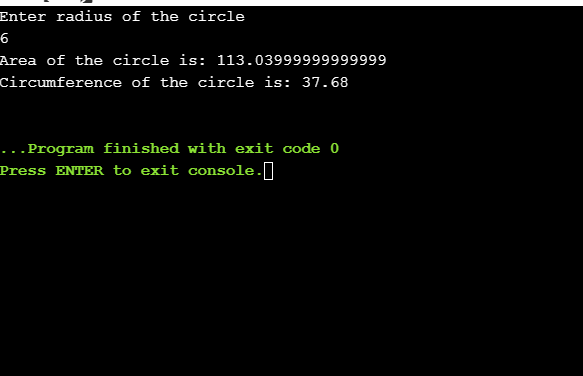
obj.getCircumference();

obj.displayArea();

obj.displayCircumference();

}

}



2. import java.util.Scanner;

class Student{

String name,address;

static int regno;

final String dept="CSE";

int sub\_mark1,sub\_mark2,sub\_mark3;

int elec1,elec2,elec3;

int num\_of\_subs;

char grade;

Student(int sub\_mark1,int sub\_mark2,int sub\_mark3,int elec1,int elec2,int elec3){

num\_of\_subs= 6;

this.sub\_mark1=sub\_mark1;

this.sub\_mark2=sub\_mark2;

this.sub\_mark3=sub\_mark3;

this.elec1=elec1;

this.elec2=elec2;

this.elec3=elec3;

}

Student(int sub\_mark1,int sub\_mark2,int sub\_mark3,int elec1,int elec2){

num\_of\_subs= 5;

this.sub\_mark1=sub\_mark1;

this.sub\_mark2=sub\_mark2;

this.sub\_mark3=sub\_mark3;

this.elec1=elec1;

this.elec2=elec2;

}

int getTotal(){

int total = sub\_mark1+sub\_mark2+sub\_mark3+elec1+elec2+elec3;

return total;

}

double getAverage(int total){

double avg =total/num\_of\_subs;

return avg;

}

char getGrade(double avg){

if (avg<50){

grade='F';

}else if(avg>=50 && avg<60){

grade='E';

}else if(avg>=60 && avg<70){

grade='D';

}else if(avg>=70 && avg<80){

grade='C';

}else if(avg>=80 && avg<90){

grade='B';

}else if(avg>=90 && avg<95){

grade='A';

}else {

grade='S';

}

return grade;

}

void display(){

System.out.println("Details of the student are:\nName: "+name+"\nRegistration number: "+regno+"\nDepartment: "+dept+"\nAddress: "+address+"\nGrade of the student is "+grade );

}

}

public class Main{

public static void main (String[] args) {

int sub\_mark1,sub\_mark2,sub\_mark3;

int elec1,elec2,elec3;

String name,address;

int regno;

String dept;

Student s[]=new Student[3];

Scanner sc=new Scanner(System.in);

for(int i=0;i<3;i++){

if(i>0){

sc.nextLine();

}

System.out.println("Is there 3 electives yes or no ");

String o=sc.nextLine();

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

name=sc.nextLine();

System.out.println("Enter registration number");

regno=sc.nextInt();

sc.nextLine();

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

address=sc.nextLine();

System.out.println("Enter sub\_mark1");

sub\_mark1=sc.nextInt();

System.out.println("Enter sub\_mark2");

sub\_mark2=sc.nextInt();

System.out.println("Enter sub\_mark3");

sub\_mark3=sc.nextInt();

System.out.println("Enter elec1");

elec1=sc.nextInt();

System.out.println("Enter elec2");

elec2=sc.nextInt();

if (o.equals("no")){

s[i] = new Student(sub\_mark1,sub\_mark2,sub\_mark3,elec1,elec2);

int n = s[i].getTotal();

double g = s[i].getAverage(n);

char grade = s[i].getGrade(g);

s[i].name=name;

s[i].regno=regno;

s[i].address=address;

}

else {

System.out.println("Enter elec3");

elec3=sc.nextInt();

s[i] = new Student(sub\_mark1,sub\_mark2,sub\_mark3,elec1,elec2,elec3);

int n = s[i].getTotal();

double g = s[i].getAverage(n);

char grade = s[i].getGrade(g);

s[i].name=name;

s[i].regno=regno;

s[i].address=address;

}

}

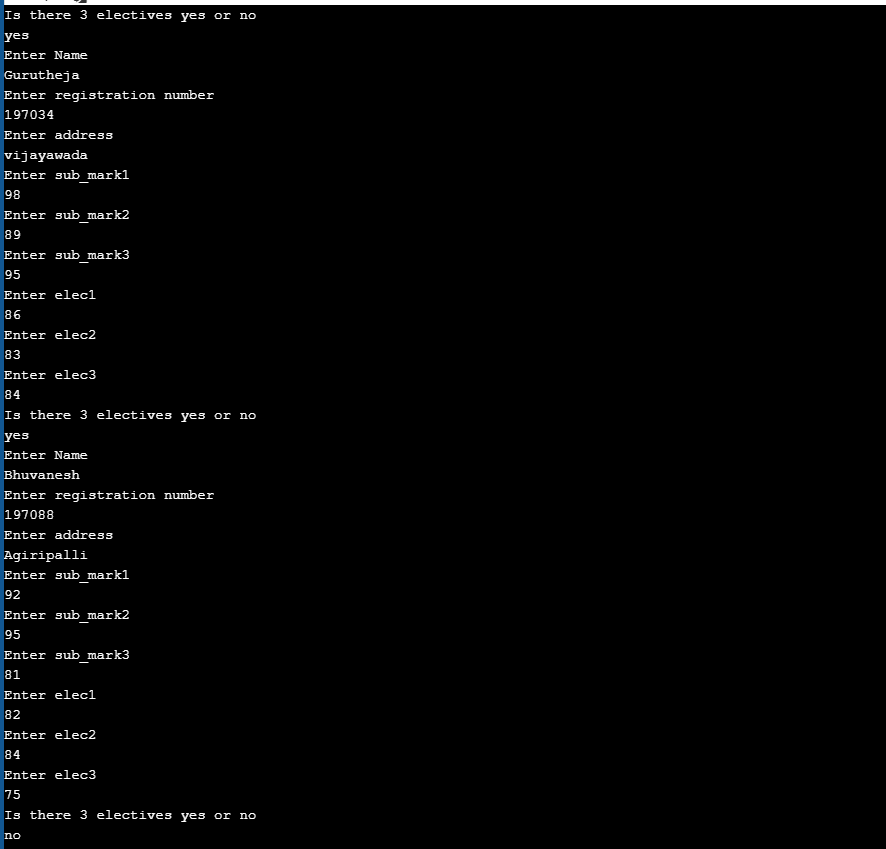
for(int j=0;j<3;j++){

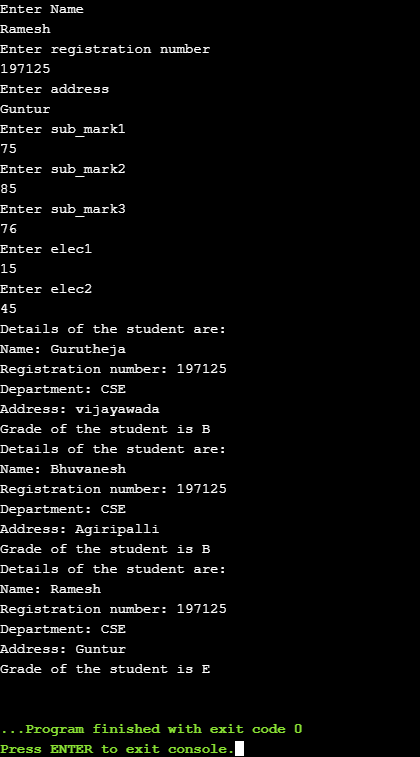
s[j].display();

}

}

}





1.

import java.util.Scanner;

class Rectangle{

int length;

int breadth;

public Rectangle(int length,int breadth){

this.length=length;

this.breadth=breadth;

}

}

class Box extends Rectangle{

int height;

Box(int length,int breadth, int h){

super(length,breadth);

height=h;

}

int volume(){

return length\*breadth\*height;

}

}

class Main{

public static void main(String[]args){

Scanner sc= new Scanner(System.in);

System.out.println("Enter Length: ");

int l = sc.nextInt();

System.out.println("Enter Breadth: ");

int b = sc.nextInt();

System.out.println("Enter Height: ");

int h = sc.nextInt();

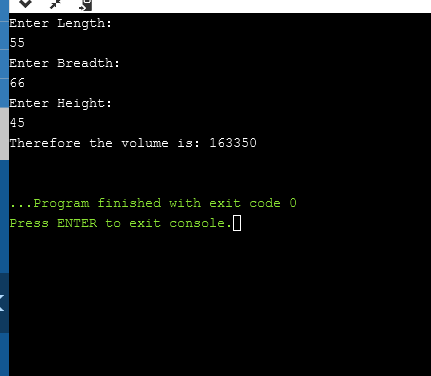
Box obj= new Box(l,b,h);

System.out.println("Therefore the volume is: "+obj.volume());

}

}

Output:



2.

import java.util.Scanner;

class Student {

private String name;

private String email;

private char gender;

public Student(String name, String email, char gender) {

this.name = name;

this.email = email;

this.gender = gender;

}

public String getName() {

return name;

}

public char getGender() {

return gender;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public String toString() {

return( "Details of the student are:\nName: "+name+"\nGender: "+gender+"\nDepartment: "+dept+"\nEmail: "+name+"@"+email);

}

}

class Department extends Student{

String dept;

Department(String name, String email, char gender, String str){

super(name,email,gender);

dept=str;

}

public String getDepartment(){

return dept;

}

}

public class Main{

public static void main(String[]args){

Scanner sc=new Scanner(System.in);

System.out.println("Enter name of the student ");

String n = sc.nextLine();

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

String e = sc.nextLine();

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

char g = sc.next().charAt(0);

sc.nextLine();

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

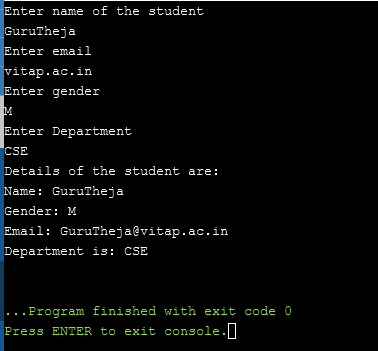
String d= sc.nextLine();

Department stu=new Department(n,e,g,d);

System.out.println(stu);

}

}



3.

import java.util.Scanner;

class Department{

String department\_name,hod\_name;

int total\_students,no\_of\_sections;

public Department(String department\_name, String hod\_name, int total\_students, int no\_of\_sections){

this.department\_name=department\_name;

this.hod\_name=hod\_name;

this.total\_students=total\_students;

this.no\_of\_sections=no\_of\_sections;

}

void showDepartmentDetails(){

System.out.println("Department\_name: "+department\_name);

System.out.println("Hod\_name: "+hod\_name);

System.out.println("Total\_students: "+total\_students);

System.out.println("No\_of\_sections: "+no\_of\_sections);

}

}

class StaffMember extends Department implements Publication {

String staff\_name,staff\_qualification,designation,experience;

int staff\_id;

public StaffMember(String department\_name, String hod\_name, int total\_students, int no\_of\_sections,String staff\_name,int staff\_id,String staff\_qualification,String designation,String experience){

super(department\_name,hod\_name,total\_students,no\_of\_sections);

this.staff\_name=staff\_name;

this.staff\_id=staff\_id;

this.staff\_qualification=staff\_qualification;

this.designation=designation;

this.experience=experience;

}

void showStaffDetails(){

System.out.println("Staff Name: "+staff\_name);

System.out.println("Staff\_id "+staff\_id);

System.out.println("Staff\_qualification: "+staff\_qualification);

System.out.println("Designation: "+designation);

System.out.println("Experience: "+experience);

}

public void show\_publication\_detail()

{

System.out.println("Number of journal count"+journalcount);

System.out.println("Number of projectcount"+projectcount);

System.out.println("Number of patterncount"+patterncount);

}

}

interface Publication{

int journalcount=5;

int projectcount=6;

int patterncount=7;

void show\_publication\_detail();

}

public class Main{

public static void main(String[]args){

String d,h,sn,sq,des,e;

int t,ns,si ;

Scanner sc=new Scanner(System.in);

System.out.println("Enter number of staff members: ");

int n=sc.nextInt();

StaffMember obj[]= new StaffMember[n];

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

System.out.println("Enter Department details ");

d=sc.nextLine();

h=sc.nextLine();

sc.nextLine();

t=sc.nextInt();

ns=sc.nextInt();

System.out.println("Enter Staff Details");

sn=sc.nextLine();

sc.nextLine();

si=sc.nextInt();

sc.nextLine();

sq=sc.nextLine();

des=sc.nextLine();

e=sc.nextLine();

obj[i]=new StaffMember(d,h,t,ns,sn,si,sq,des,e);

}

for(int j=0;j<n;j++){

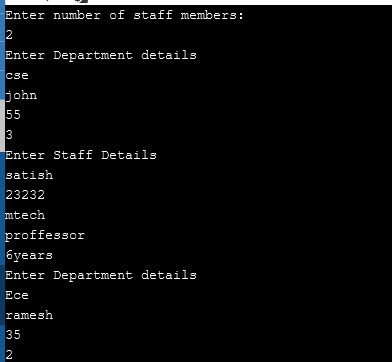
obj[j].showDepartmentDetails();

obj[j].showStaffDetails();

}

}

}



**THOTA GURUTHEJA REDDY**

**19BCD7034**

**LAB\_4**

1. Create a class 'Degree' having a method 'getDegree' that prints "I got a degree". It has two subclasses namely 'Undergraduate' and 'Postgraduate' each having a method with the same name that prints "I am an Undergraduate" and "I am a Postgraduate" respectively. Call the method by creating an object of each class and print the student details (name, regno, degree, years) by creating a method “display” in class Degree

Ans.)

import java.util.Scanner;

class Degree{

public void getDegree(){

System.out.println("I got a Degree");

}

String name, regno, degree;

int years;

void display(String name,String regno,String degree, int years){

this.name=name;

this.regno=regno;

this.degree=degree;

this.years=years;

System.out.println("Name of the Student: "+name+"\nRegistration number is: "+regno+"\nDegree: "+degree+"\nYears: "+years);

}

}

class Undergraduate extends Degree{

public void getDegree(){

System.out.println("I am an Undergraduate");

}

}

class Postgraduate extends Degree{

public void getDegree(){

System.out.println("I am an Postgraduate");

}

}

public class Main{

public static void main(String[]args){

Undergraduate a = new Undergraduate();

Postgraduate b = new Postgraduate();

Scanner sc=new Scanner(System.in);

System.out.println("Enter name of the student");

String n = sc.nextLine();

System.out.println("Enter Registration number");

String r = sc.nextLine();

System.out.println("Enter type of Degree");

String d = sc.nextLine();

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

int y = sc.nextInt();

if(d.equals("Undergraduate")){

a.display(n,r,d,y);

a.getDegree();

}

else if (d.equals("Postgraduate")){

b.display(n,r,d,y);

b.getDegree();

}

else {

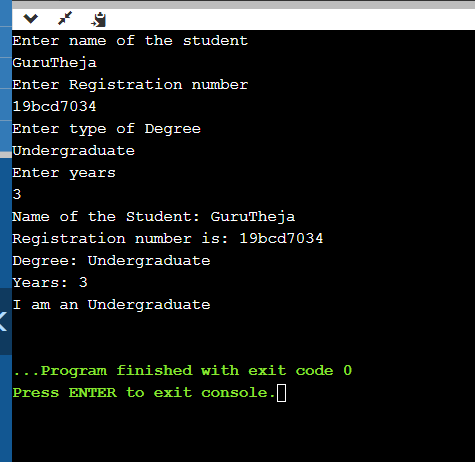
System.out.println("Invalid input");

}

}

}

Output:



2. Create a class Shapes with a method “calculatearea()” to find the area of a circle (πr2), square(a2), rectangle(w X h) and triangle(1/2 b h) using the same method name “calculatearea” in each class. Declare the input for calculation in class as private (r,a,w,h,b).

Ans.)

import java.util.Scanner;

class Shapes{

private double r,w;

private int a,h,b;

double pi=3.14;

void calculateArea(double r){

this.r=r;

double area = pi\*r\*r;

System.out.println("Area of circle is: "+area);

}

void calculateArea(int a){

this.a=a;

double area = a\*a;

System.out.println("Area of square is: "+area);

}

void calculateArea(double w,int h){

this.w=w;

this.h=h;

double area = w\*h;

System.out.println("Area of rectangle is: "+area);

}

void calculateArea(int b, int h){

this.b=b;

this.h=h;

double area = 0.5\*b\*h;

System.out.println("Area of triangle is: "+area);

}

}

class Main{

public static void main(String[]args){

Scanner sc =new Scanner(System.in);

System.out.println("Enter radius: ");

double r= sc.nextDouble();

System.out.println("Enter height: ");

int h= sc.nextInt();

System.out.println("Enter width: ");

double w= sc.nextDouble();

System.out.println("Enter length: ");

int a= sc.nextInt();

System.out.println("Enter base length: ");

int b= sc.nextInt();

Shapes obj=new Shapes();

obj.calculateArea(r);

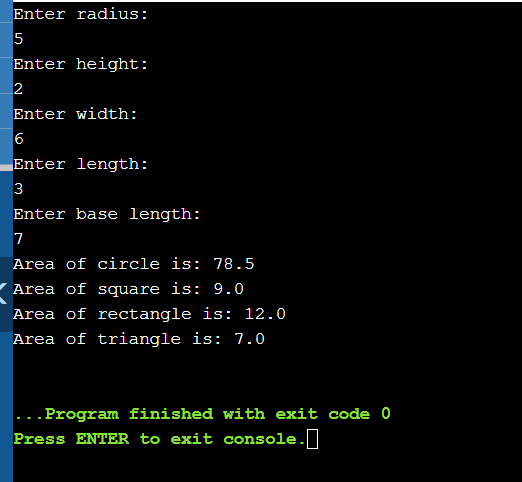
obj.calculateArea(a);

obj.calculateArea(w,h);

obj.calculateArea(b,h);

}

}

Output: 

3. Create a class to perform addition of different data types (int, double, string) using the concept of polymorphism.

int add(int,int)

add(int, double)

add(double,int)

add(double,double)

add(String,String)

add(int [])

Ans.)

import java.util.Scanner;

class addition {

void add(int a,int b)

{

int c=a+b;

System.out.println("Sum of numbers is "+ c);

}

void add(int a,double b)

{

double c=a+b;

System.out.println("Sum of numbers is "+ c);

}

void add(double a,int b)

{

double c=a+b;

System.out.println("Sum of numbers is "+c);

}

void add(double a,double b)

{

double c=a+b;

System.out.println("Sum of numbers is "+c);

}

void add(String a,String b)

{

String c=a+b;

System.out.println("Sum of strings is "+c);

}

void add(int a[])

{

int c=0;

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

{

c=c+a[i];

}

System.out.println("Sum of numbers in array is "+ c);

}

}

class Main {

public static void main(String[] args) {

addition a1=new addition();

Scanner sc=new Scanner(System.in);

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

int n= sc.nextInt();

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

double d=sc.nextDouble();

sc.nextLine();

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

String str=sc.nextLine();

System.out.println("Enter number of elements in array");

int e=sc.nextInt();

int a[]= new int[e];

for (int i=0;i<e;i++){

a[i]=sc.nextInt();

}

a1.add(n,n);

a1.add(n,d);

a1.add(d,n);

a1.add(d,d);

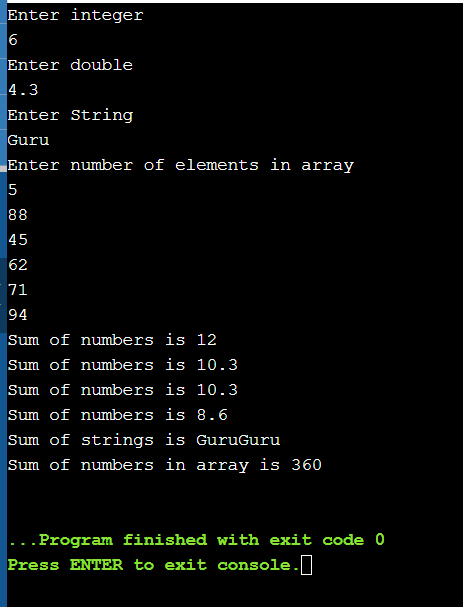
a1.add(str,str);

a1.add(a);

}

}

Output:



1. Write a java generic program to create class that takes one type parameter to print the different data type value and its class name.

Ans: class Gen <T>{

T x;

Gen(T g)

{

x=g;

}

String Display(){

return(x.getClass().getSimpleName());

}

public T getx()

{

return x;

}

}

class Main{

public static void main(String[]args){

Gen < Integer> a = new Gen<>(85);

int g =a.getx();

System.out.println("Class Name is: "+a.Display());

System.out.println(g);

Gen < Double> b = new Gen<>(88.35);

Double t =b.getx();

System.out.println("Class Name is: "+b.Display());

System.out.println(t);

Gen < String> c = new Gen<>("GTR");

String r = c.getx();

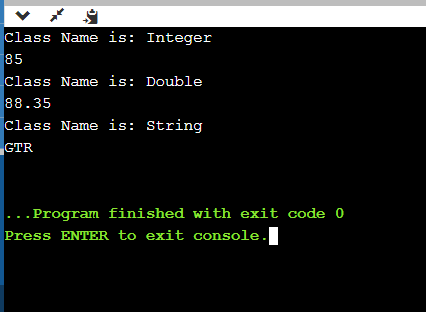
System.out.println("Class Name is: "+c.Display());

System.out.println(r);

}

}

Output:



1. Write a java generic program to print greatest of two numbers.

Ans:

import java.util.Scanner;

class Gen<T>{

public static <T extends Comparable<T>> T max(T x1, T x2){

T max = x1;

if(x2.compareTo(max)>0){

max = x2;

}

return max;

}

}

class Main{

public static void main(String args[]){

Scanner sc = new Scanner(System.in);

System.out.println("Enter the two numbers:");

Integer a=sc.nextInt();

Integer b=sc.nextInt();

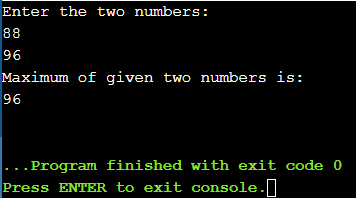
Gen < Integer> obj = new Gen<Integer>();

System.out.println("Maximum of given two numbers is: ");

System.out.println(obj.max(a,b));

}

}

OutPut:

1. Write a java generic program to print the ArrayList of different data types.

Ans:

import java.util.\*;

public class Main{

static <T> void printArray(T[] a){

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

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

}

System.out.println();

}

public static void main(String[] args) {

Integer [] a={1,2,3,4,5,6,7,8,9};

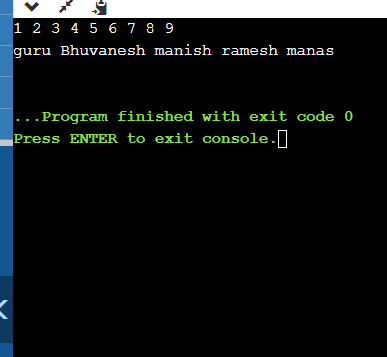
String [] str = {"guru","Bhuvanesh","manish","ramesh","manas"};

printArray(a);

printArray(str);

}

}

Output: 

1. Create a class “CountFreqEven” with two generic static method. One is “count” to find the frequency of a value in an array of any type with duplicate elements and the other is “Counteven” to check the number of even numbers in the list whose type is bound to Integer value. Create two arrays of Integer and String with duplicate elements and print the frequency of a “value” passed to method “count” along with array elements. Pass an integer array to method “Counteven” and print the number of even numbers in an integer array.

Ans:

class CountFreqEven{

public static <T> int[] count(T[]a){

int count;

int freq[]=new int[a.length];

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

{

count = 0;

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

{

if(a[k]==a[j])

{

count++;

}

}

freq[k] = count;

}

return freq;

}

public static int countEven(Integer []a){

int n=0;

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

if(a[k]%2==0){

n++;

}

}

return n;

}

public static void main(String[]args){

Integer []g={1,2,5,6,88,88,44,42};

int[]b =new int[g.length];

b=count(g);

int h=countEven(g);

System.out.println("Frequencies of each element in array are: ");

for(int y=0;y<b.length;y++){

System.out.print(b[y]+" ");

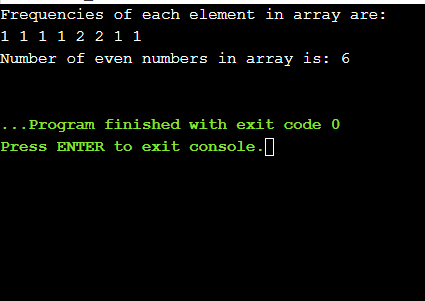
}

System.out.println("\nNumber of even numbers in array is: "+h);

}

}

Output:



1. Write a java program to demonstrate ArithmeticException, NullPointerException, ArrayIndexOutOfBoundsException and NumberFormatEception using throws.

Ans:

class Main{

public static void method1()throws ArithmeticException

{

int x=68/0;

System.out.println(x);

}

public static void method2()throws ArrayIndexOutOfBoundsException

{

int a[]={1,2,3};

System.out.println(a[3]);

}

public static void method3()throws NullPointerException

{

Object ref = null;

ref.toString();

}

public static void method4()throws NumberFormatException

{

int data = Integer.parseInt("hello");

}

public static void main(String[]args){

method1();

method2();

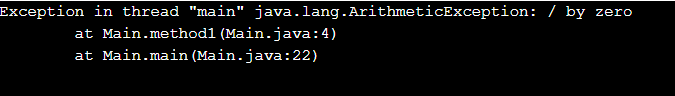
method3();

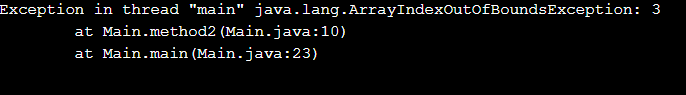
method4();

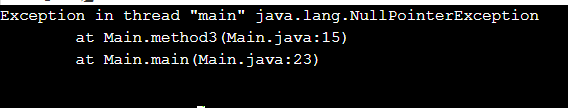
}

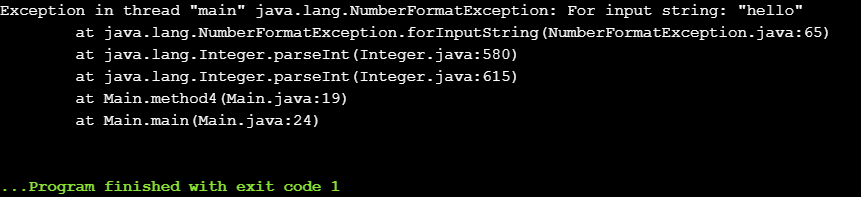
}

Output:









2. Write a java program by considering your own example to show the working of nested try catch and finally

class Main{

public static void main(String[]args){

try{

int x=68/0;

System.out.println(x);

try{

int a[]={1,2,3};

System.out.println(a[3]);

}

catch(ArrayIndexOutOfBoundsException e)

{

System.out.println("Can't access element out of the array");

}

}

catch(ArithmeticException e)

{

System.out.println("ArithmeticException can't divide number by zero");

}

finally

{

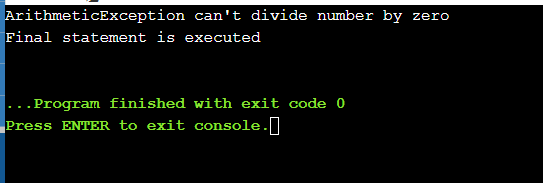
System.out.println("Final statement is executed");

}

}

}

OutPut:



3. Write a program to get the details of voter and check whether age is >18 if not raise an exception “check18”. Create your exception in the name ”check18”.

Ans:

import java.util.Scanner;

class check18 extends Exception{

public check18(){

System.out.println("Age of the voter is less than 18, votting not allowed ");

}

}

class Main {

public static void main(String[] args) {

Scanner in=new Scanner(System.in);

System.out.println("Enter the age of voter");

int n=in.nextInt();

try

{

if(n<18){

throw new check18();

}else{

System.out.println("Voter is valid for votting");

}

}

catch(Exception e)

{

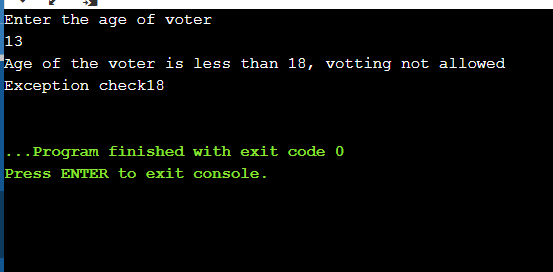
System.out.println("Exception "+ e);

}

}

}

Output:



4. Create a class “stud” and two sub class “studA” an “studB” which has a constructor to set all the variables in class stud. Create a generic class “student” and two methods, one is “getaverage()” to get the average and method “maxavg to compare the average of two students. Create a class “test” and create two objects S1 and S2 for the generic class student whose type is bound to “stud”, S1 is of type “studA” and S2 of type “studB”. Pass the details of each student via the constructor and compare their average mark. If the average mark of S1 >S2 it returns 1 and print “Student S1 has got High Average” else return 2 and print “Student S2 has got High Average”.

Ans:

import java.util.Scanner;

class Stud {

String name, regno;

int mark1,mark2;

}

class StudA extends Stud {

public StudA(String name,String regno,int mark1,int mark2)

{

this.name=name;

this.regno=regno;

this.mark1=mark1;

this.mark2=mark2;

System.out.println("Name "+name);

System.out.println("Registration number "+regno);

System.out.println("mark1 "+mark1);

System.out.println("mark2 "+mark2);

}

}

class StudB extends Stud{

public StudB(String name,String regno,int mark1,int mark2)

{

this.name=name;

this.regno=regno;

this.mark1=mark1;

this.mark2=mark2;

System.out.println("Name "+name);

System.out.println("Registration number "+regno);

System.out.println("mark1 "+mark1);

System.out.println("mark2 "+mark2);

}

}

class Student<T extends Stud>{

public int getAverage(int x,int y){

int z = (x+y)/2;

return z;

}

public void maxAvg(int a1,int a2){

if(a1>a2){

System.out.println("Student S1 has got High Average");

}else{

System.out.println("Student S2 has got High Average");

}

}

}

class Main{

public static void main(String[]args){

Scanner sc=new Scanner(System.in);

Student s1=new Student<StudA>();

Student s2=new Student<StudB>();

System.out.println("Enter details of Student-1");

System.out.println("Enter Name : ");

String name1= sc.nextLine();

System.out.println("Enter Registration number: ");

String regno1=sc.nextLine();

System.out.println("Enter mark1: ");

int m11= sc.nextInt();

System.out.println("Enter mark2: ");

int m22=sc.nextInt();

sc.nextLine();

System.out.println("Enter details of Student-2");

System.out.println("Enter Name : ");

String name2= sc.nextLine();

System.out.println("Enter Registration number: ");

String regno2=sc.nextLine();

System.out.println("Enter mark1: ");

int m31= sc.nextInt();

System.out.println("Enter mark2: ");

int m32=sc.nextInt();

System.out.println("Student-1 Details: ");

StudA st1=new StudA(name1,regno1,m11,m22);

int a1=s1.getAverage(m11,m22);

System.out.println("Student-2 Details: ");

StudB st2=new StudB(name2,regno2,m31,m32);

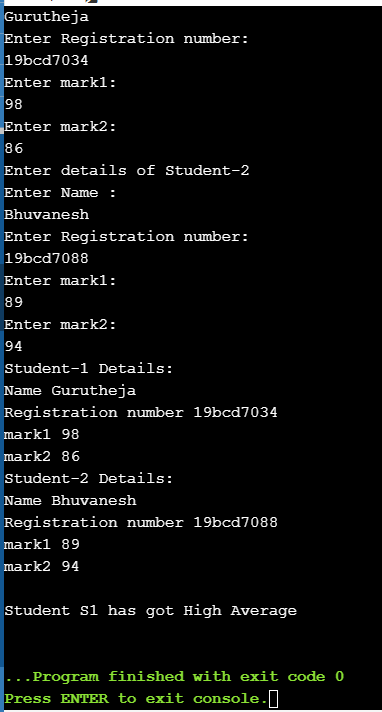
int a2=s2.getAverage(m31,m32);

System.out.println("");

s1.maxAvg(a1,a2);

}

}



1. . Create a class addMatrix to read two matrix of r1Xc1 and r2Xc2 to perform matrix addition. Check the condition for matrix addition if not possible raise an user defined exception “MatrixAdditionNotPossible” and print the message ”Matrix addition is not possible”.

Ans:

import java.util.Scanner;

class MatrixAdditionNotPossible extends Exception{

public MatrixAdditionNotPossible(){

System.out.println("Matrix addition is not possible");

}

}

class Main{

public static void main(String[]args){

Scanner sc=new Scanner(System.in);

System.out.println( "Enter number of rows in matrix1" );

int r1=sc.nextInt();

System.out.println( "Enter number of columns in matrix1" );

int c1=sc.nextInt();

System.out.println("Enter elements of matrix1");

int [][]a=new int[r1][c1];

for(int i= 0;i<r1;i++){

for(int j=0;j<c1;j++){

a[i][j]=sc.nextInt();

}

}

System.out.println( "Enter number of rows in matrix2" );

int r2=sc.nextInt();

System.out.println( "Enter number of columns in matrix2" );

int c2=sc.nextInt();

int [][]b=new int[r2][c2];

System.out.println("Enter elements of matrix2");

for(int k=0;k<r2;k++){

for(int h=0;h<c2;h++){

b[k][h]=sc.nextInt();

}

}

int c[][]= new int[r1][c1];

try

{

if(r1!=r2 ||c1!=c2)

throw new MatrixAdditionNotPossible();

else

{

for(int i=0;i<r1;i++)

{

for(int j=0;j<c1;j++)

{

c[i][j]=a[i][j]+b[i][j];

}

}

System.out.println("Elements of the resultant matrix are ");

for(int i=0;i<r1;i++) {

for(int j=0;j<c1;j++)

System.out.print(c[i][j]+" ");

System.out.println();

}

}

}

catch(MatrixAdditionNotPossible e)

{

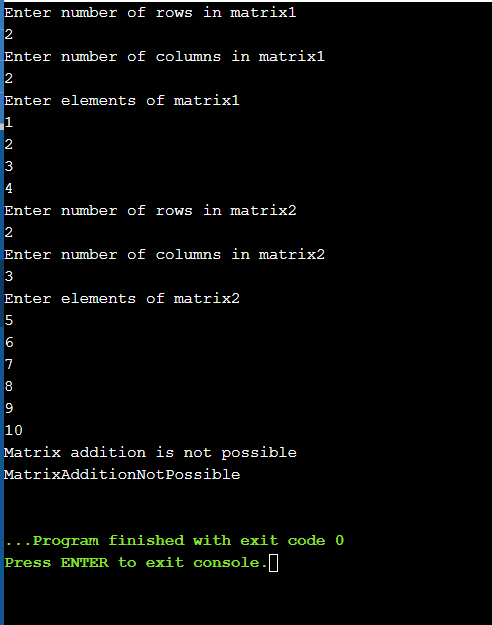
System.out.println(e);

}

}

}

Output:



2. Create a class “Student” which is the super class for the class “Mark”. Read the details of a students and calculate the entrance mark to raise a user defined exception using “Throws”.

Class : Student

Variable: Name (String)

Regno (int)

Course (String)

Method: getdetails() (\* either read the input or store it directly)

Class: Mark

Variable: Part1(int) (max 50)

Part2(int) (max 50)

Core(int) (max 100)

Total (int)(Part1+Part2+Core)

Method: getmarks() (\* either read the input or store it directly)

CalcEntranceMark

If the total is less than 100 raise a “NotEligibleException” and print “NOT ELIGIBLE”

If the total is >=100 and <150 raise a “WaitingListException” and print “WAITING LIST”

If the total is >=150 and print “ELIGIBLE”

Ans:

import java.util.Scanner;

class Student{

String name,course;

int regno;

public void getDetails(String name,int regno,String course){

this.name=name;

this.regno=regno;

this.course=course;

System.out.println("Name "+name);

System.out.println("Registration number "+regno);

System.out.println("Course "+course);

}

}

class Mark extends Student{

int part1,part2,core,total;

public void getMarks(int part1,int part2,int core){

this.part1=part1;

this.part2=part2;

this.core=core;

System.out.println("part1 "+part1);

System.out.println("part2 "+part2);

System.out.println("Core "+core);

}

public void calcentrancemark()

{

total=part1+part2+core;

System.out.println("Total "+total);

if(total>=150)

System.out.println("ELIGIBLE");

}

public void entrance1() throws NotEligibleException

{

if(total<100)

throw new NotEligibleException("NOT ELIGIBLE");

}

public void entrance2() throws WaitingListException

{

if(total>=100&total<150)

throw new WaitingListException("WAITING LIST");

}

}

class NotEligibleException extends Exception{

NotEligibleException(String s)

{

System.out.println("NOT ELIGIBLE");

}

}

class WaitingListException extends Exception{

WaitingListException(String s)

{

System.out.println("WAITING LIST");

}

}

class Main{

public static void main(String[]args){

Scanner in=new Scanner (System.in);

Mark a=new Mark();

a.getDetails("Gurutheja",197034,"CSE-DA");

a.getMarks(23,32,35);

a.calcentrancemark();

try

{

a.entrance1();

}

catch( NotEligibleException e)

{

System.out.println(e);

}

try

{

a.entrance2();

}

catch( WaitingListException e)

{

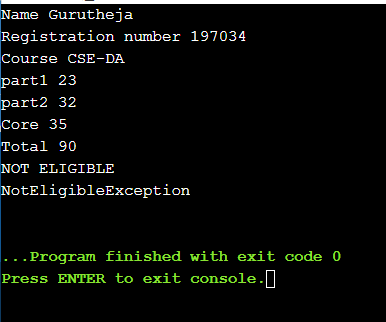
System.out.println(e);

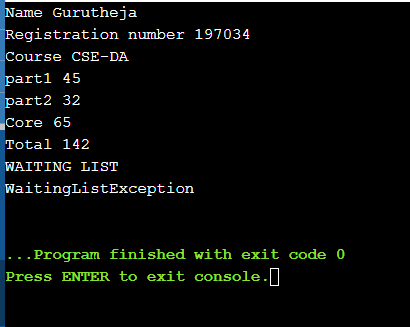
}

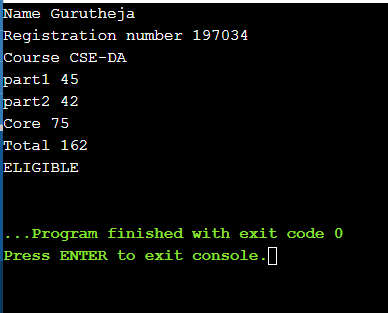
}

}

Output:







3. .Define a new exception, called ExceptionLineTooLong, that prints out the error message "The strings is too long". Write a program that reads all user entered string message and throws an exception of type ExceptionLineTooLong in the case where a string is longer than 80 characters. Handle also all exceptions that could be thrown by the program.

Ans:

import java.util.Scanner;

class ExceptionLineTooLong extends Exception{

public void ExceptionLineTooLong(){

System.out.println("The String is too long");

}

}

class Main{

public static void main(String[]args){

Scanner sc=new Scanner(System.in);

System.out.println("Enter the String");

String str=sc.nextLine();

try{

if(str.length()>=80){

throw new ExceptionLineTooLong();

}

}

catch(ExceptionLineTooLong e){

System.out.println(e);

}

}

}

Output:

Name:Thota GuruTheja Reddy

Regno:19BCD7034

Slot:L1

Lab: 8

1. Write a multi-threaded Java program to print all numbers below 100 that are both prime and Fibonacci number (some examples are 2, 3, 5, 13, etc.). Design a thread that generates prime numbers below 100 and writes them into a pipe. Design another thread that generates Fibonacci numbers and writes them to another pipe. The main thread should read both the pipes to identify numbers common to both.

Ans:

import java.util.\*;

import java.io.\*;

class Fibonacci extends Thread

{

private PipedWriter out = new PipedWriter();

public PipedWriter getPipedWriter()

{

return out;

}

public void run()

{

Thread t = Thread.currentThread();

int fibo1=0,fibo2=1,fibo=0;

while(true)

{

try

{

fibo = fibo1 + fibo2;

if(fibo>100)

{

out.close();

break;

}

out.write(fibo);

sleep(100);

}

catch(Exception e)

{

System.out.println("Fibonacci:"+e);

}

fibo1=fibo2;

fibo2=fibo;

}

}

}

class Prime extends Thread

{

private PipedWriter out1 = new PipedWriter();

public PipedWriter getPipedWriter()

{

return out1;

}

public void run()

{

Thread t= Thread.currentThread();

int prime=1;

while(true)

{

try

{

if(prime>100)

{

out1.close();

break;

}

if(isPrime(prime))

out1.write(prime);

prime++;

sleep(0);

}

catch(Exception e)

{

System.exit(0);

}

}

}

public boolean isPrime(int n)

{

int m=(int)Math.round(Math.sqrt(n));

if(n==1 || n==2)

return true;

for(int i=2;i<=m;i++)

if(n%i==0)

return false;

return true;

}

}

public class Main

{

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

{

Thread t=Thread.currentThread();

Fibonacci fibonacci = new Fibonacci();

Prime prime = new Prime();

PipedReader fpr = new PipedReader(fibonacci.getPipedWriter());

PipedReader ppr = new PipedReader(prime.getPipedWriter());

fibonacci.start();

prime.start();

int fib=fpr.read(), prm=ppr.read();

System.out.println("Numbers both Fibonacci and prime below 100 are: ");

while((fib!=-1) && (prm!=-1))

{

while(prm<=fib)

{

if(fib==prm)

System.out.println(prm);

prm=ppr.read();

}

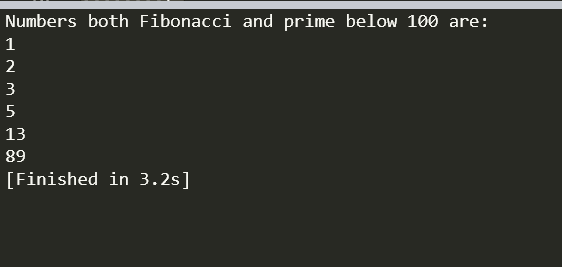
fib=fpr.read();

}

}

}

Output:



2. Create a class “A” with two methods, “arms” to check whether a given number is Armstrong number (sum of cube of individual digit is the same as the number) or not and “multiarray” ( multiply two consecutive array elements and for the last number again multiply with first number and print the value). Create 3 threads to perform the following operation simultaneously

1.multiply two consecutive array elements and print it. Eg. Input:(1 2 3 4 5) output: (2 6 12 20 5)

2. check whether the given number 153 is Armstrong number or not. 3.check whether the given number 120 is Armstrong number or not

Ans:

import java.util.Scanner;

class A extends Thread{

int n,m;

int a[];

public void arms(int n){

this.n=n;

int c=0,a,temp;

temp=n;

while(n>0){

a=n%10;

n=n/10;

c=c+(a\*a\*a);

}

if(temp==c){

System.out.println("armstrong number");

}else{

System.out.println("Not armstrong number");

}

}

public void multiarray(int a[]){

this.a=a;

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

if(i==a.length-1){

m=a[i]\*a[0];

}else{

m=a[i]\*a[i+1];

}

System.out.println(m);

}

}

}

class Thread1 extends A{

int b[];

Thread1(int[]b){

this.b=b;

}

public void run(){

super.multiarray(b);

}

}

class Thread2 extends A{

int g;

Thread2(int g){

this.g=g;

}

public void run(){

super.arms(g);

}

}

class Main{

public static void main(String[] args) {

int[]f={1,3,5,6};

Thread1 t1 = new Thread1(f);

Thread2 t2 = new Thread2(153);

Thread2 t3 = new Thread2(120);

t2.setPriority(10);

t3.setPriority(1);

t1.start();

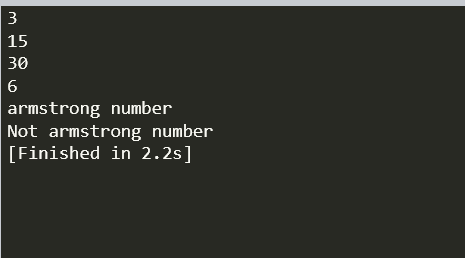
t2.start();

t3.start();

}

}

Output:



1. 3. Create a class with method to count odd and even number in 3X3 matrix. Create two threads to perform the operation concurrently. Display the odd and even number count in the matrix.(use Runnable interface)

Original array:

4 1 3

3 5 7

8 2 6

Frequency of odd numbers: 5

Frequency of even numbers: 4

Ans:

import java.util.Scanner;

class counte implements Runnable{

int arr[][];

public counte(int a[][]) {

this.arr=a;

}

public void run() {

int c=0;

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

for(int j=0;j<arr.length;j++) {

if(arr[i][j]%2==0) {

c++;

}

}

}

System.out.println("Total frequency of even numbers is "+c);

}

}

class counto implements Runnable{

int arr[][];

public counto(int a[][]) {

this.arr=a;

}

public void run() {

int c=0;

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

for(int j=0;j<arr.length;j++) {

if(arr[i][j]%2!=0) {

c++;

}

}

}

System.out.println("Total frequency of odd numbers is "+c);

}

}

public class Main

{

public static void main(String[] args) {

Scanner in=new Scanner(System.in);

System.out.println("Enter the size of matrix");

int n=in.nextInt();

int a[][]= new int[n][n];

System.out.println("Enter elements in matrix");

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

for (int j=0;j<n;j++) {

a[i][j]=in.nextInt();

}

}

counto o=new counto(a);

counte e=new counte(a);

Thread t1=new Thread(o);

t1.start();

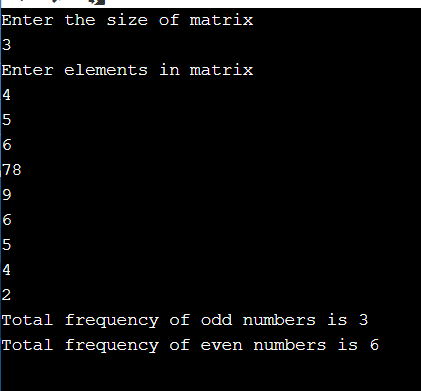
Thread t2=new Thread(e);

t2.start();

}

}

Output:



Name: Thota Gurutheja Reddy

Regno: 19BCD7034

Lab-9

1. Consider the track line in railways. If multiple trains trying to access the same line it should not allow because it will lead to collision. Now write a java program to schedule the track for the train. Create 3 threads objects for the class Train for 3 trains and schedule the Line by calling the getLine() (Synchronized method on the same Object but only one thread will be able to execute it at a time) in the Line class.

Ans:

class Line

{

synchronized public void getLine()

{

for (int i = 1; i < 4; i++)

{

System.out.println(i);

try

{

Thread.sleep(400);

}

catch (Exception e)

{

System.out.println(e);

}

}

}

}

class Train extends Thread

{

Line line = new Line();

Train(Line line)

{

this.line = line;

}

public void run()

{

line.getLine();

}

}

class Main

{

public static void main(String[] args)

{

Line obj = new Line();

Train train1 = new Train(obj);

Train train2 = new Train(obj);

Train train3 = new Train(obj);

train1.start();

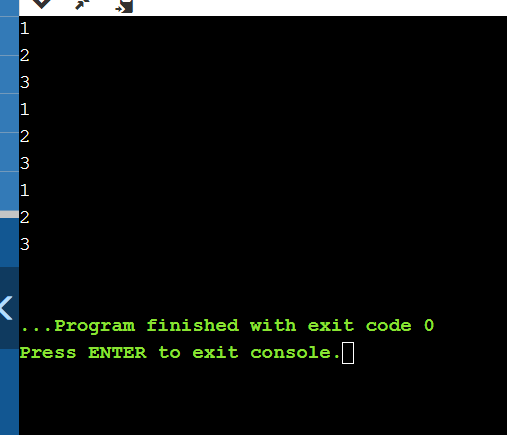
train2.start();

train3.start();

}

}

Output:



2. Create a class serieseven with two methods Series and Printeven. Series method is synchronized to print the series by reading n and generated series should have the value <2000 and Printeven to find the sum of the series generated is even. Create two threads to perform series generation and check whether sum is even or not.

Sample input and output n=2

2

8

18

32

50

100

400

900

1600

5610Sum is EVEN

n=3

3

12

27

48

75

100

400

900

1600

Sum is odd

Ans:

import java.util.\*;

class SeriesSeven extends Thread{

Random r= new Random();

int sum,p;

public void series(){

Scanner sc= new Scanner(System.in);

int n= sc.nextInt();

sum=n;

System.out.println(n);

for(int i=0;i<10;i++){

p=r.nextInt(2000)+n;

System.out.println(p);

sum=sum+p;

}

}

public void printeven(){

System.out.println(sum);

if(sum%2==0){

System.out.println("Sum is even");

}

else {

System.out.println("Sum is odd");

}

}

}

class Guru extends Thread{

SeriesSeven s=new SeriesSeven();

public void run(){

s.series();

s.printeven();

}

}

class Main{

public static void main (String[] args) {

Guru t1 = new Guru();

t1.start();

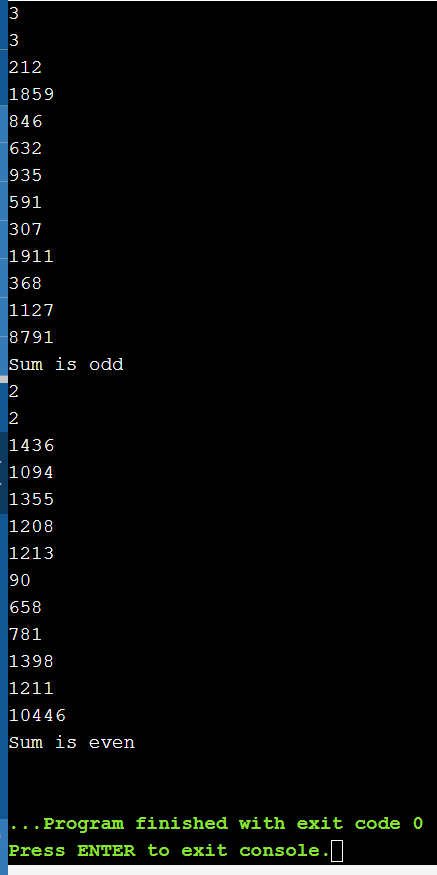
Guru t2 = new Guru();

t2.start();

}

}

Output:



3. a.Create a CeaserCipher class to perform substitution and reverse substitution of characters of a message.

- mEncryption method - substitute a character with another charcter of alphabet.

- mDecryption method - similar to mEncryption method but it performs in reverse.

Each character of message is considered as numeric value with the following mapping:**a-z to 0-25**, respectively.

The mEncryption method replaces each character of the message with another character by using the following formula:(N(ch)+k)%26, where N(ch) means Numeric value of a character 'ch', k means key value 0<=k<=25.

The mDecryption method substitutes each character with the following formula: (N(ch)-k)%26.

Inputs to each method is a message and a key and output is substituted message printed on console character by character.

(Ex: Input to mEncryption is: rama and 25 and output is: qzlz ;

Input to mDecryption is: qzlz and 25 and output is: rama )

Create a TestCeaserCipher class to test mEncryption & mDecryption methods.

b. Jennifer comes with a message "gdhrzfnncanx". She wants to perform reverse substitution using mDecryption method but not aware of key 'k'. To help her, develop a multithreaded program to create separate thread for each possible key 'k' and print all reverse substitutions. Do necessary changes to CeaserCipher class and provide synchronization for threads if the output from threads are mixed.

Ans:

a.

import java.util.Scanner;

class CeaserCipher

{

public static String a = "abcdefghijklmnopqrstuvwxyz";

public static String mEncryption(String p, int key)

{

p = p.toLowerCase();

String c = "";

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

{

int cp = a.indexOf(p.charAt(i));

int kv = (key + cp) % 26;

char rv = a.charAt(kv);

c += rv;

}

return c;

}

public static String mDecryption(String c, int key)

{

c = c.toLowerCase();

String p = "";

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

{

int cp = a.indexOf(c.charAt(i));

int kv = (cp - key) % 26;

if (kv < 0)

{

kv = a.length() + kv;

}

char rv = a.charAt(kv);

p += rv;

}

return p;

}

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in);

System.out.println("Enter the Message to encrypt");

String m =sc.nextLine();

System.out.println("Enter the Key for encryption");

int key = sc.nextInt();

String c=mEncryption(m, key);

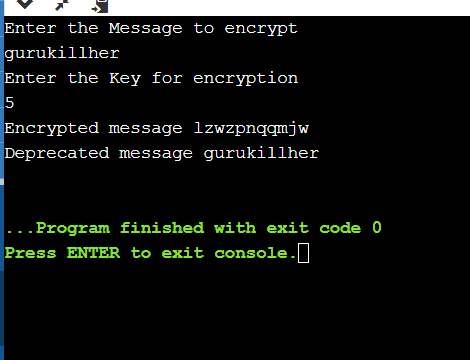
System.out.println("Encrypted message " + c);

System.out.println("Deprecated message " + mDecryption(c, key));

}

}

Output:



b.

public class Main extends Thread

{

public static String m = "gdhrzfnncanx";

public static String a = "abcdefghijklmnopqrstuvwxyz";

static class Decrypt extends Thread

{

int key;

Decrypt(int key)

{

this.key=key;

}

String l = m.toLowerCase();

String p = "";

public void run(){

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

{

int cp = a.indexOf(l.charAt(i));

int kv = (cp - key) % 26;

if (kv < 0)

{

kv = a.length() + kv;

}

char r = a.charAt(kv);

p += r;

}

System.out.println("Message for key "+key+ " is " + p);

}

}

public static void main(String[] args)

{

Main.Decrypt [] c=new Main.Decrypt[26];

for(int k=0;k<26;k++){

c[k]=new Main.Decrypt(k);

}

for(int y=0;y<26;y++){

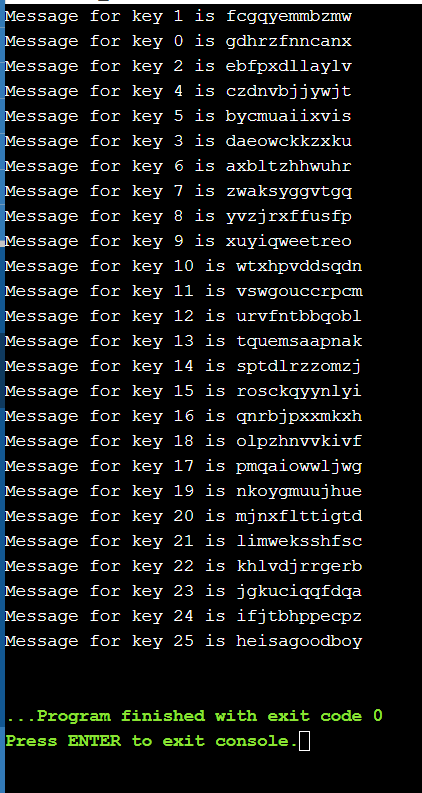
c[y].start();

}

}

}

Output:



Name: Thota GuruTheja Reddy

Regno: 19BCD7034

Lab: 10

1. Consider two friends Herbert and Schildt are chatting and discussing about next day exam preparation. Create a class chat with two methods Ask and Reply to perform this. Create two threads Herbert and Schidlt using Runnable Interface. Implement inter thread communication for the scenario and exchange the following messages

Herbert : Hi

Schildt : Hello

Herbert : Tomorrow do you have exam

Schildt : yes

Herbert : have you prepared?

Schildt : ya i am preparing

Herbert : All the best!

Schildt : Thank you

Ans:

class Chat {

boolean flag = false;

public synchronized void Ask(String msg) {

if (flag) {

try {

wait();

} catch (InterruptedException e) {

System.out.println(e);

}

}

System.out.println(msg);

flag = true;

notify();

}

public synchronized void Reply(String msg) {

if (!flag) {

try {

wait();

} catch (InterruptedException e) {

System.out.println(e);

}

}

System.out.println(msg);

flag = false;

notify();

}

}

class Herbert implements Runnable {

Chat m;

String[] s1 = { "Hi", "Tomorrow do you have exam", "Have you prepared?","All the best!" };

public Herbert(Chat m1) {

this.m = m1;

new Thread(this, "Ask").start();

}

public void run() {

for (int i = 0; i < s1.length; i++) {

m.Ask(s1[i]);

}

}

}

class Schildt implements Runnable {

Chat m;

String[] s2 = { "Hello", "Yes", "ya I am preparing", "Thank you" };

public Schildt(Chat m2) {

this.m = m2;

new Thread(this, "Reply").start();

}

public void run() {

for (int i = 0; i < s2.length; i++) {

m.Reply(s2[i]);

}

}

}

public class Main {

public static void main(String[] args) {

Chat m = new Chat();

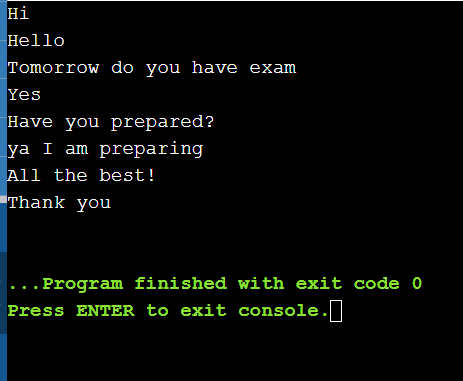
new Herbert(m);

new Schildt(m);

}

}

Output:



2. Apply Inter thread communication to solve the Producer-Consumer problem with a common or shared bounded buffer(Queue) holding up to 5 elements.

The producer consumer problem is a synchronization problem. There is a fixed size buffer and the producer produces items and enters them into the buffer. The consumer removes the items from the buffer and consumes them.

A producer should not produce items into the buffer when the consumer is consuming an item from the buffer and vice versa. So the buffer should only be accessed by the producer or consumer at a time.

Whenever buffer is filled up and no more space to add the element into the queue(buffer) producer has to wait until the buffer is emptied by consumer. Whenever the buffer is empty and no more items are available for consumption the consumer should wait for producer to produce elements. Write a solution for N elements, where N is multiple of 5 other than 0.

Ans:

import java.util.concurrent.\*;

class Producer extends Thread {

private BlockingQueue<Integer> sharedQueue;

public Producer(BlockingQueue<Integer> aQueue) {

super("PRODUCER");

this.sharedQueue = aQueue;

}

public void run() {

for (int i = 1; i < 6; i++) {

try {

System.out.println(getName() + " produced " + i);

sharedQueue.put(i);

Thread.sleep(200);

} catch (InterruptedException e) {

System.out.println(e);

}

}

}

}

class Consumer extends Thread {

private BlockingQueue<Integer> sharedQueue;

public Consumer(BlockingQueue<Integer> aQueue) {

super("CONSUMER");

this.sharedQueue = aQueue;

}

public void run() {

try {

while (true) {

Integer item = sharedQueue.take();

System.out.println(getName() + " consumed " + item);

}

} catch (InterruptedException e) {

System.out.println(e);

}

}

}

public class Main {

public static void main(String[] args) {

BlockingQueue<Integer> sharedQ = new LinkedBlockingQueue<Integer>();

Producer p = new Producer(sharedQ);

Consumer c = new Consumer(sharedQ);

p.start();

c.start();

}

}

Output:



Name: Thota GuruTheja Reddy

Regno: 19BCD7034

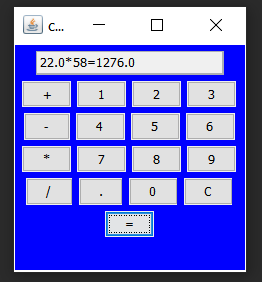
Lab: 11

1.Design a calculator to perform the operation (+,-,\*,/) on both integer and float values.

Ans:

package com.company;  
import java.awt.event.\*;  
import javax.swing.\*;  
import java.awt.\*;  
class calculator extends JFrame implements ActionListener {  
 static JFrame *f*;  
 static JTextField *l*;  
 String s0, s1, s2;  
 calculator()  
 {  
 s0 = s1 = s2 = "";  
 }  
 public static void main(String args[])  
 {  
 *f* = new JFrame("calculator");  
  
 try {  
 UIManager.*setLookAndFeel*(UIManager.*getSystemLookAndFeelClassName*());  
 }  
 catch (Exception e) {  
 System.*err*.println(e.getMessage());  
 }  
 calculator c = new calculator();  
 *l* = new JTextField(16);  
 *l*.setEditable(false);  
 JButton b0, b1, b2, b3, b4, b5, b6, b7, b8, b9, ba, bs, bd, bm, be, beq, beq1;  
 b0 = new JButton("0");  
 b1 = new JButton("1");  
 b2 = new JButton("2");  
 b3 = new JButton("3");  
 b4 = new JButton("4");  
 b5 = new JButton("5");  
 b6 = new JButton("6");  
 b7 = new JButton("7");  
 b8 = new JButton("8");  
 b9 = new JButton("9");  
 beq1 = new JButton("=");  
 ba = new JButton("+");  
 bs = new JButton("-");  
 bd = new JButton("/");  
 bm = new JButton("\*");  
 beq = new JButton("C");  
 be = new JButton(".");  
 JPanel p = new JPanel();  
 bm.addActionListener(c);  
 bd.addActionListener(c);  
 bs.addActionListener(c);  
 ba.addActionListener(c);  
 b9.addActionListener(c);  
 b8.addActionListener(c);  
 b7.addActionListener(c);  
 b6.addActionListener(c);  
 b5.addActionListener(c);  
 b4.addActionListener(c);  
 b3.addActionListener(c);  
 b2.addActionListener(c);  
 b1.addActionListener(c);  
 b0.addActionListener(c);  
 be.addActionListener(c);  
 beq.addActionListener(c);  
 beq1.addActionListener(c);  
 p.add(*l*);  
 p.add(ba);  
 p.add(b1);  
 p.add(b2);  
 p.add(b3);  
 p.add(bs);  
 p.add(b4);  
 p.add(b5);  
 p.add(b6);  
 p.add(bm);  
 p.add(b7);  
 p.add(b8);  
 p.add(b9);  
 p.add(bd);  
 p.add(be);  
 p.add(b0);  
 p.add(beq);  
 p.add(beq1);  
 p.setBackground(Color.*blue*);  
 *f*.add(p);  
 *f*.setSize(200, 220);  
 *f*.show();  
 }  
 public void actionPerformed(ActionEvent e)  
 {  
 String s = e.getActionCommand();  
 if ((s.charAt(0) >= '0' && s.charAt(0) <= '9') || s.charAt(0) == '.') {  
 if (!s1.equals(""))  
 s2 = s2 + s;  
 else  
 s0 = s0 + s;  
 *l*.setText(s0 + s1 + s2);  
 }  
 else if (s.charAt(0) == 'C') {  
 s0 = s1 = s2 = "";  
 *l*.setText(s0 + s1 + s2);  
 }  
 else if (s.charAt(0) == '=') {  
 double te;  
 if (s1.equals("+"))  
 te = (Double.*parseDouble*(s0) + Double.*parseDouble*(s2));  
 else if (s1.equals("-"))  
 te = (Double.*parseDouble*(s0) - Double.*parseDouble*(s2));  
 else if (s1.equals("/"))  
 te = (Double.*parseDouble*(s0) / Double.*parseDouble*(s2));  
 else  
 te = (Double.*parseDouble*(s0) \* Double.*parseDouble*(s2));  
 *l*.setText(s0 + s1 + s2 + "=" + te);  
 s0 = Double.*toString*(te);  
 s1 = s2 = "";  
 }  
 else {  
 if (s1.equals("") || s2.equals(""))  
 s1 = s;  
 else {  
 double te;  
 if (s1.equals("+"))  
 te = (Double.*parseDouble*(s0) + Double.*parseDouble*(s2));  
 else if (s1.equals("-"))  
 te = (Double.*parseDouble*(s0) - Double.*parseDouble*(s2));  
 else if (s1.equals("/"))  
 te = (Double.*parseDouble*(s0) / Double.*parseDouble*(s2));  
 else  
 te = (Double.*parseDouble*(s0) \* Double.*parseDouble*(s2));  
 s0 = Double.*toString*(te);  
 s1 = s;  
 s2 = "";  
 }  
 *l*.setText(s0 + s1 + s2);  
 }  
 }  
}

Output:

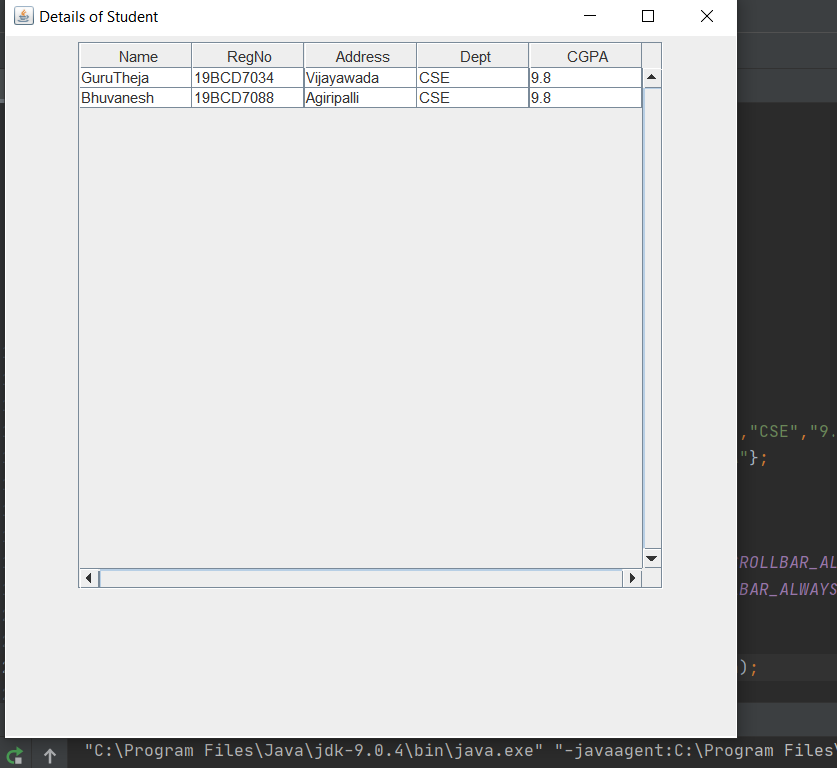


2. Write a swing program to get the details (Name, Regno, address, Dept, CGPA) of students and display in table format.

Ans:

package com.company;  
import java.awt.\*;  
import javax.swing.\*;  
class Main {  
 JFrame f;  
 JTable table;  
 JScrollPane sp;  
 public Main() {  
 f = new JFrame("Details of Student");  
 f.setSize(600,600);  
 f.setLayout(new FlowLayout());  
 f.setDefaultCloseOperation(f.*EXIT\_ON\_CLOSE*);  
 String data[][] = { {"GuruTheja","19BCD7034","Vijayawada","CSE","9.8" },{"Bhuvanesh","19BCD7088","Agiripalli","CSE","9.8"}};  
 String column[] = { "Name","RegNo","Address","Dept","CGPA"};  
 table = new JTable(data,column);  
 f.add(table);  
 sp = new JScrollPane(table);  
 sp.setHorizontalScrollBarPolicy(JScrollPane.*HORIZONTAL\_SCROLLBAR\_ALWAYS*);  
 sp.setVerticalScrollBarPolicy(JScrollPane.*VERTICAL\_SCROLLBAR\_ALWAYS*);  
 f.add(sp);  
 JTextArea textArea = new JTextArea(20, 20);  
 JScrollPane scrollableTextArea = new JScrollPane(textArea);  
 f.setVisible(true);  
 }  
 public static void main(String[] args) {  
 new Main();  
 }  
}

Output:



Name: Thota Gurutheja Reddy

Reg No: 19BCD7034

Lab 12

1. Create your own menu and display it using javaFX

Ans:

import javafx.application.Application;

import javafx.scene.Group;

import javafx.scene.Scene;

import javafx.scene.control.Menu;

import javafx.scene.control.MenuBar;

import javafx.scene.control.MenuItem;

import javafx.scene.paint.Color;

import javafx.stage.Stage;

public class MenuBar\_1 extends Application {

public void start(Stage stage) {

Menu fileMenu = new Menu("Documents");

MenuItem item1 = new MenuItem("Office Documents");

MenuItem item2 = new MenuItem("personal Documents");

fileMenu.getItems().addAll(item1, item2);

MenuBar menuBar = new MenuBar(fileMenu);

menuBar.setTranslateX(200);

menuBar.setTranslateY(20);

Group root = new Group(menuBar);

Scene scene = new Scene(root, 595, 200, Color.BEIGE);

stage.setTitle("Menu");

stage.setScene(scene);

stage.show();

}

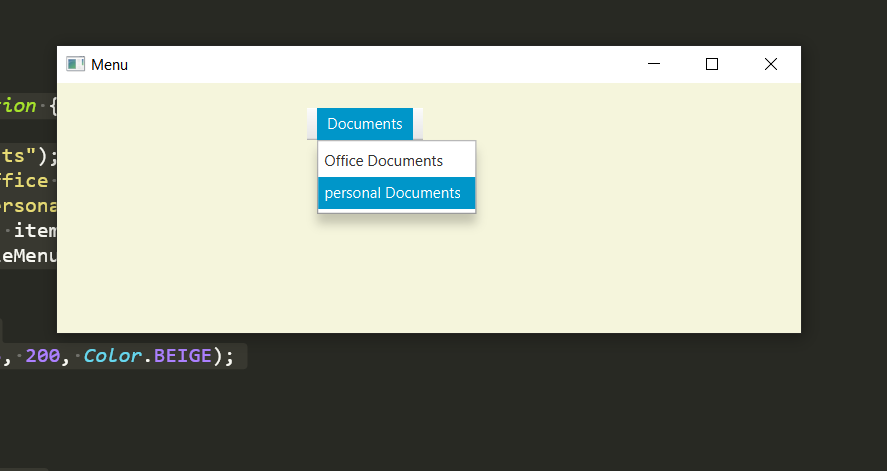
public static void main(String args[]){

launch(args);

}

}

Output:



2. Create the tree below using javaFX

Ans: import javafx.application.Application;

import javafx.geometry.Insets;

import javafx.scene.Group;

import javafx.scene.Scene;

import javafx.scene.control.TreeItem;

import javafx.scene.control.TreeView;

import javafx.scene.layout.VBox;

import javafx.scene.paint.Color;

import javafx.stage.Stage;

public class Tree extends Application {

public void start(Stage stage) {

TreeItem root1 = new TreeItem("Sales Department");

TreeItem item1 = new TreeItem("Ethan Williams");

TreeItem item2 = new TreeItem("Michael Brown");

TreeItem item3 = new TreeItem("Anna Balck");

root1.getChildren().addAll(item1, item2, item3);

TreeItem root2 = new TreeItem("IT Support");

TreeItem item4 = new TreeItem("Mike Graham");

TreeItem item5 = new TreeItem("Judy Mayer");

TreeItem item6 = new TreeItem("Grefory Smith");

root2.getChildren().addAll(item4, item5, item6);

TreeItem root3 = new TreeItem("Accounts Department");

TreeItem item7 = new TreeItem("Jacob Smith");

TreeItem item8 = new TreeItem("Isabella Jhonson");

TreeItem item9 = new TreeItem("Jhonson");

root3.getChildren().addAll(item7, item8, item9);

TreeItem<String> base = new TreeItem<String>("MyCompany Human Resources");

base.setExpanded(true);

base.getChildren().addAll(root1, root2, root3);

TreeView view = new TreeView(base);

view.setPrefHeight(300);

VBox pane = new VBox(10);

pane.setPadding(new Insets(5, 5, 5, 50));

pane.getChildren().addAll(view);

Group node = new Group(pane);

Scene scene = new Scene(node, 595, 320, Color.BEIGE);

stage.setTitle("Tree");

stage.setScene(scene);

stage.show();

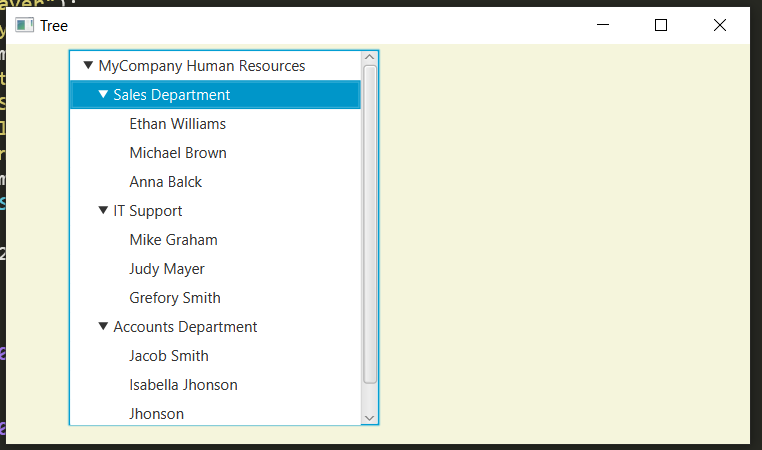
}

public static void main(String args[]){

launch(args);

}

}



Name: Thota GuruTheja Reddy

Reg No: 19BCD7034

PROJECT

Create an application using java Swings.

Ans:

import java.awt.\*;

import java.awt.event.\*;

import javax.swing.\*;

class Pin

{

JFrame optionPane\_f;

Pin()

{

optionPane\_f=new JFrame();

JOptionPane.showInputDialog(optionPane\_f,"Enter Pin");

}

}

class MyWindowAdapter extends WindowAdapter{

public void windowClosing(WindowEvent we) {

System.exit(0);

}

}

class DataType {

JTextField t1,t2,t3,t4,t5,Total;

JButton m;

}

class ClickMeListener extends DataType implements ActionListener{

public void actionPerformed(ActionEvent o) {

String [] s = new String [5];

int [] a = new int [5];

s[0] = t1.getText();

s[1] = t2.getText();

s[2] = t3.getText();

s[3] = t4.getText();

s[4] = t5.getText();

for(int i = 0;i<5;i++) {

if (s[i].equals("")) {

a[i]=0;

}

else {

a[i]= Integer.parseInt(s[i]);

}

}

String result=String.valueOf(a[0]\*2000+a[1]\*500+a[2]\*100+a[3]\*50+a[4]\*10);

Total.setText(result);

}

}

public class Denomination {

public static void main(String args[]) {

new Pin();

ClickMeListener y = new ClickMeListener();

JFrame f = new JFrame("Denomination calculator");

f.setSize(400,360);

Panel p = new Panel();

p.setLayout(new GridLayout(6,2));

JLabel l1 = new JLabel("Number of 2000 Rs notes");

p.add(l1);

y.t1 = new JTextField();

p.add(y.t1);

JLabel l2 = new JLabel("Number of 500 Rs notes");

p.add(l2);

y.t2 = new JTextField();

p.add(y.t2);

JLabel l3 = new JLabel("Number of 100 Rs notes");

p.add(l3);

y.t3 = new JTextField();

p.add(y.t3);

JLabel l4 = new JLabel("Number of 50 Rs notes");

p.add(l4);

y.t4 = new JTextField();

p.add(y.t4);

JLabel l5 = new JLabel("Number of 10 Rs notes");

p.add(l5);

y.t5 = new JTextField();

p.add(y.t5);

y.m = new JButton("Total Denomination");

y.m.addActionListener(y);

p.add(y.m);

y.Total = new JTextField();

p.add(y.Total);

f.add(p);

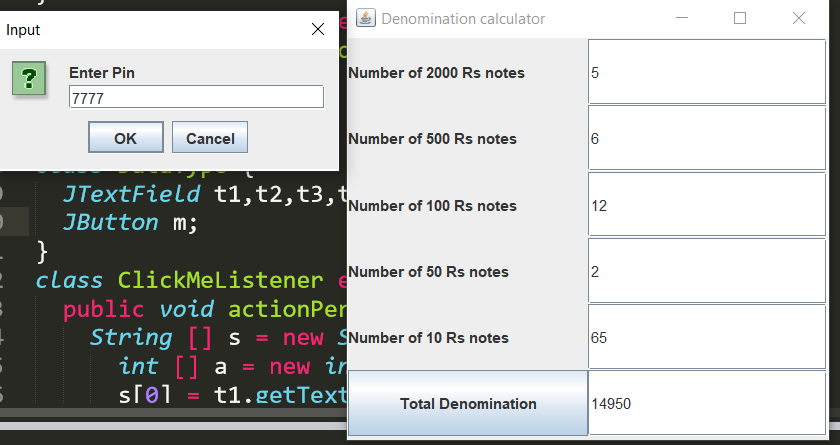
f.addWindowListener(new MyWindowAdapter());

f.setVisible(true);

}

}

Output:



 19bcd7088

1.a.

Create an application named Percentages whose main() method holds two double variables. Assign values to the variables. Pass both variables to a method named computePercent() that displays the two values and the value of the first number as a percentage of the second one. For example, if the numbers are 2.0 and 5.0, the method should display a statement similar to “2.0 is 40 percent of 5.0.” Then call the method a second time, passing thevalues in reverse order. Save the application as Percentages.java.

public class Percentages{

static void computePercent(int a, int b){

System.out.println(a+" is "+(a\*100/b)+"% of "+b);

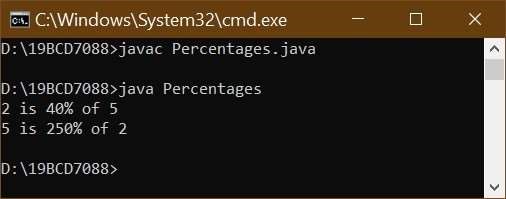
}

public static void main(String[] args) {

int a = 2; int b = 5; computePercent(2,5); computePercent(5,2);

}

}



1.b.

Modify the Percentages class to accept the values of the two doubles from auser at the keyboard. Save the file as Percentages2.java.

import java.util.\*;

public class Percentages2{

static void computePercent(double a, double b){

System.out.println(a+" is "+(a\*100/b)+"% of "+b);

}

public static void main(String[] args) {

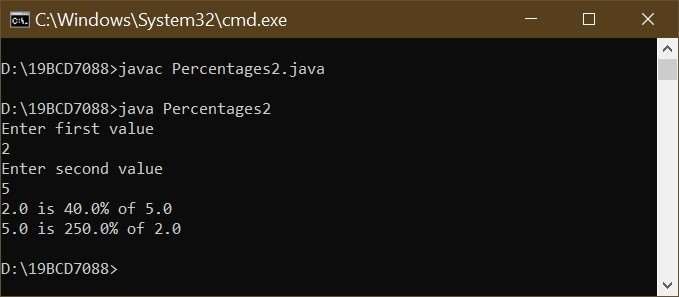
Scanner sc = new Scanner(System.in); System.out.println("Enter first value"); double a = sc.nextDouble();

System.out.println("Enter second value"); double b = sc.nextDouble(); computePercent(a,b);

computePercent(b,a);

}

}



2.

There are 12 inches in a foot and 3 feet in a yard. Create a class named InchConversion. Its main() method accepts a value in inches from a user at the keyboard, and in turn passes the entered value to two methods. One converts the value from inches to feet, and the other converts the same value from inches to yards. Each method displays the results with appropriate explanation. Save the application as InchConversion.java.

import java.util.\*; public class InchConversion{ static void InchtoFeet(double a){

System.out.println("The measurement of "+ a +" inches in feet is "+ a/12);

}

static void InchtoYards(double b){

System.out.println("The measurement of "+ b +" inches in yards is "+ b/36);

}

public static void main(String[] args) {

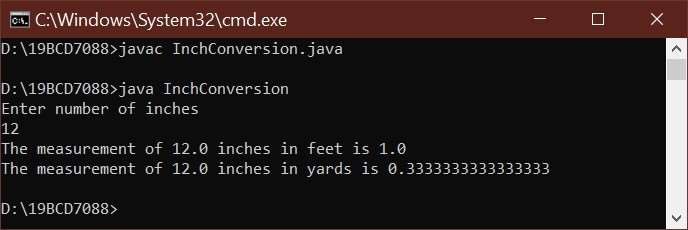
Scanner sc = new Scanner(System.in); System.out.println("Enter number of inches"); double a = sc.nextDouble();

InchtoFeet(a);

InchtoYards(a);

}

}



3.

Assume that a gallon of paint covers about 350 square feet of wall space. Create an application with a main() method that prompts the user for the length, width, and height of a rectangular room. Pass these three values to a method that does the following:

•• Calculates the wall area for a room

•• Passes the calculated wall area to another method that calculates and returns the number of gallons of paint needed

•• Displays the number of gallons needed

•• Computes the price based on a paint price of $32 per gallon, assuming that the painter can buy any fraction of a gallon of paint at the same price as a whole gallon

•• Returns the price to the main() method

The main() method displays the final price. For example, the cost to paint a 15-by-20-foot room with 10-foot ceilings is $64. Save the application as PaintCalculator.java.

import java.util.\*;

public class PaintCalculator{ static double area (double l,double b,double h){

return ((2\*l\*h)+(2\*b\*h));

}

static double gallons(double q){

return q/350;

}

static double price(double g){ return g\*32;

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter length of the room"); double l = sc.nextDouble();

System.out.println("Enter of width the room"); double b = sc.nextDouble();

System.out.println("Enter height of the room"); double h = sc.nextDouble(); double a = area(l,b,h); double gallon = gallons(a); double rate = price(gallon);

System.out.println("The cost to paint a " + l + "-by-" + b + "-foot room with " + h + "-

foot ceilings is $" + rate + ".");

}

}



4.

Herbert’s Home Repair estimates each job cost as the cost of materials plus $35 per hour while on the job, plus $12 per hour for travel time to the job site. Create a class that contains a main() method that prompts the user for the name of a job (for example, Smith bathroom remodel), the cost of materials, the number of hours of work required, and the number of hours travel time. Pass the numeric data to a method that computes estimate for the job and returns the computed value to the main() method where the job name and estimated price are

displayed. Save the program as JobPricing.java.

import java.util.\*; public class JobPricing{

static double computetion (double m,double h,double t){

return m+(h\*35)+(t\*12);

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter job title");

String str = sc.nextLine();

System.out.println("Enter the cost of material");

double m = sc.nextDouble();

System.out.println("Enter number of hours he/she worked");

double h = sc.nextDouble();

System.out.println("Enter number of hours he/she traveled");

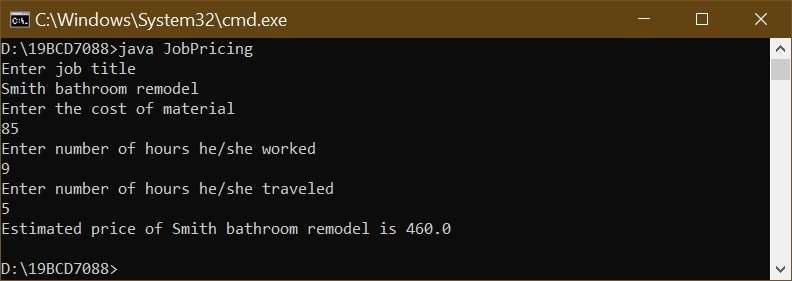
double t = sc.nextDouble();

double n = computetion(m,h,t);

System.out.println("Estimated price of " + str + " is " + n);

}

}



5.a.

Create a class named Student that has fields for an ID number, number of credit hours earned, and number of points earned. (For example, many schools compute grade point averages based on a scale of 4, so a three-credit-hour class in which a student earns an A is worth 12 points.) Include methods to assign values to all fields. A Student also has a field for grade point average. Include a method to compute the grade point average field by dividing points by credit hours earned. Write methods to display the values in each Student field. Save this class as Student.java.

public class Student{

static String str; static double credit,points; static void setId(String s){

str=s;

}

static void setCredit(int c){

credit=c;

}

static void setPoints(int p){ points=p;

}

static void average(){

double a = points/credit;

System.out.println("The Student of Id " + str + " has " + credit + " credits with " +

points + " points with average " + a + ".");

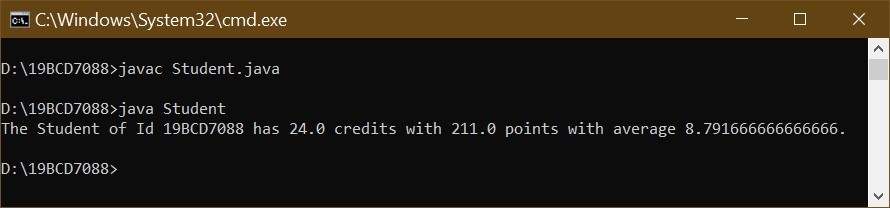
}

public static void main(String[] args) {

setId("19BCD7088"); setCredit(24); setPoints(211); average();

}

}



5.b.

Write a class named ShowStudent that instantiates a Student object from the class you created and assign values to its fields. Compute the Student grade point average, and then display all the values associated with the Student. Save the application as ShowStudent.java.

class Student{ String str;

double credit,points; void setId(String s){

str=s;

}

void setCredit(int c){

credit=c;

}

void setPoints(int p){ points=p;

}

void average(){

double a = points/credit;

System.out.println("The Student of Id " + str + " has " + credit + " credits with " +

points + " points with average " + a + ".");

}

}

public class ShowStudent{

public static void main(String[] args) { Student st = new Student(); st.setId("19BCD7110"); st.setCredit(24); st.setPoints(200);

st.average();

}

}



5.c.

Create a constructor for the Student class you created. The constructor should initialize each Student’s ID number to 9999, his or her points earned to 12, and credit hours to 3 (resulting in a grade point average of 4.0). Write a program that demonstrates that the constructor works by instantiating an object and displaying the initial values. Save the application as ShowStudent2.java.

class Student{ String str;

double credit,points;

Student(String s,int c,int p){

str=s; credit=c; points=p;

}

void mean(){ double a = points/credit;

System.out.println("The Student of Id " + str + " has " + credit + " credits with " +

points + " points with average " + a + ".");

}

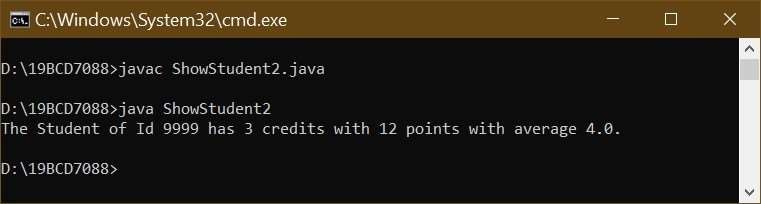
}

public class ShowStudent2{

public static void main(String[] args) { Student st = new Student("9999",3,12); st.mean();

}

}



6.a.

Create a class named Lease with fields that hold an apartment tenant’s name, apartment number, monthly rent amount, and term of the lease in months. Include a constructor that initializes the name to “XXX”, the apartment number to 0, the rent to 1000, and the term to 12. Also include methods to get and set each of the fields. Include a nonstatic method named addPetFee() that adds $10 to the monthly rent value and calls a static method named explainPetPolicy() that explains the pet fee. Save the class as Lease.java.

class Data{ String str; int a,r,m;

Data(String str,int a ,int r, int m){

this.str=str; this.a=a; this.r=r; this.m=m;

}

void setName(String s){

this.str=s;

}

void setAptNum(int a){

this.a=a;

}

void setRent(int r){

this.r=r;

}

void setMonth(int m){

this.m=m;

}

void getName(){

System.out.println("Apartment tenant’s name " + this.str);

}

void getAptNum(){

System.out.println("Apartment Number " + this.a);

}

void getMonth(){

System.out.println("Term of the lease in months " + this.m);

}

void getRent(){

System.out.println("Monthly rent amount $" + this.r);

}

void addPetFee(){ this.r= this.r+10;

}

static void explainPetPolicy(){

System.out.println("If there is any pet then 10$ will be added to rent.");

}

}

public class Lease{

public static void main(String[] args) { Data d = new Data("XXXZ",0,1000,12);

d.getName();

d.getAptNum();

d.getMonth();

d.getRent();

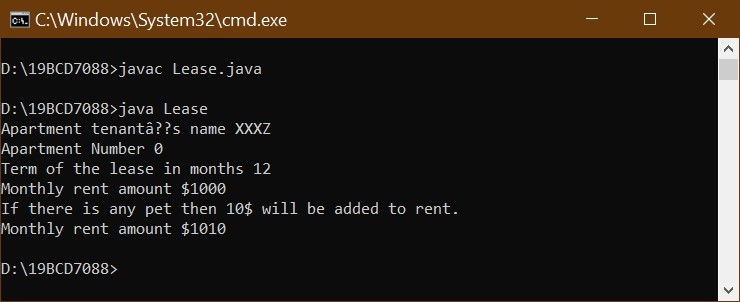
d.explainPetPolicy();

d.addPetFee();

d.getRent();

}

}



6.b.

Create a class named TestLease whose main() method declares four Lease objects. Call a getData() method three times. Within the method, prompt a user for values for each field for a Lease, and return a Lease object to the main() method where it is assigned to one of main()’s Lease objects. Do not prompt the user for values for the fourth Lease object, but let it continue to hold the default values. Then, in main(), pass one of the Lease objects to a showValues() method that displays the data. Then call the addPetFee() method using the passed Lease object and confirm that the fee explanation statement is displayed. Next, call the showValues() method for the Lease object again and confirm that the pet fee has been added to the rent. Finally, call the showValues() method with each of the other three objects; confirm that two hold the values you supplied as input and one holds the constructor default values. Save the application as TestLease.java.

import java.util.\*;

class Lease{ String str; int a,r,m;

void setName(String s){

this.str=s;

}

void setAptNum(int a){

this.a=a;

}

void setRent(int r){

this.r=r;

}

void setMonth(int m){

this.m=m;

}

void getName(){

System.out.println("Apartment tenant’s name is " + this.str);

}

void getAptNum(){

System.out.println("Apartment Number is " + this.a);

}

void getMonth(){

System.out.println("Term of the lease in months " + this.m);

}

void getRent(){

System.out.println("Monthly rent amount is $" + this.r);

}

void addPetFee(){

this.r= this.r+10;

}

Lease getData(){

Scanner sc = new Scanner(System.in);

Lease temp = new Lease();

System.out.println("Enter Apartment tenant’s name");

String e = sc.nextLine();

System.out.println("Enter Apartment Number"); int k =sc.nextInt();

System.out.println("Enter Number of Months"); int o = sc.nextInt();

System.out.println("Enter Monthly rent amount");

int l = sc.nextInt();

temp.str=e; temp.a=k; temp.r=l; temp.m=o;

return temp;

}

void showValues(){ getName(); getAptNum(); getMonth(); getRent();

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

}

static void explainPetPolicy(){

System.out.println("If there is any pet then 10$ will be added to rent.");

}

}

public class TestLease{ public static void main(String[] args) {

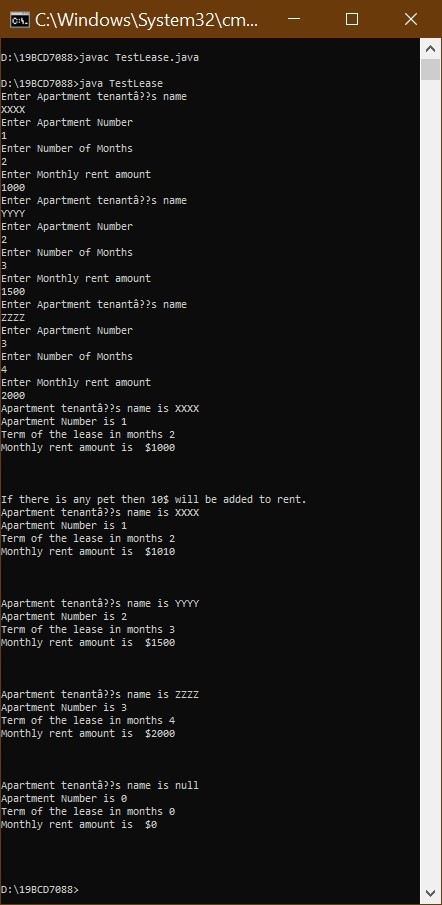
Lease l1 = new Lease();

Lease l2 = new Lease();

Lease l3 = new Lease(); Lease l4 = new Lease(); l1=l1.getData(); l2=l2.getData(); l3=l3.getData(); l1.showValues(); l1.addPetFee(); l1.explainPetPolicy(); l1.showValues(); l2.showValues(); l3.showValues(); l4.showValues();

}

}



19bcd7088

# Lab2

1.

Create a class named Billing that includes three overloaded computeBill() methods for a photo book store.

•• When computeBill() receives a single parameter, it represents the price of one photo book ordered. Add 8% tax, and return the total due.

•• When computeBill() receives two parameters, they represent the price of a photo book and the quantity ordered. Multiply the two values, add 8% tax, and return the total due.

•• When computeBill() receives three parameters, they represent the price of a photo book, the quantity ordered, and a coupon value. Multiply the quantity and price, reduce the result by the coupon value, and then add 8% tax and return the total due.

Write a main() method that tests all three overloaded methods. Save the application as Billing.java.

public class Billing{

static double computeBill(double a){

return a\*1.08;

}

static double computeBill(double a,double b){

return(a\*b)\*1.08;

}

static double computeBill(double a, double b,double c){

double n = (a\*b)-c;

return n\*1.08;

}

public static void main(String[] args) {

double k = computeBill(20); double l = computeBill(40,80); double m = computeBill(50,60,100);

System.out.println("Price of one photo book after adding 8% tax is " + k + "Rs.");

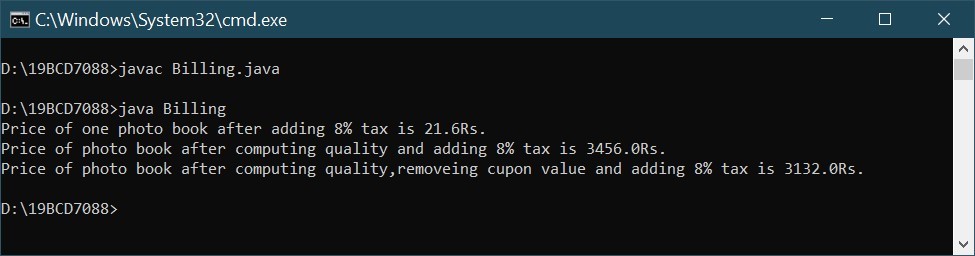
System.out.println("Price of photo book after computing quality and adding 8% tax

is " + l + "Rs.");

System.out.println("Price of photo book after computing quality,removeing cupon value and adding 8% tax is " + m + "Rs.");

}

}



2.

a.

Create a class named BloodData that includes fields that hold a blood type (the four blood types are O, A, B, and AB) and an Rh factor (the factors are 1 and –). Create a default constructor that sets the fields to O and 1, and an overloaded constructor that requires values for both fields. Include get and set methods for each field. Save this file as BloodData.java. Create an application named TestBloodData that demonstrates each method works correctly. Save the application as TestBloodData.java.

class BloodData{

String Blood,Rh; BloodData(){ this.Blood = "O"; this.Rh = "+";

}

BloodData(String Blood,String Rh){

this.Blood=Blood; this.Rh=Rh;

}

void setBlood(String Blood){

this.Blood=Blood;

}

void setRh(String Rh){

this.Rh=Rh;

}

void getBlood(){

System.out.println("Blood type is " + this.Blood);

}

void getRh(){

System.out.println("Rh factor of Blood " + this.Blood + " is " + this.Rh);

}

}

public class TestBloodData{ public static void main(String[] args) { BloodData b1 = new BloodData(); BloodData b2 = new BloodData("AB","-");

b1.getBlood(); b1.getRh(); b2.getBlood(); b2.getRh(); b1.setBlood("A"); b1.setRh("-");

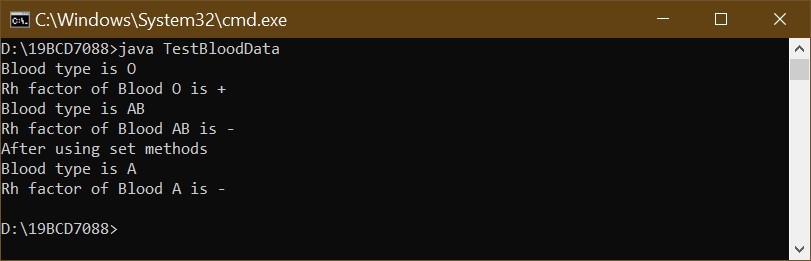
System.out.println("After using set methods");

b1.getBlood();

b1.getRh();

}

}



2.

b.

Create a class named Patient that includes an ID number, age, and BloodData. Provide a default constructor that sets the ID number to 0, the age to 0, and the BloodData values to O and 1. Create an overloaded constructor that provides values for each field. Also provide get methods for each field. Save the file as Patient.java. Create an application that demonstrates hat each method works correctly, and save it as TestPatient.java.

class Patient{

int Id,age;

String Blood,Rh; Patient(){

this.Id = 0; this.age =0; this.Rh="+";

this.Blood = "O";

}

Patient(int Id,int age,String Blood,String Rh){

this.Id = Id; this.age=age; this.Rh=Rh;

this.Blood = Blood;

}

void getId(){

System.out.println("Patient Id number is " + this.Id);

}

void getage(){

System.out.println("Age of thr Patient is " + this.age + " years old");

}

void getBlood(){

System.out.println("Blood type is " + this.Blood);

}

void getRh(){

System.out.println("Rh factor of Blood is " + this.Rh);

}

}

public class TestPatient{

public static void main(String[] args) {

Patient p1 = new Patient();

Patient p2 = new Patient(152685,48,"B","-"); System.out.println("Details of Default Patient");

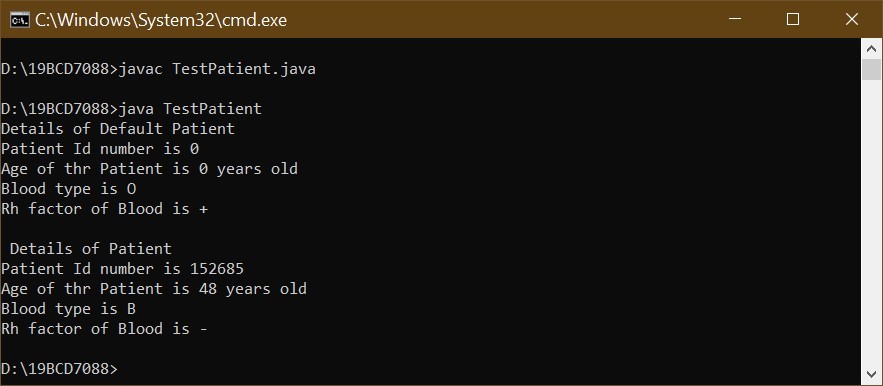
p1.getId(); p1.getage(); p1.getBlood(); p1.getRh();

System.out.println("\n Details of Patient");

p2.getId(); p2.getage(); p2.getBlood(); p2.getRh();

}

}



3.

a.

Create a class named Circle with fields named radius, diameter, and area. Include a constructor that sets the radius to 1 and calculates the other two values. Also include methods named setRadius() and getRadius(). The setRadius() method not only sets the radius, but it also calculates the other two values. (The diameter of a circle is twice the radius, and the area of a circle is pi multiplied by the square of the radius. Use the Math class PI constant for this calculation.) Save the class as Circle.java.

import java.lang.Math;

class Circle{

double r; double d; double area; Circle(){

this.r = 1; this.d = 2\*1;

this.area = Math.PI\*1\*1;

}

void setRadius(double r){

this.r = r; this.d = 2\*r;

this.area = Math.PI\*r\*r;

}

void getRadius(){

System.out.println(this.r + "radius circle has " + this.d + " diameter and " + this.area

+ " area.");

}

} 3.

b.

Create a class named TestCircle whose main() method declares several Circle objects. Using the setRadius() method, assign one Circle a small radius value, and assign another a larger radius value. Do not assign a value to the radius of the third circle; instead, retain the value assigned at construction. Display all the values for all the Circle objects. Save the application as TestCircle.java.

public class TestCircle{ public static void main(String[] args) {

Circle c1 = new Circle();

Circle c2 = new Circle();

Circle c3 = new Circle(); c1.setRadius(10); c2.setRadius(50);

System.out.println("Values of small radius circle.");

c1.getRadius();

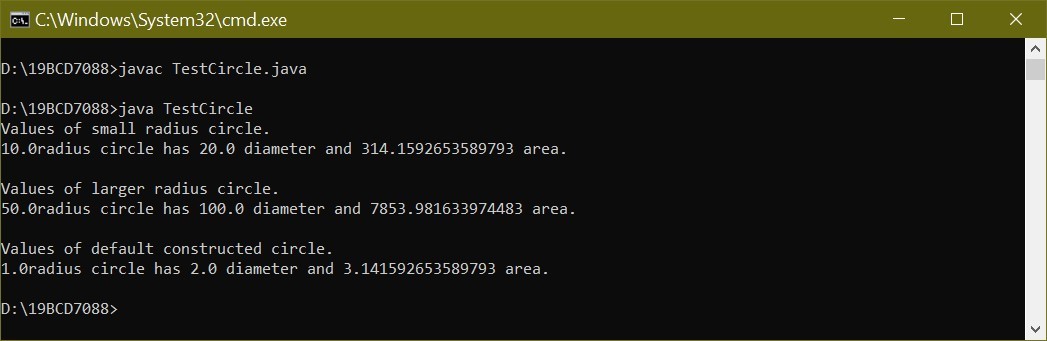
System.out.println("\nValues of larger radius circle.");

c2.getRadius();

System.out.println("\nValues of default constructed circle.");

c3.getRadius();

}



4.

Write a Java application that uses the Math class to determine the answers for each of the following:(Use java.lang.Math class)

1. The square root of 37
2. The sine and cosine of 300
3. The value of the floor, ceiling, and round of 22.8
4. The larger and the smaller of the character ‘D’ and the integer 71
5. A random number between 0 and 20 (Hint: The random() method returns a value between 0 and 1; you want a number that is 20 times larger.)

Save the application as MathTest.java.

import java.lang.Math; public class MathTest{ public static void main(String[] args) {

double a = 37;

System.out.println("Square root of " + a + " is " + Math.sqrt(a));

double b = 300; double n=Math.toRadians(b);

System.out.println("Sine of " + b + " is " + Math.sin(n)); System.out.println("Cosine of " + b + " is " + Math.cos(n));

double c = 22.8;

System.out.println("Floor value of " + c + " is " + Math.floor(c));

System.out.println("ceiling value of " + c + " is " + Math.ceil(c)); System.out.println("Round value of " + c + " is " + Math.round(c)); if ((char)(Math.max('D',71)) == 'D'){

System.out.println("Largest of Character 'D' and integer 71 is 'D'");

}

else {

System.out.println("Largest of Character 'D' and integer 71 is 71.");

}

if ((char)(Math.min('D',71)) == 'D'){

System.out.println("Smallest of Character 'D' and integer 71 is 'D'");

}

else {

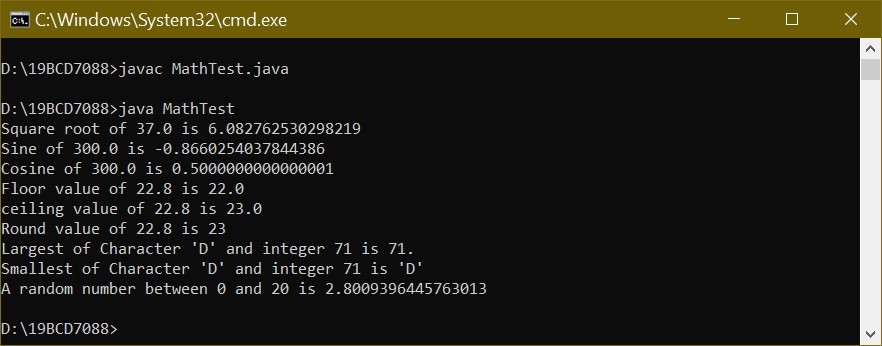
System.out.println("Smallest of Character 'D' and integer 71 is 71.");

}

System.out.println("A random number between 0 and 20 is " + 20\*Math.random());

}

}



5.

a.

Write a program that declares two LocalDate objects and assign values that represent January 31 and December 31 in the current year. Display output that demonstrates the dates displayed when

one, two, and three months are added to each of the objects. Save the application as TestMonthHandling.java.

import java.time.\*; public class TestMonthHandling{ public static void main(String[] args) {

LocalDate mon1;

LocalDate mon2;

LocalDate temp1;

LocalDate temp2;

int mo1 = 1; int day1 = 31; int year = 2020;

mon1 = LocalDate.of(year, mo1, day1);

int mo2 = 12; int day2 = 31;

mon2 = LocalDate.of(year, mo2, day2);

System.out.println("Present dates are " + mon1.getYear() +" " + mon1.getMonth()+

" " + mon1.getDayOfMonth() + " and " + mon2.getYear() +" " + mon2.getMonth()+" " + mon2.getDayOfMonth());

temp1 = mon1.plusMonths(1); temp2 = mon2.plusMonths(1);

System.out.println("Dates after adding one month to each " + temp1.getYear() +" " +

temp1.getMonth() + " " + temp1.getDayOfMonth() + " and " + temp2.getYear() +" " + temp2.getMonth()+" " + temp2.getDayOfMonth() );

temp1 = mon1.plusMonths(2); temp2 = mon2.plusMonths(2);

System.out.println("Dates after adding two month to each " + temp1.getYear() + " "+

temp1.getMonth() + " " + temp1.getDayOfMonth() + " and " + temp2.getYear() +" " + temp2.getMonth()+" " + temp2.getDayOfMonth() );

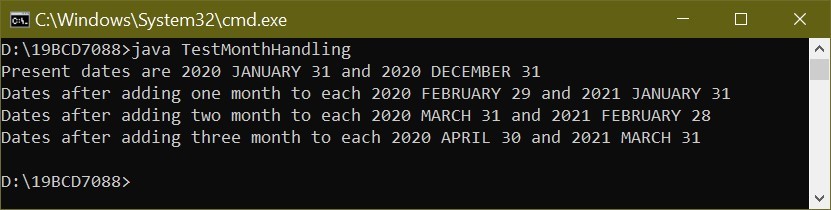
temp1 = mon1.plusMonths(3); temp2 = mon2.plusMonths(3);

System.out.println("Dates after adding three month to each " + temp1.getYear() + "

" + temp1.getMonth() + " " + temp1.getDayOfMonth() + " and " + temp2.getYear() +" " + temp2.getMonth()+" " + temp2.getDayOfMonth() );

}

}



5.

b.

Write an application that computes and displays the day on which you become (or became) 10,000 days old. Save the application as TenThousandDaysOld.java.

import java.time.\*; public class TenThousandDaysOld{ public static void main(String[] args) {

LocalDate dob; int mo = 7; int day = 22; int year = 2001;

dob = LocalDate.of(year, mo, day);

System.out.println("Born on " + dob.getYear() +" " + dob.getMonth() + " " + dob.getDayOfMonth());

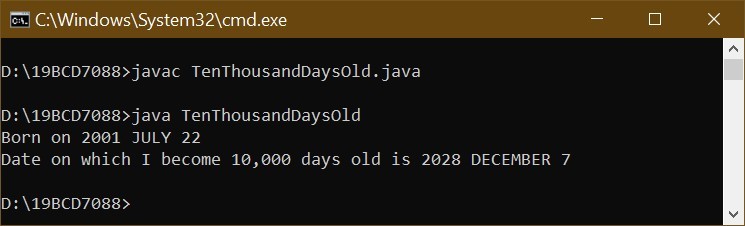
dob=dob.plusDays(10000);

System.out.println("Date on which I become 10,000 days old is " + dob.getYear() +"

" + dob.getMonth()+ " " + dob.getDayOfMonth());

}

}



5.

c.

The LocalDate class includes an instance method named lengthOfMonth() that returns the number of days in the month. Write an application that uses methods in the LocalDate class to calculate how many days are left until the first day of next month. Display the result, including the name of the next month. Save the file as DaysTilNextMonth.java.

import java.time.\*; public class DaysTilNextMonth{ public static void main(String[] args) {

LocalDate mon; LocalDate temp; int mo = 6; int day = 12; int year = 2010;

mon = LocalDate.of(year, mo, day); int remain = mon.lengthOfMonth() - day;

System.out.println("Todays date " + mon.getYear() + " " + mon.getMonth()+ " " + mon.getDayOfMonth());

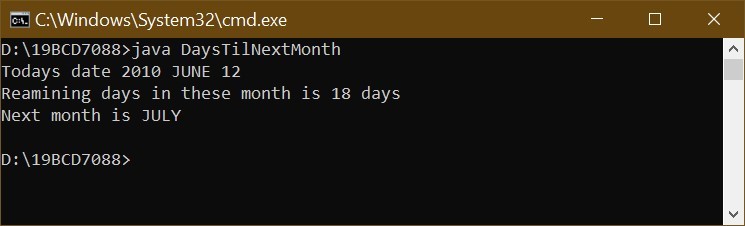
System.out.println("Reamining days in these month is " + remain + " days");

mon = mon.plusMonths(1);

System.out.println("Next month is " + mon.getMonth());

}

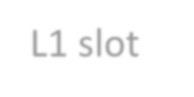
}



1.



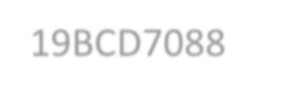
Lab3



L1 slot



V.Manikanta Bhuvanesh



19

BCD7088

Mick’s Wicks makes candles in various sizes. Create a class for the business named Candle that contains data fields for color, height, and price. Create get methods for all three fields. Create set methods for color and height, but not for price. Instead, when height is set, determine the price as $2 per inch. Create a child class named ScentedCandle that contains an additional data field named scent and methods to get and set it. In the child class, override the parent’s setHeight() method to set the price of a ScentedCandle object at $3 per inch. Write an application that instantiates an object of each type and displays the details. Save the files as Candle.java, ScentedCandle.java, and DemoCandles.java.

class Candle

{

String color; int height; double price; void getColor(){

System.out.println ("The colour of the candel is " + color);

}

void getHeight()

{

System.out.println("Height of the candel id " + height);

}

void getPrice()

{

System.out.println("Price of thr candel is " + price);

}

void setColor(String c)

{

color = c;

}

void setHeight(int h)

{

height=h;

double n = 2; price = h \* n;

}

}

class ScentedCandle extends Candle

{

String scent; void getScent()

{

System.out.println("The scent of the candel is " + scent);

}

void setScent(String s)

{

scent = s;

}

void setHeight(int h)

{

double n = 3; super.setHeight(h); price = h \* n;

}

}

public class DemoCandles

{

public static void main(String args[])

{

Candle a = new Candle(); ScentedCandle b = new ScentedCandle();

a.setColor("yellow");

a.setHeight(9);

b.setColor("orange");

b.setScent("jasmine");

b.setHeight(7);

a.getHeight();

a.getColor();

a.getPrice();

System.out.println("Scented candel");

b.getHeight();

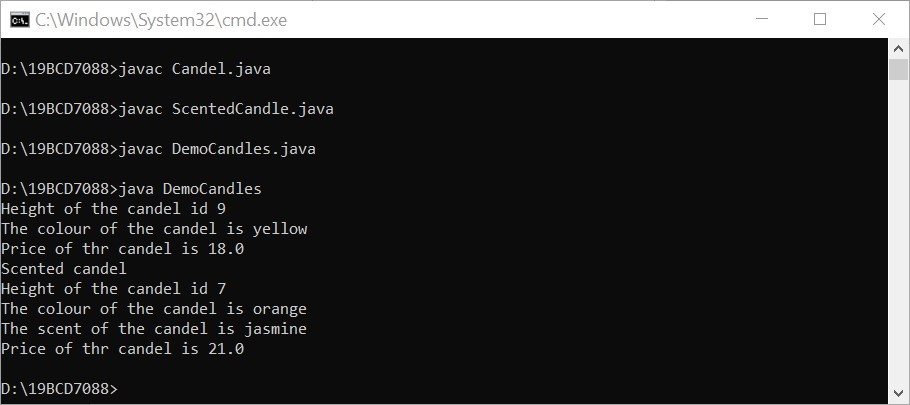
b.getColor();

b.getScent();

b.getPrice();

}

}



2.

Create a class named Poem that contains fields for the name of the poem and the number of lines in it. Include a constructor that requires values for both fields. Also include get methods to retrieve field values. Create three subclasses: Couplet, Limerick, and Haiku. The constructor for each subclass requires only a title; the lines field is set using a constant value. A couplet has two lines, a limerick has five lines, and a haiku has three lines. Create an application that demonstrates usage of an object of each type. Save the files as Poem.java, Couplet.java, Limerick.java, Haiku.java, and DemoPoems.java.

class Poem

{

String name;

int lines;

Poem(String name, int lines)

{

this.name = name;

this.lines = lines;

}

String getPoemName()

{

return name;

}

int getLines()

{

return lines ;

}

}

class Couplet extends Poem

{

Couplet(String name)

{

super(name,2);

}

}

class Haiku extends Poem

{

Haiku(String name)

{

super(name,3);

}

}

class Limerick extends Poem

{

Limerick(String name)

{

super(name,5);

}

}

public class DemoPoems

{

public static void main(String[] args)

{

Poem p1 = new Poem("You Forget Me", 84);

Couplet p2 = new Couplet("Phenomenal Woman");

Limerick p3 = new Limerick("Still I Rise");

Haiku p4 = new Haiku("The Road Not Taken");

System.out.println("Poem: " + p1.getPoemName() +" Lines: " + p1.getLines());

System.out.println("Poem: " + p2.getPoemName() +" Lines: " + p2.getLines());

System.out.println("Poem: " + p3.getPoemName() +" Lines: " + p3.getLines());

System.out.println("Poem: " + p4.getPoemName() +" Lines: " + p4.getLines()); }

}



3

The developers of a free online game named "Sugar Smash" have asked you to develop a class named SugarSmashPlayer that holds data about a single player. The class contains the following fields: the player’s integer ID number, a String screen name, and an array of integers that stores the highest score achieved in each of 10 game levels. Include get and set methods for each field. The get and set methods for the scores should each require two parameters—one that represents the score achieved and one that represents the game level to be retrieved or assigned. Display an error message if the user attempts to assign or retrieve a score from a level that is out of range for the array of scores. Additionally, no level except the first one should be set unless the user has earned at least 100 points at each previous level. If a user tries to set a score for a level that is not yet available, issue an error message. Create a class named PremiumSugarSmashPlayer that descends from SugarSmashPlayer. This class is instantiated when a user pays $2.99 to have access to 40 additional levels of play. As in the free version of the game, a user cannot set a score for a level unless the user has earned at least 100 points at all previous levels. Create a program that instantiates several objects of each type and demonstrates the methods. Save the files as SugarSmashPlayer.java, PremiumSugarSmashPlayer.java, and DemoSugarSmash.java. class SugarSmashPlayer

{ int ID;

String screenName; protected int[] scores = new int[10]; void setId(int num)

{

ID = num;

}

void setName(String player)

{

screenName = player;

}

void setScore(int score, int level)

{

if (level == 0) { scores[level] = score;} else

{

if (scores[level - 1] >= 100 && level < scores.length)

{scores[level] = score;}

else

{

System.out.println("Invalid score");

}

}

}

int getId()

{

return ID;

}

String getName(){ return screenName;

}

int getScore(int level)

{

if (level >= scores.length)

{

System.out.println("Invalid game level");

return 0;

} else{

return scores[level];

}

}

}

class PremiumSugarSmashPlayer extends SugarSmashPlayer

{

protected int[] scores = new int[50]; void setScore(int score, int level)

{

if (level == 0) { scores[level] = score;} else

{

if (scores[level - 1] >= 100 && level < scores.length)

{scores[level] = score;}

else

{

System.out.println("Invalid score");

}

}

}

int getScore(int level)

{

if (level >= scores.length)

{

System.out.println("Invalid game level");

return -1;

}

else{ return scores[level];

}

}

}

public class DemoSugarSmash

{

public static void main(String[] args)

{

SugarSmashPlayer ss = new SugarSmashPlayer();

PremiumSugarSmashPlayer ps = new PremiumSugarSmashPlayer();

ss.setName("Jhon"); ss.setId(1519); int a= 100; for(int i = 0;i<10;i++){ ss.setScore(a,i); a=a+100;

}

String name = ss.getName();

int n = ss.getId();

System.out.print(name + " of player Id " + n + " Scores are ");

int o; for(int j=0;j<10;j++){ o=ss.getScore(j);

System.out.print(o + " ");

}

System.out.println(); ps.setId(9562); ps.setName("Vicky"); int b= 100; for(int k = 0;k<50;k++){ ps.setScore(b,k); b=b+100;

}

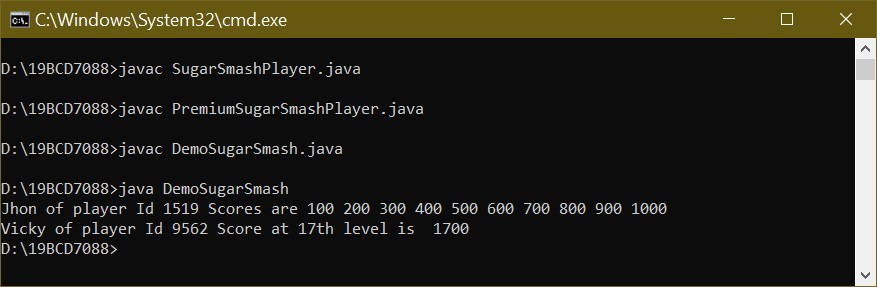
name = ps.getName();

n=ps.getId();

System.out.print(name + " of player Id " + n + " Score at 17th level is "); System.out.print(ps.getScore(16));

}

}



4.

Create a class named CollegeCourse that includes data fields that hold the department (for example,

ENG), the course number (for example, 101), the credits (for example, 3), and the fee for the course (for example, $360). All of the fields are required as arguments to the constructor, except for the fee, which is calculated at $120 per credit hour. Include a display() method that displays the course data. Create a subclass named LabCourse that adds $50 to the course fee. Override the parent class display() method to indicate that the course is a lab course and to display all the data. Write an application named UseCourse that prompts the user for course information. If the user enters a class in any of the following departments, create a LabCourse: BIO, CHM, CIS, or PHY. If the user enters any other department, create a CollegeCourse that does not include the lab fee. Then display the course data. Save the files as CollegeCourse.java, LabCourse.java, and UseCourse.java.

class CollegeCourse

{

double Ch = 120.00; String dpt; int num;

int c;

double cf;

CollegeCourse(String d, int n, int nc)

{

dpt = d.toUpperCase(); num = n;

c = nc; cf = Ch \* c;

}

String getdpt()

{

return dpt;

}

int getcourseNo()

{

return num;

}

int getCredits()

{

return c;

}

double getcourseFee()

{

return cf;

}

void display()

{

System.out.println("department " + this.getdpt());

System.out.println("Course number " + this.getcourseNo());

System.out.println("Credit hours " + this.getCredits());

System.out.println("Course fee " + this.getcourseFee());

}

}

class LabCourse extends CollegeCourse

{

double lf = 50.00; double cf;

LabCourse(String dpt, int cn, int c)

{

super(dpt, cn, c); cf = super.getcourseFee() + lf;

}

double getLabCourseFee()

{

return cf;

}

void display()

{

System.out.println("department " + super.getdpt());

System.out.println("Course number " + super.getcourseNo());

System.out.println("Credit hours " + super.getCredits());

System.out.println("Course fee " + this.getLabCourseFee());

}

}

import java.util.Scanner; public class UseCourse{ public static void main(String[] args)

{

Scanner sc = new Scanner(System.in);

System.out.print("Enter the department of the course: ");

String dept = sc.nextLine();

System.out.print("Enter the number of the courses "); int number = sc.nextInt();

System.out.print("Enter the credit hours of the courses "); int hours = sc.nextInt();

if(dept.equals("BIO") || dept.equals("CHM")|| dept.equals("CIS") || dept.equals("PHY"))

{

LabCourse l = new LabCourse(dept, number, hours);

l.display(); }

else

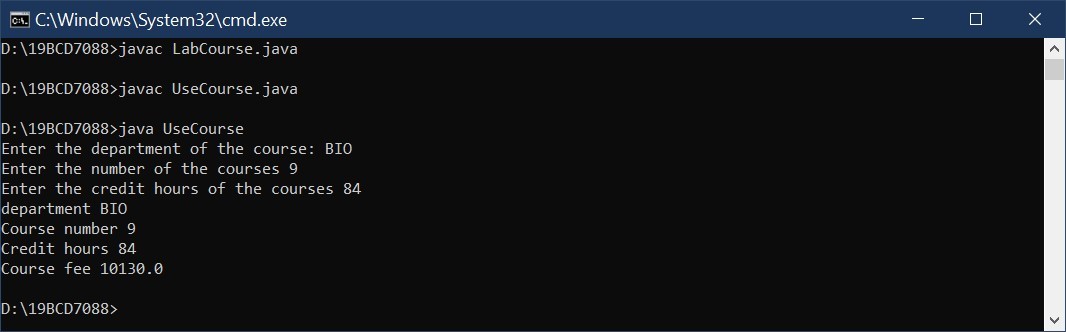
{

CollegeCourse c = new CollegeCourse(dept, number, hours); c.display();

}

}

}



5.

Develop a set of classes for a college to use in various student service andpersonnel applications. Classes you need to design include the following:

•• Person—A Person contains a first name, last name, street address, zip code,and phone number. The class also includes a method that sets each datafield, using a series of dialog boxes and a display method that displays all of a Person’s information on a single line at the command line on the screen.

•• CollegeEmployee—CollegeEmployee descends from Person. A CollegeEmployee also includes a Social Security number, an annual salary, and a department name, as well as methods that override the Person methods to accept and display all CollegeEmployee data.

•• Faculty—Faculty descends from CollegeEmployee. This class also includes Boolean field that indicates whether the Faculty member is tenured, as well as methods that override the CollegeEmployee methods to accept and display this additional piece of information.

•• Student—Student descends from Person. In addition to the fields available in Person, a Student contains a major field of study and a grade point average as well as methods that override the Person methods to accept and display these additional facts.

Write an application named CollegeList that declares an array of four “regular” CollegeEmployees, three Faculty, and seven Students. Prompt the user to specify which type of person’s data will be entered (C, F, or S), or allow the user to quit (Q). While the user chooses to continue (that is, does not quit), accept data entry for the appropriate type of Person. If the user attempts to enter data for more than four CollegeEmployees, three Faculty, or seven Students, display an error message. When the user quits, display a report on the screen listing each group of Persons under the appropriate heading of “College Employees,” “Faculty,” or “Students.” If the user has not entered data for one or more types of Persons during a session, display an appropriate message under the appropriate heading. Save the files as Person.java, CollegeEmployee.java, Faculty.java, Student.java, and CollegeList.java.

class Person{

String firstName;

String lastName; String address; int pincode; long phonenum;

void setFirstName(String firstName){ this.firstName= firstName;

}

void setLastName(String lastName){

this.lastName=lastName;

}

void setAddress(String address){

this.address = address;

}

void setPincode(int pincode){

this.pincode = pincode;

}

void setphonenum(long phonenum){

this.phonenum= phonenum;

}

void display(){

System.out.println(this.firstName + " " + this.lastName + "'s " + "Address is " + this.address + "," + this.pincode + ".Mobile number is " + this.phonenum);

}

}

class CollegeEmployee extends Person{

int sn; double a;

String dept; void setSnum(int n){

this.sn=n;

}

void setAsal(double b){

this.a=b;

}

void setdept(String dept){

this.dept=dept;

}

void display(){

super.display();

System.out.println("His Security number is "+this.sn+" and Annual salary is

"+this.a+". He belongs to Department of "+this.dept+".");

}

}

class Faculty extends CollegeEmployee

{

boolean tenured; void setTenured(boolean t){

this.tenured=t;

}

public void display()

{

super.display(); if(tenured){

System.out.println("Faculty member is tenured");

}

else{

System.out.println("Faculty member is not tenured");

}

}

}

class Student extends Person

{

String major;

double avg; void setMajor(String major)

{

this.major = major;

}

void setAvg(double avg)

{

this.avg = avg;

}

public String getMajor()

{

return major;

}

public double getAvg()

{

return avg;

}

public void display()

{

super.display();

System.out.println("His major is " + getMajor() + " and his average is " + getAvg());

}

}

import java.util.Scanner; public class CollegeList

{

public static void main(String[] args)

{

CollegeEmployee[] c = new CollegeEmployee[4];

Faculty[] f = new Faculty[3];

Student[] s = new Student[7];

Scanner sc = new Scanner(System.in);

String response,fname,lname,address,dept,major; int pin,securitynum; double salary,avg; long phone; boolean tenured; String cont ="y"; String QUIt = "Q";

int i=0; int j=0; int k=0;

System.out.println("Enter C for CollegeEmployee entry or F fot Faculty entry or S for Student entry"); response = sc.nextLine(); while(response!="Q"){ if(response.equals("C")){ while(cont.equals("y"))

{

c[i]=new CollegeEmployee();

System.out.println("Enter first name"); fname = sc.nextLine(); c[i].setFirstName(fname);

System.out.println("Enter last name"); lname = sc.nextLine(); c[i].setLastName(lname); System.out.println("Enter address"); address = sc.nextLine(); c[i].setAddress(address);

System.out.println("Enter pin code"); pin = sc.nextInt(); c[i].setPincode(pin);

System.out.println("Enter phone number"); phone = sc.nextLong(); c[i].setphonenum(phone);

System.out.println("Enter security number"); securitynum = sc.nextInt(); c[i].setSnum(securitynum);

System.out.println("Enter Anual salary"); salary = sc.nextDouble(); sc.nextLine(); c[i].setAsal(salary);

System.out.println("Enter department name");

dept = sc.nextLine(); c[i].setdept(dept);

System.out.println("Enter more entries? (y/n)"); cont = sc.nextLine();

if(i==3){

System.out.println("Entered maxium number of entries");

cont="n";

} i++;

}

System.out.println("Enter C for CollegeEmployee entry or F fot Faculty entry or S for Student entry"); response = sc.nextLine();

cont ="y";

}

if(response.equals("F")){

while(cont.equals("y"))

{

f[j]=new Faculty();

System.out.println("Enter first name: "); fname = sc.nextLine(); f[j].setFirstName(fname);

System.out.println("Enter last name"); lname = sc.nextLine(); f[j].setLastName(lname); System.out.println("Enter address"); address = sc.nextLine(); f[j].setAddress(address);

System.out.println("Enter pin code");

pin = sc.nextInt(); f[j].setPincode(pin);

System.out.println("Enter phone number"); phone = sc.nextLong(); f[j].setphonenum(phone);

System.out.println("Enter security number"); securitynum = sc.nextInt(); f[j].setSnum(securitynum);

System.out.println("Enter Anual salary"); salary = sc.nextDouble(); f[j].setAsal(salary);

System.out.println("Enter department name"); dept = sc.nextLine(); sc.nextLine(); f[j].setdept(dept);

System.out.println("Enter true if tenured or enter false if not tenured"); tenured=sc.nextBoolean(); sc.nextLine();

f[j].setTenured(tenured);

System.out.println("Enter more entries ? (y/n)"); cont = sc.nextLine();

if(j==2){

System.out.println("Entered maxium number of entries");

cont="n";

} j++;

}

System.out.println("Enter C for CollegeEmployee entry or F fot Faculty entry or S for Student entry"); response = sc.nextLine();

cont ="y";

}

if(response.equals("S")){ while(cont.equals("y"))

{

s[k]=new Student();

System.out.println("Enter first name"); fname = sc.nextLine(); s[k].setFirstName(fname);

System.out.println("Enter last name"); lname = sc.nextLine(); s[k].setLastName(lname); System.out.println("Enter address"); address = sc.nextLine(); s[k].setAddress(address); System.out.println("Enter pin code");

pin = sc.nextInt(); s[k].setPincode(pin);

System.out.println("Enter phone number");

phone = sc.nextLong();

sc.nextLine(); s[k].setphonenum(phone); System.out.println("Enter major"); major =sc.nextLine(); s[k].setMajor(major);

System.out.println("Enter average"); avg=sc.nextDouble();

sc.nextLine(); s[k].setAvg(avg);

System.out.println("Enter more entries? (y/n)"); cont = sc.nextLine();

if(k==6){

System.out.println("Entered maxium number of entries");

cont="n";

} k++;

}

System.out.println("quiting"); response="Q";

}

}

if(response.equals("Q")){

for (int p =0;p<i;p++){ c[p].display();

}

for(int q=0;q<j;q++){ f[q].display();

}

for(int r=0;r<k;r++){ s[r].display();

}

}

int h = 4-i; if(h!=0){

System.out.println(h +" CollegeEmployee data not Entered");

} h=3-j; if(h!=0){

System.out.println(h+ " Faculty data not Entered");

}

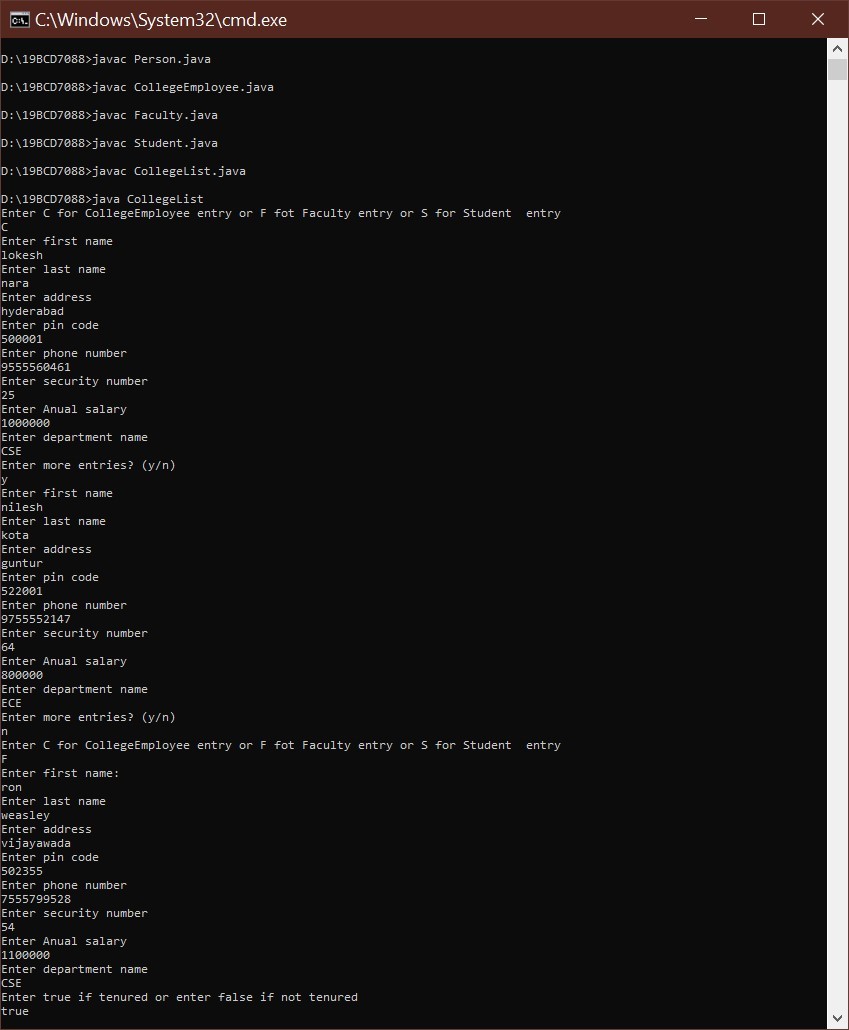
h=7-k; if(h!=0){

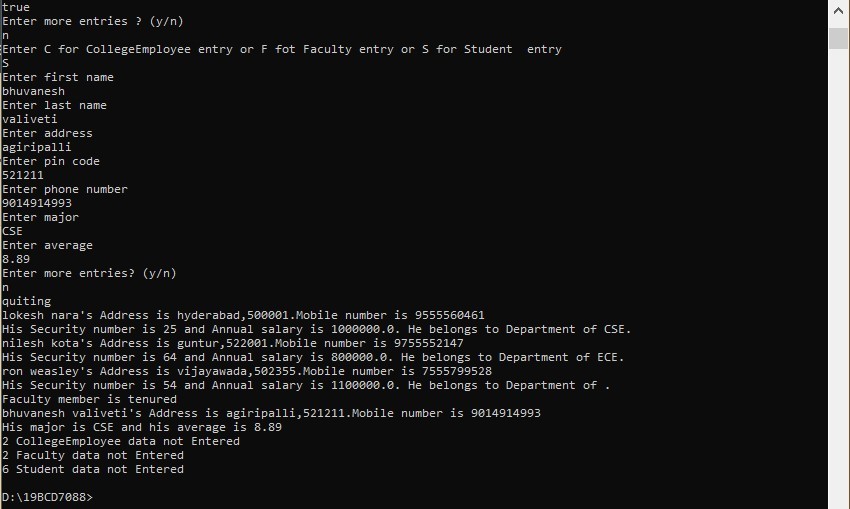
System.out.println(h + " Student data not Entered");

}

}

}



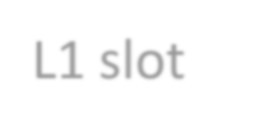


1.a.

Create an abstract class named Book. Include a String field for the book’stitle and a double field for the book’s price. Within the class, include aconstructor that requires the book title, and add two get methods—one that returns the title and one that returns the price. Include an abstract method named setPrice(). Create two child classes of Book: Fiction and NonFiction. Each must include a setPrice() method that sets the price for all Fiction Books to $24.99 and for all NonFiction Books to $37.99. Write a constructor for each subclass, and include a call to setPrice() within each. Write an application demonstrating that you can create both a Fiction and a NonFiction Book, and display their fields. Save the files as Book.java, Fiction.java, NonFiction.java, and UseBook.java.



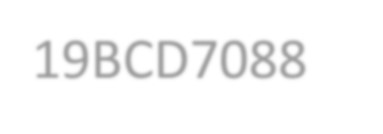
Lab4



L1 slot

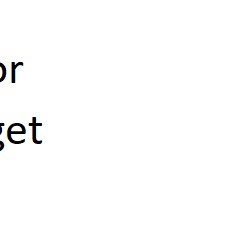


V.Manikanta Bhuvanesh



19

BCD7088



abstract class Book {

String t; double p;

Book(String title ){

t=title;

}

String gettitle(){ return t;

}

double getPrice(){

return p;

}

abstract void setPrice();

}

class Fiction extends Book{

Fiction(String title) { super(title); setPrice();

}

void setPrice(){ super.p=24.99;

}

}

class NonFiction extends Book{

NonFiction(String title) {

super(title); setPrice();

}

void setPrice(){ super.p=37.99;

}

}

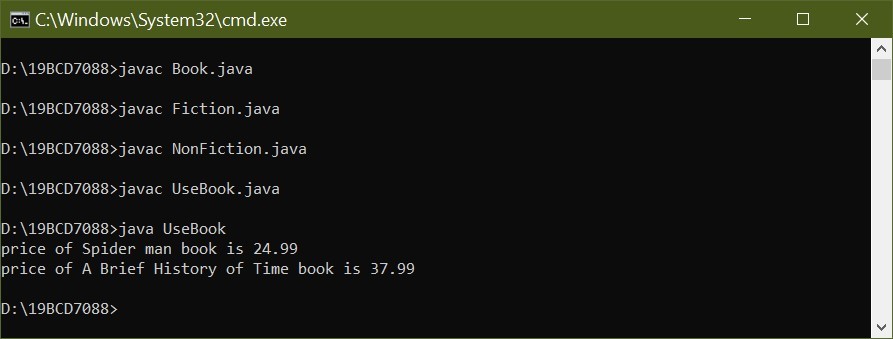
public class UseBook { public static void main(String[] args){ Book b1,b2; b1=new Fiction("Spider man");

System.out.println("price of " + b1.gettitle() + " book is " + b1.getPrice()); b2=new NonFiction("A Brief History of Time");

System.out.println("price of " + b2.gettitle() + " book is " + b2.getPrice());

}

}



1.b.

Write an application named BookArray in which you create an array that holds 10 Books, some Fiction and some NonFiction. Using a for loop, display details about all 10 books. Save the file as BookArray.java.

public class BookArray{ public static void main(String[] args) {

Book b[] = new Book[10];

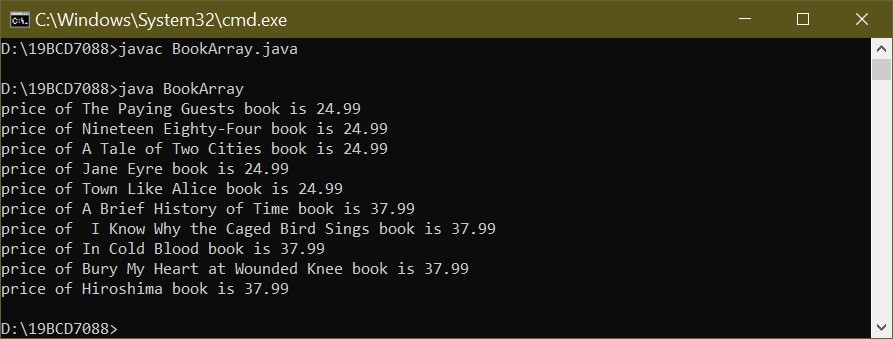
b[0] = new Fiction("The Paying Guests"); b[1] = new Fiction("Nineteen Eighty-Four"); b[2] = new Fiction("A Tale of Two Cities"); b[3] = new Fiction("Jane Eyre"); b[4] = new Fiction("Town Like Alice"); b[5] = new NonFiction("A Brief History of Time"); b[6] = new NonFiction(" I Know Why the Caged Bird Sings"); b[7] = new NonFiction("In Cold Blood"); b[8] = new NonFiction("Bury My Heart at Wounded Knee"); b[9] = new NonFiction("Hiroshima"); for(int i=0;i<10;i++){

System.out.println("price of " + b[i].gettitle() + " book is " + b[i].getPrice());

}

}

}



2.a.

The Talk-A-Lot Cell Phone Company provides phone services for its customers. Create an abstract class named PhoneCall that includes a String field for a phone number and a double field for the price of the call. Also include a constructor that requires a phone number parameter and that sets the price to 0.0. Include a set method for the price. Also include three abstract get methods—one that returns the phone number, another that returns the price of the call, and a third that displays information about the call. Create two child classes of PhoneCall: IncomingPhoneCall and OutgoingPhoneCall. The IncomingPhoneCall constructor passes its phone number parameter to its parent’s constructor and sets the price of the call to 0.02. The method that displays the phone call information displays the phone number, the rate, and the price of the call (which is the same as the rate). The OutgoingPhoneCall class includes an additional field that holds the time of the call in minutes. The constructor requires both a phone number and the time. The price is 0.04 per minute, and the display method shows the details of the call, including the phone number, the rate per minute, the number of minutes, and the total price. Write an application that demonstrates you can instantiate and display both IncomingPhoneCall and OutgoingPhoneCall objects. Save the files as PhoneCall.java, IncomingPhoneCall.java, OutgoingPhoneCall.java, and DemoPhoneCalls.java. abstract class PhoneCall

{

String phn; double price;

PhoneCall(String phn)

{

this.phn = phn; this.price = 0.0;

}

abstract String getPhoneNumber(); abstract double getPrice();

abstract void getInf(); abstract void setPrice();

}

class IncomingPhoneCall extends PhoneCall {

double r=0.02;

IncomingPhoneCall(String phoneNumber){ super(phoneNumber);

setPrice();

}

void setPrice() { super.price = 0.02;

}

void getInf(){

System.out.println("Incoming phone call for "+getPhoneNumber()+",Price for a call is $"+getPrice());

}

String getPhoneNumber()

{

return super.phn;

}

double getPrice()

{

return super.price;

}

}

class OutgoingPhoneCall extends PhoneCall {

double r = 0.04; int minutes; OutgoingPhoneCall(String phoneNumber, int minutes){ super(phoneNumber); this.minutes = minutes;

setPrice();

}

void setPrice() { super.price = 0.04;

}

void getInf() {

System.out.println("Outgoing phone call for " + getPhoneNumber() + " "+ r + " per minute at " + minutes + " minutes is $" + price\*minutes);

}

public String getPhoneNumber()

{

return super.phn;

}

public double getPrice()

{

return super.price;

}

}

public class DemoPhoneCalls {

public static void main(String [] args) {

IncomingPhoneCall in=new IncomingPhoneCall("9014914993"); OutgoingPhoneCall out=new OutgoingPhoneCall("9456552237",40);

in.getInf(); out.getInf();

}

}



2.b.

Write an application in which you assign data to a mix of eight IncomingPhoneCall and OutgoingPhoneCall objects into an array. Use a for loop to display the data. Save the file as PhoneCallArray.java.

public class PhoneCallArray{ public static void main(String[] args) {

IncomingPhoneCall in[]=new IncomingPhoneCall[3]; OutgoingPhoneCall out[]=new OutgoingPhoneCall[5]; in[0]=new IncomingPhoneCall("9856412345"); in[1]=new IncomingPhoneCall("94851265126"); in[2]=new IncomingPhoneCall("78843204566"); out[0]= new OutgoingPhoneCall("8456131361",50); out[1]= new OutgoingPhoneCall("9456123789",60); out[2]= new OutgoingPhoneCall("7541237896",70); out[3]= new OutgoingPhoneCall("7148529632",80); out[4]= new OutgoingPhoneCall("8794561235",90); for(int i=0;i<3;i++){

in[i].getInf();

}

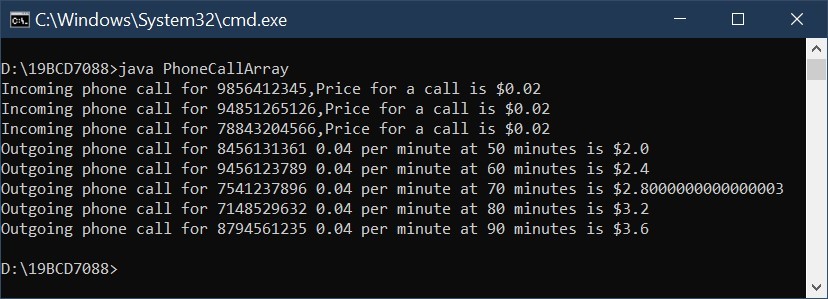
for(int j=0;j<5;j++){

out[j].getInf();

}

}

}



3.a.

Create an interface named Turner, with a single method named turn(). Create a class named Leaf that implements turn() to display Changing colors. Create a class named Page that implements turn() to display Going to the next page. Create a class named Pancake that implements turn() to display Flipping. Write an application named DemoTurners that creates one object of each of these class types and demonstrates the turn() method for each class. Save the files as Turner.java, Leaf.java, Page.java, Pancake.java, and DemoTurners.java.

interface Turner { public void turn();

}

class Leaf implements Turner{

public void turn() {

System.out.println("Changing colors");

}

}

class Page implements Turner{

public void turn() {

System.out.println("Going to the next page");

}

}

class Pancake implements Turner{

public void turn() {

System.out.println("Flipping");

}

}

public class DemoTurners {

public static void main(String[] args){

Leaf l = new Leaf();

Page p=new Page(); Pancake c=new Pancake();

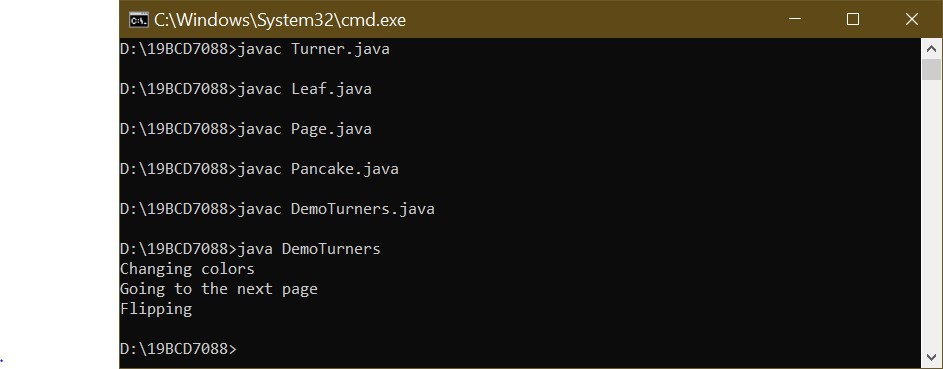
l.turn();

p.turn();

c.turn();

}

}



3.b.

Think of two more objects that use turn(), create classes for them, and then add objects to the DemoTurners application, renaming it DemoTurners2. java. Save the files, using the names of new objects that use turn(). interface Turner { public void turn();

}

class Leaf implements Turner{

public void turn() {

System.out.println("Changing colors");

}

}

class Page implements Turner{

public void turn() {

System.out.println("Going to the next page");

}

}

class Pancake implements Turner{

public void turn() {

System.out.println("Flipping");

}

}

class Lesson implements Turner{

public void turn() {

System.out.println("Changeing to next Lesson");

}

}

class Cook implements Turner{

public void turn() {

System.out.println("Cook new dish");

}

}

public class DemoTurners2 { public static void main(String[] args){

Leaf l = new Leaf();

Page p=new Page();

Pancake c=new Pancake();

Lesson le = new Lesson();

Cook co = new Cook();

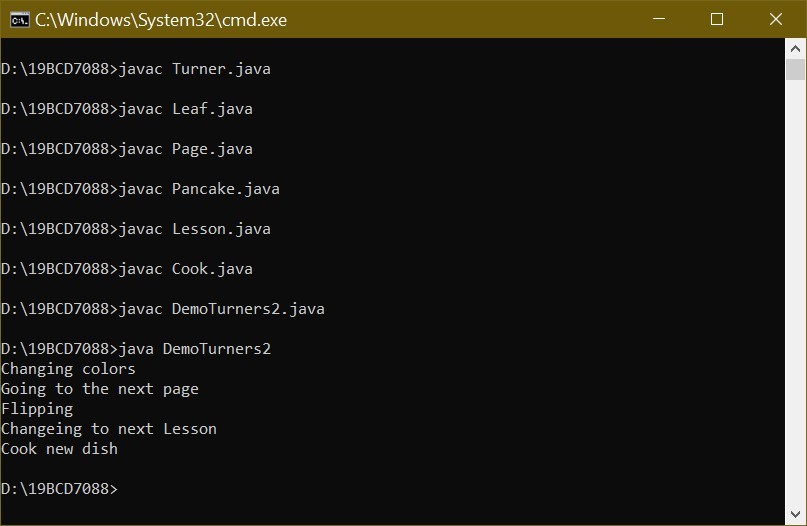
l.turn();

p.turn();

c.turn(); le.turn(); co.turn();

}

}



3.c.

Apply Dynamic method dispatch to show the power of it and name the class as DemoTurners3.

interface Turner { public void turn();

}

class Leaf implements Turner{

public void turn() {

System.out.println("Changing colors");

}

}

class Page implements Turner{

public void turn() {

System.out.println("Going to the next page");

}

}

class Pancake implements Turner{

public void turn() {

System.out.println("Flipping");

}

}

class Lesson implements Turner{

public void turn() {

System.out.println("Changeing to next Lesson");

}

}

class Cook implements Turner{

public void turn() {

System.out.println("Cook new dish");

}

}

public class DemoTurners3 {

public static void main(String[] args){ Turner t; t = new Leaf();

t.turn(); t=new Page();

t.turn(); t=new Pancake();

t.turn();

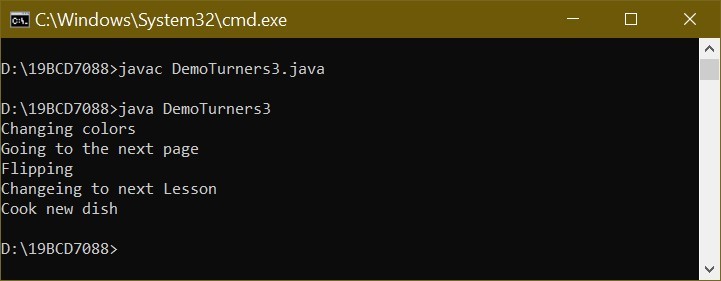
t= new Lesson();

t.turn(); t = new Cook();

t.turn();

}

}



4.a.

Create an abstract class called GeometricFigure. Each figure includes a height, a width, a figure type, and an area. Include an abstract method to determine the area of the figure. Create two subclasses called Square and Triangle. Create an application that demonstrates creating objects of both subclasses, and store them in an array. Save the files as GeometricFigure.java, Square.java, Triangle.java, and UseGeometric.java.

abstract class GeometricFigure {

int height, width; String figureType; int area; abstract void Area(int h, int w);

}

class Square extends GeometricFigure{ Square(int a, int b){ super.height=a; super.width=b;

Area(a,b);

}

void Area(int h, int w) { super.area=(h\*w);

}

}

class Triangle extends GeometricFigure{

Triangle(int a, int b){ super.height=a; super.width=b;

Area(a,b);

}

void Area(int h, int w) { area=(h\*w)/2;

}

}

public class UseGeometric { public static void main(String[] args){

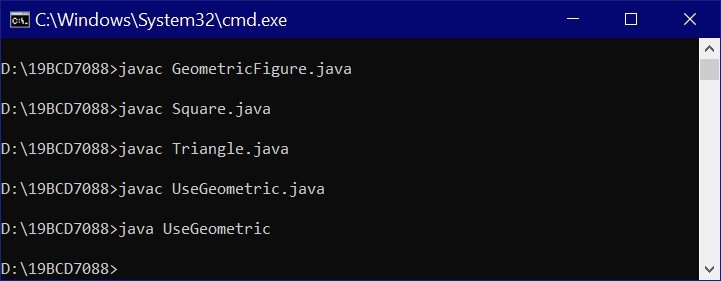
GeometricFigure f[]=new GeometricFigure[2];

Square s=new Square(20,20); Triangle t=new Triangle(10,20);

f[0]=s; f[1]=t;

}

}



4.b.

Modify 4.a., adding an interface called SidedObject that contains a method called displaySides(); this method displays the number of sides the object possesses. Modify the GeometricFigure subclasses to include the use of the interface to display the number of sides of the figure. Create an application that demonstrates the use of both subclasses. Save the files as GeometricFigure2.java, Square2.java, Triangle2.java, SidedObject.java, and UseGeometric2.java.

abstract class GeometricFigure {

int height, width; String figureType; int area,sides; abstract void Area(int h, int w);

}

interface SidedObject{

public void display();

}

class Square2 extends GeometricFigure implements SidedObject{ Square2(int a, int b){ super.height=a; super.width=b; super.sides=4;

Area(a,b);

}

void Area(int h, int w) {

super.area=(h\*w);

}

public void display(){

System.out.println("Area of the figure is " + super.area + " and number of sides are " + super.sides);

}

}

class Triangle2 extends GeometricFigure implements SidedObject{

Triangle2(int a, int b){ super.height=a; super.width=b; super.sides=3;

Area(a,b);

}

void Area(int h, int w) { super.area=(h\*w)/2;

}

public void display(){

System.out.println("Area of the figure is " + super.area + " and number of sides are " + super.sides);

}

}

public class UseGeometric2 {

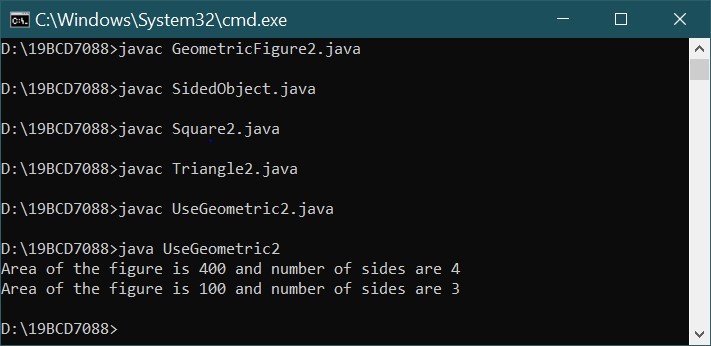
public static void main(String[] args){ Square2 s=new Square2(20,20); Triangle2 t=new Triangle2(10,20);

s.display();

t.display();

}

}



5.

Sanchez Construction Loan Co. makes loans of up to $100,000 for construction projects. There are two categories of Loans—those to businesses and those to individual applicants.

Write an application that tracks all new construction loans. The application also must calculate the total amount owed at the due date (original loan amount + loan fee). The application should include the following classes:

•• Loan—A public abstract class that implements the LoanConstants interface. A Loan includes a loan number, customer last name, amount of loan, interest rate, and term. The constructor requires data for each of the fields except interest rate. Do not allow loan amounts greater than $100,000. Force any loan term that is not one of the three defined in the LoanConstants class to a short-term, 1-year loan. Create a toString() method that displays all the loan data.

•• LoanConstants—A public interface class. LoanConstants includes constant values for short-term (1 year), medium-term (3 years), and long-term (5 years) loans. It also contains constants for the company name and the maximum loan amount.

•• BusinessLoan—A public class that extends Loan. The BusinessLoan constructor sets the interest rate to 1% more than the current prime interest rate.

•• PersonalLoan—A public class that extends Loan. The PersonalLoan constructor sets the interest rate to 2% more than the current prime interest rate.

•• CreateLoans—An application that creates an array of five Loans. Prompt the user for the current prime interest rate. Then, in a loop, prompt the user for a loan type and all relevant information for that loan. Store the created Loan objects in the array. When data entry is complete, display all the loans.

Save the files as Loan.java, LoanConstants.java, BusinessLoan.java, PersonalLoan.java, and CreateLoans.java. interface LoanConstants {

public int st = 1; public int mt = 3; public int lt = 5;

public String cn = "Sanchez Construction Loan Co."; public double max = 100000;

}

abstract class Loan implements LoanConstants {

String loanNum; String lastName; double loanAmt; double interestRate; int term;

Loan(String loanNum, String lastName, double loanAmt, int term) { this.loanNum = loanNum; this.lastName = lastName; if (loanAmt > max) {

System.out.println("Loan amount value is more than $100,000");

} else {

this.loanAmt = loanAmt;

}

if(term==st||term==mt||term==lt){

this.term=term;

}

else{ this.term=1;

}

}

public String toString() { double n =this.loanAmt+(this.loanAmt \* (this.interestRate/100));

return this.lastName + "'s loan number is " + this.loanNum + " his loan amount is " + this.loanAmt + " with intrest rate of " + this.interestRate +" and total due is " + n + " in term " + this.term;

}

}

class BusinessLoan extends Loan {

BusinessLoan(String loanNum, String lastName, double loanAmt, int term, double primeIntRate) { super(loanNum, lastName, loanAmt, term); super.interestRate = 0.01 +primeIntRate;

}

}

class PersonalLoan extends Loan {

PersonalLoan(String loanNum, String lastName, double loanAmt, int term, double primeIntRate) { super(loanNum, lastName, loanAmt, term); super.interestRate = 0.02 + primeIntRate ;

}

}

import java.util.Scanner; public class CreateLoans{ public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

Loan[] l = new Loan[5]; String ch,ln,lon,p;

double amt,ir;

int term;

for(int i=0;i<5;i++){

System.out.println("Enter b for BusinessLoan & p for personal loan");

if(i!=0){ sc.nextLine();

}

ch=sc.nextLine();

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

ln=sc.nextLine();

System.out.println("Enter loan number");

lon=sc.nextLine();

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

amt=sc.nextDouble();

System.out.println("prime interest rate");

ir=sc.nextDouble();

System.out.println("Enter term number");

term=sc.nextInt(); if(ch.equals("b")){ l[i]=new BusinessLoan(lon,ln,amt,term,ir);

}

else if(ch.equals("p")){ l[i]=new PersonalLoan(lon,ln,amt,term,ir);

}

else{

System.out.println("Invalid loan type");

}

}

for(int j = 0;j<5;j++){

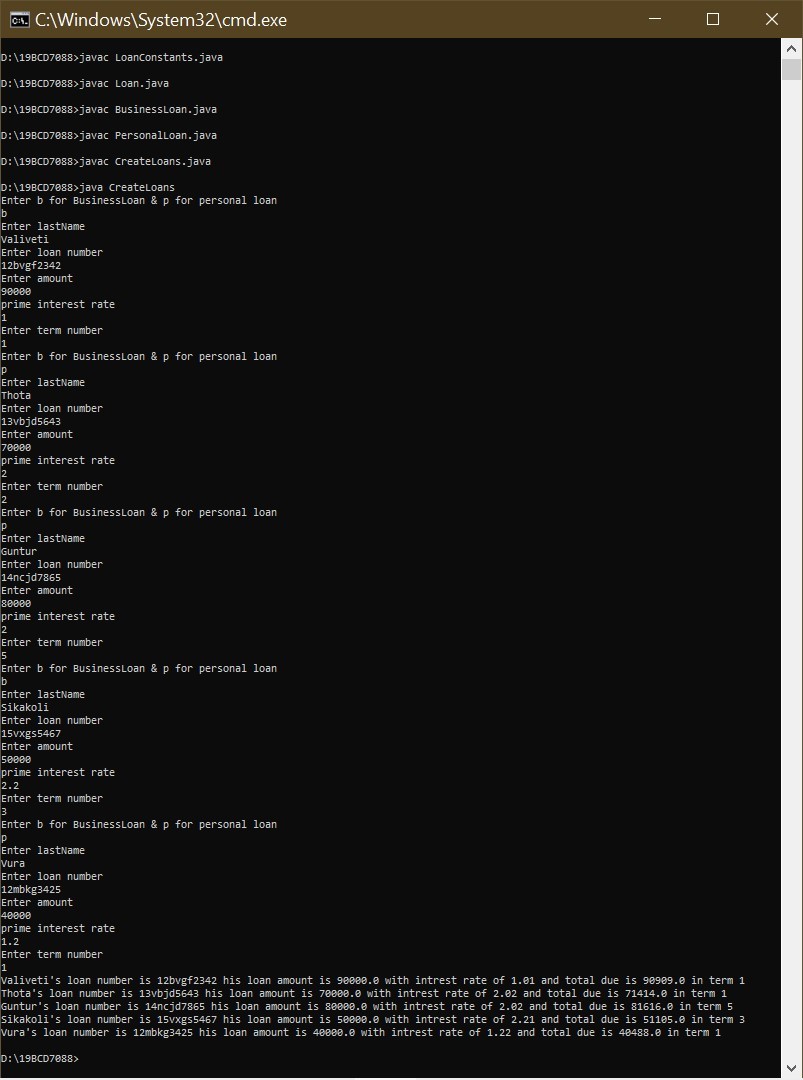
p=l[j].toString();

System.out.println(p);

}

}

}



1.

Let's say you have an integer array and a string array. You have to write a single method printArray that can print all the elements of both arrays. The method should be able to accept both integer arrays or string arrays.(Do not use overloading, use generics).Name the file ArrayPrint.java import java.util.\*; public class ArrayPrint{

static <T> void printArray(T[] a){

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

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

}

System.out.println();

}

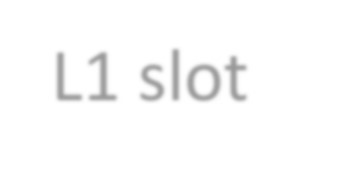
public static void main(String[] args) {

Integer [] a={1,2,3,4,5,6,7,8,9};

String [] str = {"bhuvanesh","gur","manish","ramesh","manas"};



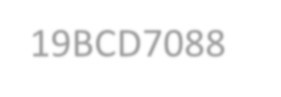
Lab 5



L1 slot



Valiveti manikanta bhuvanesh



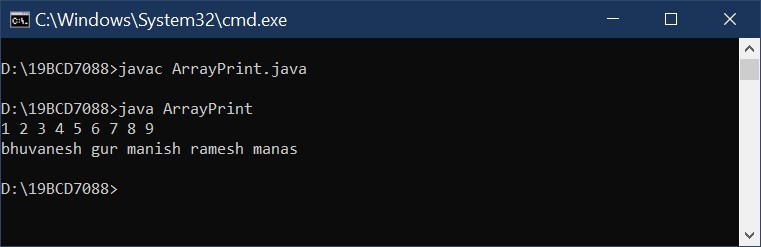
19

BCD7088

printArray(a); printArray(str);

}

}



2.

Write a generic method to perform linearSearch on array of objects. Name the file GenericLinearSearch.java

import java.util.\*; public class GenericLinearSearch{ static <T> int linearSearch(T[]a,T n){

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

if(a[i].equals(n)){

return i;

}

}

return 0;

}

static <T> void printArray(T[] a){

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

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

}

System.out.println();

}

public static void main(String[] args) {

Integer [] a={1,2,3,4,5,6,7,8,9};

Integer k1=5; printArray(a);

int k = linearSearch(a,k1);

System.out.println(k1 + " is found at index " + k);

String [] str = {"bhuvanesh","guru","manish","ramesh","manas"};

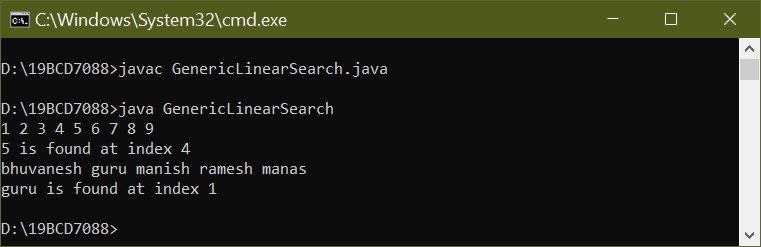
String k2="guru"; printArray(str);

int p = linearSearch(str,k2);

System.out.println(k2 + " is found at index " + p);

}

}



3.

Write a generic class called GenericStack<T> that represents a stack structure. A stack structure follow the strategy last-in-first-out, which means that the last element added to the stack, is the first to be taken out.

The GenericStack class has the following attributes and methods:

--An attribute ArrayList<T> elements which represents the elements of the stack.(All of you refer collection framework for ArrayList. or you can use an array to hold the elements of Stack.)[Refer the following links to have intro on ArrayList:

https://www.w3schools.com/java/java\_arraylist.asp, https://www.geeksforgeeks.org/arraylist-in-java/]

--A constructor that creates the ArrayList or an Array

--A method push(T e) which adds the element to the ArrayList<T> or array.

--A method pop() which removes the last element of the ArrayList<T> (last element added), if the list is not already empty and returns it.

--A method print() which prints the elements of the stack starting from the last element to the first element.

import java.util.\*; class GenericStack<T>{

int top; int max;

T [] a;

GenericStack(int size){

a=(T[]) new Object[size];

max=size; top=-1;

}

void push(T x){

if(top==(max-1)){

System.out.println("Stack is Filled");

}

else{

top++;

a[top]=x;

}

}

T pop(){ if(top==-1){

System.out.println("Stack is empty");

return null;

}

else{

return a[top--];

}

}

void print(){

for (int i=top;i>=0;i--){

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

}

System.out.println();

}

}

public class GenericStackDemo{ public static void main(String[] args) {

GenericStack<Integer> s =new GenericStack<Integer>(3); GenericStack<String> str =new GenericStack<String>(3);

s.push(11228);

s.push(220102);

s.push(310201);

s.print();

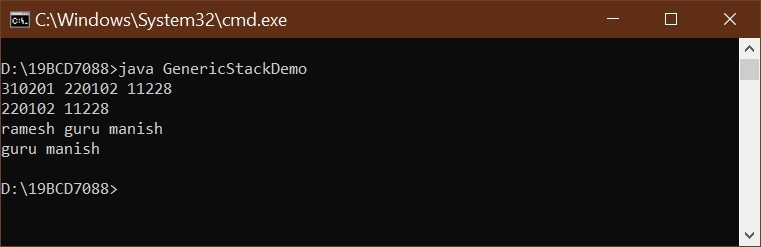
s.pop();

s.print();

str.push("manish"); str.push("guru"); str.push("ramesh"); str.print(); str.pop(); str.print();

}

}



3. b.

Create the generic interface GenericStackable<T> that contains the abstract methods:

-an abstract method push(T e) which adds the element to the ArrayList<T>

-an abstract method pop() which remove the last element of the ArrayList<T> (last element added), if the list is not already empty.

-a abstract method print() which prints the elements of the stack starting from the last element to the first element.

-a abstract method isEmpty() that would return true if the stack is empty, and false otherwise.

Modify the class GenericStack<T> such that it implements the generic interface GenericStackable<T> . Create a class GenericStackDemo2 and work with two different stacks.

import java.util.\*; interface GenericStackable<T>{

public void push(T x); public T pop(); public void print(); public boolean isEmpty();

}

class GenericStack<T> implements GenericStackable<T> {

int top; int max;

T [] a;

GenericStack(int size){

a=(T[]) new Object[size];

max=size;

top=-1;

}

public void push(T x){

if(top==(max-1)){

System.out.println("Stack is Filled");

}

else{

top++;

a[top]=x;

}

}

public T pop(){

if(top==-1){

System.out.println("Stack is empty");

return null;

}

else{

return a[top--];

}

}

public void print(){ for (int i=top;i>=0;i--){

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

}

System.out.println();

}

public boolean isEmpty(){

if(top==-1){

return true;

}

else{

return false;

}

}

}

public class GenericStackDemo2{ public static void main(String[] args) {

GenericStack<Integer> s =new GenericStack<Integer>(3); GenericStack<String> str =new GenericStack<String>(3);

s.push(11228);

s.push(220102);

s.push(310201);

s.print();

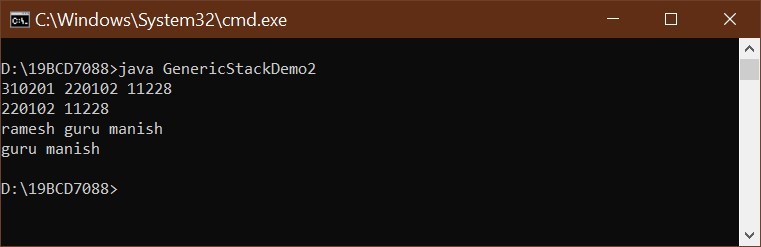
s.pop();

s.print();

str.push("manish"); str.push("guru"); str.push("ramesh"); str.print(); str.pop(); str.print();

}

}



Lab 6

L1 slot

Valiveti manikanta bhuvanesh

1.

19BCD7088

Create the generic interface GenericQueuable<T> that contains the following abstract methods:

* an abstract method insertEnd(T e) which adds the element to the Queue at end.
* an abstract method removeBegin() which removes a element from the begining of the queue.
* an abstract method printQueue() which prints the queue elements from the front of the queue to end.
* an abstract method isQueueEmpty() which returns true if the queue is empty otherwise return false.

Here 'T' should be bounded by Person and its Children.(Refer Lab3 Exercise Question 5 to know the Person hierarchy).

Create GenericQueue<T> such that it implements GenericQueuable<T>. Write a GenericQueueDemo class to test the operations of GenericQueue class with two different queues.

interface GenericQueuable<T>{ public void insertEnd(T e); public void removeBegin(); public void printQueue(); public boolean isQueueEmpty();

}

class GenericQueue<T> implements GenericQueuable<T>{ int front,rear;

int max;

T a[];

GenericQueue(int max){

this.max=max; front=rear=0;

a=(T[]) new Object[max];

}

public void insertEnd(T e){

if(rear==max){

System.out.println("Queue is full");

}

else{ a[rear]=e; rear++;

}

}

public void removeBegin(){

if(front==rear){

System.out.println("Queue is empty");

}

else{

for(int j=0;j<rear-1;j++){

a[j]=a[j+1];

}

rear--;

}

}

public void printQueue(){ for(int i=front;i<rear;i++){

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

}

System.out.println();

}

public boolean isQueueEmpty(){

if(rear==front){

return true;

}

else{

return false;

}

}

}

public class GenericQueueDemo{ public static void main(String[] args) {

GenericQueue<Person> g = new GenericQueue<Person>(3);

Person p1=new Person("guru","thota","vijayawada",502355,9555560461L);

Person p2=new CollegeEmployee("lokesh","nara","hyderabad",500001,9755552147L,64,800000,"ECE");

Person p3=new Faculty("nilesh","kota","guntur",522001,7555799528L,54,1100000,"CSE",true);

Person p4=new Student("ron","weasly","agiripalli",521211,9014914993L,"CSE",8.89);

g.insertEnd(p1);

g.insertEnd(p2);

g.insertEnd(p3);

g.printQueue();

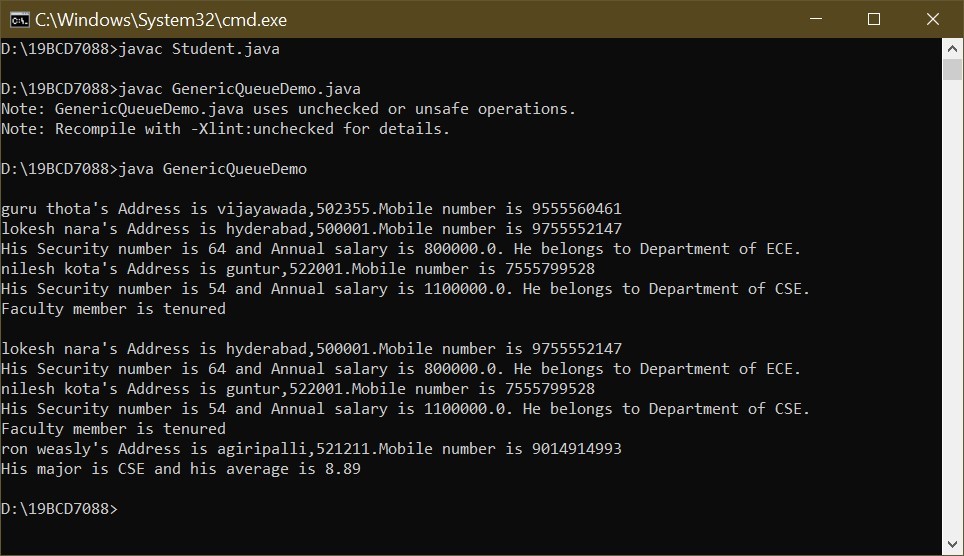
g.removeBegin();

g.insertEnd(p4);

g.printQueue();

}

}



2.a.

Create a generic Map interface MyMap<K,V> that represents a Map structure. K is the type for a key, V is the type of a value. A key is unique in a map and cannot be repeated.

The map contains the following abstract methods.

-an abstract method add(K key, V value) which adds the element to the map

-an abstract method remove(K key) which removes the element with the specified key from the map and returns the value removed.

-a abstract method size() which returns the size of the map

-a abstract method isEmpty() that would return true if the map is empty, and false otherwise.

-a abstract method keys() that returns the list of all keys.

-a abstract method print() that prints all the elements of the map.

2.b.

Create a generic class MyMapImpl that implements the interface MyMap. Use an ArrayList or array to store the keys, and another ArrayList or array to store the values.

2.c.

Create a test class "MyMapTest" that creates two Maps.

Map1: <String, Integer> where the key is a String and the value is an Integer

Map2: <Integer, Double> where the key is a Integer and the value is an Double

Add several elements. Try to add elements with the same key, and check that only one instance is added effectively with no redundancy.

interface MyMap<K,V>{ public void add(K k,V v); public void remove(K k); public int size(); public boolean isEmpty(); public K []keys(); public void print();

}

class MyMapImpl<K,V> implements MyMap<K,V>{

K a[]; V b[];

int p,max,size,ind;

boolean o; MyMapImpl(int n){

p=-1; max=n;

size = 0;

a=(K[]) new Object[n]; b=(V[]) new Object[n];

}

public boolean isPresent(K k){

for (int i=0;i<size;i++){

if(k.equals(a[i])){

o=true;

}

}

if(o){ return false;

}

else{

return true;

}

}

public void add(K k ,V v){

if(isPresent(k)){ if(size==max){

System.out.println("Map is full");

}

else{

p++;

a[p]=k; b[p]=v; size ++;

}

}

else{

System.out.println(k +" Key is already defined");

}

}

public void remove(K k){

if(size!=0){

for(int j=0;j<size;j++){

if(k.equals(a[j])){

ind=j;

}

}

for(int l=ind;l<size-ind;l++){

a[l]=a[l+1]; b[l]=b[l+1];

}

size--;

p--;

}

else{

System.out.println("Map is empty");

}

}

public int size(){

return size;

}

public boolean isEmpty(){

if(size==0){

return true;

}

else{

return false;

}

}

public K []keys(){

return a;

}

public void print(){

for(int m=0;m<size;m++){

System.out.println(a[m] + " : " + b[m]);

}

}

}

public class MyMapTest{ public static void main(String[] args) {

MyMapImpl<String, Integer> r = new MyMapImpl<String, Integer>(3);

MyMapImpl<Integer, Double> u = new MyMapImpl<Integer, Double>(3);

r.add("bhuvanesh",22222);

r.add("guru",52652);

r.add("manish",85642);

r.print();

System.out.println("After removeing one element");

r.remove("guru");

r.print();

r.add("manish",1245);

System.out.println();

u.add(7088,665.265);

u.add(7034,523.245);

u.add(7110,745.261);

u.print();

System.out.println("After removeing one element");

u.remove(7110);

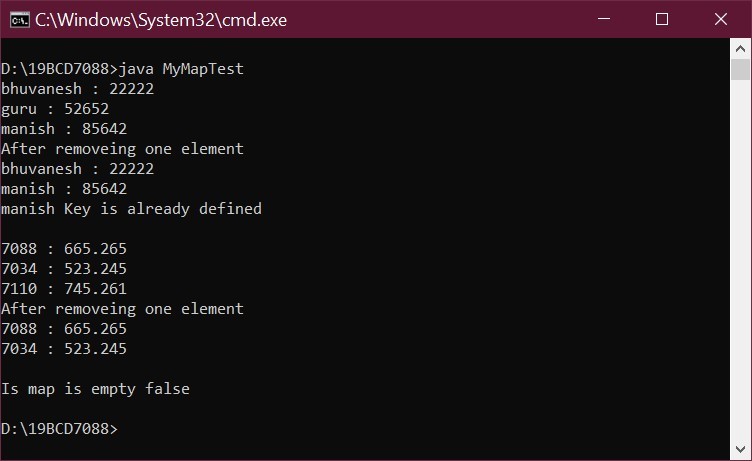
u.print();

System.out.println();

System.out.println("Is map is empty " + u.isEmpty());

}

}



1.

a.

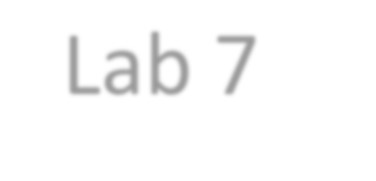
Write a program that declares a named constant to hold the number of quarts in a gallon (4). Also declare a variable to represent the number of quarts needed for a painting job, and assign an appropriate value—for example, 18. Compute and display the number of gallons and quarts needed for the job. Display explanatory text with the values—for example, A job that needs 18 quarts requires 4 gallons plus 2 quarts. Save the program as QuartsToGallons.java.

public class QuartsToGallons{ public static void main(String args[]) { int qg = 4; int n = 18;

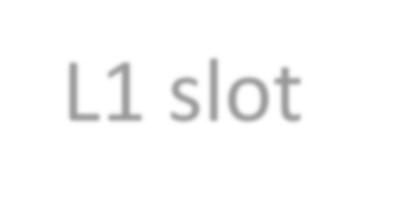
System.out.println("A job that needs " + n + " quarts required " + n/qg + " gallons plus " +n%qg + " quarts");

}

}



Lab 7



L1 slot



Valiveti manikanta bhuvanesh

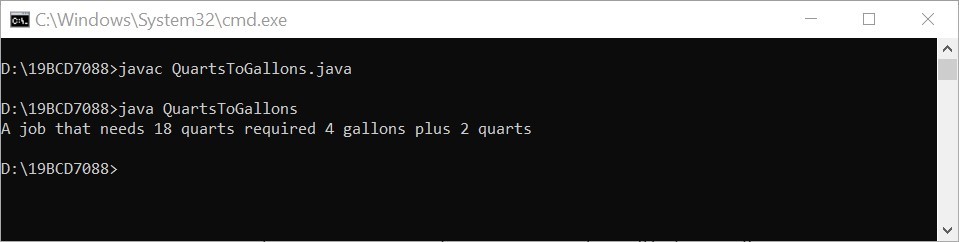


19

BCD

7088



 b.

Convert the QuartsToGallons program to an interactive application. Instead of assigning a value to the number of quarts, accept the value from the user as input. Save the revised program as QuartsToGallonsInteractive.java.

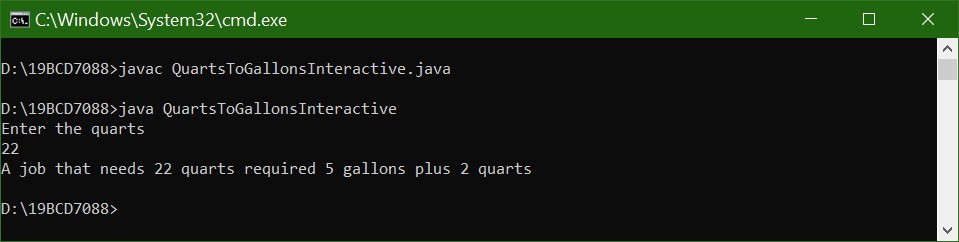
import java.util.Scanner; public class QuartsToGallonsInteractive{ public static void main(String args[]) {

Scanner sc=new Scanner (System.in); int qg = 4;

System.out.println("Enter the quarts"); int n = sc.nextInt(); System.out.println("A job that needs " + n + " quarts required " + n/qg + " gallons plus " +n%qg + " quarts");

}

}



c.

Now, add exception-handling capabilities to this program and continuously reprompt the user while any nonnumeric value is entered. Save the file as QuartsToGallonsWithExceptionHandling.java.

import java.util.\*;

public class QuartsToGallonsWithExceptionHandling{ public static void main(String[] args) {

Scanner sc=new Scanner (System.in);

int qg = 4; int p=1;

while(p!=0){

try{

System.out.println("Enter the quarts"); int n = Integer.parseInt(sc.nextLine());

System.out.println("A job that needs " + n + " quarts required " + n/qg + " gallons plus " +n%qg + " quarts");

p=0;

}

catch(Exception e){

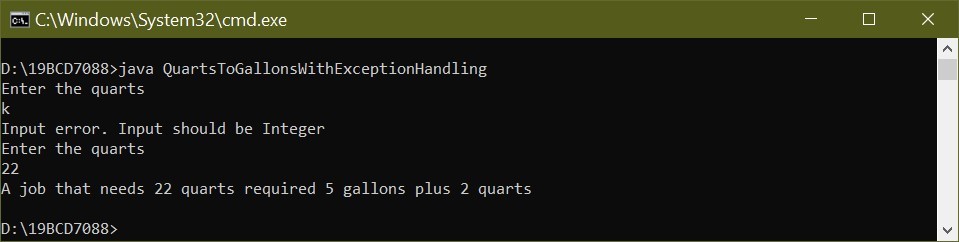
System.out.println("Input error. Input should be Integer");

}

}

}

}



2.

a.

Allow a user to enter any number of double values up to 15. The user should enter 99999 to quit entering numbers. Display an error message if the user quits without entering any numbers; otherwise, display each entered value and its distance from the average. Save the file as DistanceFromAverage.java. import java.util.Scanner; public class DistanceFromAverage{ public static void main(String[] args) {

Scanner sc = new Scanner(System.in); double[] a=new double[15];

double sum=0; int i=0,k=0; double avg,n=0;

System.out.println("For exiting enter 9999 as value");

while(i<a.length){

System.out.println("Enter the value"); a[i]=Double.parseDouble(sc.nextLine());

sum=sum+a[i];

n=a[i];

if(n==99999){ k=i;

i=a.length;

}

i++;

}

if(n==99999){ i=k;

System.out.println((a.length-i) + " values not entered");

}

else{

avg=sum/(i+1);

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

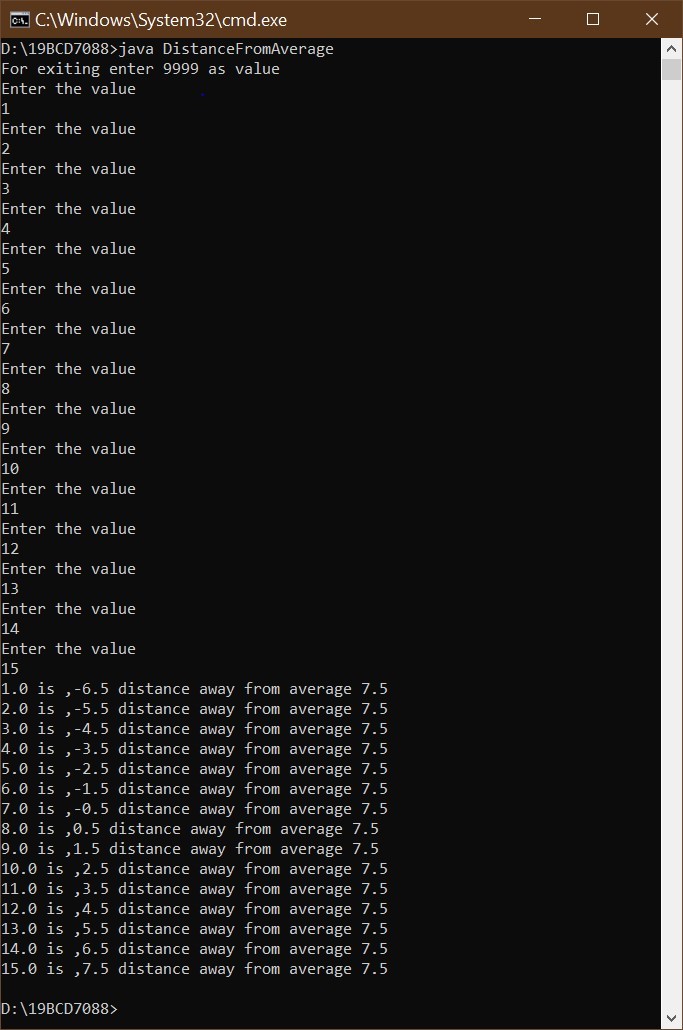
System.out.println(a[j] + " is ," + (a[j]-avg) + " distance away from average "+avg);

}

}

}

}



b.

Now, modify that program to first prompt the user to enter an integer that represents the array size. Java generates a NumberFormatException if you attempt to enter a noninteger value using nextInt(); handle this exception by displaying an appropriate error message. Create an array using the integer entered as the size.

Java generates a NegativeArraySizeException if you attempt to create an array with a negative size; handle this exception by setting the array size to a default value of five.

If the array is created successfully, use exception-handling techniques to ensure that each entered array value is a double before the program calculates each element’s distance from the average. Save the file as DistanceFromAverageWithExceptionHandling.java.

import java.util.\*; public class DistanceFromAverageWithExceptionHandling{ public static void main(String[] args) {

Scanner sc = new Scanner(System.in); double[] a=new double[5];

double sum=0; int i=0,k=0; double avg,n=0; int size=5;

int p=1;

while(p!=0){

try {

System.out.println("Enter array size"); size=Integer.parseInt(sc.nextLine());

p=0;

}

catch (NumberFormatException e) {

System.out.println("Size should be a number");

}

}

try{

a= new double[size];

}

catch(NegativeArraySizeException e){

size=5;

System.out.println("Array size can not be negative value taking 5 as default

value of size");

}

while(i<a.length){

try{

System.out.println("Enter the value"); a[i]=Double.parseDouble(sc.nextLine());

sum=sum+a[i]; n=a[i];

if(n==9999){ k=i;

i=a.length;

}

i++;

}

catch(Exception e){

System.out.println("Value should be double");

}

}

if(n==9999){ i=k;

System.out.println((a.length-i) + " values not entered");

}

else{

avg=sum/(i+1);

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

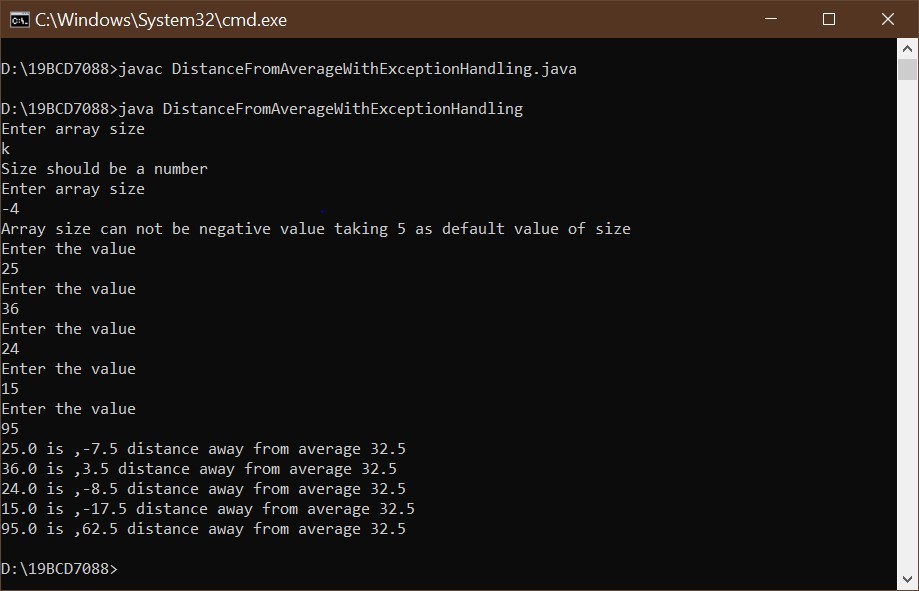
System.out.println(a[j] + " is ," + (a[j]-avg) + " distance away from average "+avg);

}

}

}

}



3.

a.

Create a CourseException class that extends Exception and whose constructor receives a String that holds a college course’s department (for example, CIS), a course number (for example, 101), and a number of credits (for example, 3). Save the file as CourseException.java. Create a Course class with the same fields and whose constructor requires values for each field. Upon construction, throw a CourseException if the department does not consist of three letters, if the course number does not consist of three digits between 100 and 499 inclusive, or if the credits are less than 0.5 or more than 6. Save the class as Course.java. Write an application that establishes an array of at least six Course objects with valid and invalid values. Display an appropriate message when a Course object is created successfully and when one is not. Save the file as ThrowCourseException.java.

class CourseException extends Exception{ CourseException(String msg) {

super(msg);

}

}

class Course {

String dept;

int cn;

double c; public Course(String dept, int cn, double c) throws CourseException {

if(dept.length()!=3 || (cn<100 || cn>499) || (c<0.5 || c>6)) { throw new CourseException("Error in given details");

}

this.dept = dept; this.cn = cn; this.c = c;

System.out.println("Created successfully");

}

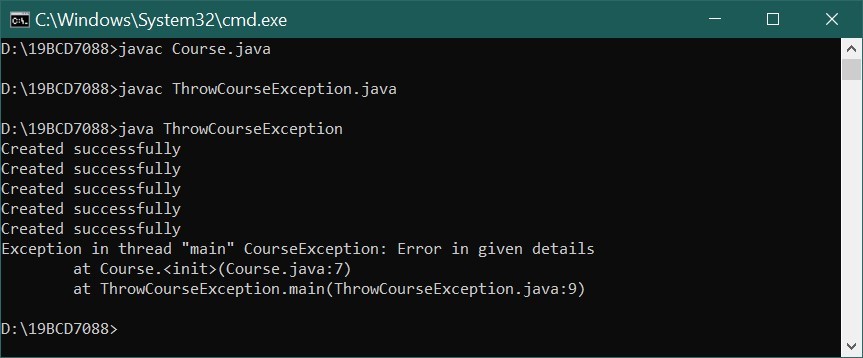
}

public class ThrowCourseException{ public static void main(String[] args) throws CourseException{

Course c[]=new Course[6]; c[0]=new Course("BCE",350,5); c[1]=new Course("BCD",260,3); c[2]=new Course("BCI",275,4); c[3]=new Course("BCN",150,0.7); c[4]=new Course("BCR",185,1); c[5]=new Course("BCB",230,7);

}

}



3.b.

Modify the CourseException class to extend RuntimeException class and identify the differences.

class CourseException extends RuntimeException{ CourseException(String msg) {

super(msg);

}

}

class Course {

String dept;

int cn; double c;

public Course(String dept, int cn, double c) throws CourseException { if(dept.length()!=3 || (cn<100 || cn>499) || (c<0.5 || c>6)) { throw new CourseException("Error in given details");

}

this.dept = dept; this.cn = cn; this.c = c;

System.out.println("Created successfully");

}

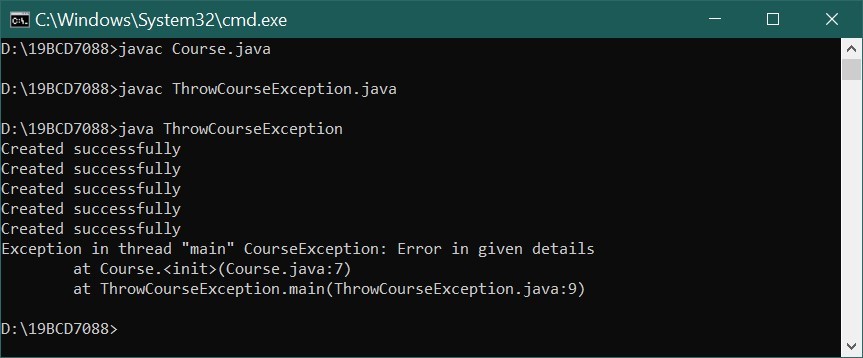
}

public class ThrowCourseException{ public static void main(String[] args) throws CourseException{

Course c[]=new Course[6]; c[0]=new Course("BCE",350,5); c[1]=new Course("BCD",260,3); c[2]=new Course("BCI",275,4); c[3]=new Course("BCN",150,0.7); c[4]=new Course("BCR",185,1); c[5]=new Course("BCB",230,7);

}

}



1.

Write a multithreaded program to compute sum of elements of NxN matrix. It should be done in two phases.

Phase I:

Create N threads to compute 'N' Columns sum, where 'i' th thread computes a 'i' th column sum.

Phase II:

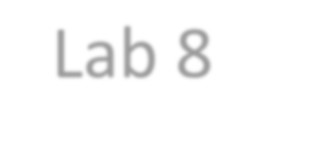
Create a thread to compute sum of 'N' Column's Sums. Finally main thread is going to print result.

Note: Main thread should wait till all threads completes their work.

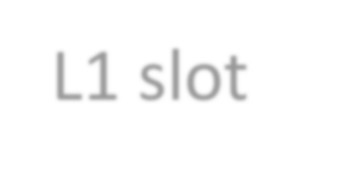
You can use Math.random() to initialize NxN matrix, in the range 0 to 100; 5<=N<=8.

Implement this application in two versions, using both Thread class & Runnable interface.

import java.lang.\*; import java.util.\*; public class Sum extends Thread{



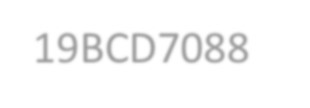
Lab 8



L1 slot



Valiveti manikanta bhuvanesh



19

BCD7088

public static int n=5; public static int sum; public static int[][]a=new int[n][n]; public static int[]s=new int[n];

static class ColSum extends Thread{

int i;

int sum=0; ColSum(int i){

this.i=i;

}

public void run(){ for(int q=0;q<n;q++){ sum=sum+a[q][i];

}

s[i]=sum;

}

}

static class Tsum extends Thread{

public void run(){

int sum1=0;

for(int m=0;m<n;m++){

sum1=sum1+s[m];

}

sum=sum1;

}

}

public static void main(String[] args) { Random r= new Random();

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

for (int j=0;j<n;j++) {

a[i][j]=r.nextInt(100);

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

}

System.out.println();

}

Sum.ColSum [] c=new Sum.ColSum[n]; for(int k=0;k<n;k++){ c[k]=new Sum.ColSum(k);

}

Sum.Tsum t2=new Sum.Tsum();

for(int y=0;y<n;y++){ c[y].start();

}

t2.start();

Thread t = Thread.currentThread();

try{

t.sleep(1500);

}

catch(Exception e){

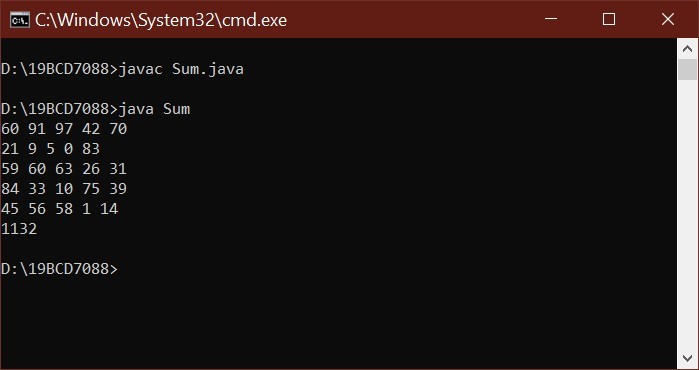
System.out.println(e);

}

System.out.println(sum);

}

}



import java.lang.\*; import java.util.\*; public abstract class Sum1 implements Runnable{

public static int n=5; public static int sum; public static int[][]a=new int[n][n]; public static int[]s=new int[n];

static class ColSum1 implements Runnable{

int i;

int sum=0;

ColSum1(int i){

this.i=i;

}

public void run(){ for(int q=0;q<n;q++){ sum=sum+a[q][i];

}

s[i]=sum;

}

}

static class Tsum implements Runnable{

public void run(){

int sum1=0;

for(int m=0;m<n;m++){

sum1=sum1+s[m];

}

sum=sum1;

}

}

public static void main(String[] args) { Random r= new Random(); for(int i=0;i<n;i++) {

for (int j=0;j<n;j++) { a[i][j]=r.nextInt(100);

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

}

System.out.println();

}

Thread [] t1=new Thread[n]; for(int k=0;k<n;k++){ t1[k]=new Thread(new Sum1.ColSum1(k)); }

Thread t2=new Thread (new Sum1.Tsum());

for(int y=0;y<n;y++){ t1[y].start();

}

t2.start();

Thread t = Thread.currentThread();

try{

t.sleep(1500);

}

catch(Exception e){

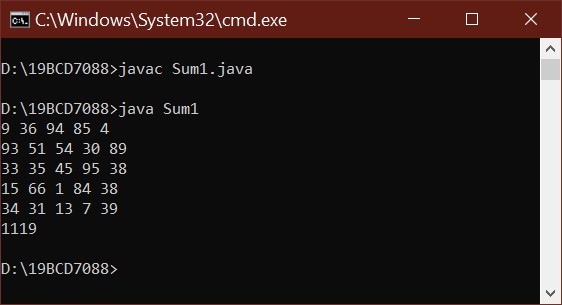
System.out.println(e);

}

System.out.println(sum);

}

}



2.

a.

Create a CeaserCipher class to perform substitution and reverse substitution of characters of a message.

* mEncryption method - substitute a character with another charcter of alphabet.
* mDecryption method - similar to mEncryption method but it perform in reverse.

Each character of message is considered as numeric value with the following mapping:a-z to 0-25, respectively.

(a-0,b-1,c-2,....z-25)

The mEncryption method replaces each character of the message with another character by using the following formula:(N(ch)+k)%26, where N(ch) means Numeric value of a character 'ch', k means key value 0<=k<=25.

The mDecryption method substitutes each character with the following formula: (N(ch)-k)%26.

Inputs to each method is a message and a key and output is substituted message printed on console character by character.

(Ex: Input to mEncryption is: rama and 25 and output is: qzlz ;

Input to mDecryption is: qzlz and 25 and output is: rama )

Create a TestCeaserCipher class to test mEncryption & mDecryption methods.

import java.util.Scanner; class TestCeaserCipher

{

public static String a = "abcdefghijklmnopqrstuvwxyz"; public static String mEncryption(String p, int key)

{

p = p.toLowerCase();

String c = ""; for (int i = 0; i < p.length(); i++) {

int cp = a.indexOf(p.charAt(i)); int kv = (key + cp) % 26; char rv = a.charAt(kv); c += rv;

}

return c;

}

public static String mDecryption(String c, int key)

{

c = c.toLowerCase();

String p = ""; for (int i = 0; i < c.length(); i++)

{

int cp = a.indexOf(c.charAt(i)); int kv = (cp - key) % 26; if (kv < 0)

{

kv = a.length() + kv;

}

char rv = a.charAt(kv); p += rv;

}

return p;

}

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in);

System.out.println("Enter the Message to encrypt");

String m =sc.nextLine();

System.out.println("Enter the Key for encryption"); int key = sc.nextInt();

String c=mEncryption(m, key);

System.out.println("Encrypted message " + c);

System.out.println("Deprecated message " + mDecryption(c, key));

}

}



2.

b.

Jennifer comes with a message "gdhrzfnncanx". She wants to perform reverse substitution using mDecryption method but not aware of key 'k'. To help her, develop a multithreaded program to create separate thread for each possible key 'k' and print all reverse substitutions. Do necessary changes to CeaserCipher class and provide synchronization for threads if the output from threads are mixed. Name the file as BruteForceCeaserCipher.java.

public class BruteForceCeaserCipher extends Thread

{

public static String m = "gdhrzfnncanx"; public static String a = "abcdefghijklmnopqrstuvwxyz"; static class Decrypt extends Thread

{

int key;

Decrypt(int key)

{

this.key=key;

}

String l = m.toLowerCase();

String p = ""; public synchronized void decrypt(){

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

{

int cp = a.indexOf(l.charAt(i)); int kv = (cp - key) % 26;

if (kv < 0)

{

kv = a.length() + kv;

}

char r = a.charAt(kv);

p += r;

}

System.out.println("Message for key "+key+ " is " + p);

}

public void run(){

decrypt();

}

}

public static void main(String[] args)

{

BruteForceCeaserCipher.Decrypt [] c=new BruteForceCeaserCipher.Decrypt[26]; for(int k=0;k<26;k++){

c[k]=new BruteForceCeaserCipher.Decrypt(k);

}

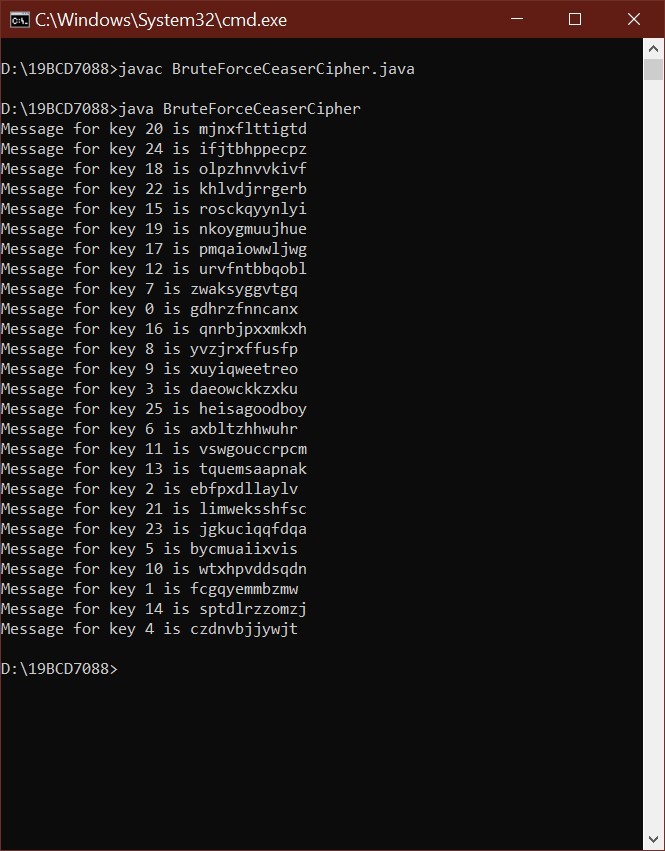
for(int y=0;y<26;y++){

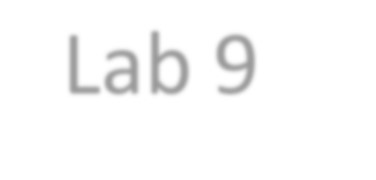
c[y].start();

}

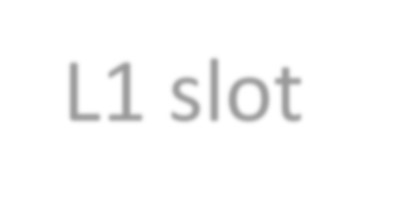
}

}





Lab 9



L1 slot



Valiveti manikanta bhuvanesh



19

BCD7088

1. Apply Interthread communication to solve the Producer-Consumer problem with a common or shared bounded buffer(Queue) holding upto 5 elements. The producer consumer problem is a synchronization problem. There is a fixed size buffer and the producer produces items and enters them into the buffer. The consumer removes the items from the buffer and consumes them. A producer should not produce items into the buffer when the consumer is consuming an item from the buffer and vice versa. So the buffer should only be accessed by the producer or consumer at a time. When ever buffer is filled up and no more space to add the element into the queue(buffer) producerhas to wait until the buffer is emptied by consumer. When ever the buffer is empty and no more items are available for consumption the consumer should wait for producer to

produceelements. Write a solution for N elements, where N is multiple of 5 or greater than 5 other than0 import java.util.\*; class Queue

{

int n;

boolean run = false; synchronized int get()

{

while(!run)

try

{

wait();

}

catch(InterruptedException e)

{

System.out.println(e);

}

System.out.println("Produced " + n); run=false;

notify();

return n;

}

synchronized void set(int n)

{

while(run)

try

{

wait();

}

catch(InterruptedException e)

{

System.out.println(e);

}

this.n = n; run = true;

System.out.println("Consumed " + n); notify();

}

}

class Producer implements Runnable

{

Queue q;

Thread t;

Producer(Queue q)

{

this.q=q; t=new Thread(this, "Producer");

}

public void run()

{ int i = 0; while (true)

{ if(i>5)

{

System.exit(0);

}

q.set(i++);

}

}

}

class Consumer implements Runnable

{

Queue q;

Thread t;

Consumer (Queue q)

{

this.q = q;

t=new Thread (this,"Consumer");

}

public void run ()

{

while(true)

{

q.get();

}

}

}

class Sell

{

public static void main(String args[])

{

Queue q =new Queue();

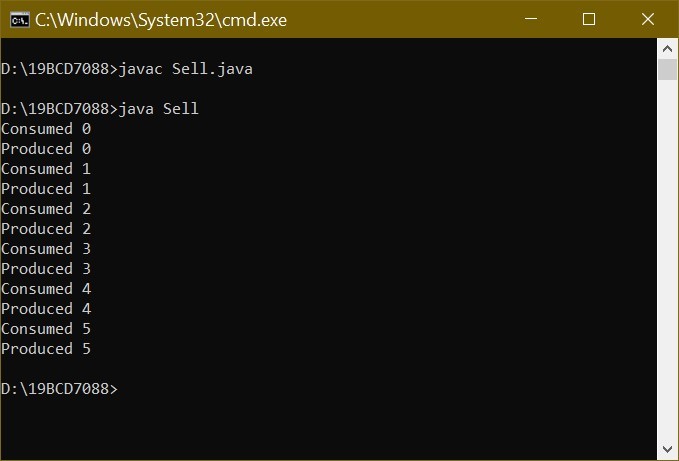
Producer p = new Producer (q); Consumer c = new Consumer (q);

p.t.start();

c.t.start();

}

}



2. Develop a simple chat application between two users using "Send-Wait-Receive" Protocol: Once a user sends a message, he waits till a message is received from other user. The users are "User1" and "User2". At initial stage of application, User1 is in sending mode and User2 is in receiving mode. These two users are sending and receiving the messages alternatively. -Create a Chat class with two methods: sendMessage and recvMessage – Create two threads to represent two users,User1 and User2. -Use Interthread communication to exchange messages. -No need to maintain any chat history. Example: User1: Hi User2: Hello User1: R u preparing for exam? User2: Yes User1: Ok. Bye User2: Bye. import java.util.\*;

class Queue1

{

Scanner sc=new Scanner(System.in);

int n;

String msg;

boolean run = false; synchronized int recvMessage()

{

while(!run)

try

{

wait();

}

catch(InterruptedException e)

{

System.out.println(e);

}

System.out.print("User2: "); msg=sc.nextLine(); run = true; System.out.println(); run=false;

notify(); return n;

}

synchronized void sendMessage(int n)

{

while(run)

try

{

wait();

}

catch(InterruptedException e)

{

System.out.println(e);

}

this.n = n; System.out.print("User1: "); msg=sc.nextLine(); run = true; System.out.println();

notify();

}

}

class User1 implements Runnable

{

Queue1 que;

Thread t;

User1(Queue1 que)

{

this.que=que; t=new Thread(this, "User1");

}

public void run()

{

int i = 0;

while (true)

{

if(i>5)

{

System.exit(0);

}

que.sendMessage(i++);

}

}

}

class User2 implements Runnable {

Queue1 que;

Thread t;

User2 (Queue1 que)

{

this.que = que; t=new Thread (this,"User2");

}

public void run ()

{

while(true)

{

que.recvMessage();

}

}

}

public class Message{ public static void main(String args[])

{

Queue1 que =new Queue1();

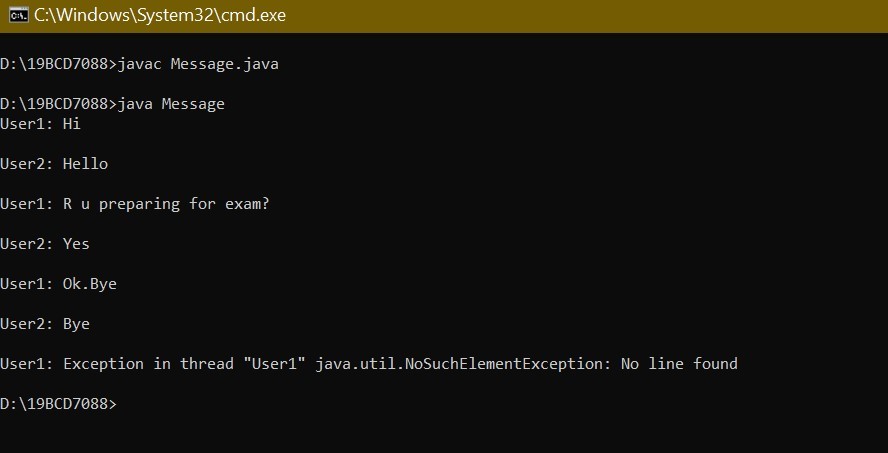
User1 u1 = new User1 (que);

User2 u2 = new User2 (que);

u1.t.start(); u2.t.start();

}

}

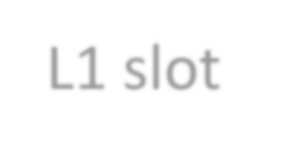


1.

Create an application for Paula’s Portraits, a photography studio. The application allows users to compute the price of a photography session. Paula’s base price is $40 for an in-studio photo session with one person. The in-studio fee is $75 for a session with two or more subjects, and $95 for a session with a pet. A $90 fee is added to take photos on location instead of in the studio. Include a set of mutually exclusive check boxes to select the portrait subject and another set for the session location. Include labels as appropriate to explain the application’s functionality. Save the file as JPhotoFrame.java. import java.awt.\*; import java.awt.event.\*; import javax.swing.\*;



Lab 10



L1 slot



Valiveti manikanta bhuvanesh



19

BCD7088

class MyWindowAdapter extends WindowAdapter{ public void windowClosing(WindowEvent we) {

System.exit(0);

}

}

class DataType{

int single=40; int dual=75; int pet=95; int onl=90; int sum=0;

JCheckBox ib,ob,sb,tb,pb;

JTextField total;

}

class CheckMeListener extends DataType implements ItemListener{ public void itemStateChanged(ItemEvent e){ Object source=e.getSource(); int select=e.getStateChange();

if(source==ib)

{

if(select==ItemEvent.SELECTED){

sum+=single;

}

else{

sum-=single;

}

}

else if(source==ob)

{

if(select==ItemEvent.SELECTED){

sum+=onl;

}

else{ sum-=onl;

}

}

else if(source==sb)

{

if(select==ItemEvent.SELECTED){

sum+=single;

}

else{

sum-=single;

}

}

else if(source==tb)

{

if(select==ItemEvent.SELECTED){

sum+=dual;

}

else{

sum-=dual;

}

}

else if(source==pb)

{

if(select==ItemEvent.SELECTED){

sum+=pet;

}

else{ sum-=pet;

}

}

total.setText("$"+sum);

}

}

public class JPhotoFrame{ public static void main(String[] args) {

CheckMeListener y=new CheckMeListener(); JFrame f = new JFrame("Photo price calculator");

f.setSize(400,150); Panel p = new Panel();

p.setLayout(new FlowLayout());

y.ib=new JCheckBox("In studio");

y.ob=new JCheckBox("out studio");

y.sb=new JCheckBox("One Person");

y.tb=new JCheckBox("Two Subjects");

y.pb=new JCheckBox("Pet");

y.total=new JTextField("TOTAL",15);

ButtonGroup lg=new ButtonGroup();

lg.add(y.ib); lg.add(y.ob);

p.add(new JLabel("Select one location"));

p.add(y.ib);

p.add(y.ob);

ButtonGroup m=new ButtonGroup();

m.add(y.sb);

m.add(y.tb);

m.add(y.pb);

p.add(new JLabel("Select one Subject"));

p.add(y.sb);

p.add(y.tb);

p.add(y.pb);

y.ib.addItemListener(y);

y.ob.addItemListener(y);

y.sb.addItemListener(y);

y.tb.addItemListener(y);

y.pb.addItemListener(y);

p.add(y.total);

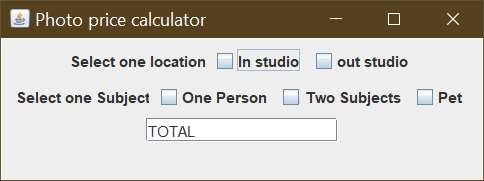
f.add(p);

f.addWindowListener(new MyWindowAdapter());

f.setVisible(true);

}

}



2.

Write an application for Lambert’s Vacation Rentals. Use separate ButtonGroups to allow a client to select one of three locations, the number of bedrooms, and whether meals are included in the rental. Assume that the locations are parkside for $600 per week, poolside for $750 per week, or lakeside for $825 per week. Assume that the rentals have one, two, or three bedrooms and that each bedroom greater than one adds $75 to the base price. Assume that if meals are added, the price is $200 more per rental. Save the file as JVacationRental.java.

import java.awt.\*; import java.awt.event.\*; import javax.swing.\*;

class MyWindowAdapter extends WindowAdapter{ public void windowClosing(WindowEvent we) {

System.exit(0);

}

}

class Listeners { static int pars= 600; static int pos = 750; static int ls = 825; static int b1= 75; static int b2 = 150; static int b3 = 225; static int m = 200; static int lr = 0; static int br = 0; static int mc = 0;

static JRadioButton park,pool,lake,one,two,three,yes,no;

static JButton Cal; static JTextField total;

static class ClickMeListener implements ActionListener{

public void actionPerformed(ActionEvent e)

{

Object source = e.getSource(); if(source == Cal){ double totalRent = lr + br + mc; total.setText("Total amount $ " + totalRent);

}

}

}

static class blistener implements ItemListener{ public void itemStateChanged(ItemEvent e){ Object source = e.getItem(); if(source == one){ br = b1;

}

else if(source == two){

br = b2;

}

else if(source == lake){ br=b3;

}

else{ br=0;

}

}

}

static class llistener implements ItemListener{ public void itemStateChanged(ItemEvent e)

{

Object source = e.getItem();

if(source == park){

lr = pars;

}

else if(source == pool){ lr = pos;

}

else if(source == lake){

lr = ls;

}

else{ lr=0;

}

}

}

static class mlistener implements ItemListener{ public void itemStateChanged(ItemEvent e)

{

Object source = e.getItem(); if(source == yes){ mc = m;

}

else if(source == no){ mc = 0;

}

else{ mc= 0;

}

}

}

}

public class JVacationRental{

public static void main(String[] args) {

Listeners y = new Listeners();

Frame f = new JFrame("Rental price calculator");

f.setSize(350,160); Panel p = new Panel();

p.setLayout(new FlowLayout());

y.park= new JRadioButton("park side");

y.pool= new JRadioButton("pool side");

y.lake= new JRadioButton("lake side");

y.one= new JRadioButton("one room");

y.two= new JRadioButton("two room");

y.three= new JRadioButton("three room");

y.yes= new JRadioButton("Yes");

y.no= new JRadioButton("No");

y.Cal= new JButton("Total");

y.total=new JTextField(15); ButtonGroup l=new ButtonGroup();

l.add(y.park);

l.add(y.pool);

l.add(y.lake);

ButtonGroup b=new ButtonGroup();

b.add(y.one);

b.add(y.two);

b.add(y.three);

ButtonGroup yn=new ButtonGroup(); yn.add(y.yes); yn.add(y.no);

p.add(new JLabel("Location"));

p.add(y.park);

p.add(y.pool);

p.add(y.lake);

p.add(new JLabel("Rooms"));

p.add(y.one);

p.add(y.two);

p.add(y.three);

p.add(new JLabel("Meals"));

p.add(y.yes);

p.add(y.no);

p.add(y.Cal);

p.add(y.total);

y.park.addItemListener(new Listeners.llistener());

y.pool.addItemListener(new Listeners.llistener());

y.lake.addItemListener(new Listeners.llistener());

y.one.addItemListener(new Listeners.blistener());

y.two.addItemListener(new Listeners.blistener());

y.three.addItemListener(new Listeners.blistener());

y.yes.addItemListener(new Listeners.mlistener());

y.no.addItemListener(new Listeners.mlistener());

y.Cal.addActionListener(new Listeners.ClickMeListener());

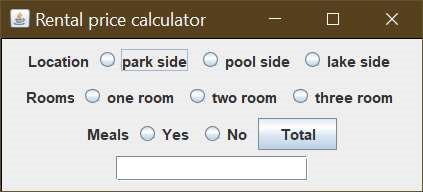
f.add(p);

f.addWindowListener(new MyWindowAdapter());

f.setVisible(true);

}

}



1.

Write a JavaFX application that allows the user to choose insurance options. Use a ToggleGroup to allow the user to select only one of two 8 types—HMO (health maintenance organization) or PPO (preferred provider organization). Use CheckBoxes for dental insurance and vision insurance options; the user can select one option, both options, or neither option. As the user selects each option, display its name and price in a text field; the HMO costs $200 per month, the PPO costs $600 per month, the dental coverage adds $75 per month, and the vision care adds $20 per month. Save the application as FXInsurance.java.

import java.awt.\*; import java.awt.event.\*; import javax.swing.\*;

class MyWindowAdapter extends WindowAdapter{ public void windowClosing(WindowEvent we) {

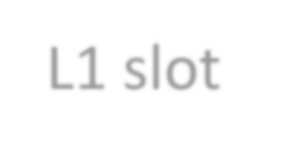
System.exit(0);

}

}



Lab 11



L1 slot



Valiveti manikanta bhuvanesh



19

BCD7088

class CheckMeListener implements ItemListener {

JCheckBox hb,pb,db,vb; JTextField disp; public void itemStateChanged(ItemEvent e){ Object source=e.getSource(); int select=e.getStateChange();

if(source==hb){ if(select==ItemEvent.SELECTED){ disp.setText("HMO costs $200 per month");

}

else{

disp.setText("");

}

}

else if(source==pb){

if(select==ItemEvent.SELECTED){ disp.setText("PPO costs $600 per month");

}

else{

disp.setText("");

}

}

else if(source==db){ if(select==ItemEvent.SELECTED){ disp.setText("dental coverage adds $75 per month");

}

else{

disp.setText("");

}

}

else if(source== vb){ if(select==ItemEvent.SELECTED){ disp.setText("vision care adds $20 per month");

}

else{

disp.setText("");

}

}

else{

disp.setText("");

}

}

}

public class FXInsurance{ public static void main(String[] args) { CheckMeListener y=new CheckMeListener(); JFrame f = new JFrame("Amount displayer");

f.setSize(400,150); Panel p = new Panel();

p.setLayout(new FlowLayout());

y.hb=new JCheckBox("HMO");

y.pb=new JCheckBox("PPO");

y.db=new JCheckBox("Dental");

y.vb=new JCheckBox("Vision");

y.disp=new JTextField("displayer",20); ButtonGroup bg=new ButtonGroup();

bg.add(y.hb); bg.add(y.pb);

p.add(new JLabel("Select one type of insurance"));

p.add(y.hb);

p.add(y.pb);

p.add(new JLabel("Select one or both or neither of them"));

p.add(y.db);

p.add(y.vb);

p.add(y.disp);

y.hb.addItemListener(y);

y.pb.addItemListener(y);

y.db.addItemListener(y);

y.vb.addItemListener(y);

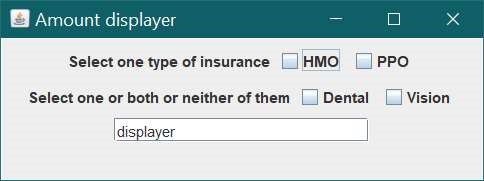
f.add(p);

f.addWindowListener(new MyWindowAdapter());

f.setVisible(true);

}

}



2.

Design a JavaFX application for the Sublime Sandwich Shop. The user makes sandwich order choices from list boxes, and the application displays the price. The user can choose from three main sandwich ingredients of your choice (for example, chicken) at three different prices. The user also can choose from three different bread types (for example, rye) from a list of at least three options.

Save the application as FXSandwich.java.

import java.awt.\*; import java.awt.event.\*; import javax.swing.\*; import javax.swing.event.\*; class MyWindowAdapter extends WindowAdapter{ public void windowClosing(WindowEvent we) {

System.exit(0);

}

}

class Listener implements ListSelectionListener {

int ing,bread;

JList ingl,breads; JTextField total; public void valueChanged(ListSelectionEvent e){ int indexing = ingl.getSelectedIndex(); int indexbread = breads.getSelectedIndex();

if(indexing==0){

ing=200;

}

else if(indexing==1){

ing=180;

}

else if(indexing==2){

ing=118;

}

else{

ing=0;

}

if(indexbread==0){

bread=30;

}

else if(indexbread==1){

bread=50;

}

else if(indexbread==2){

bread=70;

}

else{ bread=0;

}

double sum= ing+bread;

total.setText("Total price $"+sum);

}

}

public class FXSandwich{ public static void main(String[] args) {

String Ingredients [] = {"Panner", "Mushroom", "Potato"};

String breadTypes[] = {"Wheat", "Rye", "Brioche"}; Listener y = new Listener();

Frame f = new JFrame("Sublime Sandwich Shop");

f.setSize(350,160); Panel p = new Panel();

p.setLayout(new FlowLayout());

y.total=new JTextField(15);

y.ingl = new JList<String>(Ingredients);

y.breads = new JList<String>(breadTypes);

p.add(new JLabel("Choose Ingredients"));

p.add(y.ingl);

p.add(new JLabel("Choose Bread"));

p.add(y.breads);

p.add(y.total);

y.ingl.addListSelectionListener(y);

y.breads.addListSelectionListener(y);

f.add(p);

f.addWindowListener(new MyWindowAdapter());

f.setVisible(true);

}

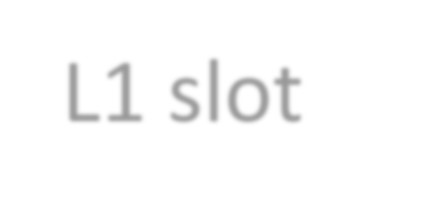
}



Develop a Menu Based GUI using Swings:



Lab12



L1 slot



Valiveti manikanta bhuvanesh



19

BCD7088

Menus:

-Country

-Font

-Color

Menu Items:

-Any 7 country names under Country.

-Name,Type,Size are Menu items under Font.

---Any four font names under Name Menu

---Type should contain Bold,Italic

---Size may contain 12,14,16,18

-Any standard seven colors as Menu items under Color Menu.

If a user clicks any menu item, It should be displayed with specified color and font in window area.

Provide shortcut keys to every menus and menu items like ctrl+o, ctrl+s.

Provide a popup menu to contain two options, clear and exit. When user clicks clear, it should clears the window area and when user clicks exit, it should terminate the application.

Save the file as MenuWithSwings.java import java.awt.\*; import java.awt.event.\*; import javax.swing.\*; import javax.swing.event.\*; public class MenuWithSwing implements ActionListener{

static JFrame f; static Panel p; static JMenuBar mb; static JMenu country,font,name,type,size,colour;

static JMenuItem

times,dia,sansserif,courier,bold,italic,size12,size14,size16,size18,india,aus,brazil,germ,ice,mal,rus,blu

,bro,gray,green,org,red,yel,clc,exit;

static JLabel disp; static JPopupMenu pop;

public static void main(String[] args) {

MenuWithSwing m = new MenuWithSwing(); f = new JFrame("MenuWithSwings ");

f.setSize(150,250); Panel p = new Panel();

p.setLayout(new FlowLayout()); pop =new JPopupMenu(); clc=new JMenuItem("Clear"); clc.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_C, KeyEvent.CTRL\_MASK)); exit=new JMenuItem("Exit"); exit.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_E, KeyEvent.CTRL\_MASK));

pop.add(clc); clc.addActionListener(m); pop.add(exit); exit.addActionListener(m); mb = new JMenuBar(); country =new JMenu("Country");

font = new JMenu("Font"); colour = new JMenu("Colour"); name = new JMenu("Name"); type = new JMenu("Type"); size = new JMenu("Size"); country.setMnemonic('C'); font.setMnemonic('F'); colour.setMnemonic('L'); name.setMnemonic('N'); type.setMnemonic('T'); size.setMnemonic('S'); times=new JMenuItem("TimesRoman"); dia=new JMenuItem("Dialog"); sansserif = new JMenuItem("SansSerif"); courier=new JMenuItem("Courier"); bold = new JMenuItem("Bold"); italic = new JMenuItem("Italic"); size12 = new JMenuItem("Size 12"); size14 = new JMenuItem("Size 14"); size16 = new JMenuItem("Size 16"); size18 = new JMenuItem("Size 18"); india = new JMenuItem("India"); aus =new JMenuItem("Australia"); brazil = new JMenuItem("Brazil"); germ = new JMenuItem("Germany"); ice = new JMenuItem("Iceland"); mal = new JMenuItem("Malaysia"); rus = new JMenuItem("Russia"); blu = new JMenuItem("Blue"); bro =new JMenuItem("Brown"); gray = new JMenuItem("Gray"); green = new JMenuItem("Green"); org = new JMenuItem("Orange"); red = new JMenuItem("Red"); yel=new JMenuItem("Yellow"); times.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_I, KeyEvent.CTRL\_MASK)); dia.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_D, KeyEvent.CTRL\_MASK)); sansserif.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_A, KeyEvent.CTRL\_MASK)); courier.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_U, KeyEvent.CTRL\_MASK)); bold.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_B, KeyEvent.CTRL\_MASK)); italic.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_R, KeyEvent.CTRL\_MASK)); size12.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_F2, 0)); size14.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_F4, 0)); size16.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_F6, 0)); size18.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_F8, 0)); india.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_G, KeyEvent.CTRL\_MASK)); aus.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_H, KeyEvent.CTRL\_MASK)); brazil.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_J, KeyEvent.CTRL\_MASK)); germ.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_K, KeyEvent.CTRL\_MASK)); ice.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_M, KeyEvent.CTRL\_MASK)); mal.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_P, KeyEvent.CTRL\_MASK)); rus.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_Q, KeyEvent.CTRL\_MASK)); blu.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_V, KeyEvent.CTRL\_MASK)); bro.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_W, KeyEvent.CTRL\_MASK)); gray.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_X, KeyEvent.CTRL\_MASK)); green.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_Y, KeyEvent.CTRL\_MASK)); org.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_Z, KeyEvent.CTRL\_MASK)); red.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_F1, 0)); yel.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_F3, 0));

disp = new JLabel(); country.add(india); country.add(aus); country.add(brazil); country.add(germ); country.add(ice); country.add(mal); country.add(rus); mb.add(country); name.add(times); name.add(dia); name.add(sansserif); name.add(courier); font.add(name); type.add(bold); type.add(italic); font.add(type); size.add(size12); size.add(size14); size.add(size16); size.add(size18); font.add(size); mb.add(font); colour.add(blu); colour.add(bro); colour.add(gray); colour.add(green); colour.add(org); colour.add(red); colour.add(yel); mb.add(colour); india.addActionListener(m); aus.addActionListener(m); brazil.addActionListener(m); germ.addActionListener(m); ice.addActionListener(m); mal.addActionListener(m); rus.addActionListener(m); times.addActionListener(m); dia.addActionListener(m); sansserif.addActionListener(m); courier.addActionListener(m); bold.addActionListener(m); italic.addActionListener(m); size12.addActionListener(m); size14.addActionListener(m); size16.addActionListener(m); size18.addActionListener(m); blu.addActionListener(m); bro.addActionListener(m); gray.addActionListener(m); green.addActionListener(m); org.addActionListener(m); red.addActionListener(m); yel.addActionListener(m);

p.addMouseListener(new MouseAdapter() {

public void mouseClicked(MouseEvent e) { pop.show(p , e.getX(), e.getY());

}

});

p.add(mb);

p.add(disp);

p.add(pop);

f.add(p);

f.setVisible(true);

}

public void actionPerformed(ActionEvent o) {

if(o.getSource()==clc){

disp.setText("");

}

if(o.getSource()==exit){

System.exit(0);

}

if(o.getSource()==times){ disp.setFont(new

Font("TimesRoman",disp.getFont().getStyle(),disp.getFont().getSize()));

}

if(o.getSource()==dia){

disp.setFont(new

Font("Dialog",disp.getFont().getStyle(),disp.getFont().getSize()));

}

if(o.getSource()==sansserif){

disp.setFont(new

Font("SansSerif",disp.getFont().getStyle(),disp.getFont().getSize()));

}

if(o.getSource()==courier){

disp.setFont(new

Font("Courier",disp.getFont().getStyle(),disp.getFont().getSize()));

}

if(o.getSource()==bold){

disp.setFont(new

Font(disp.getFont().getName(),Font.BOLD,disp.getFont().getSize()));

}

if(o.getSource()==italic){

disp.setFont(new

Font(disp.getFont().getName(),Font.ITALIC,disp.getFont().getSize()));

}

if(o.getSource()==size12){

disp.setFont(new

Font(disp.getFont().getName(),disp.getFont().getStyle(),12));

}

if(o.getSource()==size14){

disp.setFont(new

Font(disp.getFont().getName(),disp.getFont().getStyle(),14));

}

if(o.getSource()==size16){

disp.setFont(new

Font(disp.getFont().getName(),disp.getFont().getStyle(),16));

}

if(o.getSource()==size18){

disp.setFont(new

Font(disp.getFont().getName(),disp.getFont().getStyle(),18));

}

if(o.getSource()==blu){ disp.setForeground(new Color(0,0,255));

}

if(o.getSource()==bro){ disp.setForeground(new Color(165,42,42));

}

if(o.getSource()==gray){ disp.setForeground(new Color(128,128,128));

}

if(o.getSource()==green){ disp.setForeground(new Color(0,255,0));

}

if(o.getSource()==org){ disp.setForeground(new Color(255,165,0));

}

if(o.getSource()==red){ disp.setForeground(new Color(255,0,0));

}

if(o.getSource()==yel){ disp.setForeground(new Color(255,255,0));

}

if(o.getSource()==india||o.getSource()==aus||o.getSource()==brazil||o.getSource()==germ|

|o.getSource()==ice||o.getSource()==mal||o.getSource()==rus){

disp.setText(o.getActionCommand());

}

}

}



2.

Develop Question 1 using JavaFX. Save the file as MenuWithJavaFX.java import javafx.application.Application; import javafx.scene.Scene; import javafx.scene.control.Label; import javafx.scene.layout.\*; import javafx.event.ActionEvent; import javafx.event.EventHandler; import javafx.scene.control.\*; import javafx.stage.Stage; import javafx.scene.text.\*; import javafx.scene.text.FontWeight; import javafx.scene.paint.\*; import javafx.scene.input.\*; public class MenuWithJavaFX extends Application{

MenuBar mb;

Menu country,font,name,type,size,colour;

MenuItem

times,dia,sansserif,courier,bold,italic,size12,size14,size16,size18,india,aus,brazil,germ,ice,mal,rus,blu ,bro,gray,green,org,red,yel,clc,exit;

Label disp;

ContextMenu pop;

public static void main(String args[]){

launch(args);

}

public void start(Stage s){

s.setTitle("creating MenuBar"); pop= new ContextMenu(); clc=new MenuItem("Clear");

clc.setAccelerator(new KeyCodeCombination(KeyCode.C, KeyCombination.CONTROL\_DOWN));

exit=new MenuItem("Exit");

exit.setAccelerator(new KeyCodeCombination(KeyCode.E, KeyCombination.CONTROL\_DOWN)); pop.getItems().addAll(clc,exit);

mb = new MenuBar();

country =new Menu("Country"); country.setAccelerator(new KeyCodeCombination(KeyCode.C, KeyCombination.ALT\_DOWN));

font = new Menu("Font"); font.setAccelerator(new KeyCodeCombination(KeyCode.F, KeyCombination.ALT\_DOWN)); colour = new Menu("Colour"); colour.setAccelerator(new KeyCodeCombination(KeyCode.L, KeyCombination.ALT\_DOWN)); name = new Menu("Name"); name.setAccelerator(new KeyCodeCombination(KeyCode.N, KeyCombination.ALT\_DOWN)); type = new Menu("Type"); type.setAccelerator(new KeyCodeCombination(KeyCode.T, KeyCombination.ALT\_DOWN)); size = new Menu("Size");

size.setAccelerator(new KeyCodeCombination(KeyCode.S, KeyCombination.ALT\_DOWN)); times=new MenuItem("TimesRoman");

dia=new MenuItem("Dialog"); sansserif = new MenuItem("SansSerif"); courier=new MenuItem("Courier"); bold = new MenuItem("Bold"); italic = new MenuItem("Italic"); size12 = new MenuItem("Size 12"); size14 = new MenuItem("Size 14"); size16 = new MenuItem("Size 16"); size18 = new MenuItem("Size 18"); india = new MenuItem("India"); aus =new MenuItem("Australia"); brazil = new MenuItem("Brazil"); germ = new MenuItem("Germany"); ice = new MenuItem("Iceland"); mal = new MenuItem("Malaysia"); rus = new MenuItem("Russia"); blu = new MenuItem("Blue"); bro =new MenuItem("Brown"); gray = new MenuItem("Gray"); green = new MenuItem("Green"); org = new MenuItem("Orange"); red = new MenuItem("Red"); yel=new MenuItem("Yellow"); disp = new Label();

times.setAccelerator(new KeyCodeCombination(KeyCode.I, KeyCombination.CONTROL\_DOWN)); dia.setAccelerator(new KeyCodeCombination(KeyCode.D, KeyCombination.CONTROL\_DOWN));

sansserif.setAccelerator(new KeyCodeCombination(KeyCode.A, KeyCombination.CONTROL\_DOWN));

courier.setAccelerator(new KeyCodeCombination(KeyCode.U,

KeyCombination.CONTROL\_DOWN));

bold.setAccelerator(new KeyCodeCombination(KeyCode.B, KeyCombination.CONTROL\_DOWN));

italic.setAccelerator(new KeyCodeCombination(KeyCode.R, KeyCombination.CONTROL\_DOWN));

size12.setAccelerator(new KeyCodeCombination(KeyCode.F2)); size14.setAccelerator(new KeyCodeCombination(KeyCode.F4)); size16.setAccelerator(new KeyCodeCombination(KeyCode.F6)); size18.setAccelerator(new KeyCodeCombination(KeyCode.F8));

india.setAccelerator(new KeyCodeCombination(KeyCode.G, KeyCombination.CONTROL\_DOWN)); aus.setAccelerator(new KeyCodeCombination(KeyCode.H, KeyCombination.CONTROL\_DOWN));

brazil.setAccelerator(new KeyCodeCombination(KeyCode.J, KeyCombination.CONTROL\_DOWN));

germ.setAccelerator(new KeyCodeCombination(KeyCode.K, KeyCombination.CONTROL\_DOWN)); ice.setAccelerator(new KeyCodeCombination(KeyCode.M, KeyCombination.CONTROL\_DOWN)); mal.setAccelerator(new KeyCodeCombination(KeyCode.P, KeyCombination.CONTROL\_DOWN)); rus.setAccelerator(new KeyCodeCombination(KeyCode.Q, KeyCombination.CONTROL\_DOWN)); blu.setAccelerator(new KeyCodeCombination(KeyCode.V, KeyCombination.CONTROL\_DOWN)); bro.setAccelerator(new KeyCodeCombination(KeyCode.W, KeyCombination.CONTROL\_DOWN)); gray.setAccelerator(new KeyCodeCombination(KeyCode.X, KeyCombination.CONTROL\_DOWN));

green.setAccelerator(new KeyCodeCombination(KeyCode.Y, KeyCombination.CONTROL\_DOWN)); org.setAccelerator(new KeyCodeCombination(KeyCode.Z, KeyCombination.CONTROL\_DOWN)); red.setAccelerator(new KeyCodeCombination(KeyCode.F1)); yel.setAccelerator(new KeyCodeCombination(KeyCode.F3));

country.getItems().add(india); country.getItems().add(aus); country.getItems().add(brazil); country.getItems().add(germ); country.getItems().add(ice); country.getItems().add(mal); country.getItems().add(rus); mb.getMenus().add(country); name.getItems().add(times); name.getItems().add(dia); name.getItems().add(sansserif); name.getItems().add(courier); font.getItems().add(name); type.getItems().add(bold); type.getItems().add(italic); font.getItems().add(type); size.getItems().add(size12); size.getItems().add(size14); size.getItems().add(size16); size.getItems().add(size18); font.getItems().add(size); mb.getMenus().add(font); colour.getItems().add(blu); colour.getItems().add(bro); colour.getItems().add(gray); colour.getItems().add(green); colour.getItems().add(org); colour.getItems().add(red); colour.getItems().add(yel); mb.getMenus().add(colour);

EventHandler<ActionEvent> m = new EventHandler<ActionEvent>() {

Font font;

FontWeight fw;

FontPosture fw1;

public void handle(ActionEvent o)

{

if(o.getSource()==clc){

disp.setText("");

}

if(o.getSource()==exit){

System.exit(0);

}

if(o.getSource()==times){

FontPosture str = FontPosture.findByName(disp.getFont().getStyle());

FontWeight str1=FontWeight.findByName(disp.getFont().getStyle());

font=Font.font("TimesRoman",str1,str,disp.getFont().getSize());

disp.setFont(font);

}

if(o.getSource()==dia){

FontPosture str = FontPosture.findByName(disp.getFont().getStyle());

FontWeight str1=FontWeight.findByName(disp.getFont().getStyle()); font=Font.font("Dialog",str1,str,disp.getFont().getSize());

disp.setFont(font);

}

if(o.getSource()==sansserif){

FontPosture str = FontPosture.findByName(disp.getFont().getStyle());

FontWeight str1=FontWeight.findByName(disp.getFont().getStyle()); font=Font.font("SansSerif",str1,str,disp.getFont().getSize());

disp.setFont(font);

}

if(o.getSource()==courier){

FontPosture str = FontPosture.findByName(disp.getFont().getStyle());

FontWeight str1=FontWeight.findByName(disp.getFont().getStyle()); font=Font.font("Courier",str1,str,disp.getFont().getSize());

disp.setFont(font);

}

if(o.getSource()==bold){

fw=FontWeight.EXTRA\_BOLD;

disp.setFont(Font.font(disp.getFont().getName(),fw,disp.getFont().getSize()));

}

if(o.getSource()==italic){

fw1=FontPosture.ITALIC;

disp.setFont(Font.font(disp.getFont().getName(),fw1,disp.getFont().getSize()));

}

if(o.getSource()==size12){

FontPosture str = FontPosture.findByName(disp.getFont().getStyle());

FontWeight str1=FontWeight.findByName(disp.getFont().getStyle()); disp.setFont(Font.font(disp.getFont().getName(),str1,str,12));

}

if(o.getSource()==size14){

FontPosture str = FontPosture.findByName(disp.getFont().getStyle());

FontWeight str1=FontWeight.findByName(disp.getFont().getStyle()); disp.setFont(Font.font(disp.getFont().getName(),str1,str,14));

}

if(o.getSource()==size16){

FontPosture str = FontPosture.findByName(disp.getFont().getStyle());

FontWeight str1=FontWeight.findByName(disp.getFont().getStyle()); disp.setFont(Font.font(disp.getFont().getName(),str1,str,16));

}

if(o.getSource()==size18){

FontPosture str = FontPosture.findByName(disp.getFont().getStyle());

FontWeight str1=FontWeight.findByName(disp.getFont().getStyle());

disp.setFont(Font.font(disp.getFont().getName(),str1,str,18));

}

if(o.getSource()==blu){

disp.setTextFill(Color.web("#0000FF"));

}

if(o.getSource()==bro){ disp.setTextFill(Color.web("#A52A2A"));

}

if(o.getSource()==gray){ disp.setTextFill(Color.web("#808080"));

}

if(o.getSource()==green){ disp.setTextFill(Color.web("#008000"));

}

if(o.getSource()==org){ disp.setTextFill(Color.web("#FFA500"));

}

if(o.getSource()==red){ disp.setTextFill(Color.web("#FF0000"));

}

if(o.getSource()==yel){ disp.setTextFill(Color.web("#FFFF00"));

}

if(o.getSource()==india||o.getSource()==aus||o.getSource()==brazil||o.getSource()==germ|

|o.getSource()==ice||o.getSource()==mal||o.getSource()==rus){

disp.setText(((MenuItem)o.getSource()).getText());

}

}

};

india.setOnAction(m); aus.setOnAction(m); brazil.setOnAction(m); germ.setOnAction(m); ice.setOnAction(m); mal.setOnAction(m); rus.setOnAction(m); times.setOnAction(m); dia.setOnAction(m); sansserif.setOnAction(m); courier.setOnAction(m); bold.setOnAction(m); italic.setOnAction(m); size12.setOnAction(m); size14.setOnAction(m); size16.setOnAction(m); size18.setOnAction(m); blu.setOnAction(m); bro.setOnAction(m); gray.setOnAction(m); green.setOnAction(m); org.setOnAction(m); red.setOnAction(m); yel.setOnAction(m); clc.setOnAction(m); exit.setOnAction(m); TilePane tilePane = new TilePane(mb); mb.setContextMenu(pop);

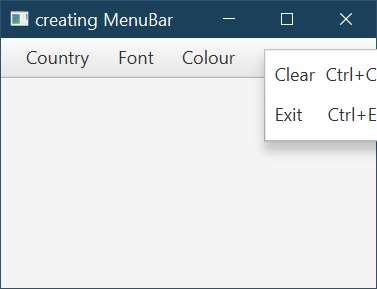
VBox vb = new VBox(mb,disp,tilePane); Scene sc = new Scene(vb, 300, 200);

s.setScene(sc);

s.show();

}

}



2. Write a program to create a class and create 5 objects using array of objects.

Read the input from the user, calculate percentage of marks and display the

student details.

Class: Student

Instance variables: name, regno, dept, year, addr, m1,m2,m3,

percentage(m2+m3+m1/3)

method:

getdetail()

display()

Ans: import java.util.Scanner;

class Student{

String name,regno,dept,address;

int m1,m2,m3,year;

double percentage;

void getDetails()

{

Scanner sc=new Scanner(System.in);

System.out.println("Enter Details of student");

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

name=sc.nextLine();

System.out.println("Enter registration number");

regno=sc.nextLine();

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

dept=sc.nextLine();

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

address=sc.nextLine();

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

year=sc.nextInt();

System.out.println("Marks in subject 1");

m1=sc.nextInt();

System.out.println("Marks in subject 2");

m2=sc.nextInt();

System.out.println("Marks in subject 3");

m3=sc.nextInt();

percentage=(m1+m2+m3)/3;

}

void display()

{

System.out.println("\n Details of the student are: \n Name of the Student is "+name+"\n Registration number of Student is "+regno+"\n Department of Student is "+dept+"\n year "+year+"\n Address of Student "+address+"\n Marks of the Student are "+m1+","+m2+","+m3+"\n Percentage of Student is " +percentage);

}

}

class Main {

public static void main (String[] args) {

Student s[]=new Student[5];

for(int i=0;i<5;i++)

{

s[i]=new Student();

}

for(int j=0;j<5;j++)

{

s[j].getDetails();

}

for(int k=0;k<5;k++)

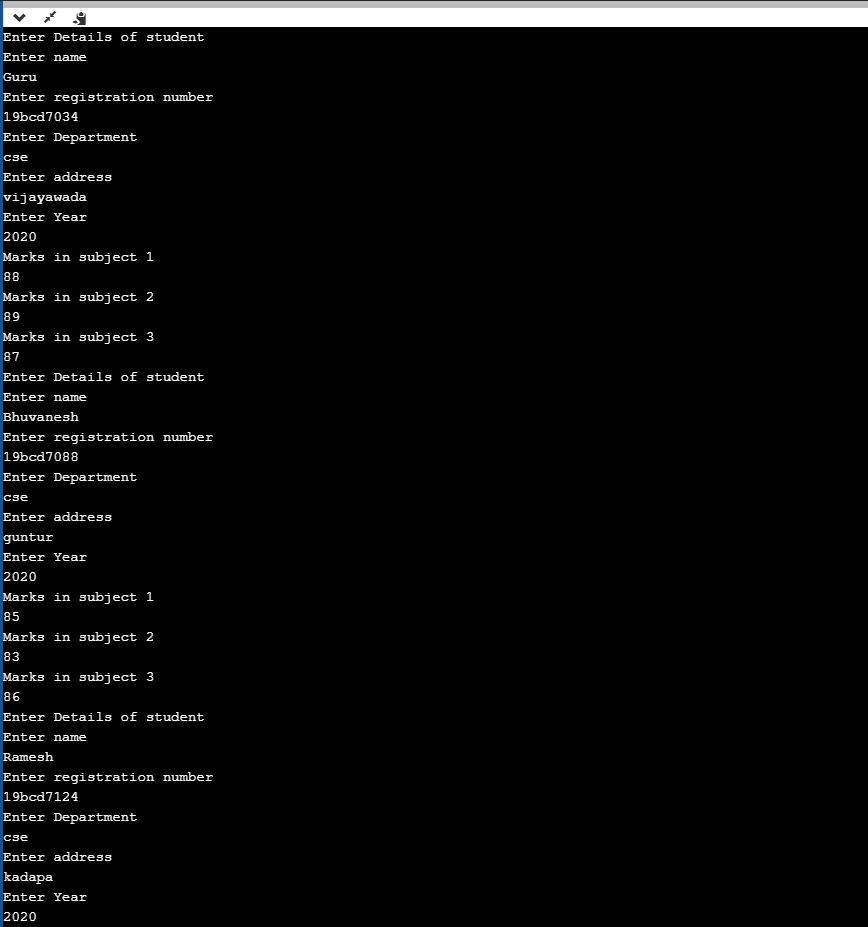
{

s[k].display();

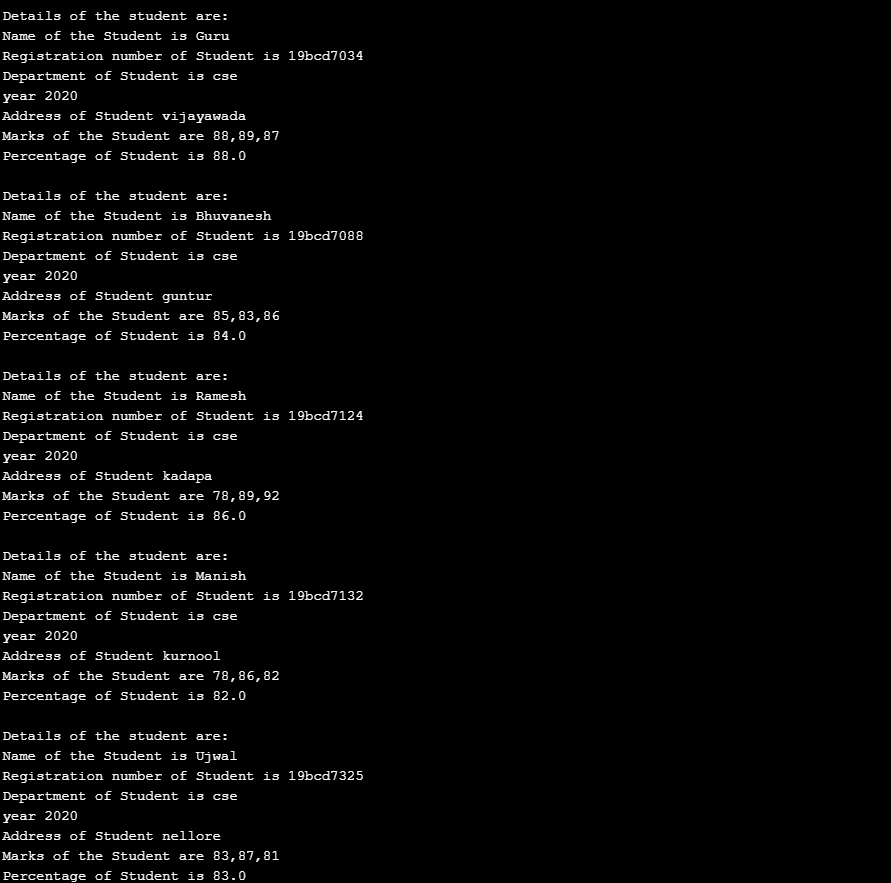
}

}

OUTPUT:

}





3.Mr.Arun gives k integer numbers as key to open a secret locker. The locker will

open only when the sum of all keys in alternative index is a palindrome otherwise

not. Write a java program to implement it and display whether locker opened or

not.

Ans: import java.util.Scanner;

public class Main {

public static void main(String []args)

{

Scanner sc= new Scanner(System.in);

int sum= 0;

int j= 0;

int k= sc.nextInt();

int a[]= new int[k];

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

{

a[i]=sc.nextInt();

}

while(j<k)

{

sum=sum+a[j];

j=j+2;

}

int temp= sum;

int n= temp;

int r;

sum=0;

while(n>0){

r=n%10;

sum=(sum\*10)+r;

n=n/10;

}

if(temp==sum)

System.out.println("palindrome number \n Door opened");

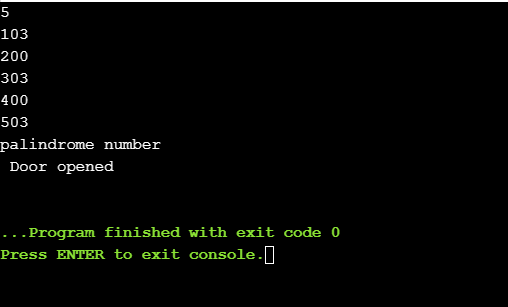
else

System.out.println("not palindrome \n Door not opened");

}

}

OUTPUT:



1. write a program to read n strings and sort in ascending order.

Ans: import java.util.Scanner;

public class Main

{

public static void main(String[] args)

{

int n;

String temp;

Scanner s = new Scanner(System.in);

System.out.print("Number of strings =");

n = s.nextInt();

String strings[] = new String[n];

Scanner gc = new Scanner(System.in);

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

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

{

strings[i] = gc.nextLine();

}

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

{

for (int j = i + 1; j < n; j++)

{

if (strings[i].compareTo(strings[j])>0)

{

temp = strings[i];

strings[i] = strings[j];

strings[j] = temp;

}

}

}

System.out.print("Strings in ascending order are: ");

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

{

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

}

System.out.print(strings[n - 1]);

}

}

OUTPUT:

