

Oop Lab Assesment -1

CSE2005-L1

P.Harsha vardhan

19BCI7039

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 the values in reverse order. Save the application as Percentages.java.

code :

```
import java.util.*;
public class Percentages{
    static void computePercent(double a,double b)
    {
        double c;
        c=((a/b)*100);
        System.out.println(a+" is "+c+" precent of "+b);
    }
    public static void main(String args[])
    {
        double a=2;
        double b=5;
        computePercent(a,b);
        computePercent(b,a);
    }
}
```

Output :

```
C:\Users\user\Downloads>cd 19bci7039
C:\Users\user\Downloads\19BCI7039>javac Percentages.java
C:\Users\user\Downloads\19BCI7039>java Percentages
2.0 is 40.0 precent of 5.0
5.0 is 250.0 precent of 2.0
```

1.b

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

code :

```
import java.util.*;
public class Percentages2{
    static void computePercent(double a,double b)
    {
        double c;
        c=(a/b)*100;
        System.out.println(a+" is "+c+" percent of "+b);
    }
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        double a=sc.nextInt();
        double b=sc.nextInt();
        computePercent(a,b);
        computePercent(b,a);
    }
}
```

Output :

```
C:\Users\user\Downloads\19BCI7039>javac percentages2.java
C:\Users\user\Downloads\19BCI7039>java Percentages2
5 5
5.0 is 100.0 percent of 5.0
5.0 is 100.0 percent of 5.0
```

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.

Code:

```
import java.util.*;
public class InchConversion
{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();
        Inchtfeet(n);
        Inchtoyard(n);
    }
    static void Inchtfeet(int n)
    {
        double f;
        f=((double)n)/12;
        System.out.println(n+"(inches) - "+f+"(feet)");
    }
    static void Inchtoyard(int n)
    {
        double y;
        y=((double)n)/36;
        System.out.println(n+"(inches) - "+y+"(Yards)");
    }
}
```

Output:

```
C:\Users\user\Downloads\19BCI7039>javac InchConversion.java
C:\Users\user\Downloads\19BCI7039>java InchConversion
36
36(inches) - 3.0(feet)
36(inches) - 1.0(Yards)
```

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.

Code :

```
import java.util.*;
public class PaintCalculator
{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        int l=sc.nextInt();
        int w=sc.nextInt();
        int h=sc.nextInt();
        int wa=wallarea(l,w,h);
        int ng=numgall(wa);
        int pp=paintprice(ng);
        System.out.println("$"+pp);
    }
    static int wallarea(int l,int w,int h)
    {
        int wk=2*((l*h)+(w*h));
        return wk;
    }
    static int numgall(int wa)
    {
        if(wa%350==0)
        {
            return wa/350;
        }
        else
        {
            return ((wa/350)+1);
        }
    }
    static int paintprice(int ng)
    {
        int p=(ng*32);
        return p;
    }
}
```

Output:

```
C:\Users\user\Downloads\19BCI7039>javac PaintCalculator.java  
C:\Users\user\Downloads\19BCI7039>java PaintCalculator  
15  
20  
10  
$64
```

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.

Code:

```
import java.util.*;  
public class JobPricing {  
    public static void main(String args[])  
    {  
        Scanner sc=new Scanner(System.in);  
        String job=sc.nextLine();  
        int com=sc.nextInt();  
        int hours=sc.nextInt();  
        int travelhours=sc.nextInt();  
        int price=pricing(com,hours,travelhours);  
        System.out.println(job+" - "+price);  
    }  
    static int pricing(int c,int h,int th)  
    {  
        int p=c+(h*35)+(th*12);  
        return p;  
    }  
}
```

Output:

```
C:\Users\user\Downloads\19BCI7039>javac JobPricing.java  
C:\Users\user\Downloads\19BCI7039>java JobPricing  
Tilesfixing  
1  
1  
1  
Tilesfixing - 48
```

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.

Code:

```
class Student  
{  
    String id;  
    int nch,np;  
    void Stud(String id,int nch,int np)  
    {  
        this.id=id;  
        this.nch=nch;  
        this.np=np;  
        System.out.println("Student id - "+id);  
        System.out.println("Number of credithours - "+nch);  
        System.out.println("Number of points - "+np);  
    }  
    void grade_point_average()  
    {  
        int gp=np/nch;  
        System.out.println("Grade point average - "+gp);  
    }  
}
```

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.

Code:

```
import java.util.*;
public class ShowStudent
{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        String id=sc.nextLine();
        int nch=sc.nextInt();
        int np=sc.nextInt();
        Student s=new Student();
        s.Stud(id,nch,np);
        s.grade_point_average();
    }
}
```

Combined output for 5.a,5.b :

```
C:\Users\user\Downloads\19BCI7039>javac ShowStudent.java
C:\Users\user\Downloads\19BCI7039>java ShowStudent
Harsha
4
24
Student id - Harsha
Number of credithours - 4
Number of points - 24
Grade point average - 6
```

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.

Code:

```
import java.util.*;
public class ShowStudent2
{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        Student s=new Student('9999',3,12);
        s.grade_point_average();
```

```

    }
}
class Student
{
    String id;
    int nch,np;
    Student(String id,int nch,int np)
    {
        System.out.println("The constructor starts");
        this.id=id;
        this.nch=nch;
        this.np=np;
        System.out.println("Student id - "+id);
        System.out.println("Number of credithours - "+nch);
        System.out.println("Number of points - "+np);
    }
    void grade_point_average()
    {
        int gp=np/nch;
        System.out.println("Grade point average - "+gp);
    }
}

```

Output :

```

C:\Users\user\Downloads\19BCI7039>javac ShowStudent2.java

C:\Users\user\Downloads\19BCI7039>java ShowStudent2
The constructor starts
Student id - 9999
Number of credithours - 3
Number of points - 12
Grade point average - 4

```

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.

Code:

```

import java.util.*;

public class Lease {

```

```
String tenant_name;  
  
int apartment_number;  
  
double rent_amount;  
  
double lease_term;  
  
Lease() {  
  
    this.tenant_name = "XXX";  
  
    this.apartment_number = 0;  
  
    this.rent_amount = 1000;  
  
    this.lease_term = 12;  
  
}  
  
public void getData() {  
  
    System.out.println("Tenant name : "+tenant_name);  
  
    System.out.println("Apartment number : "+apartment_number);  
  
    System.out.println("Lease term : "+lease_term);  
  
    System.out.println("Rent Amount : "+rent_amount);  
  
}  
  
public void setData(String tenant_name,double lease_term,double rent_amount,int  
apartment_number) {  
  
    this.tenant_name = tenant_name;  
  
    this.lease_term = lease_term;  
  
    this.apartment_number = apartment_number;  
  
    this.rent_amount = rent_amount;  
  
}  
  
public void addPetFee() {
```

```
        this.rent_amount += 10;

        System.out.println("The rentamount after adding the petfee is "+rent_amount);

        explainPetPolicy();

    }

    public static void explainPetPolicy() {

        System.out.println("An amount of Rs.10 is added to the rent if there is a pet in
the house");

    }

    public static void main(String args[])

    {

        Lease l=new Lease();

        l.getData();

        Scanner sc=new Scanner(System.in);

        String s=sc.nextLine();

        double t=sc.nextDouble();

        double r=sc.nextDouble();

        int an=sc.nextInt();

        l.setData(s,t,r,an);

        l.getData();

        l.addPetFee();

    }

}
```

```
C:\Users\user\Downloads\19BCI7039>javac Lease.java
C:\Users\user\Downloads\19BCI7039>java Lease
Tenant name : XXX
Appartment number : 0
Lease term : 12.0
Rent Amount : 1000.0
Harsha
10
120
5
Tenant name : Harsha
Appartment number : 5
Lease term : 10.0
Rent Amount : 120.0
The rentamount after adding the petfee is 130.0
An amount of Rs.10 is added to the rent if there is a pet in the house
```

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.

Code:

```
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();

}

}
```

Output:

```
C:\Users\user\Downloads\19BCI7039>javac TestLease.java  
C:\Users\user\Downloads\19BCI7039>java TestLease  
Enter Apartment tenantâ??s name  
HArsha  
Enter Apartment Number  
5  
Enter Number of Months  
12  
Enter Monthly rent amount  
10000  
Enter Apartment tenantâ??s name  
Bavita  
Enter Apartment Number  
4  
Enter Number of Months  
10  
Enter Monthly rent amount  
2000  
Enter Apartment tenantâ??s name  
Vishnu  
Enter Apartment Number  
205  
Enter Number of Months  
24  
Enter Monthly rent amount  
3000  
Apartment tenantâ??s name is HArsha  
Apartment Number is 5  
Term of the lease in months 12  
Monthly rent amount is $10000
```

```
If there is any pet then 10$ will be added to rent.  
Apartment tenantâ??s name is HArsha  
Apartment Number is 5  
Term of the lease in months 12  
Monthly rent amount is $10010
```

```
Apartment tenantâ??s name is Bavita  
Apartment Number is 4  
Term of the lease in months 10  
Monthly rent amount is $2000
```

Oop Lab Assessment-2

CSE2005–L1

P. Harsha vardhan

19BCI7039

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.

Code:

```
import java.util.*;
public class Billing{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        int a=sc.nextInt();
        int b=sc.nextInt();
        int c=sc.nextInt();
        float sv = computeBill(a);
        float sv2 = computeBill(a,b);
        float sv3 = computeBill(a,b,c);
        System.out.println("1 variable method - "+sv);
        System.out.println("2 variable method - "+sv2);
        System.out.println("3 variable method - "+sv3);
    }
    static float computeBill(int a)
    {
        float tax=((float)a*8)/100;
        return ((float)a+tax);
    }
}
```

```

static float computeBill(int a,int b)
{
    float tax=(((float)a*(float)b)*8)/100;
    return ((float)a*(float)b)+tax;
}
static float computeBill(int a,int b,int c)
{
    float tax=((((float)a*(float)b)-(float)c)*8)/100;
    return (((float)a*(float)b)-(float)c)+tax;
}
}

```

Output:

```

C:\Users\user\Downloads\19BCI7039>javac Billing.java

C:\Users\user\Downloads\19BCI7039>java Billing
200
3
50
1 variable method - 216.0
2 variable method - 648.0
3 variable method - 594.0

```

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.

Code:

```

import java.util.*;
public class TestBloodData
{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        BloodData bd=new BloodData();
        String bt=sc.nextLine();
        String rh=sc.nextLine();
        BloodData bd1=new BloodData(bt,rh);
        System.out.println("Setting data to AB and '-'");
        bd1.setdata("AB","-");
        System.out.println("BloodType : "+bd1.getBt()+" \nRhfactor : "+bd1.getRh());
    }
}

```

```

    }
}
class BloodData
{
String BloodType,Rhfact;
BloodData(){
    System.out.println("Default Constructor");
    BloodType="O";
    Rhfact="1";
    System.out.println("BloodType : "+BloodType+"\nRhfactor : "+Rhfact);
}
BloodData(String Bt, String Rf)
{
    System.out.println("Constructor");
    BloodType=Bt;
    Rhfact=Rf;
    System.out.println("BloodType : "+BloodType+"\nRhfactor : "+Rhfact);
}
void setdata(String Bt,String Rf)
{
    BloodType=Bt;
    Rhfact=Rf;
}
String getBt()
{
    return BloodType;
}
String getRh()
{
    return Rhfact;
}
}

```

Output :

```

C:\Users\user\Downloads\19BCI7039>javac TestBloodData.java

C:\Users\user\Downloads\19BCI7039>java TestBloodData
Default Constructor
BloodType : O
Rhfactor : 1
A
0
Constructor
BloodType : A
Rhfactor : 0
Setting data to AB and '-'
BloodType : AB
Rhfactor : -

```

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 that each method works correctly, and save it as TestPatient.java.

Code:

```
import java.util.*;
public class TestPatient
{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        String idno=sc.nextLine();
        String bt=sc.nextLine();
        String rhf=sc.nextLine();
        int age=sc.nextInt();
        Patient p=new Patient();
        Patient p1=new Patient(idno,bt,age,rhf);
        System.out.println("Using Getter method");
        p.getData(idno,bt,age,rhf);
    }
}
class Patient
{
    String idnum;
    int age;
    Patient()
    {
        System.out.println("Default constructor");
        BloodData bd=new BloodData("O","1");
        idnum="0";
        age=0;
        System.out.println("Idnumber : "+idnum+"\nAge : "+age);
    }
    Patient(String idn,String bt,int age,String rhf)
    {
        System.out.println("Constructor");
        BloodData bd1=new BloodData(bt,rhf);
        idnum=idn;
        this.age=age;
        System.out.println("Idnumber : "+idn+"\nAge : "+age);
    }
}
```

```

void getData(String idno,String bt,int age,String rhf)
{
    System.out.println("Idnumber : "+idno+"\nAge : "+age+"\nBloodType : "+bt+"\nRhfactor : "+rhf);
}
}

```

Output:

```

C:\Users\user\Downloads\19BCI7039>javac TestPatient.java
C:\Users\user\Downloads\19BCI7039>java TestPatient
12
A
0
5
Default constructor
BloodType : O
Rhfactor : 1
Idnumber : 0
Age : 0
Constructor
BloodType : A
Rhfactor : 0
Idnumber : 12
Age : 5
Using Getter method
Idnumber : 12
Age : 5
BloodType : A
Rhfactor : 0

```

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.

Code :

```

import java.util.*;
import java.lang.Math;
public class Circle
{
    double rad,dia,area;
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        int r=sc.nextInt();
        Circle c=new Circle();
        c.setData(r);
        c.getData();
    }
}

```

```

}
Circle()
{
    rad=1;
    dia=2*rad;
    area=(Math.PI)*rad*rad;
    System.out.println("Constructor set values");
    System.out.println("rad : "+rad+"\ndia : "+dia+"\narea : "+area);
}
public void setData(int r)
{
    System.out.println("Setting the radius using setter method");
    rad=r;
    dia=2*rad;
    area=(Math.PI)*rad*rad;
}
public void getData()
{
    System.out.println("rad : "+rad+"\ndia : "+dia+"\narea : "+area);
}
}

```

Output :

```

C:\Users\user\Downloads\19BCI7039>javac Circle.java

C:\Users\user\Downloads\19BCI7039>java Circle
5
Constructor set values
rad : 1.0
dia : 2.0
area : 3.141592653589793
Setting the radius using setter method
rad : 5.0
dia : 10.0
area : 78.53981633974483

C:\Users\user\Downloads\19BCI7039>

```

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.

Code :

```
import java.util.*;
import java.lang.Math;
public class TestCircle
{
    public static void main(String[] args) {
        Scanner sc=new Scanner(System.in);
        int r=sc.nextInt();
        int r1=sc.nextInt();
        Circle c=new Circle();
        Circle c1=new Circle();
        Circle c2=new Circle();
        if(r>r1)
        {
            System.out.println("Circle object 1");
            c.setData(r1);
            System.out.println("Circle object 2");
            c1.setData(r);
        }
        else
        {
            System.out.println("Circle object 1");
            c.setData(r);
            c.getData();
            System.out.println("Circle object 2");
            c1.setData(r1);
            c1.getData();
        }
        System.out.println("Circle object 3");
        c2.getData();
    }
}
class Circle
{
    double rad,dia,area;
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        int r=sc.nextInt();
        Circle c=new Circle();
        c.setData(r);
        c.getData();
    }
    Circle()
    {
        rad=1;
```

```

        dia=2*rad;
        area=(Math.PI)*rad*rad;
    }
    public void setData(int r)
    {
        System.out.println("Setting the radius using setter method");
        rad=r;
        dia=2*rad;
        area=(Math.PI)*rad*rad;
    }
    public void getData()
    {
        System.out.println("radius : "+rad+"\ndiameter : "+dia+"\narea : "+area);
    }
}

```

Output :

```

C:\Users\user\Downloads\19BCI7039>javac TestCircle.java
C:\Users\user\Downloads\19BCI7039>java TestCircle
4
5
Circle object 1
Setting the radius using setter method
radius : 4.0
diameter : 8.0
area : 50.26548245743669
Circle object 2
Setting the radius using setter method
radius : 5.0
diameter : 10.0
area : 78.53981633974483
Circle object 3
radius : 1.0
diameter : 2.0
area : 3.141592653589793

```

4.a

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

- The square root of 37
- The sine and cosine of 300
- The value of the floor, ceiling, and round of 22.8
- The larger and the smaller of the character ‘D’ and the integer 71
- 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.

Code :

```
import java.util.*;
```

```

import java.lang.Math;
public class MathTest
{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        double m=Math.sqrt(37);
        System.out.println("Square root of 37 is "+m);
        System.out.println("Sine and Cosine values of 300 are : ");
        System.out.println("Sine value is : "+Math.sin(300));
        System.out.println("Cosine vlaue is : "+Math.cos(300));
        System.out.println("Ceiling of number 22.8 is "+Math.ceil(22.8));
        System.out.println("Floor of number 22.8 is "+Math.floor(22.8));
        System.out.println("Round of number 22.8 is "+Math.round(22.8));
        if('D'>71)
        {
            System.out.println("D is the Larger one");
            System.out.println("71 is the Smaller one");
        }
        else
        {
            System.out.println("D is the Smaller one");
            System.out.println("71 is the Larger one");
        }
        System.out.println("Random number between 0 and 20 is "+(20*(Math.random())));
    }
}

```

Output :

```

C:\Users\user\Downloads\19BCI7039>javac MathTest.java

C:\Users\user\Downloads\19BCI7039>java MathTest
Square root of 37 is 6.082762530298219
Sine and Cosine values of 300 are :
Sine value is : -0.9997558399011495
Cosine vlaue is : -0.022096619278683942
Ceiling of number 22.8 is 23.0
Floor of number 22.8 is 22.0
Round of number 22.8 is 23
D is the Smaller one
71 is the Larger one
Random number between 0 and 20 is 12.426425269618942

```

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.

Code :

```
import java.time.*;
public class TestMonthHandling{
    public static void main(String args[]){
        LocalDate date1=LocalDate.of(2020,1,31);
        LocalDate date2=LocalDate.of(2020,12,31);
        System.out.println("Local date 1 after adding 1 month is:"+ " "+date1.plusMonths(1));
        System.out.println("Local date 1 after adding 2 months is:"+ " "+date1.plusMonths(2));
        System.out.println("Local date 1 after adding 3 months is:"+ " "+date1.plusMonths(3));
        System.out.println("");
        System.out.println("Local date 2 after adding 1 month is:"+ " "+date2.plusMonths(1));
        System.out.println("Local date 2 after adding 2 months is:"+ " "+date1.plusMonths(2));
        System.out.println("Local date 2 after adding 3 months is:"+ " "+date1.plusMonths(3));
    }
}
```

Output :

```
C:\Users\user\Downloads\19BCI7039>javac TestMonthHandling.java

C:\Users\user\Downloads\19BCI7039>java TestMonthHandling
Local date 1 after adding 1 month is: 2020-02-29
Local date 1 after adding 2 months is: 2020-03-31
Local date 1 after adding 3 months is: 2020-04-30

Local date 2 after adding 1 month is: 2021-01-31
Local date 2 after adding 2 months is: 2020-03-31
Local date 2 after adding 3 months is: 2020-04-30
```

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.

Code :

```
import java.time.*;
public class TenThousandDaysOld{
    public static void main(String args[]){
        LocalDate date=LocalDate.of(2002,03,23);
        System.out.println("my birthday is on :"+" "+date);
        System.out.println("Local date after adding 10000 days is"+ " "+date.plusDays(10000));
    }
}
```

Output:

```
C:\Users\user\Downloads\19BCI7039>javac TenThousandDaysOld.java  
C:\Users\user\Downloads\19BCI7039>java TenThousandDaysOld  
my birthday is on : 2002-03-23  
Local date after adding 10000 days is 2029-08-08
```

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.

Code :

```
import java.time.*;  
import java.util.Scanner;  
public class DaysTilNextMonth{  
public static void main(String args[]){  
Scanner in= new Scanner(System.in);  
System.out.println("enter year");  
int year=in.nextInt();  
System.out.println("enter month ");  
int month=in.nextInt();  
System.out.println("enter date");  
int date=in.nextInt();  
LocalDate d=LocalDate.of(year,month,date);  
System.out.println("Date is"+ " "+d);  
int no_of_days=d.lengthOfMonth();  
int days_left= (no_of_days)-date;  
System.out.println("No of days left :"+" "+days_left);  
LocalDate next_month=d.plusDays(days_left + 1);  
System.out.println("Next month is"+ " "+next_month.getMonth());  
}  
}
```

Output:

```
C:\Users\user\Downloads\19BCI7039>javac DaysTilNextMonth.java  
C:\Users\user\Downloads\19BCI7039>java DaysTilNextMonth  
enter year  
2002  
enter month  
03  
enter date  
23  
Date is 2002-03-23  
No of days left : 8  
Next month is APRIL
```

Oops Assessment Lab – 3

CSE2005-L1

P. Harsha vardhan

19BCI7039

1.

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.

Code :

```
import java.util.*;
class Candle
{
    String Color;
    int Height,Price;
    void setColor(String c)
    {
        Color=c;
    }
    void setHeight(int h)
    {
        Height=h;
        Price=2*h;
    }
    String getColor()
    {
        return Color;
    }
    int getHeight()
    {
        return Height;
    }
    int getPrice()
    {
        return Price;
    }
}
class ScentedCandle extends Candle
```

```

{
String scent;
void setScent(String s)
{
    scent=s;
}
String getScent()
{
    return scent;
}
void setHeight(int Height)
{
    System.out.println("Method Overridden");
    this.Height=Height;
    Price=3*Height;
}
}
public class DemoCandles
{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        String Scent=sc.nextLine();
        String color=sc.nextLine();
        int Height=sc.nextInt();
        Candle c=new Candle();
        c.setColor(color);
        c.setHeight(Height);
        System.out.println("The color of the candle is "+c.getColor());
        System.out.println("The Height of the Cnadle is "+c.getHeight());
        System.out.println("The Price of the candle is "+c.getPrice());
        ScentedCandle sc1=new ScentedCandle();
        sc1.setScent(Scent);
        sc1.setHeight(Height);
        System.out.println("The Scent of the Candle is "+sc1.getScent());
        System.out.println("The Price of the Scented Candle is "+sc1.getPrice());
    }
}

```

Output :

```

C:\Users\user\Downloads\19BCI7039>javac DemoCandles.java

C:\Users\user\Downloads\19BCI7039>java DemoCandles
Lavendar
Green
5
The color of the candle is Green
The Height of the Cnadle is 5
The Price of the candle is 10
Method Overridden
The Scent of the Candle is Lavendar
The Price of the Scented Candle is 15

```

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.

Code :

```
import java.util.*;
class Poem{
    String poemname;
    int numline;
    Poem(String pname,int nooflines)
    {
        poemname=pname;
        numline=nooflines;
    }
    String getpoemname()
    {
        return poemname;
    }
    int getnumline()
    {
        return numline;
    }
}
class Couplet extends Poem
{
    Couplet(String cname)
    {
        super(cname,2);
        System.out.println("Couplet \nname : "+cname+"\nnumber of lines : "+numline);
    }
}
class Limerick extends Poem
{
    String Lname;
    Limerick(String Lname)
    {
        super(Lname,5);
        System.out.println("Limerick \nname : "+Lname+"\nnumber of lines : "+numline);
    }
}
class Haiku extends Poem
{
```

```

String Hname;
Haiku(String Hname)
{
    super(Hname,3);
    System.out.println("Haiku \nname : "+Hname+"\nnumber of lines : "+numline);
}
}
public class DemoPoems
{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter the poem name");
        String poemname=sc.nextLine();
        System.out.println("Enter no of lines in the poem : ");
        int numlines=sc.nextInt();
        Poem p=new Poem(poemname,numlines);
        System.out.println("The name of the poem is : "+p.getpoemname());
        System.out.println("The number of lines in the poem is : "+p.getnumline());
        System.out.println("Enter the name of the Couplet : ");
        String c=sc.nextLine();
        String Copname=sc.nextLine();
        Couplet c1=new Couplet(Copname);
        System.out.println("Enter the name of the Limerick : ");
        String Limename=sc.nextLine();
        Limerick l1=new Limerick(Limename);
        System.out.println("Enter the name of the Haiku : ");
        String Haiku=sc.nextLine();
        Haiku h1=new Haiku(Haiku);
    }
}

```

Output :

```

C:\Users\user\Downloads\19BCI7039>javac DemoPoems.java

C:\Users\user\Downloads\19BCI7039>java DemoPoems
Enter the poem name
Henry
Enter no of lines in the poem :
23
The name of the poem is : Henry
The number of lines in the poem is : 23
Enter the name of the Couplet :
Harrypoter
Couplet
name : Harrypoter
number of lines : 2
Enter the name of the Limerick :
percyjackson
Limerick
name : percyjackson
number of lines : 5
Enter the name of the Haiku :
Aloutte
Haiku
name : Aloutte
nnumber of lines : 3

```

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.

Code :

```
import java.util.*;
class SugarSmashPlayer
{
    int idnum,level;
    String screenname;
    int hs[];
    SugarSmashPlayer()
    {
        hs=new int[10];
    }
    void setidnum(int idno)
    {
        idnum=idno;
    }
    void setscreenname(String sname)
    {
        screenname=sname;
    }
    void setscores(int score,int lev)
    {
        if(level==(hs.length-1)&&score==100)
        {
            System.out.println("All levels completed");
        }
        else if(score==100)
        {
            if(level==lev-2||level==0)
            {
```

```

        level=level+1;
    }
    else
    {
        System.out.println("The entered level is not yet reached by the player.");
    }
}
else if(score>hs[level]&&score!=100)
{
    if(level==lev-1 || level==0)
    {
        hs[level]=score;
    }
    else
    {
        System.out.println("The entered level is not yet reached by the player.");
    }
}
else
{
    System.out.println("Wrong information is given");
}
}
String getscreenname()
{
    return screenname;
}
int getidnum()
{
    return idnum;
}
void getscores()
{
    System.out.println("Your current level is "+level);
    System.out.println("Your current score in this level is "+hs[level]);
}
}
class PremiumSugarSmashPlayer
{
    double pay=0;
    PremiumSugarSmashPlayer()
    {
        int hs[] = new int[40];
    }
    void setpay()
    {
        pay=2.99;
        System.out.println("You paid $2.99 to unleash the premium level");
    }
    void getpay()
    {
        System.out.println("Premium level unleashed");
    }
}

```

```

}
public class DemoSugarSmash
{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter your Screen name : ");
        String scrnam=sc.nextLine();
        System.out.println("Enter Your Id number : ");
        int idnum=sc.nextInt();
        SugarSmashPlayer ssp=new SugarSmashPlayer();
        ssp.setidnum(idnum);
        ssp.getidnum();
        ssp.setscreenname(scrnam);
        ssp.getscreenname();
        ssp.setscores(100,1);
        ssp.getscores();
        ssp.setscores(100,9);
        ssp.setscores(50,2);
        ssp.getscores();
        System.out.println("You should pay $2.99 to unleash the premium features.");
        PremiumSugarSmashPlayer pssh=new PremiumSugarSmashPlayer();
        boolean b=false;
        System.out.println("Do you want to unleash the premium level ? ");
        String sq=sc.nextLine();
        String v=sc.nextLine();
        if(v.charAt(0)=='y' || v.charAt(0)=='Y')
        {
            pssh.setpay();
            pssh.getpay();
        }
        else
        {
            System.out.println("you haven't paid the amount to unlock the premium levels.");
        }
    }
}

```

Output :

```

C:\Users\user\Downloads\19BCI7039>javac DemoSugarSmash.java

C:\Users\user\Downloads\19BCI7039>java DemoSugarSmash
Enter your Screen name :
Henry Pattison
Enter Your Id number :
234
Your current level is 1
Your current score in this level is 0
The entered level is not yet reached by the player.
Your current level is 1
Your current score in this level is 50
You should pay $2.99 to unleash the premium features.
Do you want to unleash the premium level ?
yes
You paid $2.99 to unleash the premium level
Premium level unleashed

```

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.

Code :

```
import java.util.*;
class CollegeCourse
{
    String dep;
    int coursenum,credits,fee;
    CollegeCourse(int cn,int cred,String dept)
    {
        coursenum=cn;
        credits=cred;
        dep=dept;
        fee=120*cred;
    }
    void display()
    {
        System.out.println("Department : "+dep);
        System.out.println("Coursenumber : "+coursenum);
        System.out.println("Credits : "+credits);
        System.out.println("Fees : "+fee);
    }
}
class LabCourse extends CollegeCourse
{
    LabCourse(int cnum,int cred,String deptm)
    {
        super(cnum,cred,deptm);
        fee=fee+50;
    }
    void display()
    {
        System.out.println("The course has a labcourse");
        System.out.println("Department : "+this.dep);
        System.out.println("Coursenumber : "+this.coursenum);
        System.out.println("Credits : "+this.credits);
        System.out.println("Fees : "+this.fee);
    }
}
```

```

public class UseCourse
{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        String deptm=sc.nextLine();
        int cnum=sc.nextInt();
        int cred=sc.nextInt();
        CollegeCourse c1=new CollegeCourse(cnum,cred,deptm);
        c1.display();
        if(deptm.equals("BIO")||deptm.equals("CHM")||deptm.equals("CIS")||deptm.equals("PHY"))
        {
            LabCourse l1=new LabCourse(cnum,cred,deptm);
            l1.display();
        }
        else
        {
            System.out.println("There is no lab to this course");
        }
    }
}

```

Output :

```

C:\Users\user\Downloads\19BCI7039>javac UseCourse.java
C:\Users\user\Downloads\19BCI7039>java UseCourse
phy
20
15
Department : phy
Coursenumber : 20
Credits : 15
Fees : 1800
There is no lab to this course

C:\Users\user\Downloads\19BCI7039>java UseCourse
BIO
10
15
Department : BIO
Coursenumber : 10
Credits : 15
Fees : 1800
The course has a labcourse
Department : BIO
Coursenumber : 10
Credits : 15
Fees : 1850

C:\Users\user\Downloads\19BCI7039>

```

5.

Develop a set of classes for a college to use in various student service and personnel 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 data field, 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 a 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.

Code:

```

import java.util.*;
import javax.swing.*;
class Person{
    String firstname,lastname,streetaddress,zipcode,phnnumber;
    void setdata(String firstname,String lastname,String streetaddress,String zipcode,String phnnumber)
    {
        this.firstname=firstname;
        this.lastname=lastname;
        this.streetaddress=streetaddress;
        this.zipcode=zipcode;
        this.phnnumber=phnnumber;
    }
    String getFirstname()
    {
        return firstname;
    }
    String getLastname()
    {
        return lastname;
    }
    String getAddress()
    {
        return streetaddress;
    }
    String getZipCode()
    {
        return zipcode;
    }
}

```

```

String getPhoneNumber()
{
    return phnnumber;
}
void display()
{
    JOptionPane.showMessageDialog(null,firstname + lastname + " lives in " + streetadress + ", " + zipcode +
"Phnno: " + phnnumber,"Information",JOptionPane.QUESTION_MESSAGE);
}
}
class CollegeEmployee extends Person
{
    String secnum,department;
    double annsalary;
    CollegeEmployee(String firstname,String lastname,String streetadress,String zipcode,String phnnumber)
    {
        super.setdata(firstname,lastname,streetadress,zipcode,phnnumber);
    }
    void setdata(String secnum,String department,double annsalary)
    {
        this.secnum=secnum;
        this.department=department;
        this.annsalary=annsalary;
    }
    String getsecnum()
    {
        return secnum;
    }
    double getAnnSalary()
    {
        return annsalary;
    }
    String getDepartment()
    {
        return department;
    }
    void display()
    {
        super.display();
        JOptionPane.showMessageDialog(null, "\nSectionNumber: " + getsecnum() + "\nAnnual Salary: " +
getAnnSalary() + "\nDepartment: " + getDepartment(),"Information",JOptionPane.QUESTION_MESSAGE);
    }
}
class Faculty extends CollegeEmployee
{
    boolean tenured;
    Faculty(String firstname,String lastname,String streetadress,String zipcode,String phnnumber)
    {
        super(firstname,lastname,streetadress,zipcode,phnnumber);
    }
    void setdata(boolean tenured,String sn,String dept,double ai)
    {
        super.setdata(sn,dept,ai);
    }
}

```

```

        this.tenured=tenured;
    }
    boolean gettenured()
    {
        return tenured;
    }
    void display()
    {
        super.display();
        JOptionPane.showMessageDialog(null, "Tenured: " + tenured, "Information",
JOptionPane.QUESTION_MESSAGE);
    }
}
class Student extends Person
{
    String field;
    double gradepointavg;
    Student(String firstname,String lastname,String streetadress,String zipcode,String phnnumber)
    {
        super.setdata(firstname,lastname,streetadress,zipcode,phnnumber);
    }
    void setdata(String field,double gradepointavg)
    {
        this.field=field;
        this.gradepointavg=gradepointavg;
    }
    String getfield()
    {
        return field;
    }
    double getGPA()
    {
        return gradepointavg;
    }
    void display()
    {
        super.display();
        JOptionPane.showMessageDialog(null, "\nField of study: " + getfield() + "\nGPA: " + getGPA(), "Information",
JOptionPane.QUESTION_MESSAGE);
    }
}
public class CollegeList
{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        CollegeEmployee c[]=new CollegeEmployee[4];
        Faculty f[]=new Faculty[3];
        Student s[]=new Student[7];
        String firstname,lastname,streetadress,zipcode,phnnumber;
        System.out.println("Enter the fields firstname,lastname,streetadress,zipcode,phnnumber");
        firstname=sc.nextLine();
        lastname=sc.nextLine();
    }
}
```

```

streetadress=sc.nextLine();
zipcode=sc.nextLine();
phnnumber=sc.nextLine();
Person p=new Person();
p.setdata(firstname,lastname,streetadress,zipcode,phnnumber);
p.display();
int c1=0,f1=0,s1=0,q=0;
while(q==0)
{
    System.out.println("Enter the first letter of the field you have to enter \n1.College employee \n2.Faculty
\n3.Student \n4.Quit");
    String s2=sc.nextLine();
    switch(s2.charAt(0))
    {
        case 'C':
        if(c1<3)
        {
            String firstname1,lastname1,streetadress1,zipcode1,phnnumber1;
            System.out.println("Enter the fields
firstname,lastname,streetaddress,zipcode,phnnumber,sectionnumber,department,annualsalary");
            firstname1=sc.nextLine();
            lastname1=sc.nextLine();
            streetadress1=sc.nextLine();
            zipcode1=sc.nextLine();
            phnnumber1=sc.nextLine();
            String sn=sc.nextLine();
            String dept=sc.nextLine();
            double as=sc.nextDouble();
            String w=sc.nextLine();
            CollegeEmployee c3=new CollegeEmployee(firstname1,lastname1,streetadress1,zipcode1,phnnumber1);
            c3.setdata(sn,dept,as);
            c3.display();
            c[c1]=c3;
            c1=c1+1;
        }
        else{
            System.out.println("CollegeEmployeearray is out of bounds");
        }
        break;
        case 'F':
        if(f1<2)
        {
            String firstname2,lastname2,streetadress2,zipcode2,phnnumber2;
            System.out.println("Enter the fields firstname,lastname,streetaddress,zipcode,phnnumber,aection
number,department,Anuual salary,Tenured");
            firstname2=sc.nextLine();
            lastname2=sc.nextLine();
            streetadress2=sc.nextLine();
            zipcode2=sc.nextLine();
            phnnumber2=sc.nextLine();
            boolean tenure=sc.nextBoolean();
            String w1=sc.nextLine();
            String sn1=sc.nextLine();
        }
    }
}

```

```

String dept1=sc.nextLine();
double as1=sc.nextDouble();
String w=sc.nextLine();
Faculty f3=new Faculty(firstname2,lastname2,streetaddress2,zipcode2,phnnumber2);
f3.setdata(tenure,sn1,dept1,as1);
f3.display();
f[f1]=f3;
f1=f1+1;
}
else{
    System.out.println("Facultyarray is out of bounds");
}
break;
case 'S':
if(s1<6)
{
    String firstname3,lastname3,streetaddress3,zipcode3,phnnumber3;
    System.out.println("Enter the fields firstname,lastname,streetaddress,zipcode,phnnumber,field,GPA");
    firstname3=sc.nextLine();
    lastname3=sc.nextLine();
    streetaddress3=sc.nextLine();
    zipcode3=sc.nextLine();
    phnnumber3=sc.nextLine();
    String field=sc.nextLine();
    double Gpa=sc.nextDouble();
    String w2=sc.nextLine();
    Student s3=new Student(firstname3,lastname3,streetaddress3,zipcode3,phnnumber3);
    s3.setdata(field,Gpa);
    s3.display();
    s[s1]=s3;
    s1=s1+1;
}
else{
    System.out.println("Student array out of bounds");
}
break;
case 'Q':
q=1;
break;
default :
    System.out.println("Wrong input given");
}
}
System.out.println("CollegeEmployee");
for(int i=0;i<4;i++){
if(c[i]!=null)
{
    System.out.println("Firstname :" +c[i].getFirstname());
    System.out.println("Lastname :" +c[i].getLastname());
    System.out.println("streetaddress :" +c[i].getAddress());
    System.out.println("Zipcode :" +c[i].getZipCode());
    System.out.println("Phn number :" +c[i].getPhoneNumber());
    System.out.println("Section number :" +c[i].getsecnum());
    System.out.println("Department :" +c[i].getDepartment());
}
}

```

```

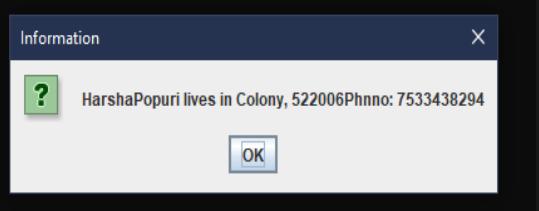
        System.out.println("Annual Salary : "+c[i].getAnnSalary());
    }
    else
    {
        System.out.println("Further objecs for CollegeEmployee class are not yet created");
        break;
    }
}
System.out.println("Faculty");
for(int i=0;i<3;i++)
{
    if(f[i]!=null)
    {
        System.out.println("Firstname : "+f[i].getFirstname());
        System.out.println("Lastname : "+f[i].getLastName());
        System.out.println("streetadress : "+f[i].getAddress());
        System.out.println("Zipcode : "+f[i].getZipCode());
        System.out.println("Phn number : "+f[i].getPhoneNumber());
        System.out.println("Section number : "+f[i].getsecnum());
        System.out.println("Department : "+f[i].getDepartment());
        System.out.println("Annual Salary : "+f[i].getAnnSalary());
        System.out.println("Tenured : "+f[i].gettenured());
    }
    else
    {
        System.out.println("Further objecs for Faculty class are not yet created");
        break;
    }
}
System.out.println("Student");
for(int i=0;i<7;i++)
{
    if(s[i]!=null)
    {
        System.out.println("Firstname : "+s[i].getFirstname());
        System.out.println("Lastname : "+s[i].getLastName());
        System.out.println("streetadress : "+s[i].getAddress());
        System.out.println("Zipcode : "+s[i].getZipCode());
        System.out.println("Phn number : "+s[i].getPhoneNumber());
        System.out.println("field :"+s[i].getfield());
        System.out.println("GPA :" +s[i].getGPA());
    }
    else
    {
        System.out.println("Further objecs for student class are not yet created");
        break;
    }
}
}

```

Output:

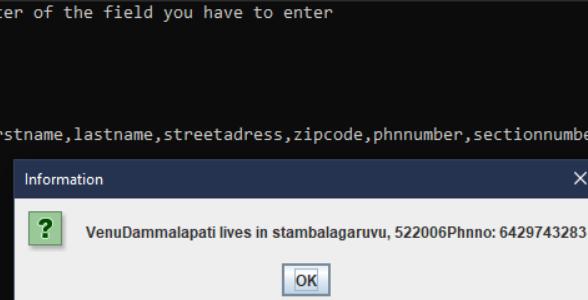
Person :

```
C:\Users\user\Downloads\19BCI7039>javac CollegeList.java
C:\Users\user\Downloads\19BCI7039>java CollegeList
Enter the fields firstname,lastname,streetaddress,zipcode,phnnumber
Harsha
Popuri
Colony
522006
7533438294
```

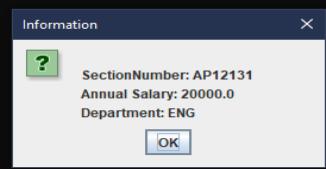


CollegeEmployee :

```
Enter the first letter of the field you have to enter
1.College employee
2.Faculty
3.Student
4.Quit
C
Enter the fields firstname,lastname,streetaddress,zipcode,phnnumber,sectionnumber,department,annualsalary
Venu
Dammalapati
stambalagaruvu
522006
6429743283
AP12131
ENG
20000
```

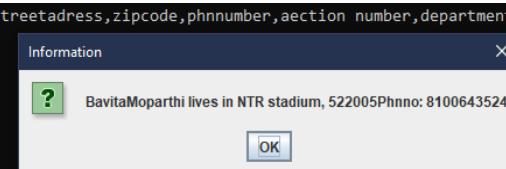


```
Enter the fields firstname,lastname,streetaddress,zipcode,phnnumber,sectionnumber,department,annualsalary
Venu
Dammalapati
stambalagaruvu
522006
6429743283
AP12131
ENG
20000
```

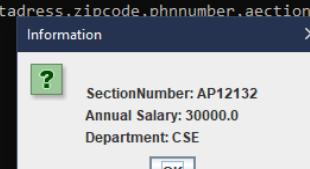


Faculty :

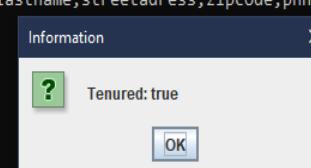
```
Enter the fields firstname,lastname,streetaddress,zipcode,phnnumber,aection number,department,Anual salary,Tenured
Bavita
Moparthi
NTR stadium
522005
8100643524
true
AP12132
CSE
30000
```



```
Enter the fields firstname,lastname,streetaddress.zipcode.phnnumber.aection number,department,Anual salary,Tenured
Bavita
Moparthi
NTR stadium
522005
8100643524
true
AP12132
CSE
30000
```



```
Enter the fields firstname,lastname,streetaddress,zipcode,phnnumber,aection number,department,Anual salary,Tenured
Bavita
Moparthi
NTR stadium
522005
8100643524
true
AP12132
CSE
30000
```

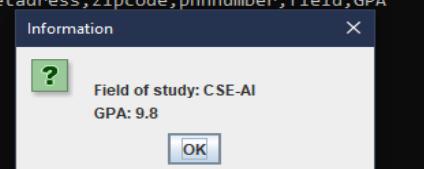


Student :

```
Enter the fields firstname,lastname,streetaddress,zipcode,phnnumber,field,GPA
Sirisha
Maddala
Brodipet
522012
7621912912
CSE-AI
9.8
```



```
Enter the fields firstname,lastname,streetaddress,zipcode,phnnumber,field,GPA
Sirisha
Maddala
Brodipet
522012
7621912912
CSE-AI
9.8
```



Final Output along with the inputs given above :

```
C:\Users\user\Downloads\198CI7039>javac Collegelist.java
C:\Users\user\Downloads\198CI7039>java Collegelist
Enter the fields firstname,lastname,streetaddress,zipcode,phnnumber
Harsha
popuri
Colony
522006
7533438294
Enter the first letter of the field you have to enter
1.College employee
2.Faculty
3.Student
4.Quit
C
Enter the fields firstname,lastname,streetaddress,zipcode,phnnumber,sectionnumber,department,annualsalary
Venu
Dammalapati
stambalagaruvu
522006
6429743283
AP12131
ENG
20000
Enter the first letter of the field you have to enter
1.College employee
2.Faculty
3.Student
4.Quit
C
Enter the fields firstname,lastname,streetaddress,zipcode,phnnumber,sectionnumber,department,annualsalary
Raja
Narru
brindavangardens
522019
6410318264
AP12132
CSE
50000
Enter the first letter of the field you have to enter
1.College employee
2.Faculty
3.Student
4.Quit
```

```

C
Enter the fields firstname,lastname,streetaddress,zipcode,phnnumber,sectionnumber,department,annualsalary
Jayanth
Kalyanam
Donkaroad
522021
7130061037
AP12133
ECE
25000
Enter the first letter of the field you have to enter
1.College employee
2.Faculty
3.Student
4.Quit
C
Enter the fields firstname,lastname,streetaddress,zipcode,phnnumber,sectionnumber,department,annualsalary
Rajesh
Malhotra
Rajebdrnagar
522019
7583926724
AP12134
Maths
35000
Enter the first letter of the field you have to enter
1.College employee
2.Faculty
3.Student
4.Quit
C
CollegeEmployeearray is out of bounds
Enter the first letter of the field you have to enter
1.College employee
2.Faculty
3.Student
4.Quit
F
Enter the fields firstname,lastname,streetaddress,zipcode,phnnumber,section number,department,Anuual salary,Tenured
Bavita
Moparthi
NTR stadium
522005
8100643524
true
AP12135

Enter the first letter of the field you have to enter
1.College employee
2.Faculty
3.Student
4.Quit
S
Enter the fields firstname,lastname,streetaddress,zipcode,phnnumber,field,GPA
Sirisha
Maddala
Brodipet
522012
7621912912
CSE-AI
9.8
Enter the first letter of the field you have to enter
1.College employee
2.Faculty
3.Student
4.Quit
Q
CollegeEmployee
Firstname : Venu
Lastname : Dammalapati
streetaddress : stambalagaruvu
Zipcode : 522006
Phn number : 6429743283
Section number : AP12131
Department : ENG
Annual Salary : 20000.0
Firstname : Raja
Lastname : Narra
streetaddress : brindavangardens
Zipcode : 522019
Phn number : 6410318264
Section number : AP12132
Department : CSE
Annual Salary : 50000.0
Firstname : Jayanth
Lastname : Kalyanam
streetaddress : Donkaroad
Zipcode : 522021
Phn number : 7130061037
Section number : AP12133
Department : ECE
Annual Salary : 25000.0

```

```
Firstname : Rajesh
Lastname : Malhotra
streetaddress : Rajebdrnagar
Zipcode : 522019
Phn number : 7583926724
Section number : AP12134
Department : Maths
Annual Salary : 35000.0
Faculty
Firstname : Bavita
Lastname : Moparthi
streetaddress : NTR stadium
Zipcode : 522005
Phn number : 8100643524
Section number : AP12135
Department : CSE
Annual Salary : 30000.0
Tenured : true
Further objecs for Faculty class are not yet created
Student
Firstname : Sirisha
Lastname : Maddala
streetaddress : Brodipet
Zipcode : 522012
Phn number : 7621912912
field :CSE-AI
GPA :9.8
Further objecs for student class are not yet created
```

Oops Lab Assessment-4

CSE2005-L1

P. Harsha vardhan

19BCI7039

1.

Create an abstract class named Book. Include a String field for the book's title and a double field for the book's price. Within the class, include a constructor 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.

Code :

```
import java.util.*;
public class UseBook {
public static void main(String args[])
{
    Scanner sc=new Scanner(System.in);
    System.out.println("Enter the title of the book");
    String s=sc.nextLine();
    System.out.println("Enter Whether the book is fictional or not");
    String s2=sc.nextLine();
    if(s2.charAt(0)=='F' | |s2.charAt(0)=='f')
    {
        Fiction f=new Fiction(s);
        f.setPrice();
        System.out.println("Title of the fictional book is "+f.gettitle());
        System.out.println("price of the fictional book is "+f.getprice());
    }
    else if(s2.charAt(0)=='N' | |s2.charAt(0)=='n')
    {
        NonFiction nf=new NonFiction(s);
        nf.setPrice();
        System.out.println("Title of the Non fictional book is "+nf.gettitle());
```

```
        System.out.println("price of the non fictional book is "+nf.getPrice());
    }
} else
{
    System.out.println("Wrong input given");
}
sc.close();
}
}

abstract class Book
{
    String title;
    double price;
    abstract void setPrice();
    Book(String title)
    {
        this.title=title;
    }
    String getTitle()
    {
        return title;
    }
    double getPrice()
    {
        return price;
    }
}
class Fiction extends Book
{
    Fiction(String title)
    {
        super(title);
    }
    void setPrice()
    {
        price=24.99;
    }
}
class NonFiction extends Book
{
    NonFiction(String title)
    {
        super(title);
    }
}
```

```

void setPrice()
{
    price=37.99;
}
}

```

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.

Code :

```

import java.util.*;
public class BookArray
{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        String ba[]=new String[10];
        for(int i=0;i<10;i++)
        {
            System.out.println("Enter fictional or not");
            String b1=sc.nextLine();
            if(b1.charAt(0)=='f' | |b1.charAt(0)=='F')
            {
                System.out.println("Enter the title of the fictional book");
                String title=sc.nextLine();
                Fiction f=new Fiction(title);
                f.setPrice();
                ba[i]=title+" - Fictional ";
            }
            else
            {
                System.out.println("Enter the title of the non fictional book");
                String title=sc.nextLine();
                NonFiction nf=new NonFiction(title);
                nf.setPrice();
                ba[i]=title+" - notfictional";
            }
        }
        for(int l=0;l<10;l++)
        {
            System.out.println(ba[l]);
        }
    }
}

```

```
}
```

Output :

```
C:\Users\user\Downloads\19BCI7039>javac BookArray.java
```

```
C:\Users\user\Downloads\19BCI7039>java BookArray
```

```
Enter fictional or not
```

```
fictional
```

```
Enter the title of the fictional book
```

```
lightening theif
```

```
Enter fictional or not
```

```
not fictional
```

```
Enter the title of the non fictional book
```

```
wings of fire
```

```
Enter fictional or not
```

```
fictional
```

```
Enter the title of the fictional book
```

```
harry potter
```

```
Enter fictional or not
```

```
not fictional
```

```
Enter the title of the non fictional book
```

```
Demigod
```

```
Enter fictional or not
```

```
fictional
```

```
Enter the title of the fictional book
```

```
socererersstone
```

```
Enter fictional or not
```

```
not fictional
```

```
Enter the title of the non fictional book
```

```
blackking
```

```
Enter fictional or not
```

```
fictional
```

```
Enter the title of the fictional book
```

```
lord of rings
```

```
Enter fictional or not
```

```
fictional
```

```
Enter the title of the fictional book
```

```
ironman
```

```
Enter fictional or not
```

```
notfictional
```

```
Enter the title of the non fictional book
```

```
malaysia
```

```
Enter fictional or not
```

```
fictional
```

```
Enter the title of the fictional book
```

```
King of wizards
```

```
class Fiction
Title : lightening theif
price : 24.99
class NonFiction
Title : wings of fire
price : 37.99
class Fiction
Title : harry potter
price : 24.99
class NonFiction
Title : Demigod
price : 37.99
class Fiction
Title : sacerersstone
price : 24.99
class NonFiction
Title : blackking
price : 37.99
class Fiction
Title : lord of rings
price : 24.99
class Fiction
Title : ironman
price : 24.99
class NonFiction
Title : malaysia
price : 37.99
class Fiction
Title : king of wizards
price : 24.99
```

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.

Code :

```
import java.util.*;
public class DemoPhoneCalls {
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter the phone number of the Incoming call");
        String phnno=sc.nextLine();
        IncomingPhoneCall ipc=new IncomingPhoneCall(phnno);
        ipc.display();
        System.out.println("Enter the phone number of the Outgoing call");
        String phnno1=sc.nextLine();
        double time=sc.nextDouble();
        OutgoingPhoneCall opc=new OutgoingPhoneCall(phnno1,time);
        opc.display();
        sc.close();
    }
}
abstract class PhoneCall{
    String phnnum;
    double price;
    PhoneCall(String phnno)
    {
        phnnum=phnno;
        price=0.0;
    }
    abstract String getphnnum();
    abstract double getprice();
    abstract void display();
}
class IncomingPhoneCall extends PhoneCall
{
    IncomingPhoneCall(String phnno)
    {
        super(phnno);
        price=0.02;
    }
    public String getphnnum()
    {
        return phnnum;
    }
    public double getprice()
    {
        return price;
    }
}
```

```
public void display()
{
    System.out.println("Incoming call");
    System.out.println("Phone number is : "+getphnnum());
    System.out.println("Price of the incoming call is : "+getprice());
}
}

class OutgoingPhoneCall extends PhoneCall
{
    double time;
    OutgoingPhoneCall(String phnno,double time)
    {
        super(phnno);
        this.time=time;
        price=time*0.04;
    }
    public String getphnnum()
    {
        return phnnum;
    }
    public double getprice()
    {
        return price;
    }
    public void display()
    {
        System.out.println("Outgoing call");
        System.out.println("Phone number is : "+getphnnum());
        System.out.println("Time : "+time);
        System.out.println("Rate per minute is : 0.04");
        System.out.println("Price of the incoming call is : "+getprice());
    }
}
```

Output :

```
C:\Users\user\Downloads\19BCI7039>javac DemoPhoneCalls.java

C:\Users\user\Downloads\19BCI7039>java DemoPhoneCalls
Enter the phone number of the Incoming call
9963015238
Incoming call
Phone number is : 9963015238
Price of the incoming call is : 0.02
Enter the phone number of the Outgoing call
7702772066
65
Outgoing call
Phone number is : 7702772066
Time : 65.0
Rate per minute is : 0.04
Price of the incoming call is : 2.6

C:\Users\user\Downloads\19BCI7039>
```

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.

Code :

```
import java.util.*;
public class PhoneCallArray {
public static void main(String args[])
{
    Scanner sc=new Scanner(System.in);
    PhoneCall a[]=new PhoneCall[8];
    for(int i=0;i<8;i++)
    {
        System.out.println("Enter incoming or outgoing");
        String b1=sc.nextLine();
        if(b1.charAt(0)=='i' | b1.charAt(0)=='I')
        {
            System.out.println("Enter the phonenumber to the incoming call");
            String title=sc.nextLine();
            IncomingPhoneCall ipc=new IncomingPhoneCall(title);
            a[i]=ipc;
        }
        else
        {
            System.out.println("Enter the phonenumber and the time taken for the outgoing
call");
            String title=sc.nextLine();
            double time=sc.nextDouble();
            String num=sc.nextLine();
        }
    }
}
```

```
        OutgoingPhoneCall opc=new OutgoingPhoneCall(title,time);
        a[i]=opc;
    }
}
for(int i=0;i<8;i++)
{
    a[i].display();
}
sc.close();
}
}
```

Output :

```
C:\Users\user\Downloads\19BCI7039>javac PhoneCallArray.java

C:\Users\user\Downloads\19BCI7039>java PhoneCallArray
Enter incoming or outgoing
incoming
Enter the phonenumber to the incoming call
9738173011
Enter incoming or outgoing
incoming
Enter the phonenumber to the incoming call
6412812931
Enter incoming or outgoing
outgoing
Enter the phonenumber and the time taken for the outgoing call
1297319381
50.03
Enter incoming or outgoing
incomin
Enter the phonenumber to the incoming call
9131387128
Enter incoming or outgoing
outgoing
Enter the phonenumber and the time taken for the outgoing call
1831037931
5
Enter incoming or outgoing
outgoing
Enter the phonenumber and the time taken for the outgoing call
8310361937
40.32
Enter incoming or outgoing
incoming
Enter the phonenumber to the incoming call
8730741832
Enter incoming or outgoing
incoming
Enter the phonenumber to the incoming call
8160324613
Incoming call
Phone number is : 9738173011
Price of the incoming call is : 0.02
Incoming call
Phone number is : 6412812931
Price of the incoming call is : 0.02
Outgoing call
```

```
Outgoing call
Phone number is : 1297319381
Time : 50.03
Rate per minute is : 0.04
Price of the incoming call is : 2.0012
Incoming call
Phone number is : 9131387128
Price of the incoming call is : 0.02
Outgoing call
Phone number is : 1831037931
Time : 5.0
Rate per minute is : 0.04
Price of the incoming call is : 0.2
Outgoing call
Phone number is : 8310361937
Time : 40.32
Rate per minute is : 0.04
Price of the incoming call is : 1.6128
Incoming call
Phone number is : 8730741832
Price of the incoming call is : 0.02
Incoming call
Phone number is : 8160324613
Price of the incoming call is : 0.02
```

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.

Codes:

```
import java.util.*;
interface Turner{
    public void turn();
}
class Leaf implements Turner{
    public void turn() {
        System.out.println("colour");
    }
}
class Page implements Turner{
    public void turn() {
        System.out.println("Going to next page");
    }
}
class Pancake implements Turner{
```

```

        public void turn() {
            System.out.println("Flipping");
        }
    }
class DemoTurners{
    public static void main(String args[]) {
        Leaf l=new Leaf();
        l.turn();
        Page p=new Page();
        p.turn();
        Pancake k=new Pancake();
        k.turn();
    }
}

```

Output :

```

C:\Users\user\Downloads\19BCI7039>javac DemoTurners.java

C:\Users\user\Downloads\19BCI7039>java DemoTurners
colour
Going to next page
Flipping

```

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().

Code :

```

import java.util.*;
interface Turner{
    public void turn();
}
class Car implements Turner
{
    public void turn()
    {
        System.out.println("Rotate the steering either clockwise or anti clockwise");
    }
}
class Bike implements Turner
{
    public void turn()
    {
        System.out.println("Change the direction of the handle in the required way");
    }
}

```

```

}
public class DemoTurners2
{
    public static void main(String args[])
    {
        Car c=new Car();
        c.turn();
        Bike b=new Bike();
        b.turn();
    }
}

```

Output :

```

C:\Users\user\Downloads\19BCI7039>javac DemoTurners2.java

C:\Users\user\Downloads\19BCI7039>java DemoTurners2
Rotate the steering either clockwise or anti clockwise
Change the direction of the handle in the required way

C:\Users\user\Downloads\19BCI7039>

```

3.c

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

Code:

```

//Apply Dynamic method dispatch to show the power of it and name the class as
DemoTurners3.
import java.util.*;
interface Turner{
    public void turn();
}
class Leaf implements Turner{
    public void turn() {
        System.out.println("colour");
    }
}
class Page implements Turner{
    public void turn() {
        System.out.println("Going to next page");
    }
}
class Pancake implements Turner{
    public void turn() {
        System.out.println("Flipping");
    }
}

```

```

class DemoTurners3{
    public static void main(String args[]) {
        Turner t;
        Leaf leaf=new Leaf();
        Page page=new Page();
        Pancake pancake=new Pancake();
        t=leaf;
        t.turn();
        t=page;
        t.turn();
        t=pancake;
        t.turn();
    }
}

```

Output :

```

C:\Users\user\Downloads\19BCI7039>javac DemoTurners3.java

C:\Users\user\Downloads\19BCI7039>java DemoTurners3
colour
Going to next page
Flipping

```

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.

Code :

```

//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.
import java.util.*;
abstract class GeometricFigure
{
    String figuretype;
    int height;
    int width;
    GeometricFigure(String type,int hght,int width)
    {
        figuretype=type;
    }
}

```

```

        height=hght;
        this.width=width;
    }
    abstract void area();
}
class Square extends GeometricFigure
{
    Square(String type,int side)
    {
        super(type,side,side);
    }
    void area()
    {
        System.out.println("The area of "+figuretype+" is
"+((double)(height)*(double)(height)));
    }
}
class Triangle extends GeometricFigure
{
    Triangle(String type,int hght,int width)
    {
        super(type,hght,width);
    }
    void area()
    {
        System.out.println("The area of "+figuretype+" is
"+(0.5*(double)(height)*(double)(width)));
    }
}
public class UseGeometric
{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter the side of the square");
        int s=sc.nextInt();
        Square s2=new Square("Square",s);
        s2.area();
        System.out.println("Enter the height and width of the triangle");
        int hght=sc.nextInt();
        int width=sc.nextInt();
        Triangle t=new Triangle("Triangle",hght,width);
        t.area();
    }
}

```

```
}
```

Output :

```
C:\Users\user\Downloads\19BCI7039>javac UseGeometric.java
C:\Users\user\Downloads\19BCI7039>java UseGeometric
Enter the side of the square
5
The area of Square is 25.0
Enter the height and width of the triangle
3
4
The area of Triangle is 6.0
```

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.

Code:

```
import java.util.*;
interface SidedObject
{
    public void displaySides();
}
abstract class GeometricFigure2 implements SidedObject
{
    String figuretype;
    int height;
    int width;
    GeometricFigure2(String type,int hght,int width)
    {
        figuretype=type;
        height=hght;
        this.width=width;
    }
    abstract void area();
}
class Square2 extends GeometricFigure2
{
    Square2(String type,int side)
    {
        super(type,side,side);
    }
```

```

void area()
{
    System.out.println("The area of "+figuretype+" is
"+((double)(height)*(double)(height)));
}
public void displaySides()
{
    System.out.println("No of sides : 4");
}
}
class Triangle2 extends GeometricFigure2
{
    Triangle2(String type,int hght,int width)
    {
        super(type,hght,width);
    }
    void area()
    {
        System.out.println("The area of "+figuretype+" is
"+(0.5*(double)(height)*(double)(width)));
    }
    public void displaySides()
    {
        System.out.println("No of sides : 3");
    }
}
public class UseGeometric2
{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter the side of the square");
        int s=sc.nextInt();
        Square2 s2=new Square2("Square",s);
        s2.area();
        s2.displaySides();
        System.out.println("Enter the height and width of the triangle");
        int hght=sc.nextInt();
        int width=sc.nextInt();
        Triangle2 t=new Triangle2("Triangle",hght,width);
        t.area();
        t.displaySides();
    }
}

```

Output :

```
C:\Users\user\Downloads\19BCI7039>javac UseGeometric2.java
C:\Users\user\Downloads\19BCI7039>java UseGeometric2
Enter the side of the square
4
The area of Square is 16.0
No of sides : 4
Enter the height and width of the triangle
2
3
The area of Triangle is 3.0
No of sides : 3
```

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**.

Code :

```
import java.util.*;
```

```

abstract class Loans implements LoanConstants{
    String loannum,lastname;
    double loanamount,interestrate,term;
    Loans(double loanamount,double term,String loannum,String lastname)
    {
        if(loanamount>maximumloanamount)
        {
            System.out.println("Loan is not given as the loan amount is greater than 100000");
        }
        else
        {
            this.loanamount=loanamount;
            this.lastname=lastname;
            this.loannum=loannum;
            if(term!=shortterm&&term!=mediumterm&&term!=longterm)
            {
                this.term=shortterm;
            }
            else
            {
                this.term=term;
            }
        }
    }
    public String toString()
    {
        if(loanamount==0)
        {
            return "Nothing to display as the loan is not sanctioned";
        }
        else
        {
            return "Type of loan - "+getClass()+"\nLoan number : "+loannum+"\nLast name : "+lastname+"\nLoan amount : "+loanamount+"\nInterest rate : "+interestrate+"\nTerm : "+term;
        }
    }
    public void amountowed(double days)
    {
        double loanfee=(loanamount*interestrate)*days;
        double amount =loanfee+loanamount;
        System.out.println("The amount owed till today is "+amount);
    }
}

```

```

interface LoanConstants
{
    final int shortterm=1;
    final int mediumterm=3;
    final int longterm=5;
    final String companyname="Sanchez Construction Loan Co.";
    final int maximumloanamount=100000;
}
class BusinessLoan extends Loans
{
    BusinessLoan(double loanamount,double term,String loannum,String lastname,double cpir)
    {
        super(loanamount,term,loannum,lastname);
        interestrate=cpir*0.01;
    }
}
class PersonalLoan extends Loans
{
    PersonalLoan(double loanamount,double term,String loannum,String lastname,double cpir)
    {
        super(loanamount,term,loannum,lastname);
        interestrate=cpir*0.02;
    }
}
public class CreateLoans{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        Loans l[]=new Loans[5];
        System.out.println("Enter the current prime interest rate");
        double cpir=sc.nextDouble();
        String cop=sc.nextLine();
        for(int i=0;i<5;i++)
        {
            System.out.println("Enter the loannum,lastname,loanamount,term");
            String loannum=sc.nextLine();
            String lastname=sc.nextLine();
            double loanamount=sc.nextDouble();
            double term=sc.nextDouble();
            String q=sc.nextLine();
            System.out.println("Enter the type of loan : ");
            String type=sc.nextLine();
            if(type.charAt(0)=='p'||type.charAt(0)=='P')
            {

```

```
PersonalLoan pl=new PersonalLoan(loanamount,term,loannum,lastname,cpir);
l[i]=pl;
}
if(type.charAt(1)=='b' | type.charAt(0)=='B')
{
    BusinessLoan bl=new BusinessLoan(loanamount,term,loannum,lastname,cpir);
    l[i]=bl;
}
}
for(int i=0;i<5;i++)
{
    System.out.println(l[i].toString());
    System.out.println("Enter the no of days till now from the begining of the loan period");
    double days=sc.nextDouble();
    l[i].amountowed(days);
}
}
```

Output :

```
C:\Users\user\Downloads\19BCI7039>javac CreateLoans.java

C:\Users\user\Downloads\19BCI7039>java CreateLoans
Enter the current prime interest rate
5
Enter the loannum,lastname,loanamount,term
12deo
popuri
10000
5
Enter the type of loan :
Personal
Enter the loannum,lastname,loanamount,term
13deo
moparthi
20000
1
Enter the type of loan :
Business
Enter the loannum,lastname,loanamount,term
14deo
Nittoor
150000
3
Enter the type of loan :
Business
Loan is not given as the loan amount is greater than 100000
Enter the loannum,lastname,loanamount,term
15deo
Maddala
30000
3
Enter the type of loan :
Business
Enter the loannum,lastname,loanamount,term
16deo
Darsi
12300
5
Enter the type of loan :
Business
```

```
Type of loan - class PersonalLoan
Loan number : 12deo
Last name : popuri
Loan amount : 10000.0
Interest rate : 0.1
Term : 5.0
Enter the no of days till now from the begining of the loan period
30
The amount owed till today is 40000.0
Type of loan - class BusinessLoan
Loan number : 13deo
Last name : moparthi
Loan amount : 20000.0
Interest rate : 0.05
Term : 1.0
Enter the no of days till now from the begining of the loan period
15
The amount owed till today is 35000.0
Nothing to display as the loan is not sanctioned
Enter the no of days till now from the begining of the loan period
16
The amount owed till today is 0.0
Type of loan - class BusinessLoan
Loan number : 15deo
Last name : Maddala
Loan amount : 30000.0
Interest rate : 0.05
Term : 3.0
Enter the no of days till now from the begining of the loan period
25
The amount owed till today is 67500.0
Type of loan - class BusinessLoan
Loan number : 16deo
Last name : Darsi
Loan amount : 12300.0
Interest rate : 0.05
Term : 5.0
Enter the no of days till now from the begining of the loan period
27
The amount owed till today is 28905.0
```

```
C:\Users\user\Downloads\19BCI7039>
```

Oops Lab Assessment-5

CSE2005-L1

P. Harsha vardhan

19BCI7039

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

Code :

```
import java.util.*;
import java.lang.reflect.Method;
public class ArrayPrint
{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        Printer myPrinter = new Printer();
        System.out.println("Enter the sizes of string array and integer arrays : ");
        int n=sc.nextInt();
        int n1=sc.nextInt();
        Integer a[]=new Integer[n1];

        String s1=sc.nextLine();
        String s[]=new String[n];
        System.out.println("Enter the elements of the string array");
        for(int i=0;i<n;i++)
        {
            s[i]=sc.nextLine();
        }
        System.out.println("Enter the elements of the integer array");
        for(int i=0;i<n1;i++)
        {
            a[i]=sc.nextInt();
        }
    }
}
```

```

        System.out.print("Elements of the Integer array : ");
        myPrinter.printArray( a );
        System.out.print("Elements of the String array : ");
        myPrinter.printArray( s );
    }
}

class Printer {
    public <inputType> void printArray(inputType[] array) {
        for (int i = 0; i < array.length-1; i++) {
            System.out.print(array[i]+", ");
        }
        System.out.println(array[array.length-1]);
    }
}

```

Output :

```

C:\Users\user\Downloads\19BCI7039>javac ArrayPrint.java

C:\Users\user\Downloads\19BCI7039>java ArrayPrint
Enter the sizes of string array and integer arrays :
4
3
Enter the elements of the string array
Venu
Mustaq
Harsha
Raja
Enter the elements of the integer array
7039
23
1923
Elements of the Integer array : 7039, 23, 1923
Elements of the string array : Venu, Mustaq, Harsha, Raja

```

2.

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

Code :

```

public class GenericLinearSearch {
    public static void main(String args[])
    {
        Object a[]=new Object[2]
        //The below objects are taken from the labassgn3 UseBook.java file
        Fiction f=new Fiction("Harsha");
        NonFiction nf=new NonFiction("Venu");
        Fiction f1=new Fiction("King");
        a[0]=f;
        a[1]=nf;
    }
}
```

```

        linearsearchDemo lsd=new linearsearchDemo();
        lsd.linearsearch(a,f);
        lsd.linearsearch(a,f1);
    }
}

class linearsearchDemo
{
    public <Object>void linearsearch(Object a[],Object b)
    {
        int j=0;
        for(int i=0;i<a.length;i++)
        {
            if(a[i]==b)
            {
                j=j+1;
            }
        }
        if(j>0)
        {
            System.out.println("Object "+b+" is found in the array of objects");
        }
        else
        {
            System.out.println("Object "+b+" is not found in the array of objects");
        }
    }
}

```

Output:

```

C:\Users\user\Downloads>cd 19BCI7039

C:\Users\user\Downloads\19BCI7039>javac GenericLinearSearch.java

C:\Users\user\Downloads\19BCI7039>java GenericLinearSearch
Object Fiction@61a52fdb is found in the array of objects
Object Fiction@7dbc5d3 is not found in the array of objects

C:\Users\user\Downloads\19BCI7039>

```

3.a

3.

Write a generic class called `GenericStack<T>` that represents a stack structure. A stack structure follows 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.

Create a class GenericStackDemo, in which you create two stacks, one stack of String and one Stack of students, add elements, print, then remove all elements and then print.

Code:

```
import java.util.*;
import javax.swing.*;
public class GenericStackDemo {
    public static void main(String args[])
    {
        System.out.println("Stack of Strings");
        GenericStack<String> gs = new GenericStack<String>();
        gs.push("Harsha");
        gs.push("Varsha");
        gs.push("Madhurya");
        gs.print();
        gs.pop();
        gs.pop();
        gs.pop();
        gs.print();
        System.out.println("Stack of Students");
        GenericStack<Student> gs1=new GenericStack<Student>();
        Student s1=new Student(1);
        gs1.push(s1);
        Student s2=new Student(2);
        gs1.push(s2);
        Student s3=new Student(3);
        gs1.push(s3);
```

```

        gs1.print();
        gs1.pop();
        gs1.pop();
        gs1.pop();
        gs1.print();
    }
}
class GenericStack<Type>
{
    ArrayList<Type> a;
    public GenericStack()
    {
        a=new ArrayList<Type>();
    }
    int i=-1;
    void push(Type e)
    {
        a.add(++i,e);
        System.out.println(e+" is added to the stack");
    }
    void pop()
    {
        if(i!=-1)
        {
            System.out.println(a.get(i)+" is removed from the stack");
            a.remove(i);
            i--;
        }
        else
        {
            System.out.println("Stack is empty");
        }
    }
    void print()
    {
        if(i!=-1)
        {
            for(int j=i;j>=0;j--)
            {
                System.out.println(a.get(j));
            }
        }
        else
        {

```

```

        System.out.println("Stack is empty");
    }
}
class Student
{
    int i;
    public Student(int i)
    {
        this.i=i;
    }
    public String toString()
    {
        return "Student object "+i;
    }
}

```

Output :

```

C:\Users\user\Downloads\19BCI7039>javac GenericStackDemo.java

C:\Users\user\Downloads\19BCI7039>java GenericStackDemo
Stack of Strings
Harsha is added to the stack
Varsha is added to the stack
Madhurya is added to the stack
Madhurya
Varsha
Harsha
Madhurya is removed from the stack
Varsha is removed from the stack
Harsha is removed from the stack
Stack is empty
Stack of Students
Student object 1 is added to the stack
Student object 2 is added to the stack
Student object 3 is added to the stack
Student object 3
Student object 2
Student object 1
Student object 3 is removed from the stack
Student object 2 is removed from the stack
Student object 1 is removed from the stack
Stack is empty

```

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.

Code :

```
import java.util.*;  
interface GenericStackable<Type>  
{  
    abstract void push(Type e);  
    abstract void pop();  
    abstract void print();  
    abstract boolean isEmpty();  
}  
public class GenericStackDemo2 {  
    public static void main(String args[])  
    {  
        System.out.println("Stack of Strings");  
        GenericStack<String> gs = new GenericStack<String>();  
        if(gs.isEmpty())  
        {  
            System.out.println("Stack is empty");  
        }  
        else{  
            System.out.println("Stack is not empty");  
        }  
        gs.push("Harsha");  
        gs.push("Varsha");  
        gs.push("Madhurya");  
        if(gs.isEmpty())  
        {  
            System.out.println("Stack is empty");  
        }  
        else{  
            System.out.println("Stack is not empty");  
        }  
        gs.print();  
        gs.pop();
```

```

        gs.pop();
        gs.pop();
        gs.print();
        System.out.println("Stack of Students");
        GenericStack<Student> gs1=new GenericStack<Student>();
        if(gs.isEmpty())
        {
            System.out.println("Stack is empty");
        }
        else{
            System.out.println("Stack is not empty");
        }
        Student s1=new Student(1);
        gs1.push(s1);
        Student s2=new Student(2);
        gs1.push(s2);
        Student s3=new Student(3);
        gs1.push(s3);
        if(gs.isEmpty())
        {
            System.out.println("Stack is empty");
        }
        else{
            System.out.println("Stack is not empty");
        }
        gs1.print();
        gs1.pop();
        gs1.pop();
        gs1.pop();
        gs1.print();
    }
}
class GenericStack<Type>implements GenericStackable<Type>
{
    ArrayList<Type> a;
    public GenericStack()
    {
        a=new ArrayList<Type>();
    }
    int i=-1;
    public void push(Type e)
    {
        a.add(++i,e);
        System.out.println(e+" is added to the stack");
    }
}

```

```
    }
    public void pop()
    {
        if(i!=-1)
        {
            System.out.println(a.get(i)+" is removed from the stack");
            a.remove(i);
            i--;
        }
        else
        {
            System.out.println("Stack is empty");
        }
    }
    public void print()
    {
        if(i!=-1)
        {
            for(int j=i;j>=0;j--)
            {
                System.out.println(a.get(j));
            }
        }
        else
        {
            System.out.println("Stack is empty");
        }
    }
    public boolean isEmpty()
    {
        if(i==-1)
        {
            return true;
        }
        else
        {
            return false;
        }
    }
}
class Student
{
    int i;
    public Student(int i)
```

```
{  
    this.i=i;  
}  
public String toString()  
{  
    return "Student object "+i;  
}  
}
```

Output :

```
C:\Users\user\Downloads\19BCI7039>javac GenericStackDemo2.java  
  
C:\Users\user\Downloads\19BCI7039>java GenericStackDemo2  
Stack of Strings  
Stack is empty  
Harsha is added to the stack  
Varsha is added to the stack  
Madhurya is added to the stack  
Stack is not empty  
Madhurya  
Varsha  
Harsha  
Madhurya is removed from the stack  
Varsha is removed from the stack  
Harsha is removed from the stack  
Stack is empty  
Stack of Students  
Stack is empty  
Student object 1 is added to the stack  
Student object 2 is added to the stack  
Student object 3 is added to the stack  
Stack is empty  
Student object 3  
Student object 2  
Student object 1  
Student object 3 is removed from the stack  
Student object 2 is removed from the stack  
Student object 1 is removed from the stack  
Stack is empty  
  
C:\Users\user\Downloads\19BCI7039>
```

Oops Lab Assessment-6

CSE2005-L1

P. Harsha vardhan

19BCI7039

1.

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 beginning 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.

Code :

```
import java.util.*;  
interface GenericQueuable<Type>  
{  
    abstract <Type extends Person> void insertEnd(Type e);  
    abstract void removeBegin();  
    abstract void printQueue();  
    abstract boolean isQueueEmpty();  
}  
class GenericQueue<Type> implements GenericQueuable<Type>  
{  
    int i=0;  
    Object queue[];  
    GenericQueue(Type a[])  
    {  
        queue=a;  
    }  
    public <Type extends Person>void insertEnd(Type e)  
    {  
        queue[i]=e;  
    }  
}
```

```

        System.out.println(e+" is inserted in the queue");
        i++;
    }
    public void removeBegin()
    {
        System.out.println(queue[0]+" is removed from the queue");
        for(int i=0;i<queue.length-1;i++)
        {
            queue[i]=queue[i+1];
        }
        queue[queue.length-1]=null;
    }
    public void printQueue()
    {
        if(!isQueueEmpty())
        {
            for(int i=0;queue[i]!=null;i++)
            {
                System.out.println(queue[i]);
            }
        }
        else
        {
            System.out.println("Queue is empty");
        }
    }
    public boolean isQueueEmpty()
    {
        if(i==0)
        {
            return true;
        }
        else
        {
            return false;
        }
    }
}
public class GenericQueueDemo
{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        Person p[]=new Person[10];
        CollegeEmployee ce[]=new CollegeEmployee[10];
        Student s[]=new Student[10];
        GenericQueue g=new GenericQueue(p);
        GenericQueue g1=new GenericQueue(ce);
    }
}

```

```

GenericQueue g2=new GenericQueue(s);
g.printQueue();
for(int i=0;i<10;i++)
{
    Person c=new Person(i);
    g.insertEnd(c);
}
g.removeBegin();
g.removeBegin();
g.removeBegin();
g.removeBegin();
g.removeBegin();
g.printQueue();
g.isEmpty();
g1.printQueue();
for(int i=0;i<10;i++)
{
    CollegeEmployee c1=new CollegeEmployee(i);
    g1.insertEnd(c1);
}
g1.removeBegin();
g1.removeBegin();
g1.removeBegin();
g1.removeBegin();
g1.removeBegin();
g1.printQueue();
g1.isEmpty();
g2.printQueue();
for(int i=0;i<10;i++)
{
    Student s2=new Student(i);
    g2.insertEnd(s2);
}
g2.removeBegin();
g2.removeBegin();
g2.removeBegin();
g2.removeBegin();
g2.removeBegin();
g2.printQueue();
g2.isEmpty();
sc.close();
}

}

```

Output :

```
C:\Users\user\Downloads\19BCI7039>javac GenericQueueDemo.java
Note: GenericQueueDemo.java uses unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.

C:\Users\user\Downloads\19BCI7039>java GenericQueueDemo
Queue is empty
Person 1 is inserted in the queue
Person 2 is inserted in the queue
Person 3 is inserted in the queue
Person 4 is inserted in the queue
Person 5 is inserted in the queue
Person 6 is inserted in the queue
Person 7 is inserted in the queue
Person 8 is inserted in the queue
Person 9 is inserted in the queue
Person 10 is inserted in the queue
Person 1 is removed from the queue
Person 2 is removed from the queue
Person 3 is removed from the queue
Person 4 is removed from the queue
Person 5 is removed from the queue
Person 6
Person 7
Person 8
Person 9
Person 10
Queue is empty
CollegeEmployee object 1 is inserted in the queue
CollegeEmployee object 2 is inserted in the queue
CollegeEmployee object 3 is inserted in the queue
CollegeEmployee object 4 is inserted in the queue
CollegeEmployee object 5 is inserted in the queue
CollegeEmployee object 6 is inserted in the queue
CollegeEmployee object 7 is inserted in the queue
CollegeEmployee object 8 is inserted in the queue
CollegeEmployee object 9 is inserted in the queue
CollegeEmployee object 10 is inserted in the queue
CollegeEmployee object 1 is removed from the queue
CollegeEmployee object 2 is removed from the queue
CollegeEmployee object 3 is removed from the queue
CollegeEmployee object 4 is removed from the queue
CollegeEmployee object 5 is removed from the queue
CollegeEmployee object 6
CollegeEmployee object 7
CollegeEmployee object 8
CollegeEmployee object 9
CollegeEmployee object 10
Queue is empty
Student object 1 is inserted in the queue
Student object 2 is inserted in the queue
Student object 3 is inserted in the queue
Student object 4 is inserted in the queue
Student object 5 is inserted in the queue
Student object 6 is inserted in the queue
Student object 7 is inserted in the queue
Student object 8 is inserted in the queue
Student object 9 is inserted in the queue
Student object 10 is inserted in the queue
Student object 1 is removed from the queue
Student object 2 is removed from the queue
Student object 3 is removed from the queue
Student object 4 is removed from the queue
Student object 5
Student object 6
Student object 7
Student object 8
Student object 9
Student object 10

C:\Users\user\Downloads\19BCI7039>
```

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.

Code :

```
import java.util.*;  
interface MyMap<Key,Value>  
{  
    abstract void add(Key k,Value v);  
    abstract Value remove(Key k);  
    abstract int size();  
    abstract boolean isEmpty();  
    abstract ArrayList<Key> keys();  
    abstract void print();  
}  
class MyMapImpl<Key, Value> implements MyMap<Key,Value>  
{  
    ArrayList<Key> al1;  
    ArrayList<Value> al2;  
    MyMapImpl(ArrayList<Key> al1,ArrayList<Value> al2)  
    {  
        this.al1=al1;  
        this.al2=al2;  
    }  
    public void add(Key k,Value v)  
    {
```

```

        System.out.println("Key : "+k+" Value : "+v+" are added to the map");
        al1.add(k);
        al2.add(v);
    }
    public Value remove(Key k)
    {
        Value v=null;
        for(int i=0;i<al1.size();i++)
        {
            if(al1.get(i)==k)
            {
                v=al2.get(i);
                System.out.println("key : "+al1.get(i)+" Value : "+al2.get(i)+" are
removed from the map");
                al1.remove(i);
                al2.remove(i);
            }
        }
        return v;
    }
    public int size()
    {
        return al1.size();
    }
    public boolean isEmpty()
    {
        if(al1.size()==0)
        {
            return true;
        }
        else
        {
            return false;
        }
    }
    public ArrayList<Key> keys()
    {
        return al1;
    }
    public void print()
    {
        for(int i=0;i<al1.size();i++)
        {
            System.out.print("Key : "+al1.get(i));
        }
    }
}

```

```

        System.out.println(" Value : "+al2.get(i));
    }
}
public class MyMapTest
{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        Map<String, Integer> m1=new HashMap<String, Integer>();
        Map<Integer, Double> m2=new HashMap<Integer, Double>();
        for(int i=0;i<5;i++)
        {
            System.out.println("Enter the data for Map1 : ");
            String s=sc.nextLine();
            int n=sc.nextInt();
            m1.put(s, n);
        System.out.println("Enter the data for Map2 : ");
            int n1=sc.nextInt();
            double n2=sc.nextDouble();
            String s1=sc.nextLine();
            m2.put(n1, n2);
        }
        ArrayList<Integer> al1=new ArrayList<Integer>(m1.values());
        ArrayList<String> al2=new ArrayList<String>(m1.keySet());
        MyMapImpl<String, Integer> mmi=new MyMapImpl<String, Integer>(al2,al1);
        System.out.println("Enter the values to be added to the Map1");
        String s=sc.nextLine();
        int n3=sc.nextInt();
        mmi.add(s, n3);
        mmi.remove(s);
        System.out.println(mmi.size());
        System.out.println(mmi.keys());
        if(mmi.isEmpty())
        {
            System.out.println("Map doesnot have any elements");
        }
        else
        {
            System.out.println("Map has "+al1.size()+" elements");
        }
        for(int i=0;i<al1.size();i++)
        {
            mmi.remove(al2.get(i));
        }
    }
}
```

```

    }
    if(mmi.isEmpty())
    {
        System.out.println("Map doesnot have any elements");
    }
    else
    {
        System.out.println("Map has "+al1.size()+" elements");
    }
    ArrayList<Integer> al3=new ArrayList<Integer>(m2.keySet());
    ArrayList<Double> al4=new ArrayList<Double>(m2.values());
    MyMapImpl<Integer,Double> mmi2=new MyMapImpl<Integer,Double>(al3,al4);
    System.out.println("Enter the values to be added to the Map2");
    int n7=sc.nextInt();
    double n8=sc.nextDouble();
    mmi2.add(n7,n8);
    mmi2.remove(n7);
    System.out.println(mmi2.size());
    System.out.println(mmi2.keys());
    if(mmi2.isEmpty())
    {
        System.out.println("Map doesnot have any elements");
    }
    else
    {
        System.out.println("Map has "+al1.size()+" elements");
    }
    for(int i=0;i<al3.size();i++)
    {
        mmi2.remove(al3.get(i));
    }
    if(mmi2.isEmpty())
    {
        System.out.println("Map doesnot have any elements");
    }
    else
    {
        System.out.println("Map has "+al1.size()+" elements");
    }
}
}

```

Output :

```
C:\Users\user\Downloads\19BCI7039>javac MyMapTest.java

C:\Users\user\Downloads\19BCI7039>java MyMapTest
Enter the data for Map1 :
Harsha
125
Enter the data for Map2 :
23
45.4
Enter the data for Map1 :
Bavita
58
Enter the data for Map2 :
21
22.67
Enter the data for Map1 :
Vishnu
67
Enter the data for Map2 :
82
182.93
Enter the data for Map1 :
Sirisha
7019
Enter the data for Map2 :
8291
638.21
Enter the data for Map1 :
Gopi
7910
Enter the data for Map2 :
721
829.89
Enter the values to be added to the Map1
Venu
0829
Key : Venu Value : 829 are added to the map
key : Venu Value : 829 are removed from the map
5
[Gopi, Vishnu, Bavita, Sirisha, Harsha]
Map has 5 elements
key : Gopi Value : 7910 are removed from the map
key : Bavita Value : 58 are removed from the map
key : Harsha Value : 125 are removed from the map
Map has 2 elements
Enter the values to be added to the Map2
198
3821.21
Key : 198 Value : 3821.21 are added to the map
6
[721, 82, 8291, 21, 23, 198]
Map has 2 elements
key : 721 Value : 829.89 are removed from the map
key : 8291 Value : 638.21 are removed from the map
key : 23 Value : 45.4 are removed from the map
Map has 2 elements

C:\Users\user\Downloads\19BCI7039>
```

Oop Lab Assessment-7

Slot-L1

RegNo : 19BCI7039

P. Harsha vardhan

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.

Code:

```
public class QuartstoGallons {  
    public static void main(String args[])  
    {  
        final int quarts=4;  
        int noofquarts=39;  
        System.out.println("A job that needs "+noofquarts+" quarts requires  
"+(noofquarts/quarts)+" gallons plus "+(noofquarts%quarts)+" quarts");  
    }  
}
```

Output :

```
C:\Users\user\Downloads\19BCI7039>javac QuartstoGallons.java
```

```
C:\Users\user\Downloads\19BCI7039>java QuartstoGallons  
A job that needs 39 quarts requires 9 gallons plus 3 quarts
```

1.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.

Code :

```
import java.util.*;  
public class QuartstoGallonsInteractive  
{  
    public static void main(String args[])  
    {
```

```

Scanner sc=new Scanner(System.in);
final int quarts=18;
System.out.println("Enter the number of quarts : ");
int noofquarts=sc.nextInt();
System.out.println("A job that needs "+noofquarts+" quarts requires "+(noofquarts/quarts)+" gallons plus "+(noofquarts%quarts)+" quarts");
}
}

```

Output :

```

C:\Users\user\Downloads\19BCI7039>javac QuartstoGallonsInteractive.java

C:\Users\user\Downloads\19BCI7039>java QuartstoGallonsInteractive
Enter the number of quarts :
45
A job that needs 45 quarts requires 2 gallons plus 9 quarts

```

1.c

Now, add exception-handling capabilities to this program and continuously reprompt the user while any nonnumeric value is entered. Save the file as QuartstoGallonsInteractive.java

Code :

```

import java.util.Scanner;
class QuartstoGallonswithExceptionHandling
{
    public static void main(String[] args)
    {
        Scanner input = new Scanner(System.in);
        final int quarts = 4;
        int noofquarts;
        int i=0;
        while(i==0)
        {
            System.out.println("Enter number of quarts");
            try
            {
                noofquarts = Integer.parseInt(input.nextLine());
                int gallons = noofquarts/quarts;
                System.out.println("A job that needs "+noofquarts+" quarts requires "+gallons+" gallons
plus "+(noofquarts%quarts)+" quarts");
                i=1;
            }
            catch(Exception e)
            {
                System.out.println("Exception: " + " NumberFormatException");
            }
        }
    }
}

```

```
    }
}
}
```

Output :

```
C:\Users\user\Downloads\19BCI7039>javac QuartstoGallonswithExceptionHandling.java
C:\Users\user\Downloads\19BCI7039>java QuartstoGallonswithExceptionHandling
Enter number of quarts
he
Exception: NumberFormatException
Enter number of quarts
eunbd
Exception: NumberFormatException
Enter number of quarts
45
A job that needs 45 quarts requires 11 gallons plus 1 quarts
```

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.

Code :

```
import java.util.*;
public class DistanceFromAverage
{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        double a[]=new double[15];
        double s=0,count=0;
        for(int i=0;i<=15;i++)
        {
            double k=sc.nextDouble();
            if(k==99999)
            {
                break;
            }
            else
            {
                a[i]=k;
                count=count+1;
                s=s+a[i];
            }
        }
        try
        {
```

```

if(s==0)
{
    throw new NotenteredanyException();
}
else
{
    double avg=s/count;
    for(int i=0;i<15;i++)
    {
        if(a[i]!=0)
        {
            System.out.println("Number : "+a[i]+" Distance from Average : "+(avg-a[i]));
        }
        else
        {
            System.out.println("Furthur values are not given by the user");
            break;
        }
    }
}
catch(NotenteredanyException e)
{
    System.out.println(e);
}
}
class NotenteredanyException extends Exception
{
    public String toString()
    {
        return "Error : Please enter atleast one element in the array other than 99999";
    }
}

```

Output :

```
C:\Users\user\Downloads\19BCI7039>javac DistanceFromAverage.java
C:\Users\user\Downloads\19BCI7039>java DistanceFromAverage
99999
Error : Please enter atleast one element in the array other than 99999
C:\Users\user\Downloads\19BCI7039>java DistanceFromAverage
1
2
3
4
5
1
43
13
38
99999
Number : 1.0 Distance from Average : 11.22222222222221
Number : 2.0 Distance from Average : 10.22222222222221
Number : 3.0 Distance from Average : 9.22222222222221
Number : 4.0 Distance from Average : 8.22222222222221
Number : 5.0 Distance from Average : 7.22222222222221
Number : 1.0 Distance from Average : 11.22222222222221
Number : 43.0 Distance from Average : -30.77777777777778
Number : 13.0 Distance from Average : -0.7777777777777786
Number : 38.0 Distance from Average : -25.77777777777778
Furthur values are not given by the user
```

2.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`.

Code :

```
import java.util.*;
public class DistanceFromAveragewithExceptionHandling
{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter the size of the array");
        int n;
        double a[];
        try
        {
            n=Integer.parseInt(sc.nextLine());
```

```

try
{
    a=new double[n];
}
catch(NegativeArraySizeException w)
{
    a=new double[5];
    n=5;
    System.out.println("The size of the array is reset to "+n);
}
double s=0,count=0;
for(int i=0;i<=n;i++)
{
    double k=sc.nextDouble();
    if(k==99999)
    {
        break;
    }
    else
    {
        a[i]=k;
        count=count+1;
        s=s+a[i];
    }
}
try
{
    if(s==0)
    {
        throw new NotenteredanyException();
    }
    else
    {
        double avg=s/count;
        for(int i=0;i<15;i++)
        {
            if(a[i]!=0)
            {
                System.out.println("Number : "+a[i]+" Distance from Average : "+(avg-a[i]));
            }
            else
            {
                System.out.println("Furthur values are not given by the user");
                break;
            }
        }
    }
}

```

```

        }
    }
}
catch(NotenteredanyException e)
{
    System.out.println(e);
}
}
catch(NumberFormatException n1)
{
    System.out.println("Input should be an integer");
}
}
}
class NotenteredanyException extends Exception
{
    public String toString()
    {
        return "Error : Please enter atleast one element in the array other than 99999";
    }
}

```

Output :

```

C:\Users\user\Downloads\19BCI7039>javac DistanceFromAveragewithExceptionHandling.java
C:\Users\user\Downloads\19BCI7039>java DistanceFromAveragewithExceptionHandling
Enter the size of the array
harsha
Input should be an integer

C:\Users\user\Downloads\19BCI7039>java DistanceFromAveragewithExceptionHandling
Enter the size of the array
-5
The size of the array is reset to 5
1
2
3
4
5
99999
Number : 1.0 Distance from Average : 2.0
Number : 2.0 Distance from Average : 1.0
Number : 3.0 Distance from Average : 0.0
Number : 4.0 Distance from Average : -1.0
Number : 5.0 Distance from Average : -2.0

C:\Users\user\Downloads\19BCI7039>java DistanceFromAveragewithExceptionHandling
Enter the size of the array
4
99999
Error : Please enter atleast one element in the array other than 99999
C:\Users\user\Downloads\19BCI7039>

```

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.

Code:

```
import java.util.*;
public class ThrowCourseException
{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        Course c[]=new Course[6];
        for(int i=0;i<6;i++)
        {
            System.out.println("Enter course deatils : ");
            String cd=sc.nextLine();
            int cn=sc.nextInt();
            float nc=sc.nextFloat();
            String s=sc.nextLine();
            c[i]=new Course(cd,cn,nc);
        }
    }
}
class CourseException extends Exception
{
    String cd;
    int cn;
    float nc;
    CourseException(String courseddept,int courseno,float noofcredits)
    {
        cd=courseddept;
        cn=courseno;
        nc=noofcredits;
    }
    public String toString()
    {
```

```
        return "Object cannot be created with the given inputs";
    }
}
class Course
{
    String cd;
    int cn;
    float nc;
    Course(String courseddept,int courseno,float noofcredits)
    {
        cd=courseddept;
        cn=courseno;
        nc=noofcredits;
        try
        {
            if((cd.length()!=3)|| (cn<100||cn>499)|| (nc<0.5||nc>6))
            {
                throw new CourseException(cd,cn,nc);
            }
            else
            {
                System.out.println("Object created successfully");
            }
        }
        catch(CourseException e)
        {
            System.out.println(e);
        }
    }
}
```

Output :

```
C:\Users\user\Downloads\19BCI7039>javac ThrowCourseException.java
C:\Users\user\Downloads\19BCI7039>java ThrowCourseException
Enter course deatils :
cse
432
2
Object created succesfully
Enter course deatils :
bf
80
1
Object cannot be create with the given inputs
Enter course deatils :
ece
709
4
Object cannot be create with the given inputs
Enter course deatils :
cis
488
9
Object cannot be create with the given inputs
Enter course deatils :
mec
489
0.2
Object cannot be create with the given inputs
Enter course deatils :
mac
347
2
Object created succesfully
C:\Users\user\Downloads\19BCI7039>
```

3.b

Modify the CourseException class to extend RuntimeException class and identify the differences.

Solution :

RuntimeException are unchecked while Exception are checked (calling code must handle them). The custom exception should extends RuntimeException if you want to make it unchecked else extend it with Exception.

If you want to write a checked exception that is automatically enforced by the Handle or Declare Rule, you need to extend the Exception class.

If you want to write a runtime exception, you need to extend the RuntimeException class

Oop Lab Assessment-8

CSE2005

RegNo : 19BCI7039

Slot-L1

P. Harsha vardhan

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; $5 \leq N \leq 8$. Implement this application in two versions, using both Thread class & Runnable interface.

Code by using Runnable interface:

```
import java.util.*;
class AdditionThread implements Runnable
{
    int num;
    Addition q;
    Thread t;
    int A1[][];
    int g;
    AdditionThread(int n, Addition q,int[][] A,int g)
    {
        num = n;
        this.q = q;
        A1=A;
        this.g=g;
        t = new Thread(this, "thread");
    }
    public synchronized void run()
    {
```

```

        if(num!=g)
        {
            q.AC(num,A1,g);
        }
        else
        {
            q.AC();
        }
    }
}

class Addition
{
    static int b[]={};
    static int i=0;
    static int sum=0;
    public synchronized void AC(int num,int[][] A,int n)
    {
        for (int i=0; i<n; i++)
        {
            b[i]=b[i]+A[num][i];
        }
    }
    public synchronized void AC()
    {
        for(int i=0;i<b.length;i++)
        {
            sum=sum+b[i];
        }
    }
}
public class Main
{
    public static void main(String[] args)
    {
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();
        try
        {
            if(n<5 || n>7)
            {
                throw new boundingException();
            }
            Random rand = new Random();
            int A[][]=new int[n][n];

```

```

for (int i =0; i < n; i++)
{
    for (int j = 0; j< n; j++)
    {
        A[i][j]=rand.nextInt(100);
    }
}
Addition q = new Addition();
for(int i=0;i<n;i++)
{
    AdditionThread ob=new AdditionThread(i,q,A,n);
    ob.t.start();
}
AdditionThread ob1=new AdditionThread(n,q,A,n);
ob1.t.start();
while(q.sum==0)
{
    try {
        Thread.currentThread().sleep(1000);
    } catch (InterruptedException e) {
        Thread.currentThread().interrupt();
    }
}
System.out.println(q.sum);
}
catch(boundingException e)
{
    System.out.println(e);
}
}
}
class boundingException extends Exception
{
    public String toString()
    {
        return "Array cannot be taken with the given size";
    }
}

```

Code by extending Thread class :

```

import java.util.*;
class AdditionThread extends Thread
{
    int num;

```

```

Addition q;
Thread t;
int A1[][];
int g;
AdditionThread(int n, Addition q,int[][] A,int g)
{
    num = n;
    this.q = q;
    A1=A;
    this.g=g;
    t = new Thread(this, "thread");
}
public synchronized void run()
{
    if(num!=g)
    {
        q.AC(num,A1,g);
    }
    else
    {
        q.AC();
    }
}
class Addition
{
    static int b[]={new int [10];
    static int i=0;
    static int sum=0;
    public synchronized void AC(int num,int[][] A,int n)
    {
        for (int i=0; i<n; i++)
        {
            b[i]=b[i]+A[num][i];
        }
    }
    public synchronized void AC()
    {
        for(int i=0;i<b.length;i++)
        {
            sum=sum+b[i];
        }
    }
}

```

```
public class Main
{
    public static void main(String[] args)
    {
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();
        try
        {
            if(n<5 || n>7)
            {
                throw new boundingException();
            }
            Random rand = new Random();
            int A[][]=new int[n][n];
            for (int i =0; i < n; i++)
            {
                for (int j = 0; j< n; j++)
                {
                    A[i][j]=rand.nextInt(100);
                }
            }
            Addition q = new Addition();
            for(int i=0;i<n;i++)
            {
                AdditionThread ob=new AdditionThread(i,q,A,n);
                ob.t.start();
            }
            AdditionThread ob1=new AdditionThread(n,q,A,n);
            ob1.t.start();
            while(q.sum==0)
            {
                try {
                    Thread.currentThread().sleep(1000);
                } catch (InterruptedException e) {
                    Thread.currentThread().interrupt();
                }
            }
            System.out.println(q.sum);
        }
        catch(boundingException e)
        {
            System.out.println(e);
        }
    }
}
```

```

}
class boundingException extends Exception
{
    public String toString()
    {
        return "Array cannot be taken with the given size";
    }
}

```

Output :

```

C:\Users\user\Downloads\19BCI7039>javac Main.java

C:\Users\user\Downloads\19BCI7039>java Main
10
Array cannot be taken with the given size

C:\Users\user\Downloads\19BCI7039>java Main
6
1628

```

2.a

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

- mEncryption method - substitute a character with another character 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. (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 $0 \leq k \leq 25$.

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

Code:

```

import java.util.*;
class CeaserCipher{
    public void mEncryption(String str, int key)
    {
        if(key>=0 && key<=25)
        {
            for(int i=0;i<str.length();i++)
            {

```

```
if(Character.isUpperCase(str.charAt(i)))
{
    char ch1 = (char)((((int)(str.charAt(i)-65))+key)%26 + 65);
    System.out.print(ch1);
}
else
{
    char ch1 = (char)((((int)(str.charAt(i)-97))+key)%26 +97);
    System.out.print(ch1);
}
}
else
{
    System.out.println("Sorry Invalid Key, incryption is not possible:");
    System.out.println("0<=key<=25");
}
}
public void mDecryption(String str, int key)
{
    if(key>0 && key<=25)
    {
        for(int i=0;i<str.length();i++)
        {
            if(Character.isUpperCase(str.charAt(i)))
            {
                char ch1 = (char)((((int)(str.charAt(i)-65))+(26-key))%26 + 65);
                System.out.print(ch1);
            }
            else
            {
                char ch1 = (char)((((int)(str.charAt(i)-97))+(26-key))%26 +97);
                System.out.print(ch1);
            }
        }
    }
    else
    {
        System.out.println("Sorry Invalid Key, incryption is not possible:");
        System.out.println("0<=key<=25");
    }
}
}
public class TestCeaserCipher{
```

```

public static void main(String args[]){
    Scanner in = new Scanner(System.in);
    System.out.println("enter text to encrypt:");
    String message1 = in.nextLine();
    System.out.println("enter key:");
    int key1 = in.nextInt();
    CeaserCipher cc = new CeaserCipher();
    System.out.print("Encrypted message is: ");
    cc.mEncryption(message1, key1);
    System.out.println();
    System.out.println("enter text to decrypt:");
    String message2 = in.next();
    System.out.println("enter key:");
    int key2 = in.nextInt();
    System.out.print("Decrypted message is: ");
    cc.mDecryption(message2, key2);
}
}

```

Output :

```

C:\Users\user\Downloads\19BCI7039>javac TestCeaserCipher.java

C:\Users\user\Downloads\19BCI7039>java TestCeaserCipher
enter text to encrypt:
Harsha
enter key:
3
Encrypted message is: Kduvkd
enter text to decrypt:
Kduvkd
enter key:
3
Decrypted message is: Harsha

```

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.

Code:

```

import java.util.*;
class CaesarCipher2 implements Runnable
{
    Decryption q;
    int num;
    String msg;
    Thread t;
    CaesarCipher2(String d, int n, Decryption q){

```

```

num = n;
msg = d;
this.q = q;
t = new Thread(this, "thread");
}
public synchronized void run(){
q.mDecryption(msg,num);
}
}
class Decryption
{
public synchronized void mDecryption(String str, int key)
{
String result="";
if(key>=0 && key<=25)
{
for(int i=0;i<str.length();i++)
{
if(Character.isUpperCase(str.charAt(i)))
{
char ch1 = (char)((((int)(str.charAt(i)-65))+(26-key))%26 + 65);
result+=""+(ch1);

}
else
{
char ch1 = (char)((((int)(str.charAt(i)-97))+(26-key))%26 + 97);
result+=""+(ch1);

}
}
System.out.println(result+" Thread " +key);
}
else
{
System.out.println("Sorry Invalid Key, incryption is not possible:");
System.out.println("0<=key<=25");
}
}
}

public class BruteForceCeaserCipher{
public static void main(String args[])
{
String message = "gdhrzfnncanx";
Decryption text = new Decryption();
CaesarCipher2 obj0 = new CaesarCipher2(message,0,text);
CaesarCipher2 obj1 = new CaesarCipher2(message,1,text);
CaesarCipher2 obj2 = new CaesarCipher2(message,2,text);
}
}

```

```
CaesarCipher2 obj3 = new CaesarCipher2(message,3,text);
CaesarCipher2 obj4 = new CaesarCipher2(message,4,text);
CaesarCipher2 obj5 = new CaesarCipher2(message,5,text);
CaesarCipher2 obj6 = new CaesarCipher2(message,6,text);
CaesarCipher2 obj7 = new CaesarCipher2(message,7,text);
CaesarCipher2 obj8 = new CaesarCipher2(message,8,text);
CaesarCipher2 obj9 = new CaesarCipher2(message,9,text);
CaesarCipher2 obj10 = new CaesarCipher2(message,10,text);
CaesarCipher2 obj11 = new CaesarCipher2(message,11,text);
CaesarCipher2 obj12 = new CaesarCipher2(message,12,text);
CaesarCipher2 obj13 = new CaesarCipher2(message,13,text);
CaesarCipher2 obj14 = new CaesarCipher2(message,14,text);
CaesarCipher2 obj15 = new CaesarCipher2(message,15,text);
CaesarCipher2 obj16 = new CaesarCipher2(message,16,text);
CaesarCipher2 obj17 = new CaesarCipher2(message,17,text);
CaesarCipher2 obj18 = new CaesarCipher2(message,18,text);
CaesarCipher2 obj19 = new CaesarCipher2(message,19,text);
CaesarCipher2 obj20 = new CaesarCipher2(message,20,text);
CaesarCipher2 obj21 = new CaesarCipher2(message,21,text);
CaesarCipher2 obj22 = new CaesarCipher2(message,22,text);
CaesarCipher2 obj23 = new CaesarCipher2(message,23,text);
CaesarCipher2 obj24 = new CaesarCipher2(message,24,text);
CaesarCipher2 obj25 = new CaesarCipher2(message,25,text);
obj0.t.start();
obj1.t.start();
obj2.t.start();
obj3.t.start();
obj4.t.start();
obj5.t.start();
obj6.t.start();
obj7.t.start();
obj8.t.start();
obj9.t.start();
obj10.t.start();
obj11.t.start();
obj12.t.start();
obj13.t.start();
obj14.t.start();
obj15.t.start();
obj16.t.start();
obj17.t.start();
obj18.t.start();
obj19.t.start();
obj20.t.start();
obj21.t.start();
obj22.t.start();
obj23.t.start();
obj24.t.start();
```

```
    obj25.t.start();
}
}
```

Output:

```
C:\Users\user\Downloads\19BCI7039>javac BruteForceCeaserCipher.java
C:\Users\user\Downloads\19BCI7039>java BruteForceCeaserCipher
gdhrzfnncanx Thread 0
jgkuciqqfdqa Thread 23
heisagoodboy Thread 25
ifjtbhppecpz Thread 24
mjnxflttigt Thread 20
kh1vdjrrgerb Thread 22
limweksshfsc Thread 21
nkoygmuujhue Thread 19
qnrbjpxxmkh Thread 16
olpzhnvvkivf Thread 18
pmqaiowwljwg Thread 17
rosckqyynlyi Thread 15
sptdlrzzomzj Thread 14
tquemsaapnak Thread 13
urvfnbbqobl Thread 12
vswgouccrpcm Thread 11
wtxhpvddsqdn Thread 10
xuyiqweetreο Thread 9
yvzjrxffusfp Thread 8
zwaksyggvtgq Thread 7
axbltzhhwuhr Thread 6
bycmuaiixvis Thread 5
czdnvbjjywjt Thread 4
daeowckzxku Thread 3
ebfpxdllaylv Thread 2
fcgqyemmbzmw Thread 1
```

Oop Lab Assessment-9

CSE2005

Slot : L1

RegNo : 19BCI7039

P. Harsha vardhan

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) producer has 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 produce elements.

Write a solution for N elements, where N is multiple of 5 or greater than 5 other than 0.

Code :

```
public class ThreadDemo {  
    public static void main(String args[])throws InterruptedException {  
        final PC pc = new PC();  
        Thread t1 = new Thread(new Runnable() {  
            public void run()  
            {  
                try {  
                    pc.produce();  
                }  
                catch (InterruptedException e) {  
                    e.printStackTrace();  
                }  
            }  
        });  
        Thread t2 = new Thread(new Runnable() {  
            public void run()
```

```

    {
        try {
            pc.consume();
        }
        catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
});

t1.start();
t2.start();
t1.join();
t2.join();
}

}

class PC {
    int a[]=new int[5];
    int value = 0;
    public void produce() throws InterruptedException
    {
        while (true) {
            synchronized (this)
            {
                if(value==a.length)
                {
                    System.out.println("Queue is full");
                }
                while (value==a.length)
                    wait();
                System.out.println("Producer produced-"+ value);
                a[value]=value+1;
                value++;
                notify();
                Thread.sleep(1000);
            }
        }
    }
    public void consume() throws InterruptedException
    {
        while (true) {
            synchronized (this)
            {
                if(value==0)
                {

```

```
        System.out.println("Queue is Empty");
    }
    while (value == 0)
        System.out.println("Queue is empty");
        wait();
    int val=a[0];
    for(int i=0;i<4;i++)
    {
        a[i]=a[i+1];
    }
    a[4]=0;
    value--;
    System.out.println("Consumer consumed-"+ val);
    notify();
    Thread.sleep(1000);
}
}
}
```

Output :

```
C:\Users\user\Downloads\19BCI7039>javac ThreadDemo.java
```

```
C:\Users\user\Downloads\19BCI7039>java ThreadDemo
Producer produced-0
Producer produced-1
Producer produced-2
Producer produced-3
Producer produced-4
Queue is full
Consumer consumed-1
Producer produced-4
Queue is full
Consumer consumed-2
Producer produced-4
Queue is full
Consumer consumed-3
Producer produced-4
Queue is full
Consumer consumed-4
Producer produced-4
Queue is full
Consumer consumed-5
Producer produced-4
Queue is full
```

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.
User1(User2):Hi
User2(User1):Hi

Code:

```
package labassign6;
import java.util.*;
class Q
{
    Scanner insendMessage=new Scanner(System.in);
    int n;
    String Message;
    boolean valueSet = false;
    synchronized int recvMessage()
    {
        while(!valueSet)
        try
        {
            wait();
        }
        catch(InterruptedException e)
        {
            System.out.println("InterruptedException caught");
        }
        System.out.println("User2 : ");
        Message=insendMessage.nextLine();
        valueSet = true;
        System.out.println("User2(User1):" + Message);
    }
}
```

```

valueSet=false;
notify();
return n;
}
synchronized void sendMessage(int n)
{
    while(valueSet) try
    {
        wait();
    }
    catch(InterruptedException e)
    {
        System.out.println("InterruptedException caught");
    }
    this.n = n;
    System.out.println("User1 : ");
    Message=insendMessage.nextLine();
    valueSet = true;
    System.out.println("User1(User2):" + Message);
    notify();
}
}
class Producer implements Runnable
{
    Q q;
    Thread t;
    Producer(Q q)
    {
        this.q=q;
        t=new Thread(this, "Producer");
    }
    public void run()
    {
        int i = 0;
        while (true)
        {
            if(i>5)
            {
                System.exit(0);
            }
            q.sendMessage(i++);
        }
    }
}

```

```
class Consumer implements Runnable
{
    Q q;
    Thread t;
    Consumer (Q q)
    {
        this.q = q;
        t=new Thread (this,"Consumer");
    }
    public void run ()
    {
        while(true)
        {
            q.recvMessage();
        }
    }
}
class Chat
{
    public static void main(String args[])
    {
        Q q =new Q();
        Producer p = new Producer (q);
        Consumer c = new Consumer (q);
        p.t.start();
        c.t.start();
        System.out.println("Press Control-C to stop. ");
    }
}
```

Output:

```
C:\Users\user\Downloads\19BCI7039>javac Chat.java
C:\Users\user\Downloads\19BCI7039>java Chat
Press Control-C to stop.
User1 :
Hi
User1(User2):Hi
User2 :
Hello
User2(User1):Hello
User1 :
R u planning to go any where to night
User1(User2):R u planning to go any where to night
User2 :
no plans.
User2(User1):no plans.
User1 :
can you join for a movie to night
User1(User2):can you join for a movie to night
User2 :
Yeah sure
User2(User1):Yeah sure
User1 :
Meet you tonight then
User1(User2):Meet you tonight then
User2 :
ok
User2(User1):ok
```

Oops Lab Assessment-10

CSE2005

Slot-L1

RegNo : 19BCI7039

P. Harsha vardhan

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.

Code:

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class JPhotoFrame extends JFrame implements ItemListener {

    final int BASE = 40;
    final int TWO = 75;
    final int PET = 95;
    final int LOCATION = 90;
    int subjectPrice;
    int locationPrice;
    JLabel description = new JLabel("Photo Session Price Calculator");
    JLabel subject = new JLabel("Choose portrait subject:");
}
```

```
JLabel sessionLocation = new JLabel("Choose session location:");

ButtonGroup subjectChoice = new ButtonGroup();

JCheckBox basePrice = new JCheckBox("1 person: $" + BASE, false);

JCheckBox twoSubjects = new JCheckBox("2 or more subjects: $" + TWO, false);

JCheckBox petPrice = new JCheckBox("Pet: $" + PET, false);

ButtonGroup locationChoice = new ButtonGroup();

JCheckBox inStudio = new JCheckBox("Studio: Free", false);

JCheckBox onLocation = new JCheckBox("On location: $" + LOCATION, false);

JLabel priceLabel = new JLabel("Total Price:");

JLabel totalPrice = new JLabel();

public JPhotoFrame()

{

    super("Photo Session Price Calculator");

    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    setLayout(new FlowLayout());

    add(description);

    add(subject);

    add(basePrice);

    add(twoSubjects);

    add(petPrice);

    add(sessionLocation);

    add(inStudio);

    add(onLocation);

    add(priceLabel);

    add(totalPrice);

    basePrice.addItemListener(this);

    twoSubjects.addItemListener(this);

    petPrice.addItemListener(this);

    onLocation.addItemListener(this);

    subjectChoice.add(basePrice);

    subjectChoice.add(twoSubjects);
```

```
    subjectChoice.add(petPrice);

    locationChoice.add(inStudio);

    locationChoice.add(onLocation);

}

public void itemStateChanged(ItemEvent a)

{

    Object choice = a.getSource();

    int chosen = a.getStateChange();

    if(choice == basePrice)

    {

        if(chosen == ItemEvent.SELECTED)

        {

            subjectPrice = BASE;

        }

        else

        {

            subjectPrice -= BASE;

        }

    }

    else if(choice == twoSubjects)

    {

        if(chosen == ItemEvent.SELECTED)

        {

            subjectPrice = TWO;

        }

        else

        {

            subjectPrice -= TWO;

        }

    }

    else if (choice == petPrice)
```

```
{  
    if(chosen == ItemEvent.SELECTED)  
    {  
        subjectPrice = PET;  
    }  
    else  
    {  
        subjectPrice -= PET;  
    }  
}  
else if (choice == onLocation)  
{  
    if(chosen == ItemEvent.SELECTED)  
    {  
        locationPrice = LOCATION;  
    }  
    else  
    {  
        locationPrice -= LOCATION;  
    }  
}  
int combinedPrice = subjectPrice + locationPrice;  
totalPrice.setText("$" + combinedPrice);  
}  
  
public static void main(String[] args)  
{  
    JPhotoFrame calc = new JPhotoFrame();  
    calc.setSize(200, 300);  
    calc.setVisible(true);  
}  
}
```

Output:

Command Prompt - java JPhotoFrame

Microsoft Windows [Version 10.0.19041.630]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\user>cd downloads

C:\Users\user\Downloads>cd 19BCI7039

C:\Users\user\Downloads\19BCI7039>javac JPhotoFrame.java

C:\Users\user\Downloads\19BCI7039>java JPhotoFrame

Photo Session Price Calculator

Choose portrait subject:

1 person: \$40

2 or more subjects: \$75

Pet: \$95

Choose session location:

Studio: Free

On location: \$90

Total Price: \$40



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.

Code:

```
import java.util.*;
import javax.swing.*;
import java.awt.event.*;

public class JPhotoFrame extends JFrame implements ActionListener

{

    final int PARK_SIDE1= 600;
    final int POOL_SIDE1 = 750;
    final int LAKE_SIDE1 = 825;
    final int ONE_BEDROOM1= 75;
    final int TWO_BEDROOMS1 = 150;
    final int THREE_BEDROOMS1 = 225;
    final int MEALS_1= 200;

    JLabel jlblLocations1 = new JLabel("LOCATION: ");
    JLabel jlblBedrooms1 = new JLabel("BEDROOMS: ");
    JLabel jlblMeals1 = new JLabel("INCLUDE MEALS: ");

    JRadioButton jrbtnPark1 = new JRadioButton("PARK SIDE");
    JRadioButton jrbtnPool1= new JRadioButton("POOL SIDE");
    JRadioButton jrbtnLake1= new JRadioButton("LAKE SIDE");
    JRadioButton jrbtnOne1 = new JRadioButton("1");
    JRadioButton jrbtnTwo1 = new JRadioButton("2");
    JRadioButton jrbtnThree1 = new JRadioButton("3");
    JRadioButton jrbtnYes1 = new JRadioButton("YES");
    JRadioButton jrbtnNo1 = new JRadioButton("NO");
```

```
ButtonGroup groupLocations1 = new ButtonGroup();
ButtonGroup groupBedrooms1 = new ButtonGroup();
ButtonGroup groupMeals1 = new ButtonGroup();
JButton jbtnCalculate1 = new JButton(" CALCULATE TOTAL RENT");
JButton jbtnReset1= new JButton("RESET ");
JLabel result1 = new JLabel("TOTAL RENT: $ 0.0");
JPanel panel=new JPanel();
JPanel jpnlLocations1 = new JPanel();
JPanel jpnlBedrooms1 = new JPanel();
JPanel jpnlMeals1 = new JPanel();
JPanel jpnlButton1 = new JPanel();
JPanel jpnlLabel1 = new JPanel();
JPanel pane1l = new JPanel();
private int locationRent1 = 0;
private int bedroomsRent1 = 0;
private int mealsCost1 = 0;
public JPhotoFrame()
{
    super("LAMBERT'S VACATION RENTALS ");
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    groupLocations1.add(jrbtnPark1);
    groupLocations1.add(jrbtnPool1);
    groupLocations1.add(jrbtnLake1);
    groupBedrooms1.add(jrbtnOne1);
    groupBedrooms1.add(jrbtnTwo1);
    groupBedrooms1.add(jrbtnThree1);
    groupMeals1.add(jrbtnYes1);
    groupMeals1.add(jrbtnNo1);
    jpnlLocations1.add(lblLocations1);
```

```
jpnLocations1.add(jrbtnPark1);
jpnLocations1.add(jrbtnPool1);
jpnLocations1.add(jrbtnLake1);
jpnBedrooms1.add(jlblBedrooms1);
jpnBedrooms1.add(jrbtnOne1);
jpnBedrooms1.add(jrbtnTwo1);
jpnBedrooms1.add(jrbtnThree1);
jpnMeals1.add(jlblMeals1);
jpnMeals1.add(jrbtnYes1);
jpnMeals1.add(jrbtnNo1);
jpnButton1.add(jbtnCalculate1);
jpnButton1.add(jbtnReset1);
jpnLabel1.add(result1);
panel.add(jpnLocations1);
panel.add(jpnBedrooms1);
panel.add(jpnMeals1);
panel.add(jpnButton1);
panel.add(jpnLabel1);
add(panel);
jrbtnPark1.addItemListener(new LocationsListener());
jrbtnPool1.addItemListener(new LocationsListener());
jrbtnLake1.addItemListener(new LocationsListener());
jrbtnOne1.addItemListener(new BedroomsListener());
jrbtnTwo1.addItemListener(new BedroomsListener());
jrbtnThree1.addItemListener(new
    BedroomsListener());
jrbtnYes1.addItemListener(new MealsListener());
jrbtnNo1.addItemListener(new MealsListener());
jbtnCalculate1.addActionListener(this);
```

```

        jbtnReset1.addActionListener(this);
    }

    public void actionPerformed(ActionEvent e)
    {
        Object source = e.getSource();
        if(source == jbtnCalculate1)
        {
            double totalRent1 = locationRent1 + bedroomsRent1 + mealsCost1;
            result1.setText("TOTALRENT: $ " + totalRent1);
        }
        else if(source == jbtnReset1)
        {
            groupLocations1.clearSelection();
            groupBedrooms1.clearSelection();
            groupMeals1.clearSelection();
            result1.setText("Total Rent: $ 0.0");
            locationRent1 = 0;
            bedroomsRent1 = 0;
            mealsCost1 = 0;
        }
    }

    private class LocationsListener implements ItemListener
    {
        public void itemStateChanged(ItemEvent e)
        {
            Object source = e.getItem();
            if(source == jrbtnPark1)
                locationRent1 = PARK_SIDE1;
            else if(source == jrbtnPool1)

```

```

        locationRent1 = POOL_SIDE1;
    else if(source == jrbtnLake1)
        locationRent1 = LAKE_SIDE1;
    else
        locationRent1 = 0;
    }

}

private class BedroomsListener implements ItemListener
{
    public void itemStateChanged(ItemEvent e)
    {
        Object source = e.getItem();
        if(source == jrbtnOne1)
            bedroomsRent1 = ONE_BEDROOM1;
        else if(source == jrbtnTwo1)
            bedroomsRent1 = TWO_BEDROOMS1;
        else if(source == jrbtnThree1)
            bedroomsRent1 = THREE_BEDROOMS1;
        else
            bedroomsRent1 = 0;
    }
}

private class MealsListener implements ItemListener
{
    public void itemStateChanged(ItemEvent e)
    {
        Object source = e.getItem();
        if(source == jrbtnYes1)
            mealsCost1 = MEALS_1;
    }
}

```

```

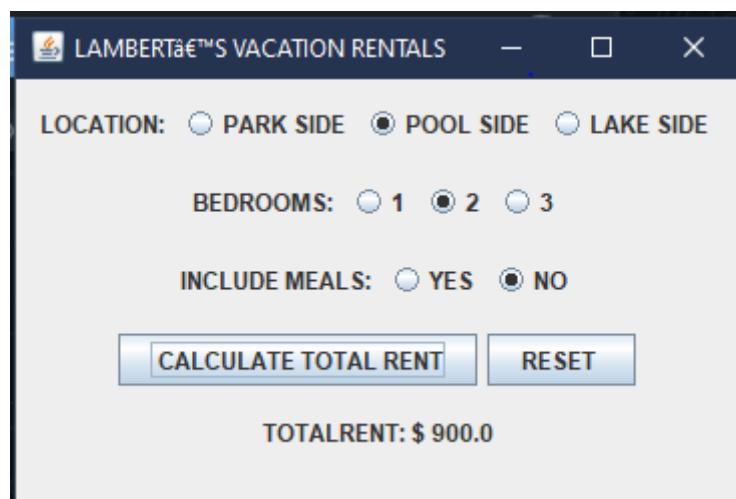
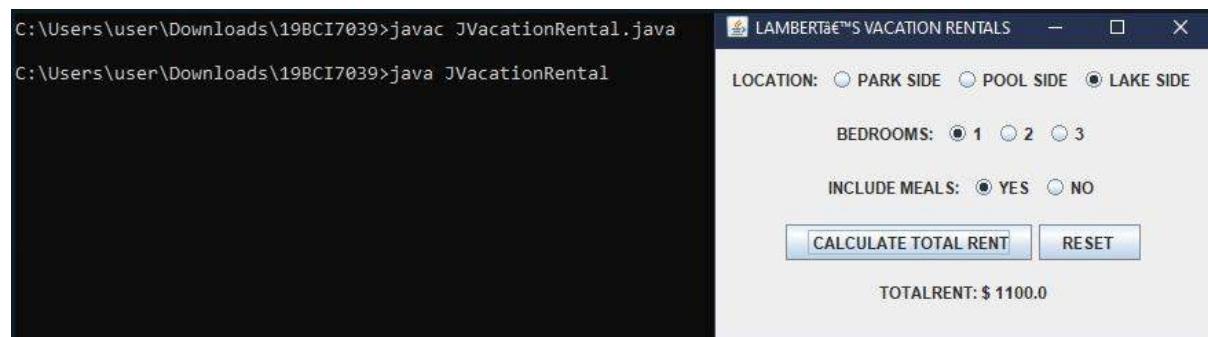
        else if(source == jrbtnNo1)
            mealsCost1 = 0;
        else
            mealsCost1= 0;
    }

}

public static void main(String[] args)
{
    JPhotoFrame frame = new JPhotoFrame();
    frame.setSize(350,250);
    frame.setVisible(true);
}

```

OUTPUT:



Oop Lab Assessment-11

CSE2005

SLOT-L1

Reg.No : 19BCI7039

P. Harsha vardhan

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 insurance 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.

Code:

```
import javafx.application.Application;  
import javafx.stage.Stage;  
import javafx.beans.value.ObservableValue;  
import javafx.scene.layout.StackPane;  
import javafx.scene.control.*;  
import javafx.scene.Scene;  
import javafx.scene.layout.*;  
import javafx.event.Event;  
import javafx.event.EventHandler;  
import javafx.event.ActionEvent;  
import javafx.scene.text.Text;  
import javafx.beans.value.*;  
public class FXInsurance extends Application  
{
```

```
ToggleGroup grp;
RadioButton hmo,ppo;
CheckBox di,vi;
Text t1,t2;
Label l1,l2,l3,l4;
Button b1;
TextField t;
final int hmocost=200;
final int ppocost=600;
final int dentalcoveragecost=75;
final int visioncareaddscost=20;
public static void main(String args[])
{
    launch(args);
}
public void start(Stage primaryStage) throws Exception
{
    l1=new Label("Select any one of the two Organizations");
    grp = new ToggleGroup();
    hmo = new RadioButton("HMO(Health Maintenance organization)");
    ppo = new RadioButton("PPO(Preferred Provider organization)");
    hmo.setToggleGroup(grp);
    ppo.setToggleGroup(grp);
    l2=new Label("Select the insurance");
    di=new CheckBox("Dental Insurance");
    vi=new CheckBox("Vision Insurance");
    l3=new Label();
    t1 = new Text();
    t2 = new Text();
```

```

l4=new Label();
b1=new Button("Calculate total cost : ");
di.selectedProperty().addListener(
(ObservableValue<? extends Boolean> ov, Boolean old_val, Boolean new_val) -> {
    t1 .setText(di.getText()+" : "+dentalcoveragecost);
});
vi.selectedProperty().addListener(
(ObservableValue<? extends Boolean> ov, Boolean old_val, Boolean new_val) -> {
    t2 .setText(vi.getText()+" : "+visioncareaddscost);
});
grp.selectedToggleProperty().addListener(new ChangeListener<Toggle>()
{
    public void changed(ObservableValue<? extends Toggle> ob,Toggle o, Toggle n)
    {
        RadioButton rb = (RadioButton)grp.getSelectedToggle();
        if (rb != null) {
            String s = rb.getText();
            if(s.equals("HMO(Health Maintenance organization)"))
            {
                l3.setText(s+" : "+hmocost);
            }
            else{
                l3.setText(s+" : "+ppocost);
            }
        }
    }
});
t=new TextField();
b1.setOnAction(new EventHandler <ActionEvent>()

```

```

{
    public void handle(ActionEvent e) {
        int price=0;
        if(hmo.isSelected())
        {
            price+= hmocost;
        }
        if(ppo.isSelected())
        {
            price+=ppocost;
        }
        if(di.isSelected())
        {
            price+=dentalcoveragecost;
        }
        if(vi.isSelected())
        {
            price+=visioncareaddscost;
        }
        String price1=Integer.toString(price);
        t.setText(price1);
    }
});

VBox root=new VBox();
root.getChildren().addAll(l1,hmo,ppo,l2,di,vi,t1,t2,l3,b1,t);
Scene scene=new Scene(root,350,250);
primaryStage.setScene(scene);
primaryStage.setTitle("Insurance form");
primaryStage.show();
}

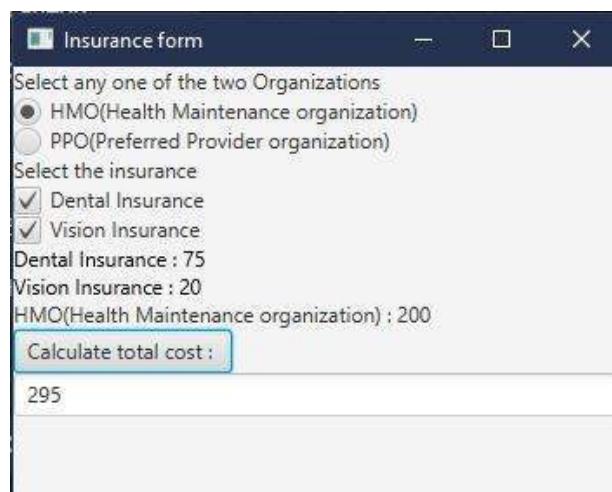
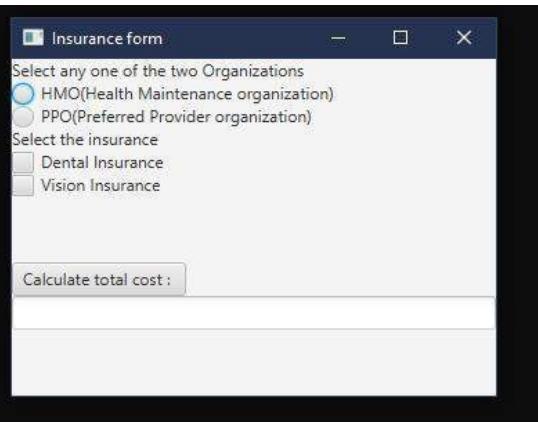
```

```
}
```

```
}
```

Output:

```
C:\Users\user\Downloads\19BCI7039>javac FXInsurance.java  
C:\Users\user\Downloads\19BCI7039>java FXInsurance
```



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.

Code:

```
import javafx.application.Application;  
  
import javafx.stage.Stage;  
  
import javafx.beans.value.ObservableValue;  
  
import javafx.collections.*;  
  
import javafx.scene.layout.StackPane;
```

```
import javafx.scene.control.*;
import javafx.scene.Scene;
import javafx.scene.layout.*;
import javafx.beans.value.*;
public class FXSandwich extends Application
{
    Label l5,l6,l7,l8;
    public void start(Stage primaryStage) throws Exception
    {
        ObservableList<String> prices =
FXCollections.observableArrayList("Chicken","Cheese","Bacon");
        ListView<String> lv = new ListView<String>(prices);
        ObservableList<String> names =
FXCollections.observableArrayList("Milkbread","Brownbread","Fruitbread");
        ListView<String> lv1 = new ListView<String>(names);
        Label l1=new Label("ingredients");
        lv.setItems(prices);
        lv.setPrefHeight(80);
        Label l2=new Label("Type of bread");
        lv1.setItems(names);
        lv1.setPrefHeight(80);
        l5=new Label();
        l6=new Label();
        l7=new Label();
        lv.getSelectionModel().selectedItemProperty().addListener(
            new ChangeListener<String>() {
                public void changed(ObservableValue<? extends String> ov,
                    String old_val, String new_val)
                {
                    if(new_val.equals("Chicken"))

```

```

{
    l5.setText("Chicken : 100");
    l7.setText("100");
}

else if(new_val.equals("Cheese"))
{
    l5.setText("Cheese : 50");
    l7.setText("50");
}

else
{
    l5.setText("Bacon : 75");
    l7.setText("75");
}
}

});

IV1.getSelectionModel().selectedItemProperty().addListener(
    new ChangeListener<String>() {
        public void changed(ObservableValue<? extends String> ov,
                            String old_val, String new_val)
        {
            if(new_val.equals("Milkbread"))
            {
                l6.setText("Milkbread : 50");
                if(l7.getText().equals(""))
                {
                    l7.setText("50");
                }else{
                    int k=Integer.parseInt(l7.getText())+50;
                }
            }
        }
    }
);

```

```

    l7.setText(Integer.toString(k));
}

else if(new_val.equals("Brownbread"))

{
    l6.setText("Brownbread : 40");
    if(l7.getText().equals(""))
    {
        l7.setText("40");
    }else{
        int k=Integer.parseInt(l7.getText())+40;
        l7.setText(Integer.toString(k));
    }
}
else
{
    l6.setText("Fruitbread : 60");
    if(l7.getText().equals(""))
    {
        l7.setText("60");
    }else{
        int k=Integer.parseInt(l7.getText())+60;
        l7.setText(Integer.toString(k));
    }
}
});

l8=new Label("Total cost : ");

VBox root=new VBox();
root.getChildren().addAll(l1,lV,l2,lV1,l5,l6,l8,l7);

Scene scene=new Scene(root);
primaryStage.setScene(scene);

```

```

        primaryStage.setTitle("Insurance form");

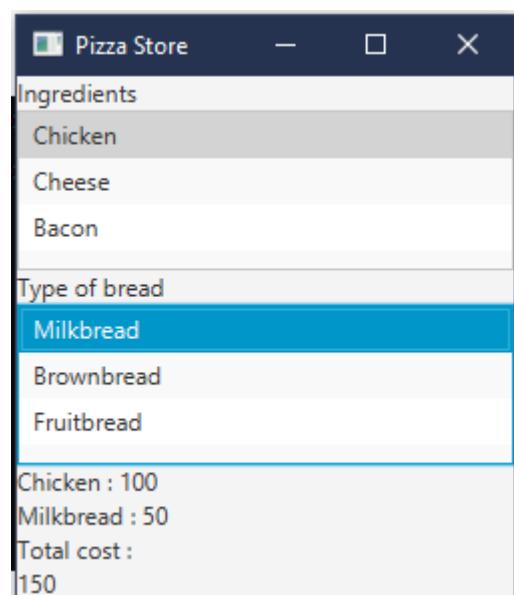
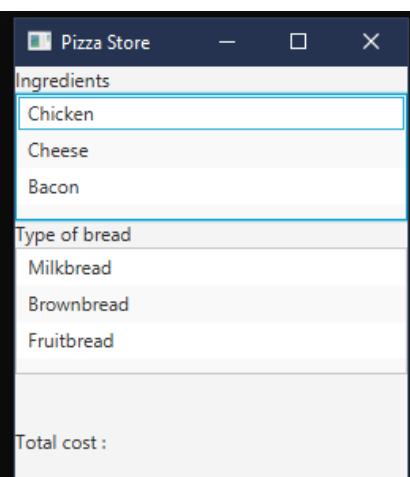
        primaryStage.show();
    }

    public static void main(String args[])
    {
        launch(args);
    }
}

```

Output:

C:\Users\user\Downloads\19BCI7039>javac FXSandwich.java
C:\Users\user\Downloads\19BCI7039>java FXSandwich



CSE-2005
Oops Lab Assessment-12
Slot-L1

RegNo : 19BCI7039

P. Harshavardhan

1.

Develop a Menu Based GUI using Swings:

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**

CODE USING SWING:

```
import static java.awt.event.InputEvent.CTRL_DOWN_MASK;  
import java.awt.*;  
import javax.swing.*;
```

```
import java.awt.event.*;
public class MenuWithSwings extends JFrame implements ActionListener{
    JMenuBar mb;
    JMenu ct,fn,cl,nam,typ,siz;
    static JTextField txt;
    JMenuItem
ct1,ct2,ct3,ct4,ct5,ct6,ct7,nam1,nam2,nam3,nam4,typ1,typ2,siz1,siz2,siz3,siz4,cl1,cl2,cl3,cl4,cl5,cl6,c
l7;
    JFrame f;
    JPopupMenu pm;
    JButton b,b1,b2;
    Font f1,f2,f3;
    MenuWithSwings()
    {
        f = new JFrame("Menu demo");
        mb = new JMenuBar();
        f.setJMenuBar(mb);
        ct=new JMenu("Country");
        fn=new JMenu("Font");
        cl=new JMenu("Color");
        nam=new JMenu("Name");
        typ=new JMenu("Type");
        siz=new JMenu("Size");
        ct1=new JMenuItem("India");
        ct2=new JMenuItem("China");
        ct3=new JMenuItem("Australia");
        ct4=new JMenuItem("Newzealand");
        ct5=new JMenuItem("England");
        ct6=new JMenuItem("Afghanistan");
        ct7=new JMenuItem("WestIndies");
        nam1=new JMenuItem("TimesRoman");
        nam2=new JMenuItem("Calibri");
        nam3=new JMenuItem("Arial Black");
    }
}
```

```
nam4=new JMenuItem("Impact");
typ1=new JMenuItem("Bold");
typ2=new JMenuItem("Italic");
siz1=new JMenuItem("12");
siz2=new JMenuItem("14");
siz3=new JMenuItem("16");
siz4=new JMenuItem("18");
cl1=new JMenuItem("GREEN");
cl2=new JMenuItem("RED");
cl3=new JMenuItem("BLUE");
cl4=new JMenuItem("PINK");
cl5=new JMenuItem("WHITE");
cl6=new JMenuItem("GRAY");
cl7=new JMenuItem("ORANGE");
ct1.setAccelerator(KeyStroke.getKeyStroke('Q', CTRL_DOWN_MASK));
ct2.setAccelerator(KeyStroke.getKeyStroke('W', CTRL_DOWN_MASK));
ct3.setAccelerator(KeyStroke.getKeyStroke('E', CTRL_DOWN_MASK));
ct4.setAccelerator(KeyStroke.getKeyStroke('R', CTRL_DOWN_MASK));
ct5.setAccelerator(KeyStroke.getKeyStroke('T', CTRL_DOWN_MASK));
ct6.setAccelerator(KeyStroke.getKeyStroke('Y', CTRL_DOWN_MASK));
ct7.setAccelerator(KeyStroke.getKeyStroke('U', CTRL_DOWN_MASK));
nam1.setAccelerator(KeyStroke.getKeyStroke('I', CTRL_DOWN_MASK));
nam2.setAccelerator(KeyStroke.getKeyStroke('O', CTRL_DOWN_MASK));
nam3.setAccelerator(KeyStroke.getKeyStroke('P', CTRL_DOWN_MASK));
nam4.setAccelerator(KeyStroke.getKeyStroke('A', CTRL_DOWN_MASK));
typ1.setAccelerator(KeyStroke.getKeyStroke('S', CTRL_DOWN_MASK));
typ2.setAccelerator(KeyStroke.getKeyStroke('D', CTRL_DOWN_MASK));
siz1.setAccelerator(KeyStroke.getKeyStroke('F', CTRL_DOWN_MASK));
siz2.setAccelerator(KeyStroke.getKeyStroke('G', CTRL_DOWN_MASK));
siz3.setAccelerator(KeyStroke.getKeyStroke('H', CTRL_DOWN_MASK));
siz4.setAccelerator(KeyStroke.getKeyStroke('J', CTRL_DOWN_MASK));
cl1.setAccelerator(KeyStroke.getKeyStroke('K', CTRL_DOWN_MASK));
```

```
cl2.setAccelerator(KeyStroke.getKeyStroke('L', CTRL_DOWN_MASK));
cl3.setAccelerator(KeyStroke.getKeyStroke('Z', CTRL_DOWN_MASK));
cl4.setAccelerator(KeyStroke.getKeyStroke('X', CTRL_DOWN_MASK));
cl5.setAccelerator(KeyStroke.getKeyStroke('C', CTRL_DOWN_MASK));
cl6.setAccelerator(KeyStroke.getKeyStroke('V', CTRL_DOWN_MASK));
cl7.setAccelerator(KeyStroke.getKeyStroke('B', CTRL_DOWN_MASK));

txt=new JTextField();
txt.setBounds(10,10,180,35);
b=new JButton("Popup");
b.addActionListener(this);
f.add(txt);

ct1.addActionListener(this);
ct2.addActionListener(this);
ct3.addActionListener(this);
ct4.addActionListener(this);
ct5.addActionListener(this);
ct6.addActionListener(this);
ct7.addActionListener(this);

nam1.addActionListener(this);
nam2.addActionListener(this);
nam3.addActionListener(this);
nam4.addActionListener(this);

typ1.addActionListener(this);
typ2.addActionListener(this);

siz1.addActionListener(this);
siz2.addActionListener(this);
siz3.addActionListener(this);
siz4.addActionListener(this);

cl1.addActionListener(this);
cl2.addActionListener(this);
cl3.addActionListener(this);
cl4.addActionListener(this);
```

```
cl5.addActionListener(this);
cl6.addActionListener(this);
cl7.addActionListener(this);
pm=new JPopupMenu();
b1=new JButton("Clear");
b2=new JButton("Exit");
pm.add(b1);
pm.add(b2);
b1.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e)
{
    txt.setText("");
}
});
b2.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e)
{
    System.exit(0);
}
});
JPanel p=new JPanel();
p.add(b);
f.add(p);
f.show();
ct.add(ct1);
ct.add(ct2);
ct.add(ct3);
ct.add(ct4);
ct.add(ct5);
ct.add(ct6);
ct.add(ct7);
nam.add(nam1);
```

```
nam.add(nam2);
nam.add(nam3);
nam.add(nam4);
typ.add(typ1);
typ.add(typ2);
siz.add(siz1);
siz.add(siz2);
siz.add(siz3);
siz.add(siz4);
fn.add(nam);
fn.add(typ);
fn.add(siz);
cl.add(cl1);
cl.add(cl2);
cl.add(cl3);
cl.add(cl4);
cl.add(cl5);
cl.add(cl6);
cl.add(cl7);
mb.add(ct);
mb.add(fn);
mb.add(cl);
f.setSize(500, 500);
f.setVisible(true);
}

public void actionPerformed(ActionEvent e) {
    if(e.getSource()==ct1)
    {
        txt.setText(ct1.getText());
    }
    else if(e.getSource()==ct2)
    {
```

```
txt.setText(ct2.getText());
}

else if(e.getSource()==ct3)
{
    txt.setText(ct3.getText());
}

else if(e.getSource()==ct4)
{
    txt.setText(ct4.getText());
}

else if(e.getSource()==ct5)
{
    txt.setText(ct5.getText());
}

else if(e.getSource()==ct6)
{
    txt.setText(ct6.getText());
}

else if(e.getSource()==ct7)
{
    txt.setText(ct7.getText());
}

else if(e.getSource()==nam1)
{
    f1=new Font(nam1.getText(),Font.PLAIN,20);
    txt.setFont(f1);
}

else if(e.getSource()==nam2)
{
    f1=new Font(nam2.getText(),Font.PLAIN,20);
    txt.setFont(f1);
}
```

```
else if(e.getSource()==nam3)
{
    f1=new Font(nam3.getText(),Font.PLAIN,20);
    txt.setFont(f1);
}

else if(e.getSource()==nam4)
{
    f1=new Font(nam4.getText(),Font.PLAIN,20);
    txt.setFont(f1);
}

else if(e.getSource()==typ1)
{
    f2=new Font(f1.getName(),Font.BOLD,20);
    txt.setFont(f2);
}

else if(e.getSource()==typ2)
{
    f2=new Font(f1.getName(),Font.ITALIC,20);
    txt.setFont(f2);
}

else if(e.getSource()==siz1)
{
    f3=new Font(f2.getName(),f2.getStyle(),12);
    txt.setFont(f3);
}

else if(e.getSource()==siz2)
{
    f3=new Font(f2.getName(),f2.getStyle(),14);
    txt.setFont(f3);
}

else if(e.getSource()==siz3)
{
```

```
f3=new Font(f2.getName(),f2.getStyle(),16);
txt.setFont(f3);

}

else if(e.getSource()==siz4)
{
    f3=new Font(f2.getName(),f2.getStyle(),18);
    txt.setFont(f3);
}

else if(e.getSource()==cl1)
{
    txt.setForeground(Color.GREEN);
}

else if(e.getSource()==cl2)
{
    txt.setForeground(Color.RED);
}

else if(e.getSource()==cl3)
{
    txt.setForeground(Color.BLUE);
}

else if(e.getSource()==cl4)
{
    txt.setForeground(Color.PINK);
}

else if(e.getSource()==cl5)
{
    txt.setForeground(Color.WHITE);
}

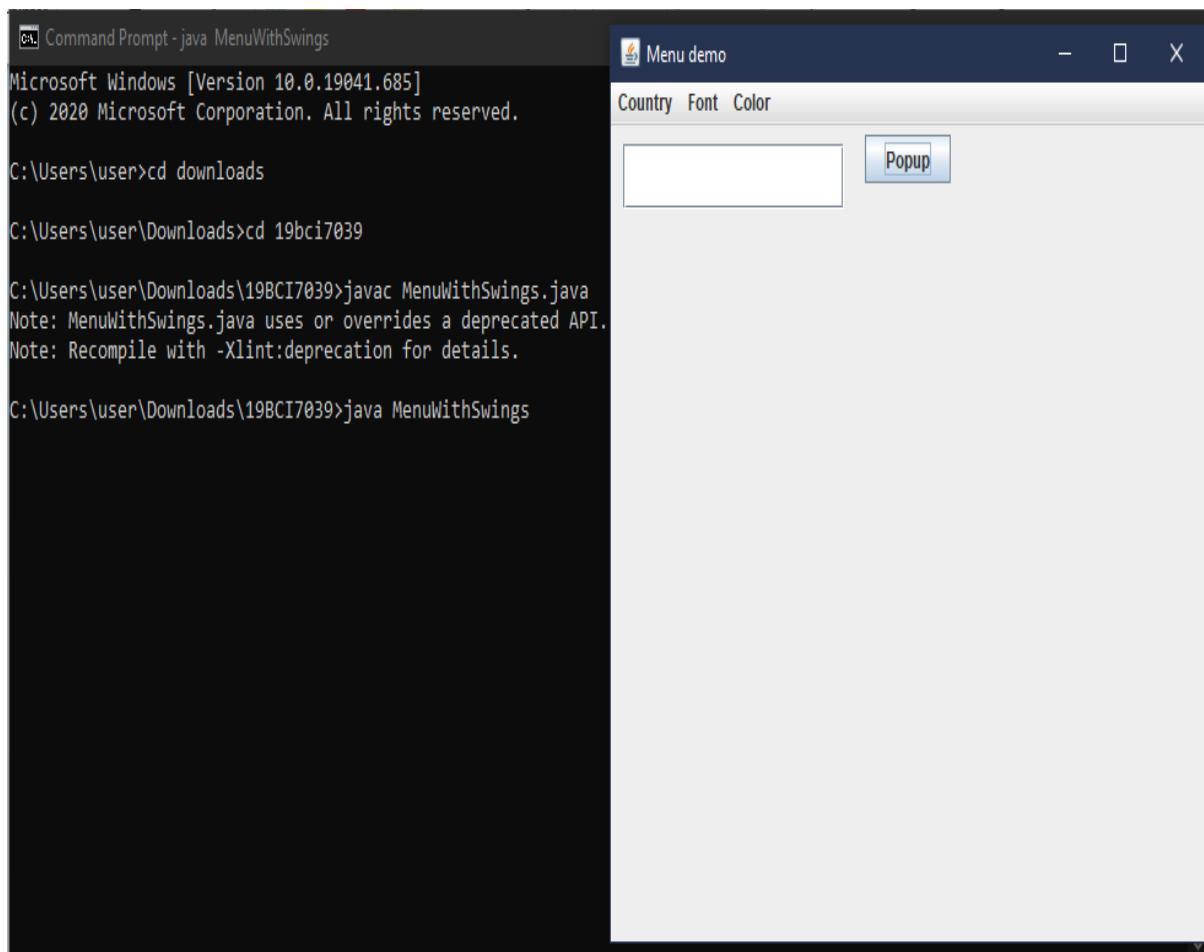
else if(e.getSource()==cl6)
{
    txt.setForeground(Color.GRAY);
}
```

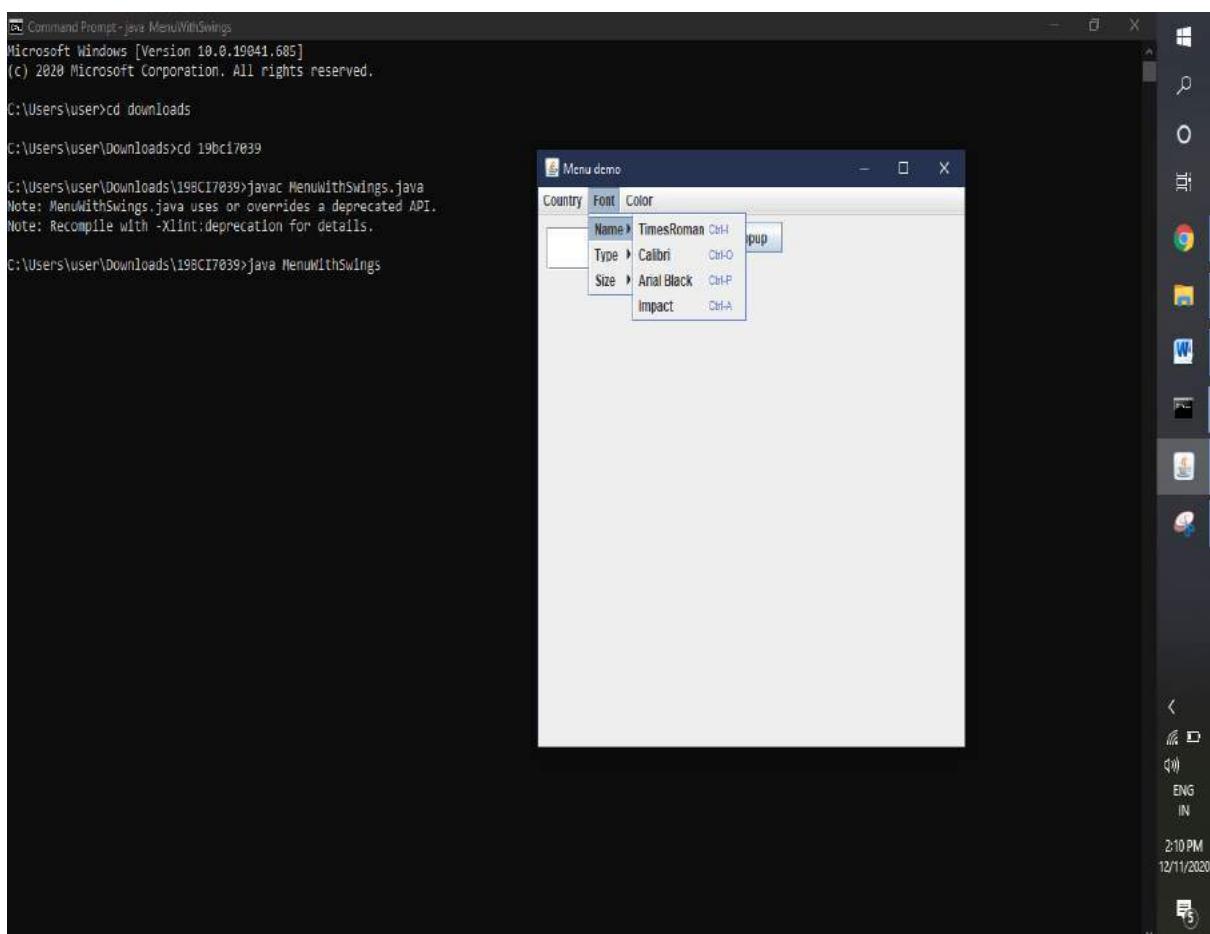
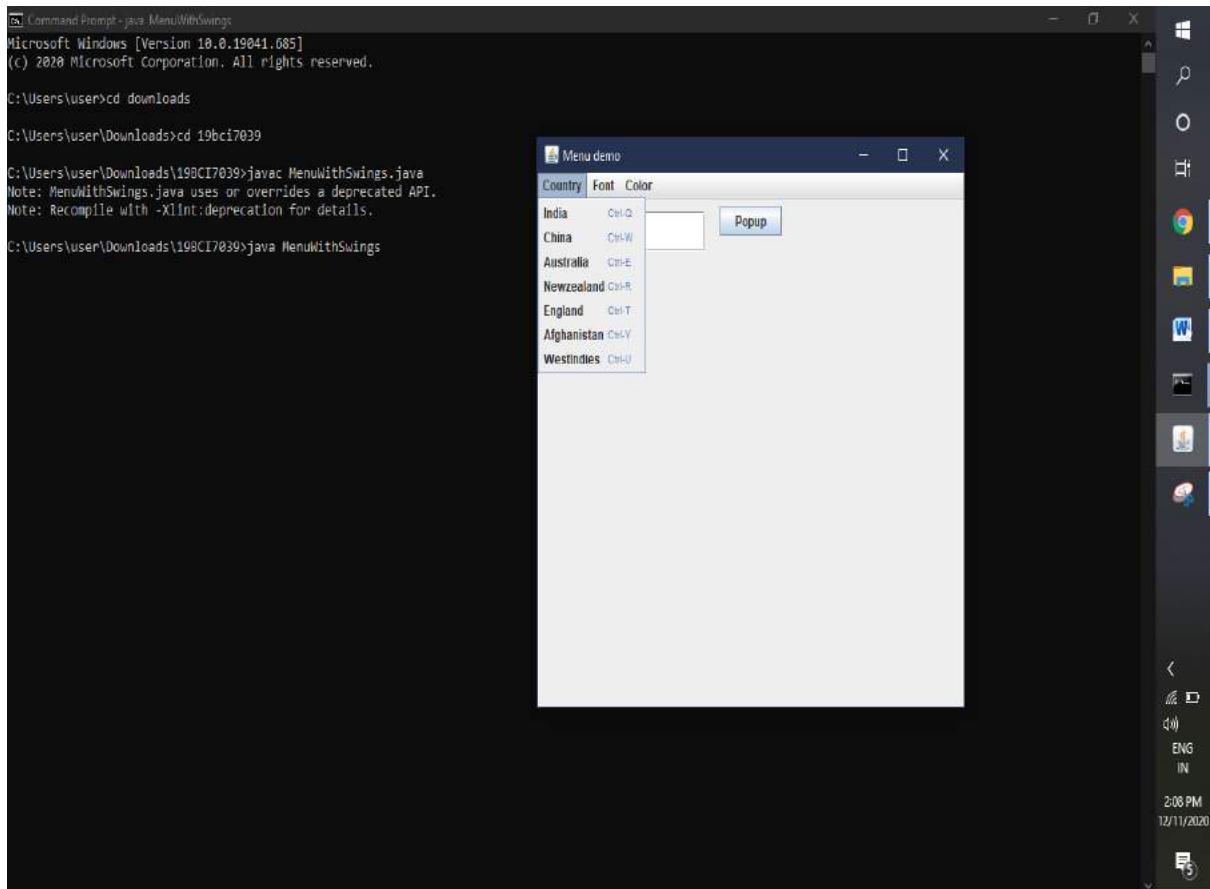
```

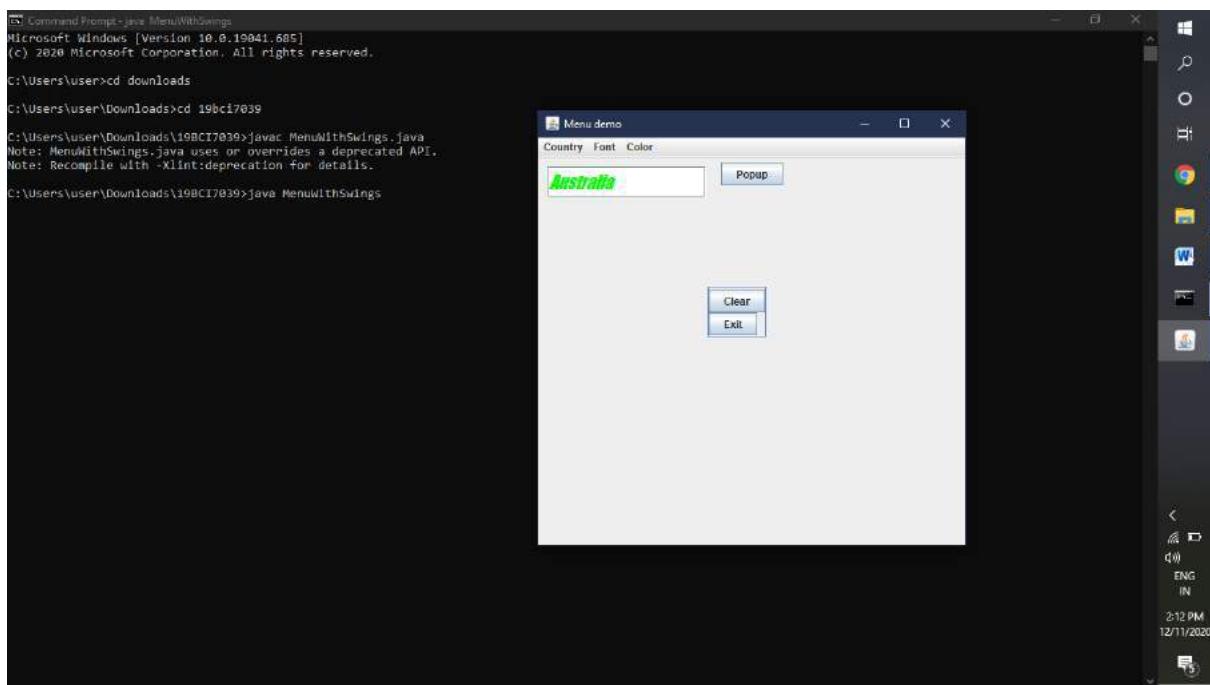
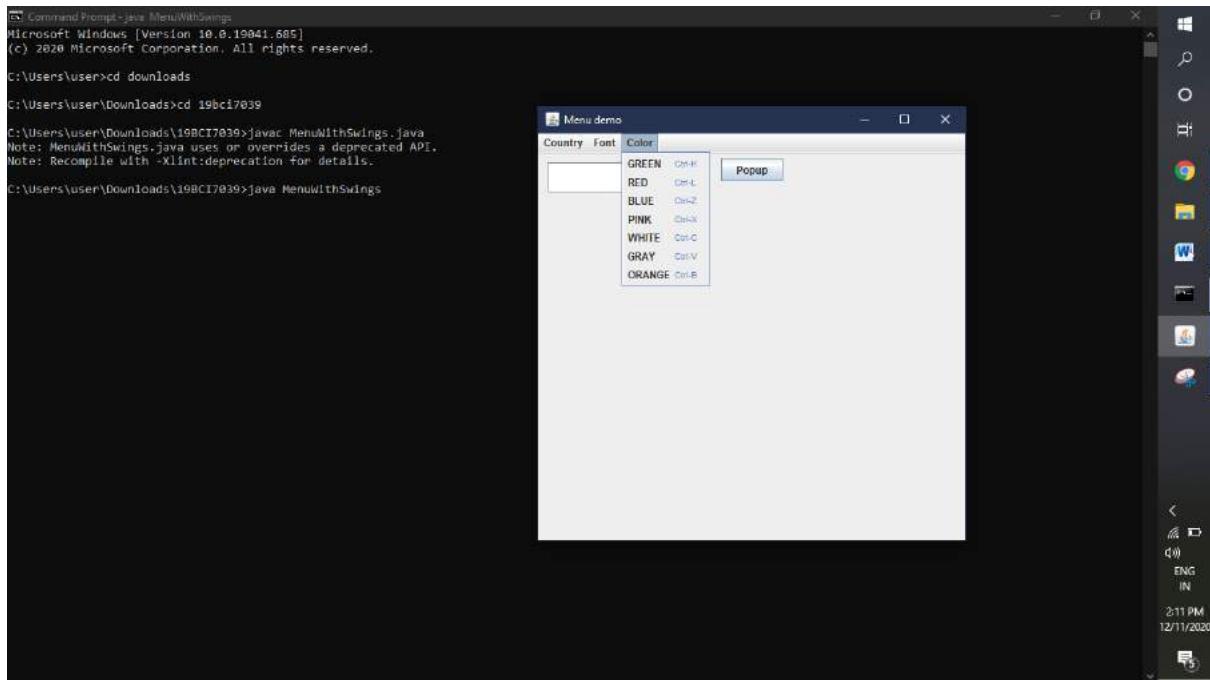
        else if(e.getSource()==cl7)
    {
        txt.setForeground(Color.ORANGE);
    }
    if(e.getSource()==b)
    {
        pm.show(f, 200, 200);
    }
}
public static void main(String args[])
{
    new MenuWithSwings();
}
}

```

Output for Swing:







2.

Develop Question 1 using JavaFX. Save the file as `MenuWithJavaFX.java`

Code using JavaFX :

```
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.beans.value.ObservableValue;
import javafx.scene.layout.StackPane;
```

```
import javafx.scene.control.*;
import javafx.scene.Scene;
import javafx.scene.layout.*;
import javafx.event.Event;
import javafx.event.EventHandler;
import javafx.scene.paint.Color;
import javafx.event.ActionEvent;
import javafx.scene.text.Text;
import javafx.scene.text.*;
import javafx.scene.text.FontWeight;
import javafx.beans.value.*;
import javafx.scene.input.KeyCombination;
public class MenuWithJavaFX extends Application implements EventHandler<ActionEvent>
{
    MenuBar mb;
    Menu ct,fn,cl,nam,typ,siz;
    static Text txt;
    ContextMenu cm;
    MenuItem
    ct1,ct2,ct3,ct4,ct5,ct6,ct7,nam1,nam2,nam3,nam4,typ1,typ2,siz1,siz2,siz3,siz4,cl1,cl2,cl3,cl4
    ,cl5,cl6,cl7;
    Button b,b1,b2;
    Font f1,f2,f3;
    VBox root;
    public void start(Stage primaryStage) throws Exception
    {
        txt=new Text();
        mb=new MenuBar();
        ct=new Menu("_Country");
        fn=new Menu("Font");
        cl=new Menu("Color");
```

```
nam=new Menu("Names");
typ=new Menu("Types");
siz=new Menu("Size");
ct1=new MenuItem("India");
ct2=new MenuItem("China");
ct3=new MenuItem("Australia");
ct4=new MenuItem("Newzealand");
ct5=new MenuItem("England");
ct6=new MenuItem("Afghanistan");
ct7=new MenuItem("WestIndies");
nam1=new MenuItem("Times New Roman");
nam2=new MenuItem("Courier New");
nam3=new MenuItem("Arial Black");
nam4=new MenuItem("Impact");
typ1=new MenuItem("Bold");
typ2=new MenuItem("Italic");
siz1=new MenuItem("12");
siz2=new MenuItem("14");
siz3=new MenuItem("16");
siz4=new MenuItem("18");
cl1=new MenuItem("GREEN");
cl2=new MenuItem("RED");
cl3=new MenuItem("BLUE");
cl4=new MenuItem("PINK");
cl5=new MenuItem("YELLOW");
cl6=new MenuItem("GRAY");
cl7=new MenuItem("ORANGE");
cm = new ContextMenu();
b=new Button("POPUP");
b1=new Button("Clear");
b2=new Button("Exit");
```

```
b.setOnAction(this);
b1.setOnAction(this);
b2.setOnAction(this);
ct.getItems().addAll(ct1,ct2,ct3,ct4,ct5,ct6,ct7);
nam.getItems().addAll(nam1,nam2,nam3,nam4);
typ.getItems().addAll(typ1,typ2);
siz.getItems().addAll(siz1,siz2,siz3,siz4);
cl.getItems().addAll(cl1,cl2,cl3,cl4,cl5,cl6,cl7);
fn.getItems().addAll(nam,typ,siz);
mb.getMenus().addAll(ct,fn,cl);
ct1.setAccelerator(KeyCombination.keyCombination("Ctrl+Q"));
ct2.setAccelerator(KeyCombination.keyCombination("Ctrl+W"));
ct3.setAccelerator(KeyCombination.keyCombination("Ctrl+E"));
ct4.setAccelerator(KeyCombination.keyCombination("Ctrl+R"));
ct5.setAccelerator(KeyCombination.keyCombination("Ctrl+T"));
ct6.setAccelerator(KeyCombination.keyCombination("Ctrl+Y"));
ct7.setAccelerator(KeyCombination.keyCombination("Ctrl+U"));
nam1.setAccelerator(KeyCombination.keyCombination("Ctrl+I"));
nam2.setAccelerator(KeyCombination.keyCombination("Ctrl+O"));
nam3.setAccelerator(KeyCombination.keyCombination("Ctrl+P"));
nam4.setAccelerator(KeyCombination.keyCombination("Ctrl+A"));
typ1.setAccelerator(KeyCombination.keyCombination("Ctrl+S"));
typ2.setAccelerator(KeyCombination.keyCombination("Ctrl+D"));
siz1.setAccelerator(KeyCombination.keyCombination("Ctrl+F"));
siz2.setAccelerator(KeyCombination.keyCombination("Ctrl+G"));
siz3.setAccelerator(KeyCombination.keyCombination("Ctrl+H"));
siz4.setAccelerator(KeyCombination.keyCombination("Ctrl+J"));
cl1.setAccelerator(KeyCombination.keyCombination("Ctrl+K"));
cl2.setAccelerator(KeyCombination.keyCombination("Ctrl+L"));
cl3.setAccelerator(KeyCombination.keyCombination("Ctrl+Z"));
cl4.setAccelerator(KeyCombination.keyCombination("Ctrl+X"));
```

```
cl5.setAccelerator(KeyCombination.keyCombination("Ctrl+C"));
cl6.setAccelerator(KeyCombination.keyCombination("Ctrl+V"));
cl7.setAccelerator(KeyCombination.keyCombination("Ctrl+B"));

ct1.setOnAction(this);
ct2.setOnAction(this);
ct3.setOnAction(this);
ct4.setOnAction(this);
ct5.setOnAction(this);
ct6.setOnAction(this);
ct7.setOnAction(this);

nam1.setOnAction(this);
nam2.setOnAction(this);
nam3.setOnAction(this);
nam4.setOnAction(this);

typ1.setOnAction(this);
typ2.setOnAction(this);

siz1.setOnAction(this);
siz2.setOnAction(this);
siz3.setOnAction(this);
siz4.setOnAction(this);

cl1.setOnAction(this);
cl2.setOnAction(this);
cl3.setOnAction(this);
cl4.setOnAction(this);
cl5.setOnAction(this);
cl6.setOnAction(this);
cl7.setOnAction(this);

root=new VBox();
root.getChildren().addAll(mb,txt,b);

Scene scene=new Scene(root);
primaryStage.setScene(scene);
```

```
primaryStage.setTitle("Menus");
primaryStage.show();
}

public void handle(ActionEvent e)
{
    if(e.getSource()==ct1)
    {
        txt.setText(ct1.getText());
    }
    else if(e.getSource()==ct2)
    {
        txt.setText(ct2.getText());
    }
    else if(e.getSource()==ct3)
    {
        txt.setText(ct3.getText());
    }
    else if(e.getSource()==ct4)
    {
        txt.setText(ct4.getText());
    }
    else if(e.getSource()==ct5)
    {
        txt.setText(ct5.getText());
    }
    else if(e.getSource()==ct6)
    {
        txt.setText(ct6.getText());
    }
    else if(e.getSource()==ct7)
    {
```

```
txt.setText(ct7.getText());
}

else if(e.getSource()==nam1)
{
    f1=new Font(nam1.getText(),20);
    txt.setFont(f1);
}

else if(e.getSource()==nam2)
{
    f1=new Font(nam2.getText(),20);
    txt.setFont(f1);
}

else if(e.getSource()==nam3)
{
    f1=new Font(nam3.getText(),20);
    txt.setFont(f1);
}

else if(e.getSource()==nam4)
{
    f1=new Font(nam4.getText(),20);
    txt.setFont(f1);
}

else if(e.getSource()==typ1)
{
    txt.setFont(Font.getFont(txt.getStyle(), FontWeight.BOLD, 20));
}

else if(e.getSource()==typ2)
{
    txt.setFont(Font.getFont(txt.getStyle(),FontPosture.ITALIC,20));
}

else if(e.getSource()==siz1)
```

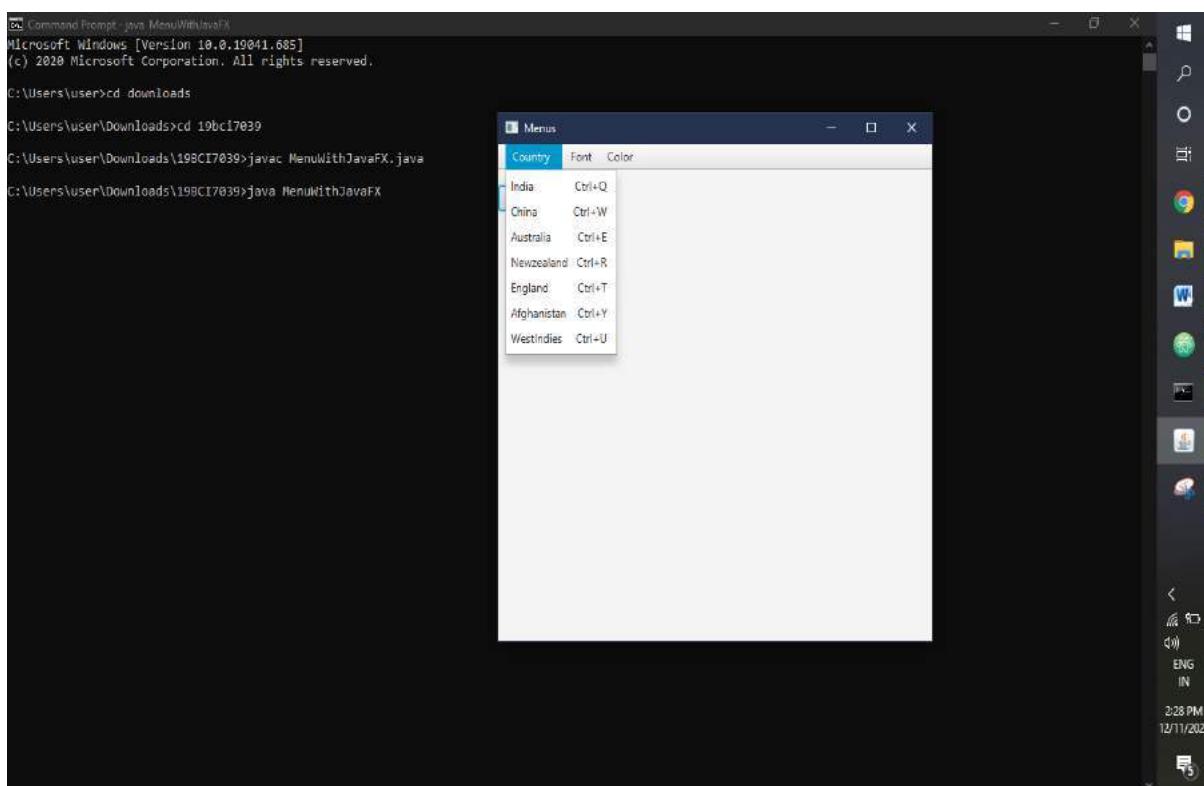
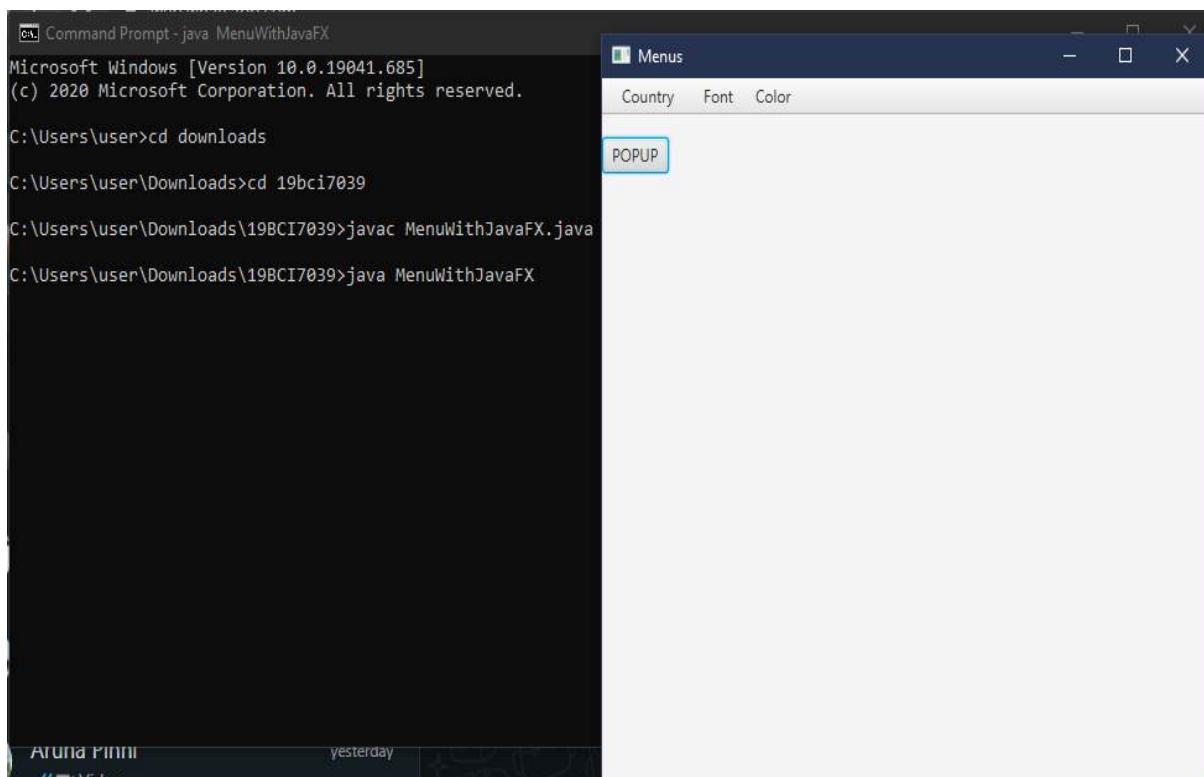
```
{  
    f2=new Font(f1.getStyle(),12);  
    txt.setFont(f2);  
}  
  
else if(e.getSource()==siz2)  
{  
    f2=new Font(f1.getStyle(),14);  
    txt.setFont(f2);  
}  
  
else if(e.getSource()==siz3)  
{  
    f2=new Font(f1.getStyle(),16);  
    txt.setFont(f2);  
}  
  
else if(e.getSource()==siz4)  
{  
    f2=new Font(f1.getStyle(),18);  
    txt.setFont(f2);  
}  
  
else if(e.getSource()==cl1)  
{  
    txt.setFill(Color.GREEN);  
}  
  
else if(e.getSource()==cl2)  
{  
    txt.setFill(Color.RED);  
}  
  
else if(e.getSource()==cl3)  
{  
    txt.setFill(Color.BLUE);  
}
```

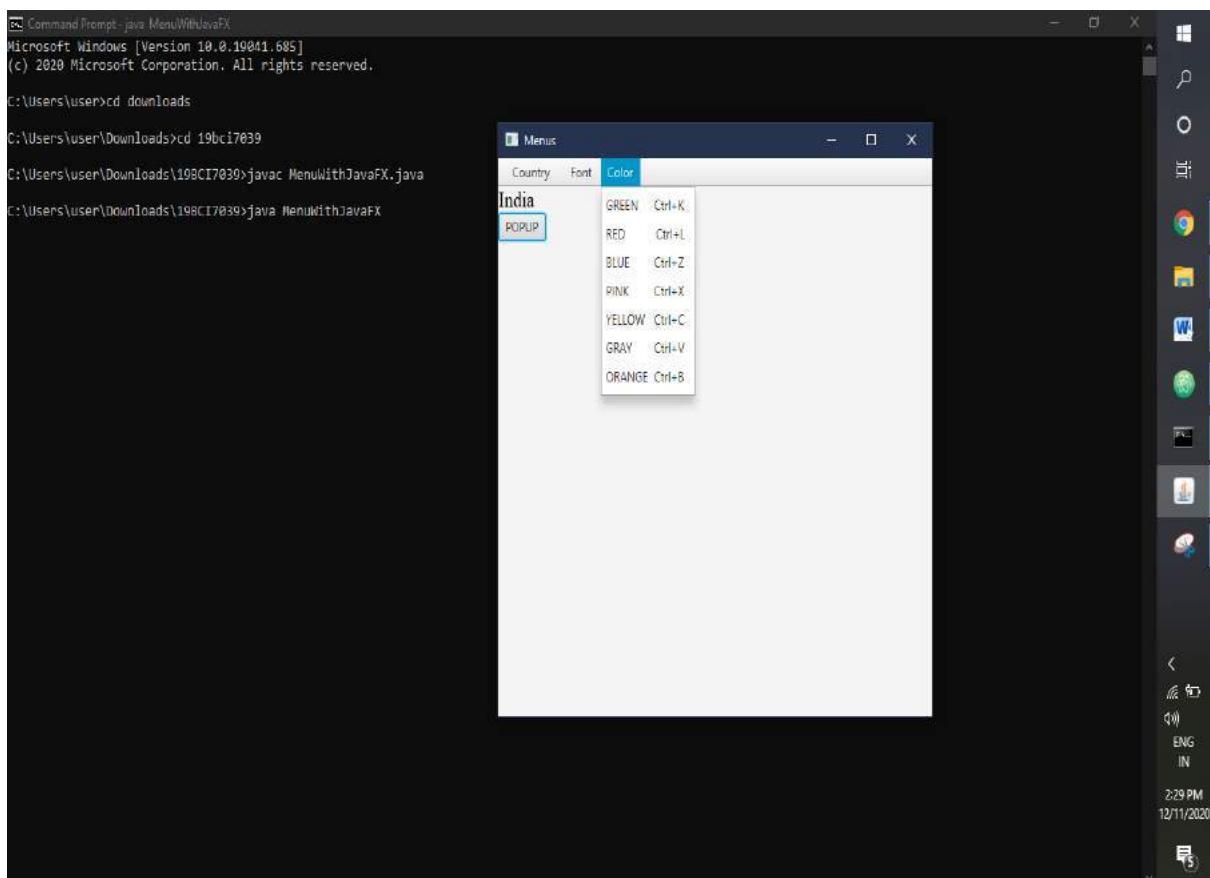
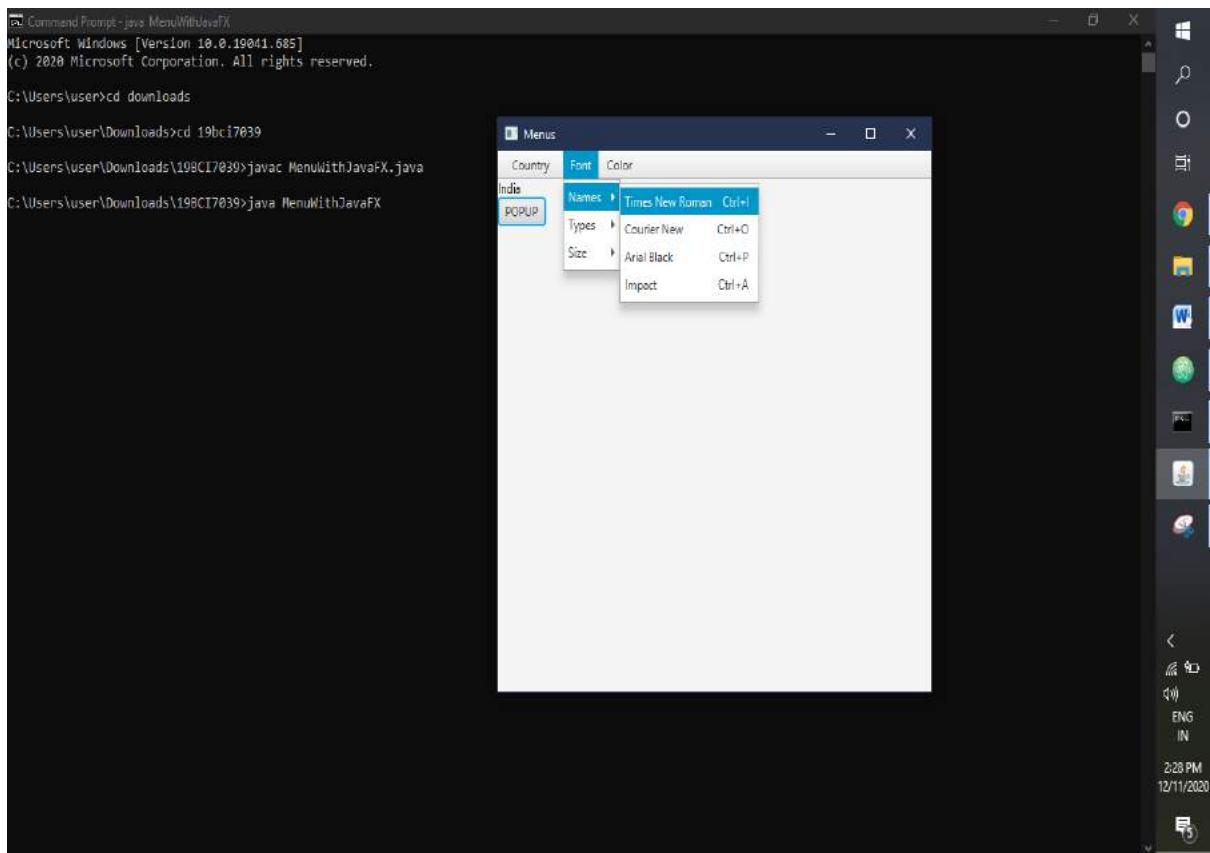
```
else if(e.getSource()==cl4)
{
    txt.setFill(Color.PINK);
}
else if(e.getSource()==cl5)
{
    txt.setFill(Color.YELLOW);
}
else if(e.getSource()==cl6)
{
    txt.setFill(Color.GRAY);
}
else if(e.getSource()==cl7)
{
    txt.setFill(Color.ORANGE);
}
else if(e.getSource()==b)
{
    root.getChildren().addAll(b1,b2);
}
else if(e.getSource()==b1)
{
    txt.setText("");
}
else if(e.getSource()==b2)
{
    System.exit(0);
}
}

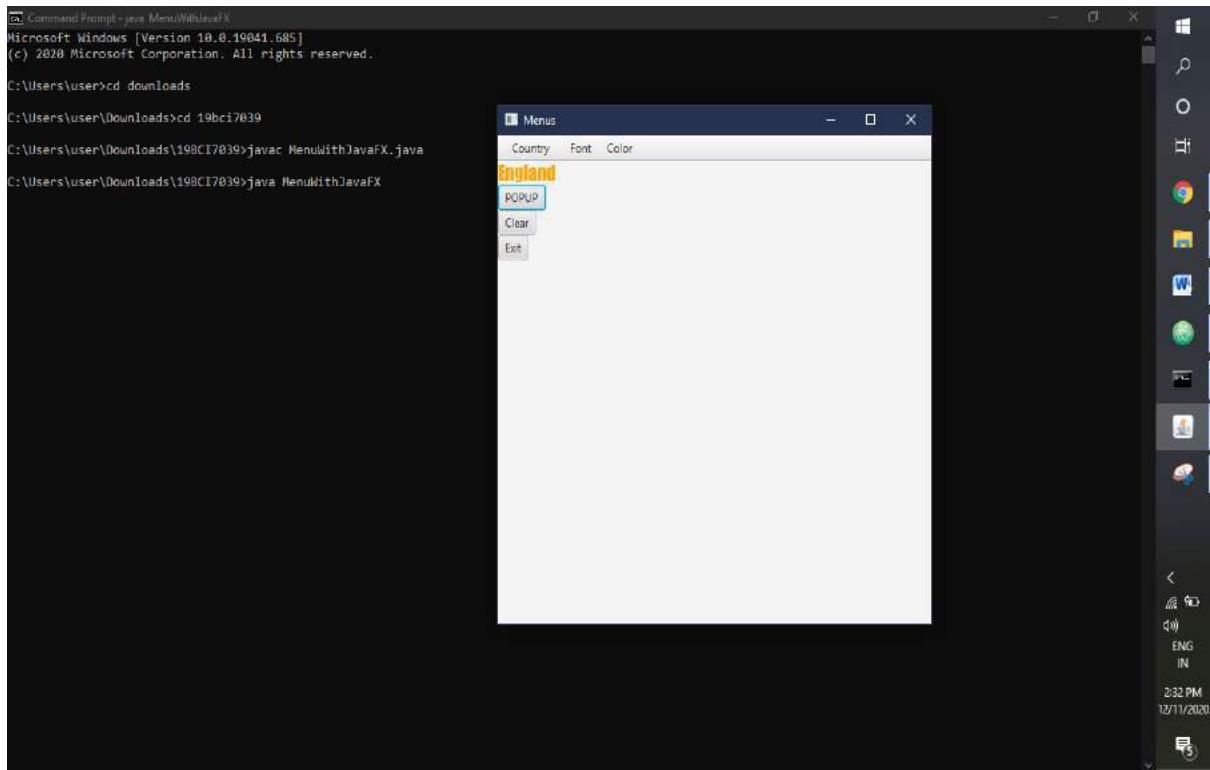
public static void main(String args[])
{
```

```
        launch(args);  
    }  
}
```

Output with JavaFX :







Module-6: Introducing JavaFX

Introduction:

To move away from that rather uninteresting text screen you have been using and build attractive windows programs for input and output, you are going to use the Java graphics package known as JavaFX.

This package provides all the graphics tools and components that you need to produce the sort of graphical interfaces that we have all become used to in modern day applications.

Small history:

From around the year 2000–2014, Swing was the main platform for producing Java Graphics. However, its look and feel became rather old-fashioned compared to today's graphics that run across multiple devices, and with the release of Java 8 in 2014 came the latest version of a new technology known as JavaFX, together with the announcement that Swing will not be developed further (although it will continue to be packaged with Java).

Some technical points:

Firstly, you need to know that a JavaFX program is referred to as an application. Your JavaFX class will extend the **Application class**, for which you need the following import statement:

import javafx.application.Application

Every JavaFX program is a child of the **Application class**.

Like other Java applications, JavaFX programs begin execution with a **main() method**. Within the **main()** method, a call to the **launch()** method sets up the application as a JavaFX application.

Most of the code executed by a JavaFX application is written within the **start() method**.

Stage:

The top-level window in which the application runs is called a stage.

When you run the program in full-screen mode, then the screen becomes the stage. Some applications can be made to run in a browser, in which case the browser is the stage.

The Stage class represents the entire window in an application.

The Stage class describes a container for an application.

In a JavaFX application, a Stage object represents the entire window. This includes the title bar and the minimize, maximize, and close buttons.

Scene:

The contents of the stage--the graphic itself--is called a scene, and is often referred to as a scene graphic.

The Scene class holds content inside a window.

A Scene resides inside the Stage and contains all of the content of the application.

Components:

The items that make up the scene are referred to as **nodes**. They are very commonly the kind of components that allow interaction with the user, such as **buttons**, **text fields**, **labels** and **check boxes**, which are often referred to collectively as **controls**.

The package **javafx.scene.control** provides all the necessary classes for the JavaFX User Interface Controls (UI Controls or just Controls) like Button, Label, TextField, etc.

Nodes can also be containers. Containers are components that hold other nodes, and each container arranges the nodes in a particular way-- for example, **vertically, horizontally, in a grid, or stacked one on top of the other.**

JavaFX Layouts or Panes:

The **JavaFX Layouts** helps manage the layout of the Scene, and contains objects, such as buttons.

Layouts are the top level container classes that define the styles for scene-graph objects.

Note: In JavaFX, the GUI Applications were constructed using a Scene Graph. A scene graph is a data structure similar to tree, in modern graphical applications. It is the starting point of the application, and it is a collection of nodes.

To display something in JavaFX you need to construct a scene graph using the nodes and set it to an object of the Stage class, the top level container of a JavaFX application.

A node is a visual/graphical primitive object of a JavaFX application.

Layout can be seen as the parent node to all the other nodes.

JavaFX provides various layout panes that support different styles of layouts.

In JavaFX, Layout defines the way in which the components are to be seen on the stage. It basically organizes the scene-graph nodes. We have several built-in layout panes in JavaFX that are **HBox**, **VBox**, **StackPane**, **FlowBox**, **AnchorPane**, etc. Each Built-in layout is represented by a separate class which needs to be instantiated in order to implement that particular layout pane.

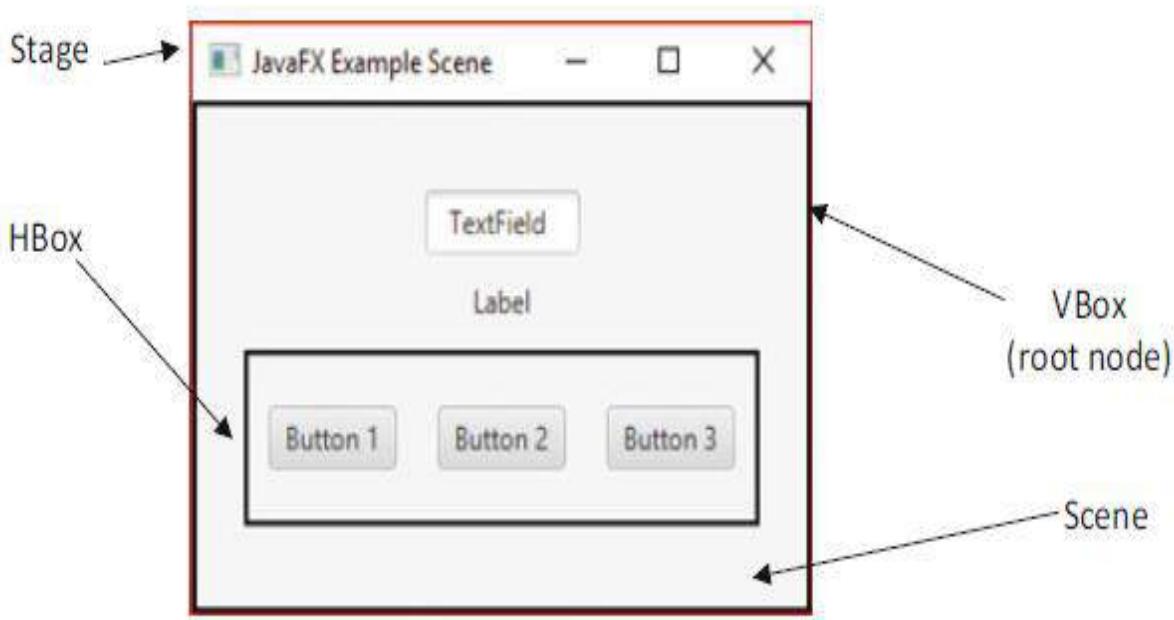
All these classes belong to **javafx.scene.layout** package.

javafx.scene.layout.Pane class is the base class for all the built-in layout classes in JavaFX.

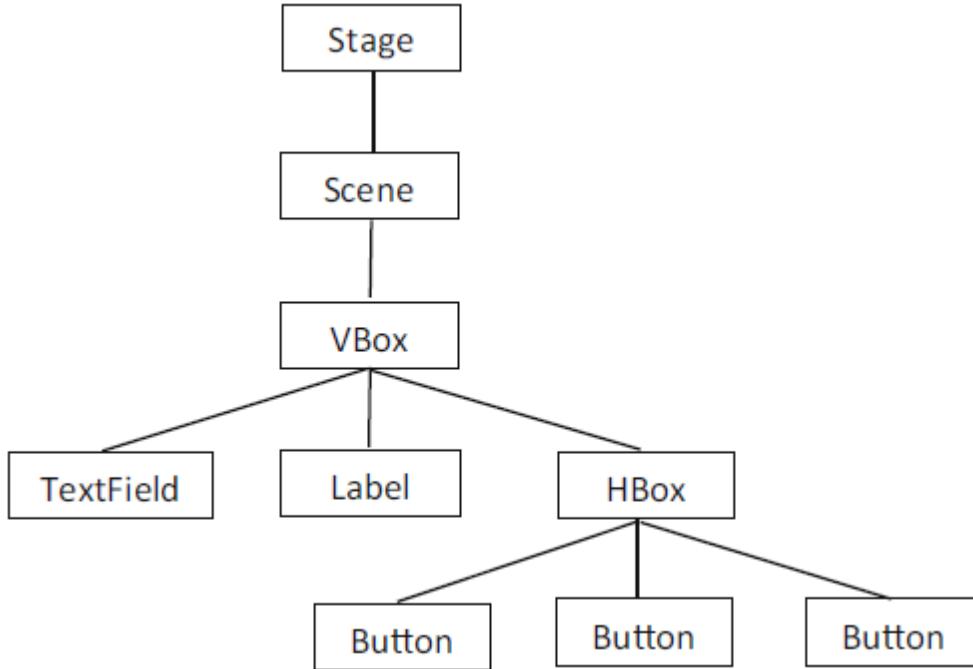
Note: Importantly, containers can contain other containers, so we can develop a hierarchy in our scene. We normally place a single **top level node** in our scene, and this is referred to as the **root node**. We use the terms **parent** and **children** for the containing and contained nodes respectively.

Here we have a sample scene in which the root node is a **VBox**—this is a container that arranges its child nodes vertically. We have given it a black border so that you can see it. The **VBox** has three children—a **TextField**, a **Label** and an **HBox**.

As you can probably guess, an **HBox** is similar to a **VBox**, but arranges its child nodes horizontally. In this case it has three child nodes which are **Buttons**.



The hierarchical structure of the scene is



abstract void start(Stage stage):

As you can see, it is an **abstract** method and therefore has to be coded. It is in this method that the code for our application is placed.

As we explained, all JavaFX programs run as an application, and we therefore have to extend the Application class. Application requires you to code the start method.

When start is called, it is automatically sent an object of the Stage class, which will be the main container for our graphic.

How do we launch a JavaFX application?

```

public static void main(String[] args)
{
    launch(args);
}
  
```

As you can see the main method calls the application's launch method, and passes to it any arguments received by the main method itself.

The **launch** method is a **static** method, and we can use it to launch a JavaFX application from another program. It is overloaded to accept the name of the compiled .class file as its first parameter.

If we want a program called, say, LaunchApplication to run an application called MyApp, we would do it like this:

```
import javafx.application.Application;
class LaunchApplication
{public static void main(String[] args)
{ Application.launch(MyApp.class, args); }
}
```

JavaFX controls:

JavaFX Button

JavaFX button control is represented by **javafx.scene.control.Button** class. A button is a component that can control the behaviour of the Application. An event is generated whenever the button gets clicked.

How to create a Button?

Button can be created by instantiating Button class. Use the following line to create button object.

Ex.1

```
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.layout.StackPane;
import javafx.stage.Stage;

public class ButtonJFX1 extends Application {

    public void start(Stage primaryStage) throws Exception {

        StackPane root = new StackPane(); // "StackPane" layout

        Button btn=new Button("This is a button"); //Button control

        Scene scene=new Scene(root,300,300); // Here we have chosen to use the constructor that
        allows us to set the size (width and height) of the initial scene.
    }
}
```

root.getChildren().add(btn); // Having created our layout as root, we need to add our control like button to it. We do this by calling a method, called **getChildren()**, which returns a list of all the child nodes.

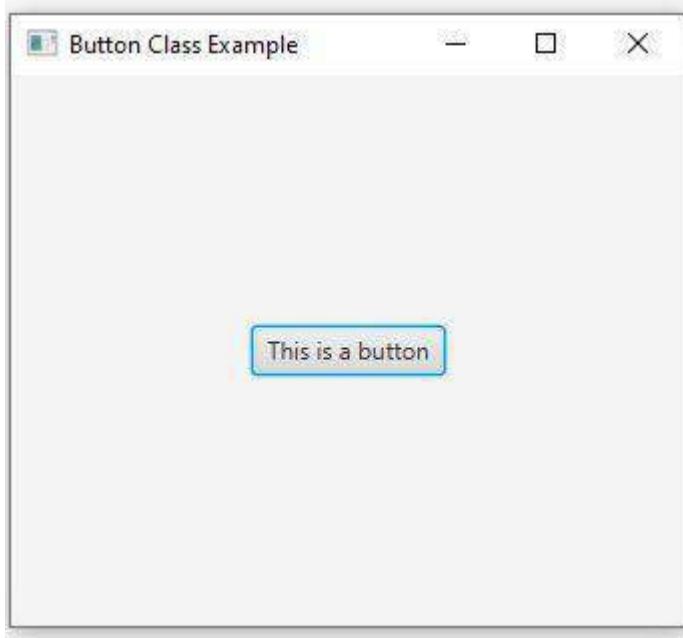
```
// This list has two methods for adding the nodes: the add() method will add a single item, and addAll() a list of items.
```

```
primaryStage.setScene(scene); // add the scene to the stage
primaryStage.setTitle("Button Class Example");
primaryStage.show(); // show the stage

}

public static void main(String[] args) {
    launch(args);
}
}
```

o/p:



Ex2.

```
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.layout.HBox;
import javafx.stage.Stage;
public class ButtonJFX2 extends Application
{
    public void start(Stage primaryStage) throws Exception {
        Button btn1 = new Button("Button 1");
        Button btn2 = new Button("Button 2");

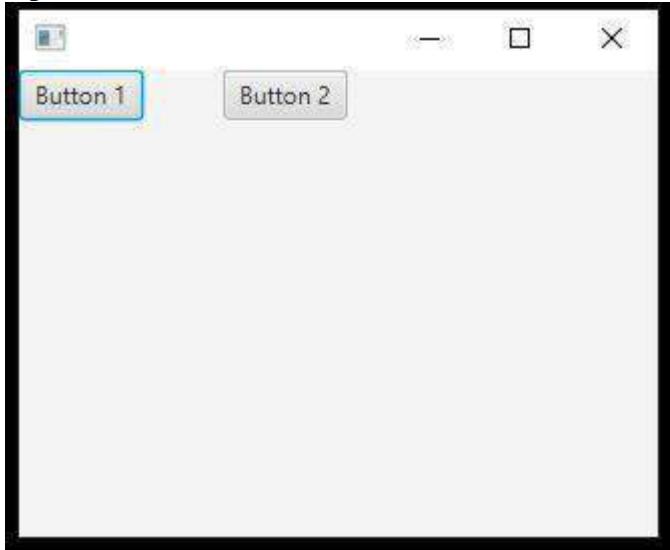
        HBox root = new HBox();
        Scene scene = new Scene(root,200,200);
```

```
root.getChildren().addAll(btn1,btn2);
root.setSpacing(40);
primaryStage.setScene(scene);
primaryStage.show();    }

public static void main(String[] args) {
    launch(args);    }

}
```

o/p:



JavaFX Label

javafx.scene.control.Label class represents label control.

Ex.

```
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.control.Label;
import javafx.scene.layout.StackPane;
import javafx.stage.Stage;

public class LabelTestJFX extends Application {

    public void start(Stage primaryStage) throws Exception {

        Label my_label=new Label("This is an example of Label");

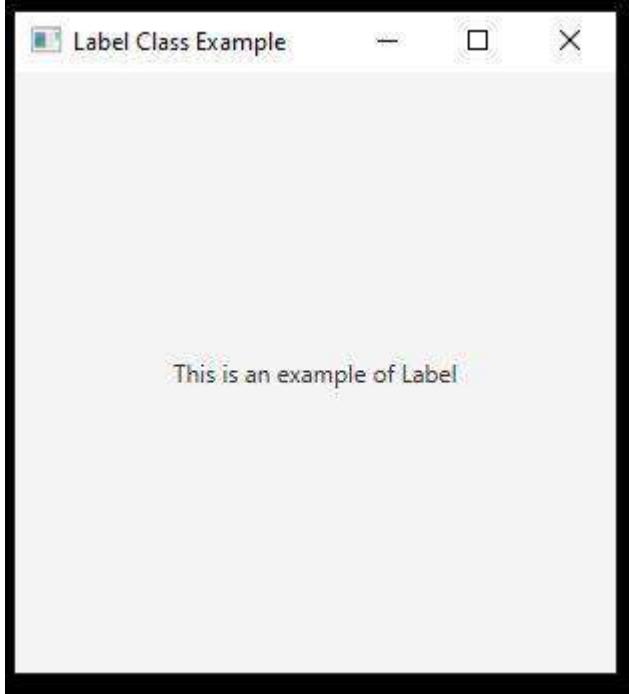
        StackPane root = new StackPane();
        Scene scene=new Scene(root,300,300);
```

```
root.getChildren().add(my_label);
primaryStage.setScene(scene);
primaryStage.setTitle("Label Class Example");
primaryStage.show();

}

public static void main(String[] args) {
    launch(args);
}
}
```

o/p:



RadioButton

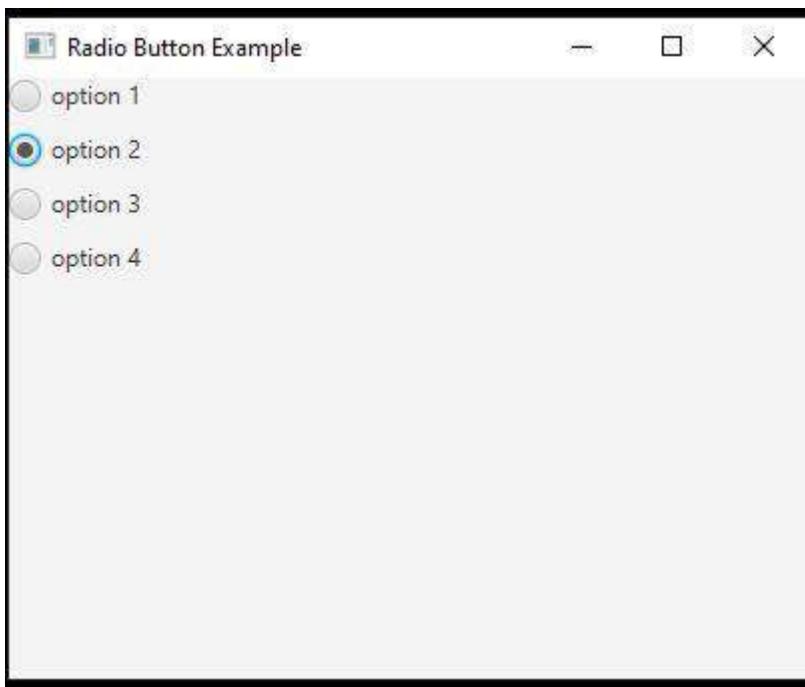
The Radio Button is used to provide various options to the user.

Ex.

```
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.control.RadioButton;
import javafx.scene.control.ToggleGroup;
import javafx.scene.layout.VBox;
import javafx.stage.Stage;
public class RadioButtonTest extends Application {
```

```
public static void main(String[] args) {  
    launch(args);  
}  
  
public void start(Stage primaryStage) throws Exception {  
  
    RadioButton button1 = new RadioButton("option 1");  
    RadioButton button2 = new RadioButton("option 2");  
    RadioButton button3 = new RadioButton("option 3");  
    RadioButton button4 = new RadioButton("option 4");  
  
    ToggleGroup group = new ToggleGroup();  
  
    button1.setToggleGroup(group);  
    button2.setToggleGroup(group);  
    button3.setToggleGroup(group);  
    button4.setToggleGroup(group);  
  
    VBox root=new VBox();  
    root.setSpacing(10);  
  
    root.getChildren().addAll(button1,button2,button3,button4);  
  
    Scene scene=new Scene(root,400,300);  
  
    primaryStage.setScene(scene);  
    primaryStage.setTitle("Radio Button Example");  
    primaryStage.show();  
}  
}
```

o/p:



JavaFX CheckBox

The Check Box is used to provide more than one choices to the user. It can be used in a scenario where the user is prompted to select more than one option or the user wants to select multiple options.

It is different from the radiobutton in the sense that, **we can select more than one checkboxes in a scenerio.**

Ex.

```
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.control.CheckBox;
import javafx.scene.control.Label;
import javafx.scene.layout.HBox;
import javafx.stage.Stage;
public class CheckBoxTestJFX extends Application {
```

```
public static void main(String[] args) {
    launch(args);
}
```

```
public void start(Stage primaryStage) throws Exception {
```

```
    Label l = new Label("What do you listen: ");
```

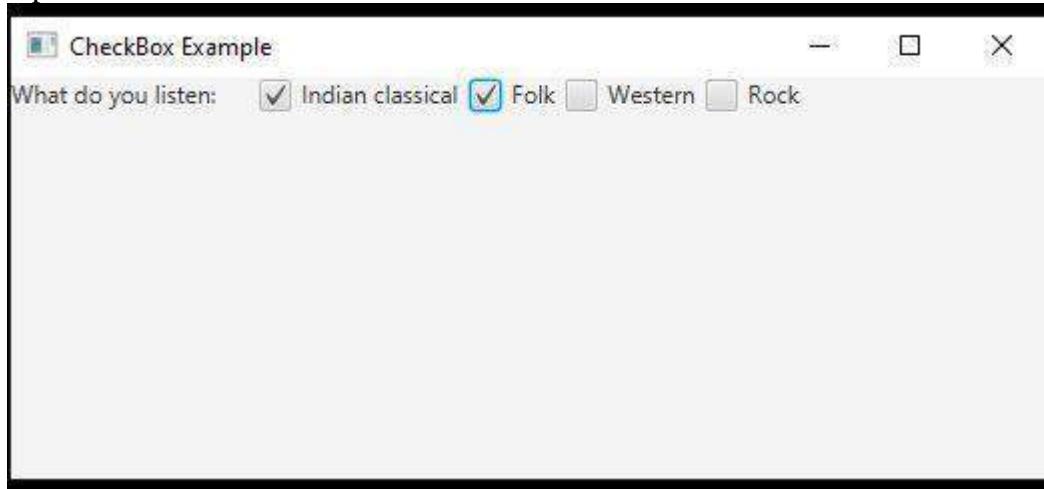
```
CheckBox c1 = new CheckBox("Indian classical");
CheckBox c2 = new CheckBox("Folk");
CheckBox c3 = new CheckBox("Western");
CheckBox c4 = new CheckBox("Rock");

HBox root = new HBox();

root.getChildren().addAll(l,c1,c2,c3,c4);

root.setSpacing(5);
Scene scene=new Scene(root,800,200);
primaryStage.setScene(scene);
primaryStage.setTitle("CheckBox Example");
primaryStage.show();
}
}
```

o/p:



JavaFX TextField

Text Field is basically used to get the input from the user in the form of text.

javafx.scene.control.TextField represents TextField. It provides various methods to deal with textfields in JavaFX. TextField can be created by instantiating TextField class.

Ex.

```
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.control.TextField;
```

```
import javafx.scene.layout.GridPane;
import javafx.stage.Stage;
public class TextFieldTestJFX extends Application {

    public static void main(String[] args) {
        launch(args);
    }

    public void start(Stage primaryStage) throws Exception {

        Label user_id=new Label("User ID");
        Label password = new Label("Password");

        TextField tf1=new TextField();
        TextField tf2=new TextField();

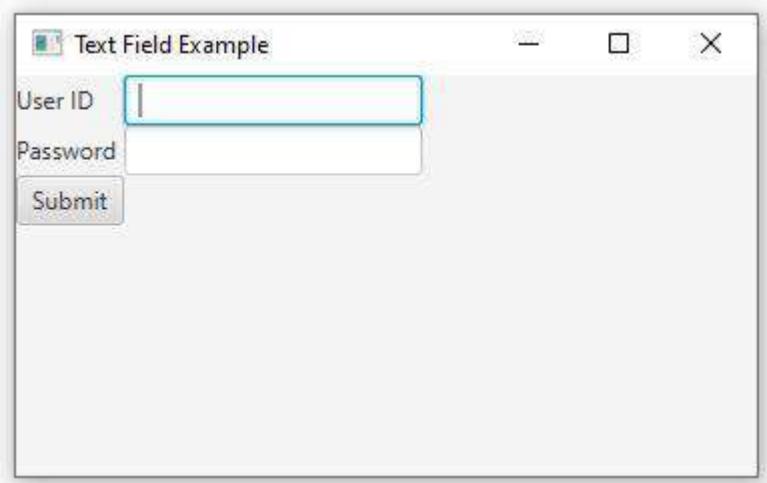
        Button b = new Button("Submit");

        GridPane root = new GridPane();

        root.addRow(0, user_id, tf1);
        root.addRow(1, password, tf2);
        root.addRow(2, b);

        Scene scene=new Scene(root,800,200);
        primaryStage.setScene(scene);
        primaryStage.setTitle("Text Field Example");
        primaryStage.show();
    }
}
```

o/p:



JavaFX ScrollPane:

ScrollPane is a scrollable component used to display a large content in a limited space. It contains horizontal and vertical scroll bars.

Ex.

```
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.control.Button;

import javafx.scene.control.ScrollPane;
import javafx.scene.control.ScrollPane.ScrollBarPolicy;

import javafx.stage.Stage;

public class ScrollPaneDemo1JFX extends Application {

    public void start(Stage primaryStage) {

        // Create a ScrollPane
        ScrollPane scrollPane = new ScrollPane();

        Button button = new Button("My Button");
        button.setPrefSize(400, 300);

        // Set content for ScrollPane
        scrollPane.setContent(button);

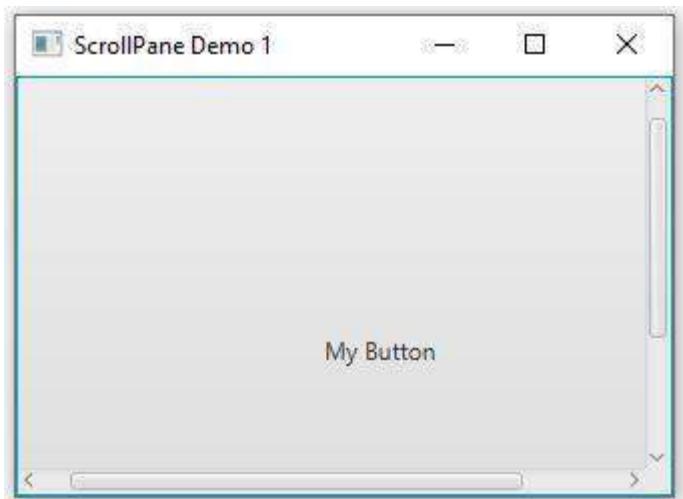
        // Always show vertical scroll bar
        scrollPane.setVbarPolicy(ScrollBarPolicy.ALWAYS);

        // Horizontal scroll bar is only displayed when needed
        scrollPane.setHbarPolicy(ScrollBarPolicy.AS_NEEDED);

        primaryStage.setTitle("ScrollPane Demo 1");
        Scene scene = new Scene(scrollPane, 550, 200);
        primaryStage.setScene(scene);
        primaryStage.show();
    }

    public static void main(String[] args) {
        launch(args);
    }
}
```

o/p:



Note:

Enum Constant Summary

Enum Constants

Enum Constant and Description

ALWAYS

Indicates that a scroll bar should always be shown.

AS_NEEDED

Indicates that a scroll bar should be shown when required.

NEVER

Indicates that a scroll bar should never be shown.

JavaFX Menu

JavaFX provides a **Menu** class to implement menus.

In JavaFX, **javafx.scene.control.Menu** class provides all the methods to deal with menus.
This class needs to be instantiated to create a Menu.

Ex.

```
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.layout.BorderPane;
import javafx.stage.Stage;

public class MenuExampleJFX extends Application {
```

```
public static void main(String[] args) {
    launch(args);
}

public void start(Stage primaryStage) throws Exception {

    BorderPane root = new BorderPane(); //create the empty layout
    Scene scene = new Scene(root,200,300);

    MenuBar menubar = new MenuBar();

    Menu FileMenu = new Menu("File");
    MenuItem filemenu1=new MenuItem("new");
    MenuItem filemenu2=new MenuItem("Save");
    MenuItem filemenu3=new MenuItem("Exit");

    Menu EditMenu=new Menu("Edit");
    MenuItem EditMenu1=new MenuItem("Cut");
    MenuItem EditMenu2=new MenuItem("Copy");
    MenuItem EditMenu3=new MenuItem("Paste");

    EditMenu.getItems().addAll(EditMenu1,EditMenu2,EditMenu3);
    FileMenu.getItems().addAll(filemenu1,filemenu2,filemenu3);

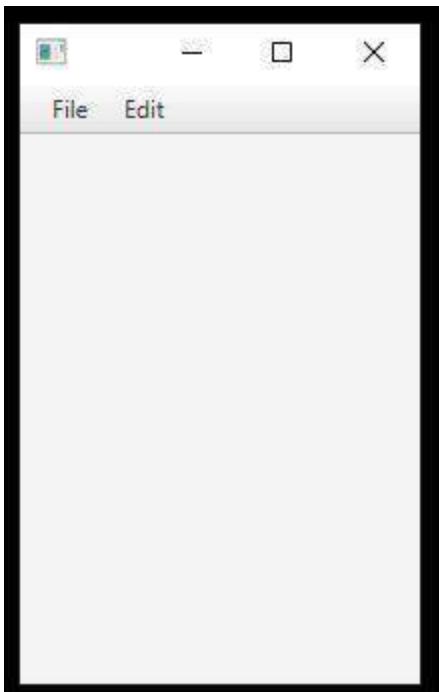
    menubar.getMenus().addAll(FileMenu>EditMenu);

    root.setTop(menubar);

    primaryStage.setScene(scene);
    primaryStage.show();
}

}
```

O/p:



ListView

List views are controls that display a list of entries from which you can select one or more.

ListView is a **generic class** that is declared like this:

```
class ListView<T>
```

Here, **T** specifies the type of entries stored in the list view. Often, these are entries of type **String**, but other types are also allowed.

ListView defines two constructors.

The first is the default constructor, which creates an empty **ListView**.

The second lets you specify the list of entries in the list.
It is shown here:

```
ListView(ObservableList<T> list)
```

Here, **list** specifies a list of the items that will be displayed.
It is an object of type **ObservableList**, which defines a list of observable objects. It inherits **java.util.List**.

ObservableList is packaged in **javafx.collections**.

To create an **ObservableList** for use in a **ListView** is to use the **method observableArrayList()**, which is a static method defined by the **FXCollections** class (which is also packaged in **javafx.collections**).

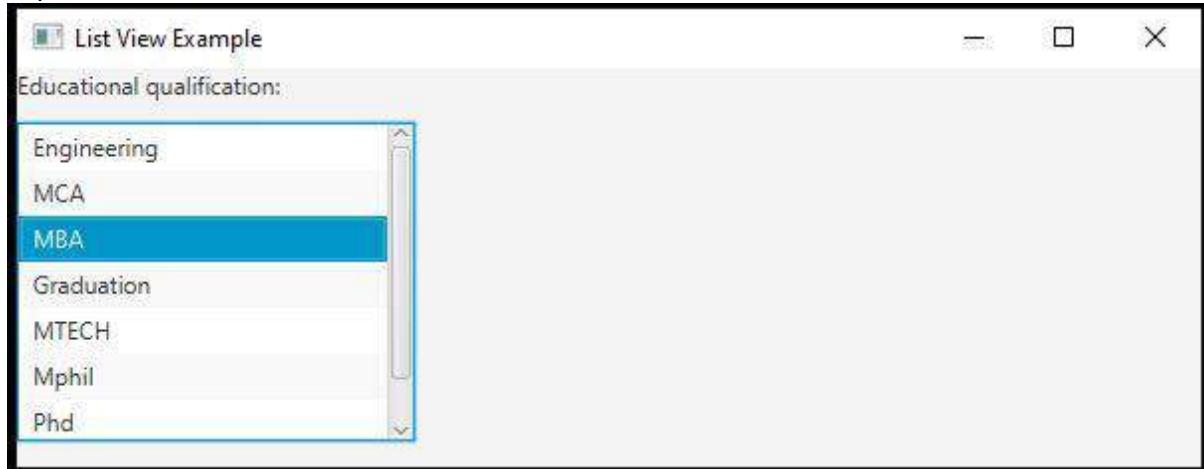
Note: One very useful feature of ListView is that when the number of items in the list exceeds the number that can be displayed within its dimensions, **scroll bars are automatically added**.

Ex.

```
import javafx.application.Application;  
  
import javafx.collections.FXCollections;  
import javafx.collections.ObservableList;  
  
import javafx.scene.Scene;  
import javafx.scene.control.Label;  
  
import javafx.scene.control.ListView;  
  
import javafx.scene.layout.VBox;  
import javafx.stage.Stage;  
public class ListViewExamplefx extends Application {  
  
    public void start(Stage stage) {  
  
        //Label for education  
        Label label = new Label("Educational qualification:");  
  
        //list View for educational qualification  
        ObservableList<String> names = FXCollections.observableArrayList("Engineering",  
"MCA", "MBA", "Graduation", "MTECH", "Mphil", "Phd");  
  
        ListView<String> listView = new ListView<String>(names);  
        listView.setMaxSize(200, 160);  
  
        //Creating the layout  
        VBox layout = new VBox(10);  
        layout.getChildren().addAll(label, listView);  
  
        //Setting the stage  
        Scene scene = new Scene(layout, 595, 200);  
        stage.setTitle("List View Example");  
        stage.setScene(scene);  
        stage.show();  
    }  
    public static void main(String args[]){
```

```
        launch(args);
    }
}
```

o/p:



TreeView

One of JavaFX's most powerful controls is the **TreeView**. It presents a hierarchical view of data in a tree-like format. In this context, the term *hierarchical* means some items are subordinate to others.

In a **TreeView**, branches can be expanded or collapsed on demand by the user.

This allows hierarchical data to be presented in a compact, yet expandable form. One very useful feature of **TreeView** is that it automatically provides scroll bars when the size of the tree exceeds the dimensions of the view. Although a fully collapsed tree might be quite small, its expanded form may be quite large. By automatically adding scroll bars as needed, **TreeView** lets you use a smaller space than would ordinarily be possible.

TreeView is a generic class that is defined like this:

```
class TreeView<T>
```

Here, T specifies the type of value held by an item in the tree. Often, this will be of type **String**.

TreeView defines two constructors. This is the one we will use:

TreeView(TreeItem<T> rootNode)

Here, `rootNode` specifies the root of the tree. Because all nodes descend from the root, it is the only one that needs to be passed to `TreeView`.

The items that form the tree are objects of type `TreeItem`. `TreeItems` are not general-purpose objects. They can be used in a `TreeView`, but not as stand-alone controls.

`TreeItem` is a generic class, as shown here:

class TreeItem<T>

Here, `T` specifies the type of value held by the `TreeItem`.

Before you can use a `TreeView`, you must construct the tree that it will display.

To do this, you must first create the root. Next, add other nodes to that root. You do this by calling either `add()` or `addAll()` on the list returned by `getChildren()`.

These other nodes can be leaf nodes or subtrees.

Ex.

```
import javafx.application.Application;
import javafx.scene.Scene;

import javafx.scene.control.TreeItem;
import javafx.scene.control.TreeView;

import javafx.scene.layout.StackPane;
import javafx.stage.Stage;

public class TreeViewDemoJFX extends Application {

    public void start(Stage primaryStage) {

        String s1="Fruit";

        String s11="Mango";

        String s12="Guava";

        // Root Item
        TreeItem<String> rootItem = new TreeItem<String>(s1);
        rootItem.setExpanded(true);

        // Mango: child Item
        TreeItem<String> itemmango = new TreeItem<String>(s11);

        // Guava: child Item
```

```
TreeItem<String> itemguava = new TreeItem<String>(s12);

// Add to Root
rootItem.getChildren().addAll(itemmango, itemguava);

TreeView<String> tree = new TreeView<String>(rootItem);

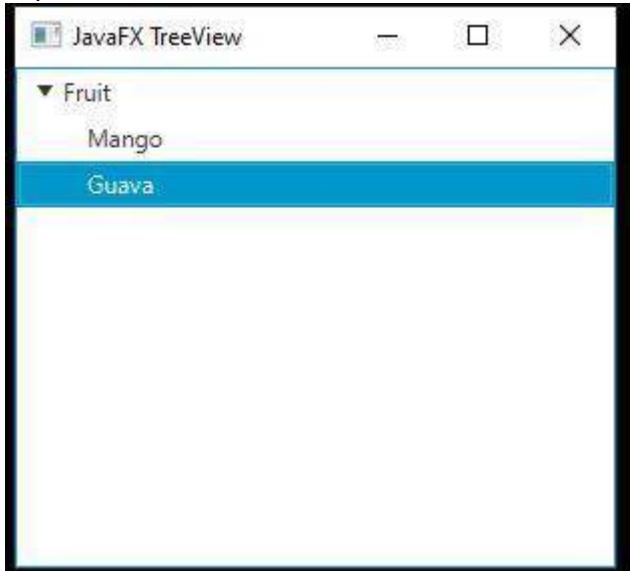
StackPane root = new StackPane();

root.getChildren().add(tree);

primaryStage.setTitle("JavaFX TreeView");
primaryStage.setScene(new Scene(root, 300, 250));
primaryStage.show();
}

public static void main(String[] args) {
    launch(args);
}
}
```

o/p:



ToggleButton

ToggleButton is a control with a Boolean indicating whether it is selected.

Typically a `ToggleButton` is rendered similarly to a `Button`. However, they are two different types of Controls. A `Button` is a "command" button which invokes a function when clicked. A `ToggleButton` on the other hand is simply a control with a Boolean indicating whether it has been selected.

`ToggleButton` can also be placed in groups. By default, a `ToggleButton` is not in a group. When in groups, only one `ToggleButton` at a time within that group can be selected. To put two `ToggleButtons` in the same group, simply assign them both the same value for [ToggleGroup](#).

Note: If a `ToggleButton` is selected, clicking on it will cause it to become unselected. With `RadioButton`, clicking on the selected button in the group will have no effect.

Ex.

```
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.control.Label;

import javafx.scene.control.ToggleButton;
import javafx.scene.control.ToggleGroup;

import javafx.scene.layout.HBox;
import javafx.stage.Stage;

public class ToggleButtonExamplefx extends Application {

    public void start(Stage stage)
    {

        // Hbox layout
        HBox root = new HBox();

        // Gender
        root.getChildren().add(new Label("Your gender:"));

        // Creating a ToggleGroup
        ToggleGroup group = new ToggleGroup();

        // Creating new Toggle buttons.
        ToggleButton maleButton = new ToggleButton("Male");
        ToggleButton femaleButton = new ToggleButton("Female");

        // Set toggle group
        // In a group, maximum only one button is selected
```

```
maleButton.setToggleGroup(group);
femaleButton.setToggleGroup(group);

root.getChildren().addAll(maleButton, femaleButton);

// create the scene
Scene scene = new Scene(root, 450, 300);

stage.setTitle("Toggle Button");
stage.setScene(scene);
stage.show();
}

// Main Method
public static void main(String[] args)
{
    launch(args);
}
}
```

o/p:



Event Basics through event caused by a button :

The base class for JavaFX events is the [Event class](#), which is packaged in [javafx.event](#). Event inherits [java.util.EventObject](#), which means that JavaFX events share the same basic functionality as other Java events.

Several subclasses of [Event](#) are defined. The one that we will use here is [ActionEvent](#). It handles action events generated by a [button](#).

When a button is pressed, an [ActionEvent](#) is generated. [ActionEvent](#) is packaged in [javafx.event](#).

To handle an event, you must first register **the handler that acts as a listener for the event**. When the event occurs, the listener is called. It must then respond to the event and return.

You can **register a listener** for this event by using **setOnAction()**, which has this general form:

final void setOnAction(EventHandler<ActionEvent> handler)

Here, **handler** is the handler being registered. As mentioned, often you will use **an anonymous inner class for the handler**.

How do we get the listener (**handler**)?

Events are handled by implementing the **EventHandler interface**, which is also in `javafx.event` and we do it through an **anonymous inner class**. We get the **handler through the phase of** implementing the **EventHandler interface**.

It is a generic interface with the following form:

interface EventHandler<T extends Event>

Here, **T** specifies the type of event that the handler will handle.

It has a method, called **handle()**, which receives the event object as a parameter. It is shown here:

void handle(T eventObj)

Here, **eventObj** is the event that was generated.

Ex.

```
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.layout.*;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.scene.control.Label;
import javafx.stage.Stage;
public class button_1 extends Application {

    // launch the application
    public void start(Stage s)
    {
        // set title for the stage
```

```
s.setTitle("creating buttons");

// create a button
Button b = new Button("button");

// create a stack pane
TilePane r = new TilePane();

// create a label
Label l = new Label("button not selected");

//Handler for events of a specific class / type.
// anonymous inner class for implementing EventHandler interface and the handler
EventHandler<ActionEvent> event = new EventHandler<ActionEvent>() {

    // Interface EventHandler<T extends Event> has "void handle(T event)"
    public void handle(ActionEvent e) // Invoked when a specific event of the type for which this
    //handler is registered happens.
    {
        l.setText(" button selected ");
    }
};

// when button is pressed and registering a listener (or, handler)
b.setOnAction(event); // here "event" is the handler

// add button
r.getChildren().add(b);
r.getChildren().add(l);

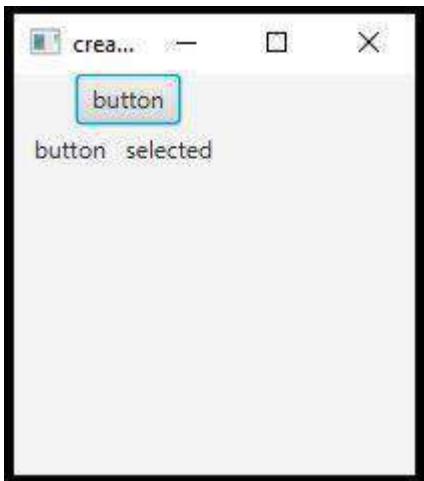
// create a scene
Scene sc = new Scene(r, 200, 200);

// set the scene
s.setScene(sc);

s.show();
}

public static void main(String args[])
{
    // launch the application
    launch(args);
}
}
```

o/p:



Note:

It is sometimes useful to know the source of an event. This is especially true if you are using one handler to handle events from different sources. You can obtain the source of the event by calling `getSource()`, which is inherited from `java.util.EventObject`. It is shown here:

```
Object getSource()
```

Note:

In JavaFX, events are processed via an **event dispatch chain**.

When an event is generated, it is passed to the root node of the chain. The event is then passed down the chain to the target of the event. After the target node processes the event, the event is passed back up the chain, thus allowing parent nodes a chance to process the event, if necessary. **This is called event bubbling**.

It is possible for a node in the chain to consume an event, which prevents it from being further processed.

An application can also **implement an event filter**, which can be used to manage events. A filter is **added to a node** by calling `addEventFilter()`, which is defined by **Node**. **A filter can consume an event, thus preventing further processing**.

Drawing Directly on a Canvas

JavaFX's graphics methods are found in the **GraphicsContext class**, which is part of

javafx.scene.canvas.

These methods can be used to draw directly on the surface of a canvas, which is encapsulated by the **Canvas** class in `javafx.scene.canvas`.

Before you can draw on a canvas, you must perform two steps.

First, you must create a **Canvas** instance.

Second, you must obtain a **GraphicsContext** object that refers to that canvas. You can then use the **GraphicsContext** to draw output on the canvas.

The **Canvas** class is derived from **Node**; thus it can be used as a node in a scene graph.

Canvas defines two constructors. One is the default constructor, and the other is the one shown here:

Canvas(double width, double height)

Here, *width* and *height* specify the dimensions of the canvas.

To obtain a **GraphicsContext** that refers to a canvas, call **getGraphicsContext2D()**. Here is its general form:

GraphicsContext getGraphicsContext2D()

The graphics context for the canvas is returned.

GraphicsContext defines a large number of methods that draw shapes, text, and images, and support effects and transforms.

If sophisticated graphics programming is in your future, you will definitely want to study its capabilities closely. For our purposes, we will use only a few of its methods, but they will give you a sense of its power. They are described here.

You can draw a line using **strokeLine()**, shown here:

void strokeLine(double startX, double startY, double endX, double endY)

It draws a line from *startX*,*startY* to *endX*,*endY*, using the current stroke, which can be a solid color or some more complex style.

To draw a rectangle, use either **strokeRect()** or **fillRect()**, shown here:

void strokeRect(double topX, double topY, double width, double height)

void fillRect(double topX, double topY, double width, double height)

The upper-left corner of the rectangle is at *topX*,*topY*. The *width* and *height* parameters specify its width and height. The **strokeRect()** method draws the outline of a rectangle using the current stroke, and **fillRect()** fills the rectangle with the current fill. The current fill can be as simple as a solid color or something more complex.

To draw an ellipse, use either **strokeOval()** or **fillOval()**, shown next:

void strokeOval(double topX, double topY, double width, double height)

void fillOval(double topX, double topY, double width, double height)

The upper-left corner of the rectangle that bounds the ellipse is at *topX,topY*. The *width* and *height* parameters specify its width and height. The **strokeOval()** method draws the outline of an ellipse using the current stroke, and **fillOval()** fills the oval with the current fill.

To draw a circle, pass the same value for *width* and *height*.

You can draw text on a canvas by using the **strokeText()** and **fillText()** methods.

We will use this version of **fillText()**:

void fillText(String str, double topX, double topY)

It displays *str* starting at the location specified by *topX,topY*, filling the text with the current fill.

Ex.

```
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.layout.*;
import javafx.stage.Stage;
import javafx.scene.canvas.*;
import javafx.scene.paint.Color;

public class CanvasJfx extends Application {

    public void start(Stage stage)
    {
        // set title for the stage
        stage.setTitle("creating canvas");

        // create a canvas
        Canvas canvas = new Canvas(800.0f, 800.0f);

        // graphics context
        GraphicsContext graphics_context = canvas.getGraphicsContext2D();
```

```
// set fill for rectangle
graphics_context.setFill(Color.RED);
graphics_context.fillRect(20, 20, 400, 300);

// set fill for oval
graphics_context.setFill(Color.YELLOW);
graphics_context.fillOval(30, 30, 200, 200);

//draw a line
graphics_context.strokeLine(20, 20, 70, 70);

//draw text on a canvas
graphics_context.strokeText("Dealing with JavaFX canvas", 50.0f, 50.0f);

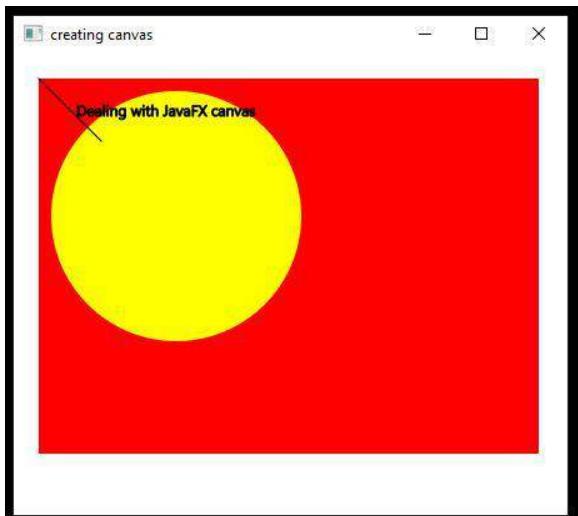
BorderPane bp= new BorderPane();
bp.getChildren().add(canvas);

Scene scene = new Scene(bp, 800, 800);
// set the scene
stage.setScene(scene);

stage.show();
}

// Main Method
public static void main(String args[])
{
    // launch the application
    launch(args);
}
}

o/p:
```



Graphical User Interfaces

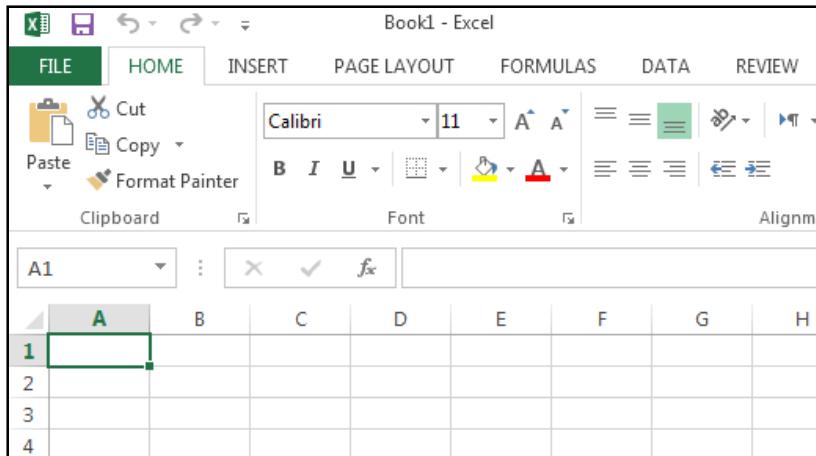
JavaFX GUI Basics, Event Programming and GUI UI Controls

CSE260, Computer Science B: Honors

Stony Brook University

<http://www.cs.stonybrook.edu/~cse260>

GUI Examples

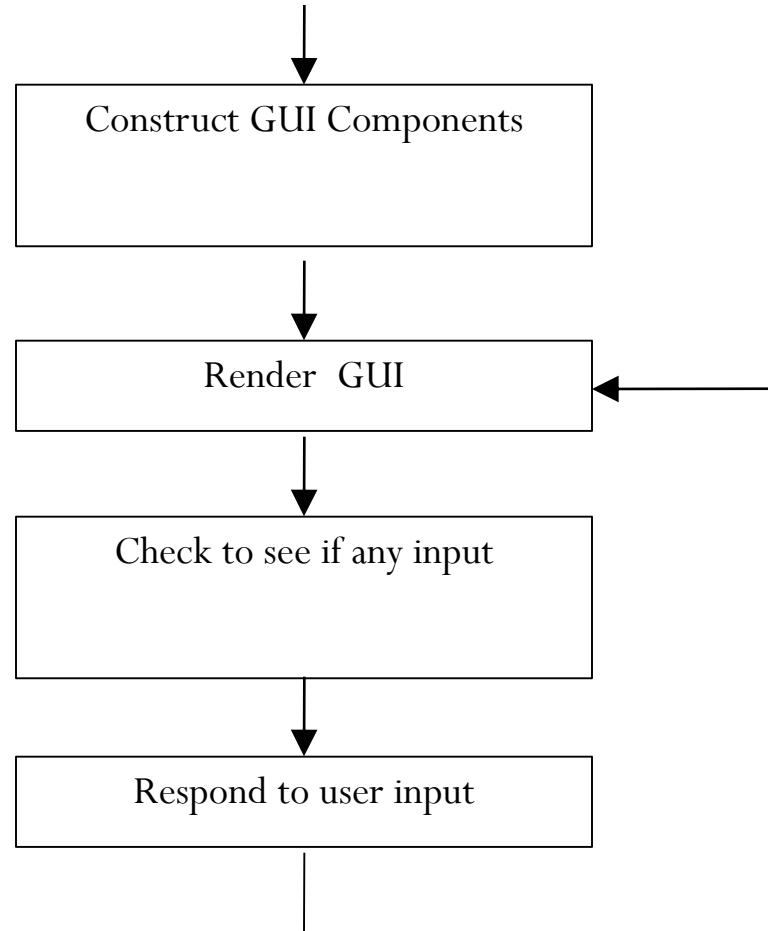


GUI

- Graphical User Interface (GUI)
 - provides user-friendly human interaction
- Building Java GUIs require use of frameworks:
 - AWT
 - Swing
 - JavaFX (part of Java since JSE 8, 2014) includes:
 - GUI components
 - Event Programming
 - Graphics

How do GUIs work?

- They loop and respond to events



Example: a mouse click on a button

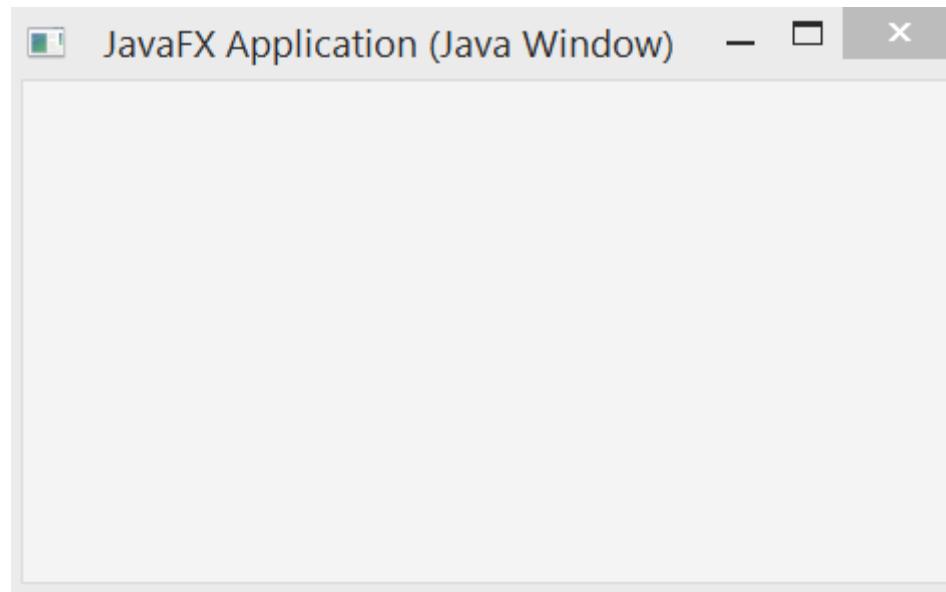
- Operating System recognizes mouse click
 - determines which window it was inside
 - notifies that program
- Program runs in loop
 - checks input buffer filled by OS
 - if it finds a mouse click:
 - determines which component in the program
 - if the click was on a relevant component
 - respond appropriately according to handler

GUI Look vs. Behavior

- Look
 - physical appearance
 - custom component design
 - containment
 - layout management
- Behavior
 - interactivity
 - event programmed response

What does a GUI framework do for you?

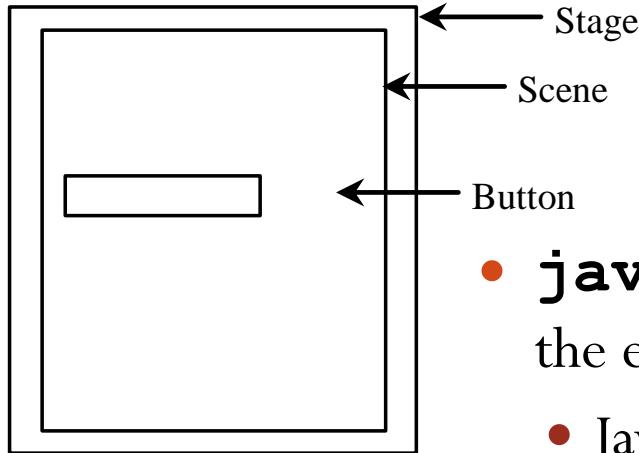
- Provides ready made visible, interactive, customizable components
 - you wouldn't want to have to code your own window



JavaFX vs Swing and AWT

- Swing and AWT are replaced by the JavaFX platform for developing rich Internet applications in JDK8 (2014)
- History:
 - When Java was introduced (1996), the GUI classes were bundled in a library known as the Abstract Windows Toolkit (AWT)
 - AWT was prone to platform-specific bugs
 - AWT was fine for developing simple graphical user interfaces, but not for developing comprehensive GUI projects
 - The AWT user-interface components were replaced by a more robust, versatile, and flexible library known as Swing components (1997)
 - **Swing components are painted directly on canvases using Java code**
 - Swing components depend less on the target platform and use less of the native GUI resource
 - **With the release of Java 8, Swing is replaced by a completely new GUI platform: JavaFX**

Basic Structure of JavaFX



- **javafx.application.Application** is the entry point for JavaFX applications
 - JavaFX creates an application thread for running the application start method, processing input events, and running animation timelines.
 - Override the start(Stage) method!
- **javafx.stage.Stage** is the top level JavaFX container.
 - The primary Stage is constructed by the platform.
- **javafx.scene.Scene** class is the container for all content in a scene graph.
- **javafx.scene.Node** is the base class for scene graph nodes.

```

import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.control.Button;

public class MyFirstJavaFX extends Application {
    @Override // Override the start method in the Application class
    public void start(Stage primaryStage) {
        // Create a button and place it in the scene
        Button btOK = new Button("OK");
        Scene scene = new Scene(btOK, 200, 250);
        primaryStage.setTitle("MyJavaFX"); // Set the stage title
        primaryStage.setScene(scene); // Place the scene in the stage
        primaryStage.show(); // Display the stage
    }

    /**
     * The main method is only needed for the IDE with limited
     * JavaFX support. Not needed for running from the command line.
     */
    public static void main(String[] args) {
        launch(args);
    }
}

```



```

// Multiple stages can be added beside the primaryStage
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.control.Button;

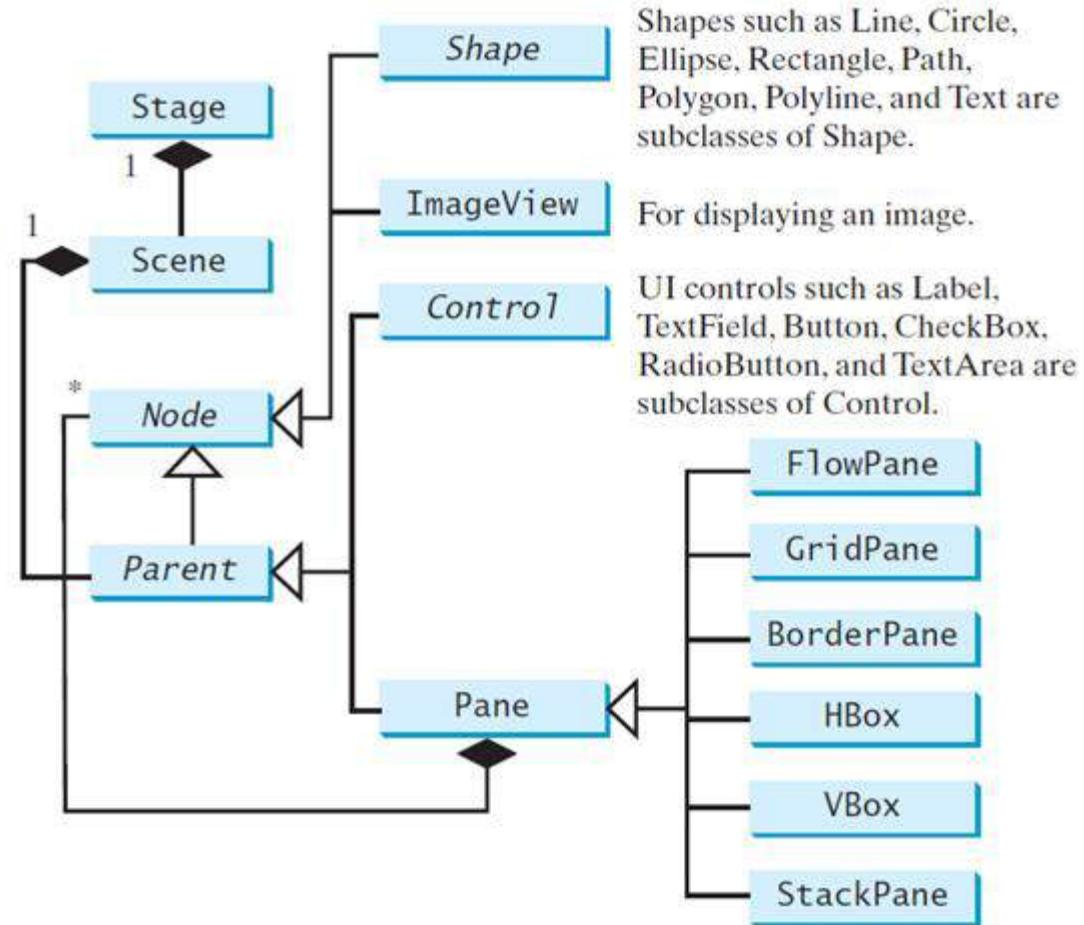
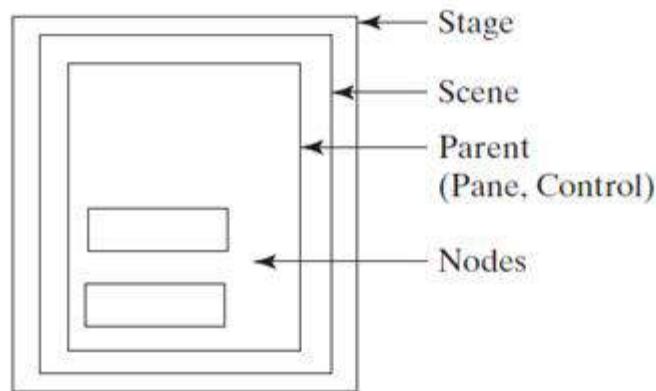
public class MultipleStageDemo extends Application {
    @Override // Override the start method in the Application class
    public void start(Stage primaryStage) {
        // Create a scene and place a button in the scene
        Scene scene = new Scene(new Button("OK"), 200, 250);
        primaryStage.setTitle("MyJavaFX"); // Set the stage title
        primaryStage.setScene(scene); // Place the scene in the stage
        primaryStage.show(); // Display the stage
        Stage stage = new Stage(); // Create a new stage
        stage.setTitle("Second Stage"); // Set the stage title
        // Set a scene with a button in the stage
        stage.setScene(new Scene(new Button("New Stage"), 100, 100));
        stage.show(); // Display the stage
    }

    public static void main(String[] args) {
        launch(args);
    }
}

```



Panes, UI Controls, and Shapes



Shapes such as Line, Circle, Ellipse, Rectangle, Path, Polygon, Polyline, and Text are subclasses of Shape.

For displaying an image.

UI controls such as Label, TextField, Button, CheckBox, RadioButton, and TextArea are subclasses of Control.

```
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.layout.StackPane;
import javafx.scene.control.Button;

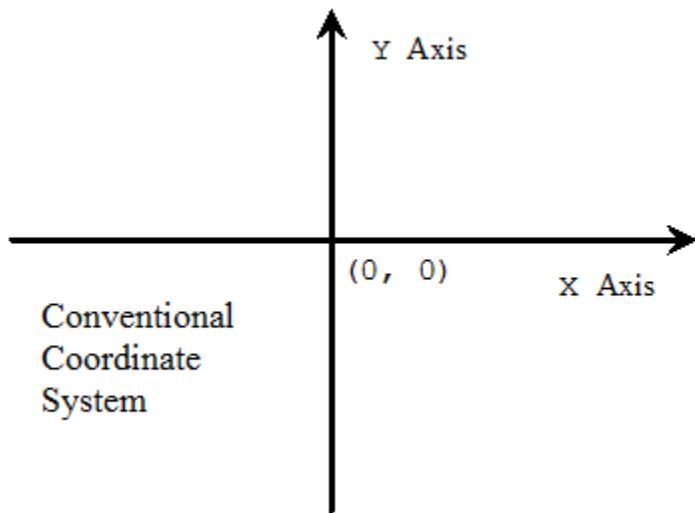
public class ButtonInPane extends Application {

    @Override // Override the start method in the Application class
    public void start(Stage primaryStage) {
        // Create a scene and place a button in the scene
        StackPane pane = new StackPane();
        pane.getChildren().add(new Button("OK"));
        Scene scene = new Scene(pane, 200, 50);
        primaryStage.setTitle("Button in a pane"); // Set the stage title
        primaryStage.setScene(scene); // Place the scene in the stage
        primaryStage.show(); // Display the stage
    }

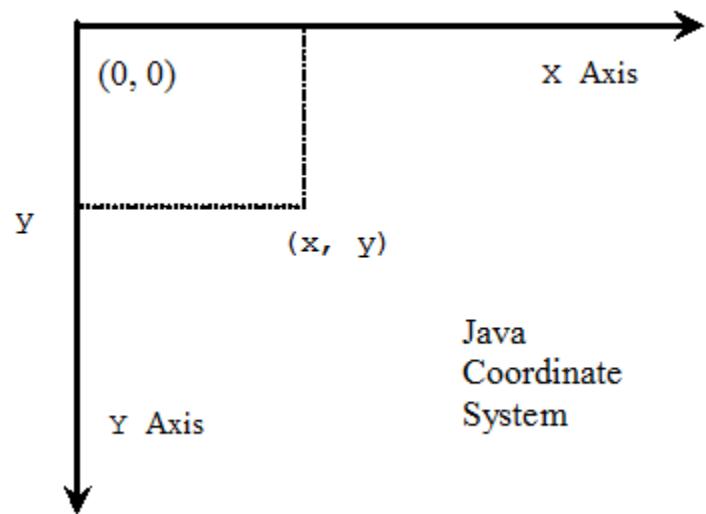
    public static void main(String[] args) {
        launch(args);
    }
}
```



Display a Shape



- Programming Coordinate Systems start from the left-upper corner



```

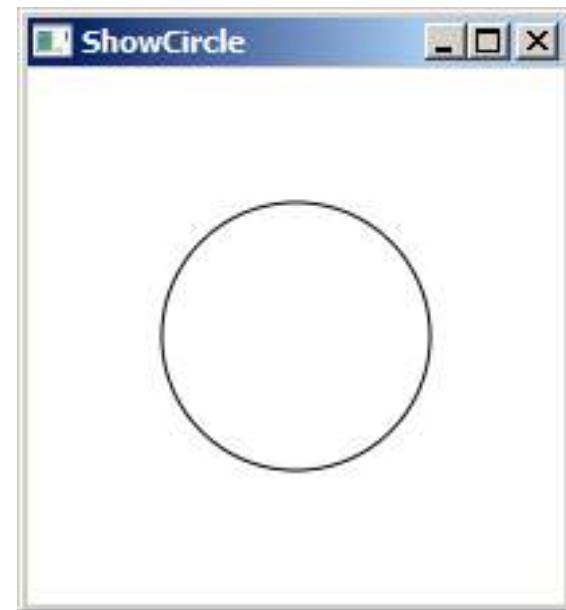
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.layout.Pane;
import javafx.scene.shape.Circle;
import javafx.scene.paint.Color;

public class ShowCircle extends Application {
    @Override // Override the start method in the Application class
    public void start(Stage primaryStage) {
        // Create a circle and set its properties
        Circle circle = new Circle();
        circle.setCenterX(100);
        circle.setCenterY(100);
        circle.setRadius(50);
        circle.setStroke(Color.BLACK);
        circle.setFill(null);
        // Create a pane to hold the circle
        Pane pane = new Pane();
        pane.getChildren().add(circle);
        // Create a scene and place it in the stage
        Scene scene = new Scene(pane, 200, 200);
        primaryStage.setTitle("ShowCircle"); // Set the stage title
        primaryStage.setScene(scene); // Place the scene in the stage
        primaryStage.show(); // Display the stage
    }

    /**
     * The main method is only needed for the IDE with limited
     * JavaFX support. Not needed for running from the command line.
     */
    public static void main(String[] args) {
        launch(args);
    }
}

```

Circle in a Pane



Binding Properties

- JavaFX introduces a new concept called ***binding property*** that enables a target object to be bound to a source object.
 - If the value in the source object changes, the target property is also changed automatically.
 - The target object is simply called a binding object or a binding property.
- Resizing the window in the previous example would cover the object:



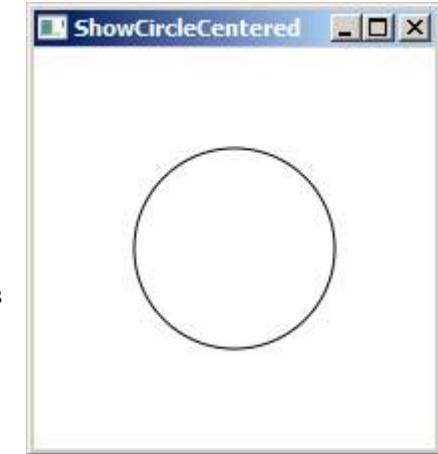
```

import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.layout.Pane;
import javafx.scene.shape.Circle;
import javafx.scene.paint.Color;

public class ShowCircleCentered extends Application {
    @Override // Override the start method in the Application class
    public void start(Stage primaryStage) {
        // Create a pane to hold the circle
        Pane pane = new Pane();
        // Create a circle and set its properties
        Circle circle = new Circle();
        circle.centerXProperty().bind(pane.widthProperty().divide(2));
        circle.centerYProperty().bind(pane.heightProperty().divide(2));
        circle.setRadius(50);
        circle.setStroke(Color.BLACK);
        circle.setFill(Color.WHITE);
        pane.getChildren().add(circle); // Add circle to the pane
        // Create a scene and place it in the stage
        Scene scene = new Scene(pane, 200, 200);
        primaryStage.setTitle("ShowCircleCentered"); // Set the stage title
        primaryStage.setScene(scene); // Place the scene in the stage
        primaryStage.show(); // Display the stage
    }

    /**
     * The main method is only needed for the IDE with limited
     * JavaFX support. Not needed for running from the command line.
     */
    public static void main(String[] args) {
        launch(args);
    }
}

```



JavaFX Beans and Binding

- Changes made to one object will automatically be reflected in another object
 - A graphical user interface automatically keeps its display synchronized with the application's underlying data: a binding observes its list of dependencies for changes, and then updates itself automatically after a change has been detected.

```
import javafx.beans.property.DoubleProperty;
import javafx.beans.property.SimpleDoubleProperty;

public class BindingDemo {
    public static void main(String[] args) {
        DoubleProperty d1 = new SimpleDoubleProperty(1);
        DoubleProperty d2 = new SimpleDoubleProperty(2);
        d1.bind(d2);
        System.out.println("d1 is " + d1.getValue()
            + " and d2 is " + d2.getValue());

        d2.setValue(70.2);
        System.out.println("d1 is " + d1.getValue()
            + " and d2 is " + d2.getValue());
    }
}
```

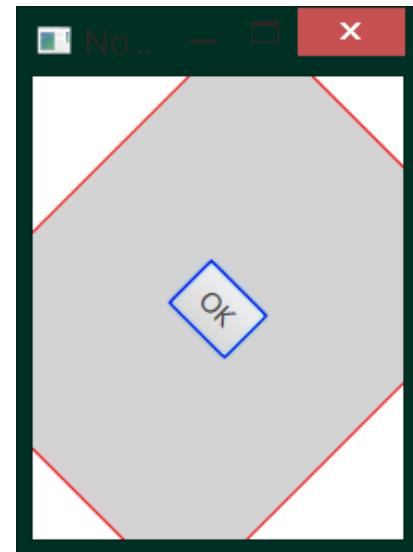
Output:

d1 is 2.0 and d2 is 2.0

d1 is 70.2 and d2 is 70.2

JavaFX CSS style and Node rotation

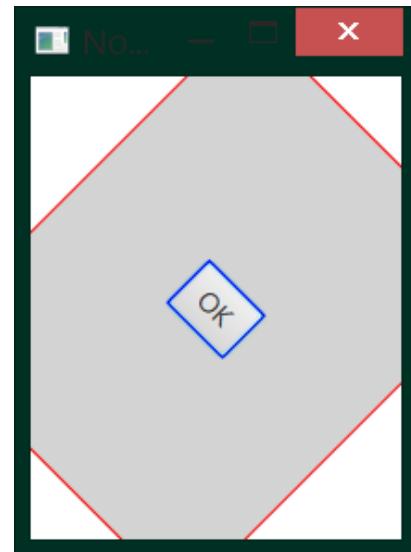
```
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.layout.StackPane;
import javafx.scene.control.Button;
public class NodeStyleRotateDemo extends Application {
    @Override
    public void start(Stage primaryStage) {
        StackPane pane = new StackPane();
        Button btOK = new Button("OK");
        btOK.setStyle("-fx-border-color: blue;");
        pane.getChildren().add(btOK);
        pane.setRotate(45);
        pane.setStyle("-fx-border-color: red; -fx-background-color: lightgray;");
        Scene scene = new Scene(pane, 200, 250);
        primaryStage.setTitle("NodeStyleRotateDemo"); // Set the stage title
        primaryStage.setScene(scene); // Place the scene in the stage
        primaryStage.show(); // Display the stage
    }
    /**
     * The main method is only needed for the IDE with limited
     * JavaFX support. Not needed for running from the command line.
     */
    public static void main(String[] args) {
        launch(args);
    }
}
```



JavaFX CSS style and Node rotation

```
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.layout.StackPane;
import javafx.scene.control.Button;

public class NodeStyleRotateDemo extends Application {
    @Override
    public void start(Stage primaryStage) {
        StackPane pane = new StackPane();
```



```
/* The StackPane layout pane places all of the nodes within
a single stack with each new node added on top of the
previous node. This layout model provides an easy way to
overlay text on a shape or image and to overlap common
shapes to create a complex shape. */
```

JavaFX External CSS style file

```
// Example to load and use a CSS style file in a scene
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.layout.BorderPane;

public class ExternalCSSFile extends Application {
    @Override
    public void start(Stage primaryStage) {
        try {
            BorderPane root = new BorderPane();
            Scene scene = new Scene(root,400,400);
            scene.getStylesheets().add(getClass()
                .getResource("application.css").toExternalForm());
            primaryStage.setScene(scene);
            primaryStage.show();
        } catch(Exception e) {
            e.printStackTrace();
        }
    }
    public static void main(String[] args) {
        launch(args);
    }
}
```

Helper classes: The Color Class

`javafx.scene.paint.Color`

```
-red: double  
-green: double  
-blue: double  
-opacity: double  
  
+Color(r: double, g: double, b:  
       double, opacity: double)  
+brighter(): Color  
+darker(): Color  
+color(r: double, g: double, b:  
       double): Color  
+color(r: double, g: double, b:  
       double, opacity: double): Color  
+rgb(r: int, g: int, b: int):  
    Color  
+rgb(r: int, g: int, b: int,  
     opacity: double): Color
```

The getter methods for property values are provided in the class, but omitted in the UML diagram for brevity.

- The red value of this Color (between 0.0 and 1.0).
- The green value of this Color (between 0.0 and 1.0).
- The blue value of this Color (between 0.0 and 1.0).
- The opacity of this Color (between 0.0 and 1.0).
- Creates a Color with the specified red, green, blue, and opacity values.
- Creates a Color that is a brighter version of this Color.
- Creates a Color that is a darker version of this Color.
- Creates an opaque Color with the specified red, green, and blue values.
- Creates a Color with the specified red, green, blue, and opacity values.
- Creates a Color with the specified red, green, and blue values in the range from 0 to 255.
- Creates a Color with the specified red, green, and blue values in the range from 0 to 255 and a given opacity.

Helper classes: The Font Class

`javafx.scene.text.Font`

```
-size: double  
-name: String  
-family: String  
  
+Font(size: double)  
+Font(name: String, size:  
      double)  
+font(name: String, size:  
      double)  
+font(name: String, w:  
      FontWeight, size: double)  
+font(name: String, w: FontWeight,  
      p: FontPosture, size: double)  
+getFamilies(): List<String>  
+getFontNames(): List<String>
```

The getter methods for property values are provided in the class, but omitted in the UML diagram for brevity.

The size of this font.

The name of this font.

The family of this font.

Creates a `Font` with the specified size.

Creates a `Font` with the specified full font name and size.

Creates a `Font` with the specified name and size.

Creates a `Font` with the specified name, weight, and size.

Creates a `Font` with the specified name, weight, posture, and size.

Returns a list of font family names.

Returns a list of full font names including family and weight.

The Image and ImageView Classes

`javafx.scene.image.Image`

-error: ReadOnlyBooleanProperty
-height: ReadOnlyBooleanProperty
-width: ReadOnlyBooleanProperty
-progress: ReadOnlyBooleanProperty

+Image(filenameOrURL: String)

The getter methods for property values are provided in the class, but omitted in the UML diagram for brevity.

Indicates whether the image is loaded correctly?
The height of the image.
The width of the image.
The approximate percentage of image's loading that is completed.

Creates an `Image` with contents loaded from a file or a URL.

`javafx.scene.image.ImageView`

-fitHeight: DoubleProperty
-fitWidth: DoubleProperty
-x: DoubleProperty
-y: DoubleProperty
-image: ObjectProperty<Image>

+ImageView()
+ImageView(image: Image)
+ImageView(filenameOrURL: String)

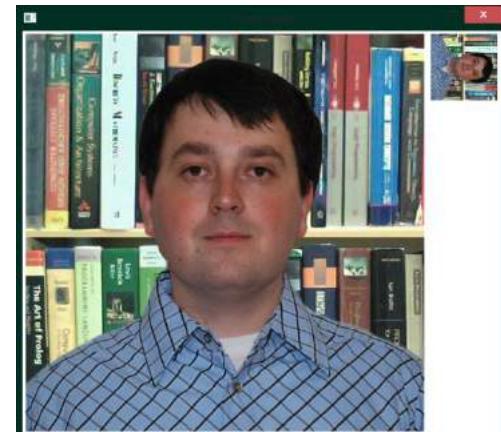
The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.

The height of the bounding box within which the image is resized to fit.
The width of the bounding box within which the image is resized to fit.
The x-coordinate of the ImageView origin.
The y-coordinate of the ImageView origin.
The image to be displayed in the image view.

Creates an `ImageView`.
Creates an `ImageView` with the specified image.
Creates an `ImageView` with image loaded from the specified file or URL.

```
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.layout.Pane;
import javafx.scene.layout.HBox;
import javafx.scene.image.Image;
import javafx.scene.image.ImageView;
import javafx.geometry.Insets;

public class ShowImage extends Application {
    @Override
    public void start(Stage primaryStage) {
        // Create a pane to hold the image views
        Pane pane = new HBox(10);
        pane.setPadding(new Insets(5, 5, 5, 5));
        Image image = new Image("paul.jpg");
        pane.getChildren().add(new ImageView(image));
        ImageView imageView2 = new ImageView(image);
        imageView2.setFitHeight(100);
        imageView2.setFitWidth(100);
        imageView2.setRotate(90);
        pane.getChildren().add(imageView2);
        Scene scene = new Scene(pane);
        primaryStage.setTitle("ShowImage");
        primaryStage.setScene(scene);
        primaryStage.show();
    }
    public static void main(String[] args) {
        launch(args);
    }
}
```



Layout Panes

- JavaFX provides many **types of panes for organizing nodes in a container.**

<i>Class</i>	<i>Description</i>
Pane	Base class for layout panes. It contains the <code>getChildren()</code> method for returning a list of nodes in the pane.
StackPane	Places the nodes on top of each other in the center of the pane.
FlowPane	Places the nodes row-by-row horizontally or column-by-column vertically.
GridPane	Places the nodes in the cells in a two-dimensional grid.
BorderPane	Places the nodes in the top, right, bottom, left, and center regions.
HBox	Places the nodes in a single row.
VBox	Places the nodes in a single column.

FlowPane

javafx.scene.layout.FlowPane

```
-alignment: ObjectProperty<Pos>
-orientation:
    ObjectProperty<Orientation>
-hgap: DoubleProperty
-vgap: DoubleProperty

+FlowPane()
+FlowPane(hgap: double, vgap:
    double)
+FlowPane(orientation:
    ObjectProperty<Orientation>)
+FlowPane(orientation:
    ObjectProperty<Orientation>,
    hgap: double, vgap: double)
```

The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.

The overall alignment of the content in this pane (default: Pos.LEFT).
The orientation in this pane (default: Orientation.HORIZONTAL).

The horizontal gap between the nodes (default: 0).
The vertical gap between the nodes (default: 0).

Creates a default FlowPane.

Creates a FlowPane with a specified horizontal and vertical gap.

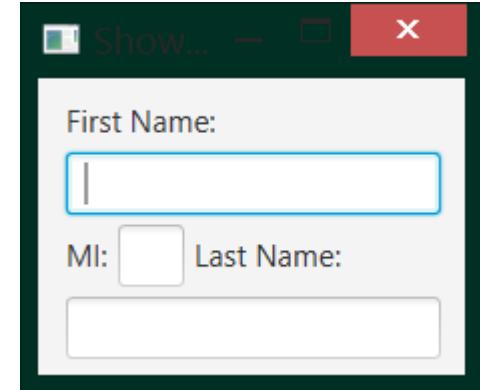
Creates a FlowPane with a specified orientation.

Creates a FlowPane with a specified orientation, horizontal gap and vertical gap.

```

import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.layout.FlowPane;
import javafx.scene.control.Label;
import javafx.scene.control.TextField;
import javafx.geometry.Insets;
public class ShowFlowPane extends Application {
    @Override
    public void start(Stage primaryStage) {
        FlowPane pane = new FlowPane();
        pane.setPadding(new Insets(11, 12, 13, 14));
        pane.setHgap(5);
        pane.setVgap(5);
        // Place nodes in the pane
        pane.getChildren().addAll(new Label("First Name:") ,
            new TextField(), new Label("MI:") );
        TextField tfMi = new TextField();
        tfMi.setPrefColumnCount(1);
        pane.getChildren().addAll(tfMi, new Label("Last Name:") ,
            new TextField());
        // Create a scene and place it in the stage
        Scene scene = new Scene(pane, 210, 150);
        primaryStage.setTitle("ShowFlowPane");
        primaryStage.setScene(scene); // Place the scene in the stage
        primaryStage.show(); // Display the stage
    }
    public static void main(String[] args) {
        launch(args);
    }
}

```



GridPane

`javafx.scene.layout.GridPane`

```
-alignment: ObjectProperty<Pos>
-gridLinesVisible:
    BooleanProperty
-hgap: DoubleProperty
-vgap: DoubleProperty

+GridPane()
+add(child: Node, columnIndex:
      int, rowIndex: int): void
+addColumn(columnIndex: int,
            children: Node...): void
+addRow(rowIndex: int,
        children: Node...): void
+getColumnIndex(child: Node):
    int
+setColumnIndex(child: Node,
                columnIndex: int): void
+getRowIndex(child: Node): int
+setRowIndex(child: Node,
             rowIndex: int): void
+setHalignment(child: Node,
               value: HPos): void
+setValignment(child: Node,
               value: VPos): void
```

The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.

The overall alignment of the content in this pane (default: Pos.LEFT).
Is the grid line visible? (default: false)

The horizontal gap between the nodes (default: 0).
The vertical gap between the nodes (default: 0).

Creates a GridPane.

Adds a node to the specified column and row.

Adds multiple nodes to the specified column.

Adds multiple nodes to the specified row.

Returns the column index for the specified node.

Sets a node to a new column. This method repositions the node.

Returns the row index for the specified node.

Sets a node to a new row. This method repositions the node.

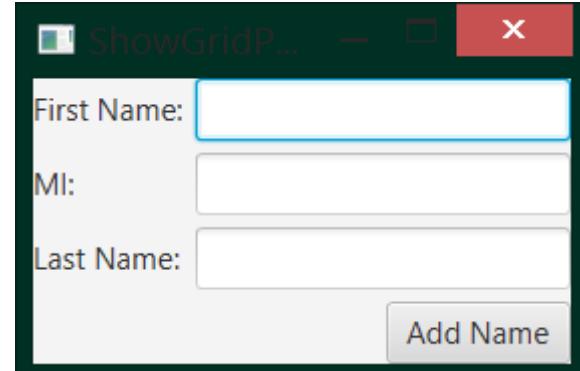
Sets the horizontal alignment for the child in the cell.

Sets the vertical alignment for the child in the cell.

```

import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.layout.GridPane;
import javafx.scene.control.*;
import javafx.geometry.*;
public class ShowGridPane extends Application {
    @Override
    public void start(Stage primaryStage) {
        // Create a pane and set its properties
        GridPane pane = new GridPane();
        pane.setAlignment(Pos.CENTER);
        pane.setHgap(5.5);
        pane.setVgap(5.5);
        // Place nodes in the pane at positions column, row
        pane.add(new Label("First Name:"), 0, 0);
        pane.add(new TextField(), 1, 0);
        pane.add(new Label("MI:"), 0, 1);
        pane.add(new TextField(), 1, 1);
        pane.add(new Label("Last Name:"), 0, 2);
        pane.add(new TextField(), 1, 2);
        Button btAdd = new Button("Add Name");
        pane.add(btAdd, 1, 3);
        GridPane.setHalignment(btAdd, HPos.RIGHT);
        // Create a scene and place it in the stage
        Scene scene = new Scene(pane);
        primaryStage.setTitle("ShowGridPane");
        primaryStage.setScene(scene); primaryStage.show(); }
    public static void main(String[] args) {
        launch(args);
    }
}

```



BorderPane

`javafx.scene.layout.BorderPane`

```
-top: ObjectProperty<Node>
-right: ObjectProperty<Node>
-bottom: ObjectProperty<Node>
-left: ObjectProperty<Node>
-center: ObjectProperty<Node>

+BorderPane()
+setAlignment(child: Node, pos: Pos)
```

The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.

The node placed in the top region (default: null).
The node placed in the right region (default: null).
The node placed in the bottom region (default: null).
The node placed in the left region (default: null).
The node placed in the center region (default: null).

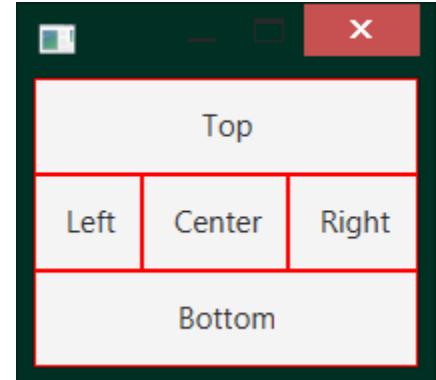
Creates a BorderPane.

Sets the alignment of the node in the BorderPane.

```

import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.layout.BorderPane;
import javafx.scene.layout.StackPane;
import javafx.scene.control.Label;
import javafx.geometry.Insets;
public class ShowBorderPane extends Application {
    @Override
    public void start(Stage primaryStage) {
        BorderPane pane = new BorderPane();
        pane.setTop(new CustomPane("Top"));
        pane.setRight(new CustomPane("Right"));
        pane.setBottom(new CustomPane("Bottom"));
        pane.setLeft(new CustomPane("Left"));
        pane.setCenter(new CustomPane("Center"));
        Scene scene = new Scene(pane);
        primaryStage.setScene(scene); primaryStage.show();
    }
    public static void main(String[] args) {
        launch(args);
    }
}
class CustomPane extends StackPane {
    public CustomPane(String title) {
        getChildren().add(new Label(title));
        setStyle("-fx-border-color: red");
        setPadding(new Insets(11.5, 12.5, 13.5, 14.5));
    }
}

```



Hbox and VBox

javafx.scene.layout.HBox

```
-alignment: ObjectProperty<Pos>  
-fillHeight: BooleanProperty  
-spacing: DoubleProperty  
  
+HBox()  
+HBox(spacing: double)  
+setMargin(node: Node, value: Insets): void
```

The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.

The overall alignment of the children in the box (default: Pos.TOP_LEFT).
Is resizable children fill the full height of the box (default: true).
The horizontal gap between two nodes (default: 0).

Creates a default HBox.

Creates an HBox with the specified horizontal gap between nodes.
Sets the margin for the node in the pane.

javafx.scene.layout.VBox

```
-alignment: ObjectProperty<Pos>  
-fillWidth: BooleanProperty  
-spacing: DoubleProperty  
  
+VBox()  
+VBox(spacing: double)  
+setMargin(node: Node, value: Insets): void
```

The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.

The overall alignment of the children in the box (default: Pos.TOP_LEFT).
Is resizable children fill the full width of the box (default: true).
The vertical gap between two nodes (default: 0).

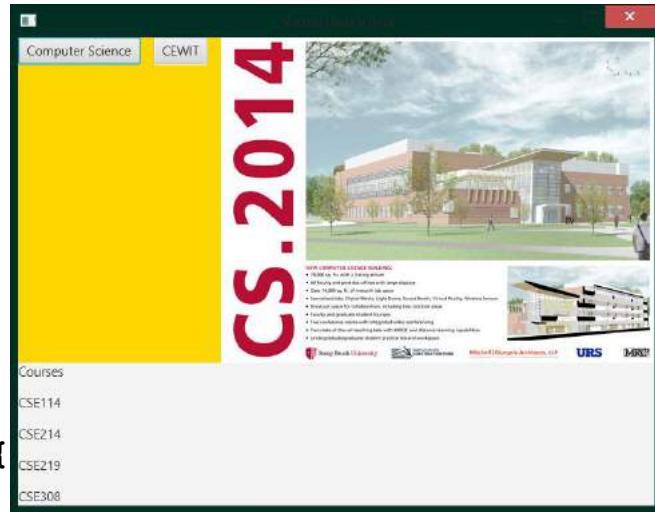
Creates a default VBox.

Creates a VBox with the specified horizontal gap between nodes.
Sets the margin for the node in the pane.

```

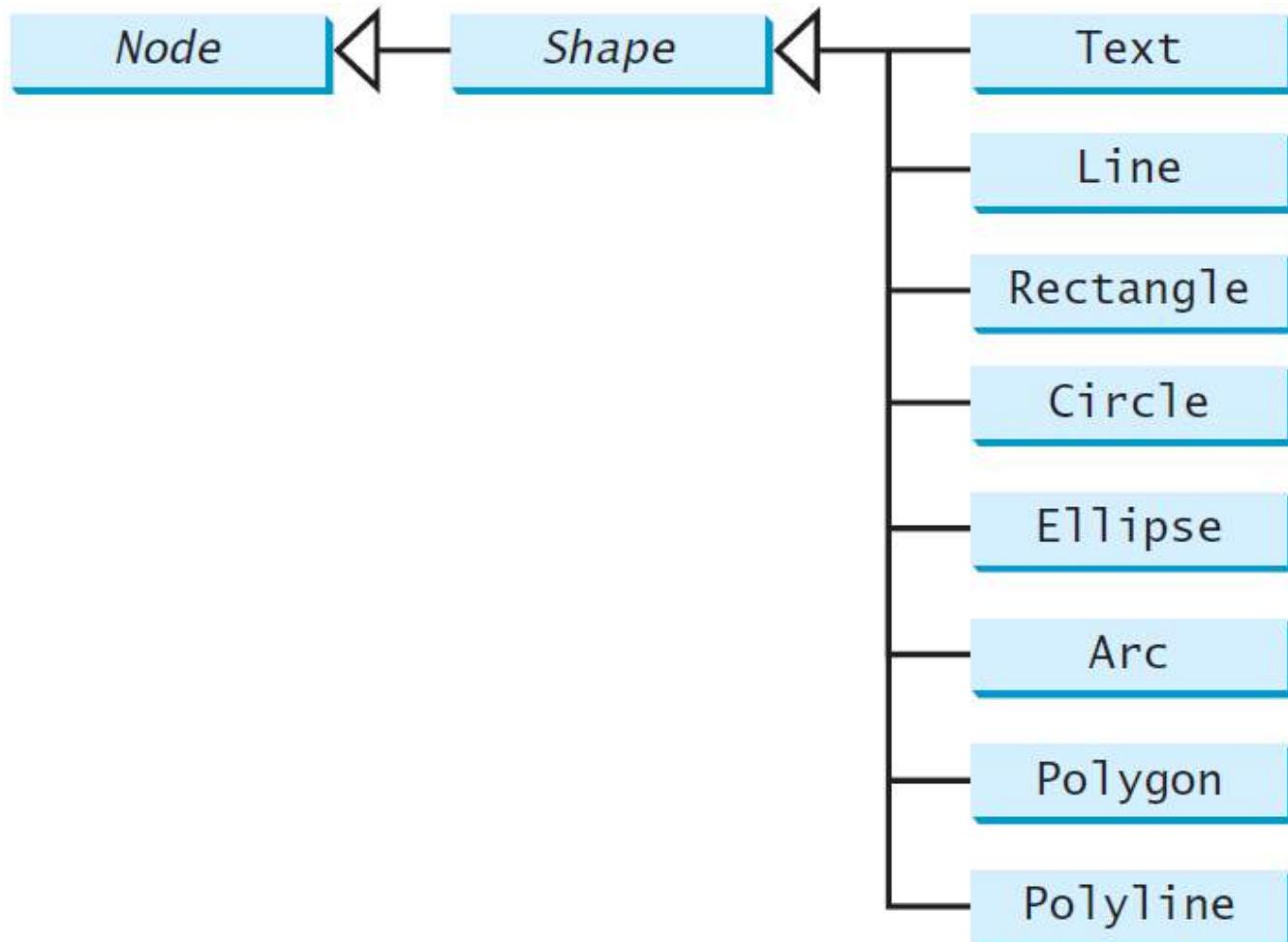
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.layout.BorderPane;
import javafx.scene.layout.HBox;
import javafx.scene.layout.VBox;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.image.Image;
import javafx.scene.image.ImageView;
public class ShowHBoxVBox extends Application {
    @Override
    public void start(Stage primaryStage) {
        BorderPane pane = new BorderPane();
        HBox hBox = new HBox(15);
        hBox.setStyle("-fx-background-color: gold");
        hBox.getChildren().add(new Button("Computer Science"));
        hBox.getChildren().add(new Button("CEWIT"));
        ImageView imageView = new ImageView(new Image("cs14.jpg"));
        hBox.getChildren().add(imageView);
        pane.setTop(hBox);
        VBox vBox = new VBox(15);
        vBox.getChildren().add(new Label("Courses"));
        Label[] courses = {new Label("CSE114"), new Label("CSE214"),
            new Label("CSE219"), new Label("CSE308")};
        for (Label course: courses) {
            vBox.getChildren().add(course);
        }
        pane.setLeft(vBox);
        Scene scene = new Scene(pane); primaryStage.setScene(scene);
        primaryStage.show(); (c) Paul Fodor and Pearson Inc.
    }
}

```



Shapes

JavaFX provides many shape classes for drawing texts, lines, circles, rectangles, ellipses, arcs, polygons, and polylines.



Text

javafx.scene.text.Text

```
-text: StringProperty  
-x: DoubleProperty  
-y: DoubleProperty  
-underline: BooleanProperty  
-strikethrough: BooleanProperty  
-font: ObjectProperty<Font>
```

```
+Text()  
+Text(text: String)  
+Text(x: double, y: double,  
      text: String)
```

The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.

Defines the text to be displayed.

Defines the x-coordinate of text (default 0).

Defines the y-coordinate of text (default 0).

Defines if each line has an underline below it (default `false`).

Defines if each line has a line through it (default `false`).

Defines the font for the text.

Creates an empty Text.

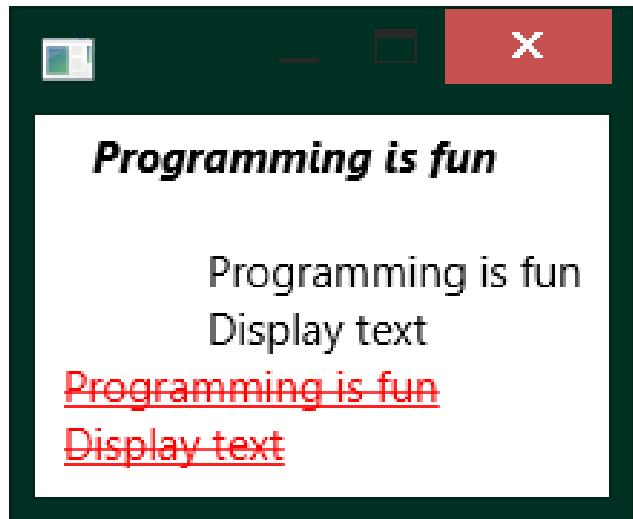
Creates a Text with the specified text.

Creates a Text with the specified x-, y-coordinates and text.

```

import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.layout.Pane;
import javafx.scene.paint.Color;
import javafx.geometry.Insets;
import javafx.scene.text.Text;
import javafx.scene.text.Font;
import javafx.scene.text.FontWeight;
import javafx.scene.text.FontPosture;
public class ShowText extends Application {
    @Override
    public void start(Stage primaryStage) {
        Pane pane = new Pane();
        pane.setPadding(new Insets(5, 5, 5, 5));
        Text text1 = new Text(20, 20, "Programming is fun");
        text1.setFont(Font.font("Courier", FontWeight.BOLD,
            FontPosture.ITALIC, 15));
        pane.getChildren().add(text1);
        Text text2 = new Text(60, 60, "Programming is fun\nDisplay text");
        pane.getChildren().add(text2);
        Text text3 = new Text(10, 100, "Programming is fun\nDisplay text");
        text3.setFill(Color.RED);
        text3.setUnderline(true);
        text3.setStrikethrough(true);
        pane.getChildren().add(text3);
        Scene scene = new Scene(pane, 600, 800);
        primaryStage.setScene(scene); primaryStage.show();
    }
}

```



(0, 0)

(x, y)

→ text is displayed

Line

`javafx.scene.shape.Line`

`-startX: DoubleProperty`
`-startY: DoubleProperty`
`-endX: DoubleProperty`
`-endY: DoubleProperty`

`+Line()`
`+Line(startX: double, startY: double, endX: double, endY: double)`

The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.

The x-coordinate of the start point.

The y-coordinate of the start point.

The x-coordinate of the end point.

The y-coordinate of the end point.

Creates an empty `Line`.

Creates a `Line` with the specified starting and ending points.

`(0, 0)`

`(getWidth(), 0)`

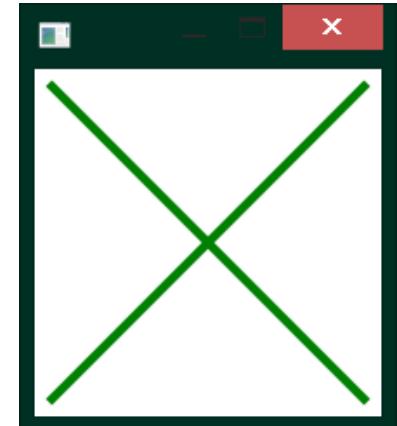
`(startX, startY)`

`(endX, endY)`

`(0, getHeight())`

`(getWidth(), getHeight())`

```
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.layout.Pane;
import javafx.scene.shape.Line;
import javafx.scene.paint.Color;
public class ShowLine extends Application {
    @Override
    public void start(Stage primaryStage) {
        Pane pane = new Pane();
        Line line1 = new Line(10, 10, 10, 10);
        line1.endXProperty().bind(pane.widthProperty().subtract(10));
        line1.endYProperty().bind(pane.heightProperty().subtract(10));
        line1.setStrokeWidth(5);
        line1.setStroke(Color.GREEN);
        pane.getChildren().add(line1);
        Line line2 = new Line(10, 10, 10, 10);
        line2.startXProperty().bind(pane.widthProperty().subtract(10));
        line2.endYProperty().bind(pane.heightProperty().subtract(10));
        line2.setStrokeWidth(5);
        line2.setStroke(Color.GREEN);
        pane.getChildren().add(line2);
        Scene scene = new Scene(pane, 200, 200);
        primaryStage.setScene(scene);
        primaryStage.show();
    }
    public static void main(String[] args) {
        launch(args);
    }
}
```



Rectangle

`javafx.scene.shape.Rectangle`

-x: DoubleProperty
-y: DoubleProperty
-width: DoubleProperty
-height: DoubleProperty
-arcWidth: DoubleProperty
-arcHeight: DoubleProperty

+Rectangle()

+Rectangle(x: double, y: double, width: double, height: double)

The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.

The x-coordinate of the upper-left corner of the rectangle (default 0).
The y-coordinate of the upper-left corner of the rectangle (default 0).
The width of the rectangle (default: 0).
The height of the rectangle (default: 0).
The **arcWidth** of the rectangle (default: 0). **arcWidth** is the horizontal diameter of the arcs at the corner (see Figure 14.31a).
The **arcHeight** of the rectangle (default: 0). **arcHeight** is the vertical diameter of the arcs at the corner (see Figure 14.31a).

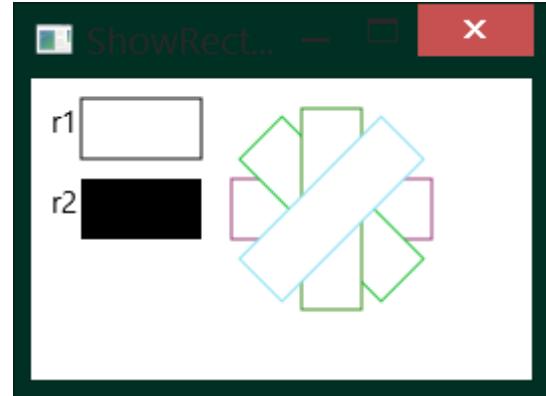
Creates an empty `Rectangle`.

Creates a `Rectangle` with the specified upper-left corner point, width, and height.

```

import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.layout.Pane;
import javafx.scene.text.Text;
import javafx.scene.shape.Rectangle;
import javafx.scene.paint.Color;
import java.util.Collections;
public class ShowRectangle extends Application {
    public void start(Stage primaryStage) {
        Pane pane = new Pane();
        Rectangle r1 = new Rectangle(25, 10, 60, 30);
        r1.setStroke(Color.BLACK);
        r1.setFill(Color.WHITE);
        pane.getChildren().add(new Text(10, 27, "r1"));
        pane.getChildren().add(r1);
        Rectangle r2 = new Rectangle(25, 50, 60, 30);
        pane.getChildren().add(new Text(10, 67, "r2"));
        pane.getChildren().add(r2);
        for (int i = 0; i < 4; i++) {
            Rectangle r = new Rectangle(100, 50, 100, 30);
            r.setRotate(i * 360 / 8);
            r.setStroke(Color.color(Math.random(), Math.random(),
                Math.random()));
            r.setFill(Color.WHITE);
            pane.getChildren().add(r);
        }
        Scene scene = new Scene(pane, 250, 150);
        primaryStage.setScene(scene); primaryStage.show();
    }
    ...// main
}

```



Circle

javafx.scene.shape.Circle

-centerX: DoubleProperty
-centerY: DoubleProperty
-radius: DoubleProperty

+Circle()
+Circle(x: double, y: double)
+Circle(x: double, y: double,
radius: double)

The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.

The x-coordinate of the center of the circle (default 0).
The y-coordinate of the center of the circle (default 0).
The radius of the circle (default: 0).

Creates an empty **Circle**.

Creates a **Circle** with the specified center.

Creates a **Circle** with the specified center and radius.

Ellipse

javafx.scene.shape.Ellipse

-centerX: DoubleProperty
-centerY: DoubleProperty
-radiusX: DoubleProperty
-radiusY: DoubleProperty

+Ellipse()
+Ellipse(x: double, y: double)
+Ellipse(x: double, y: double,
radiusX: double, radiusY:
double)

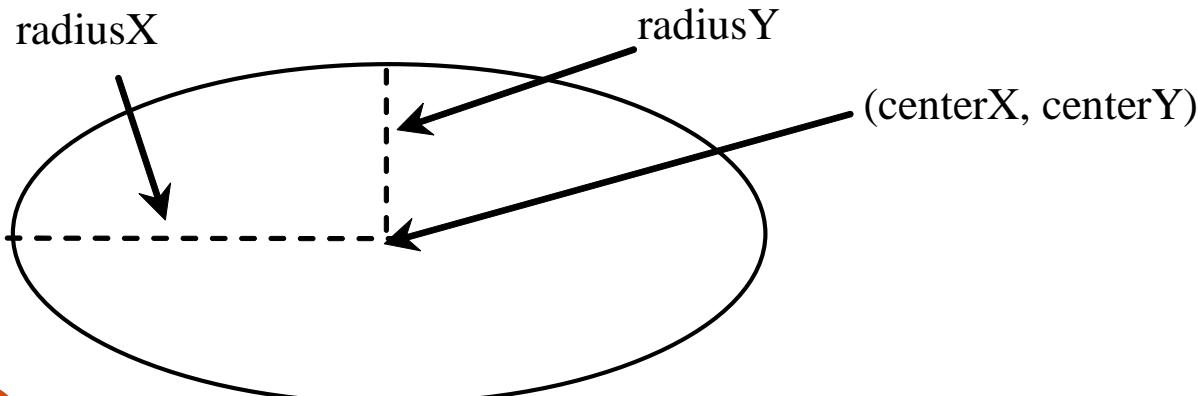
The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.

The x-coordinate of the center of the ellipse (default 0).
The y-coordinate of the center of the ellipse (default 0).
The horizontal radius of the ellipse (default: 0).
The vertical radius of the ellipse (default: 0).

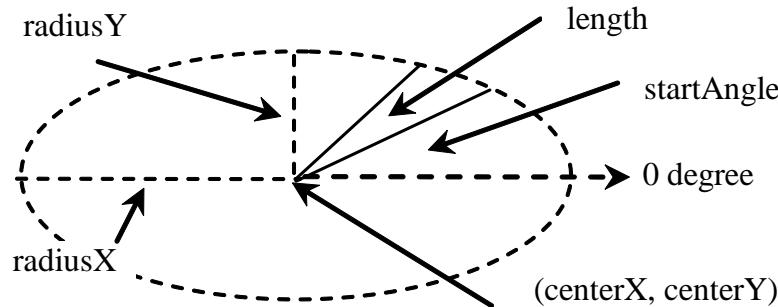
Creates an empty Ellipse.

Creates an Ellipse with the specified center.

Creates an Ellipse with the specified center and radii.



Arc



`javafx.scene.shape.Arc`

-centerX: DoubleProperty
-centerY: DoubleProperty
-radiusX: DoubleProperty
-radiusY: DoubleProperty
-startAngle: DoubleProperty
-length: DoubleProperty
-type: ObjectProperty<ArcType>

+Arc()

+Arc(x: double, y: double,
radiusX: double, radiusY:
double, startAngle: double,
length: double)

The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.

The x-coordinate of the center of the ellipse (default 0).

The y-coordinate of the center of the ellipse (default 0).

The horizontal radius of the ellipse (default: 0).

The vertical radius of the ellipse (default: 0).

The start angle of the arc in degrees.

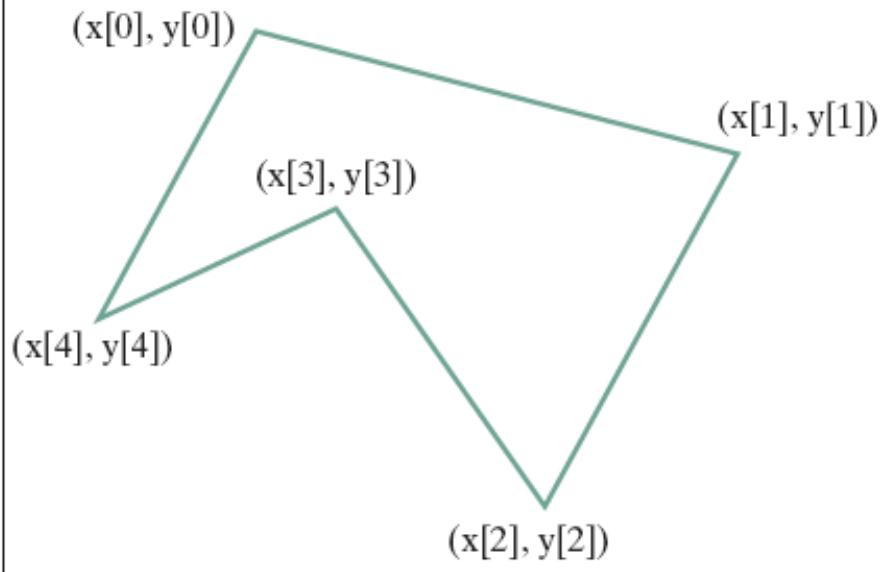
The angular extent of the arc in degrees.

The closure type of the arc (ArcType.OPEN, ArcType.CHORD, ArcType.ROUND).

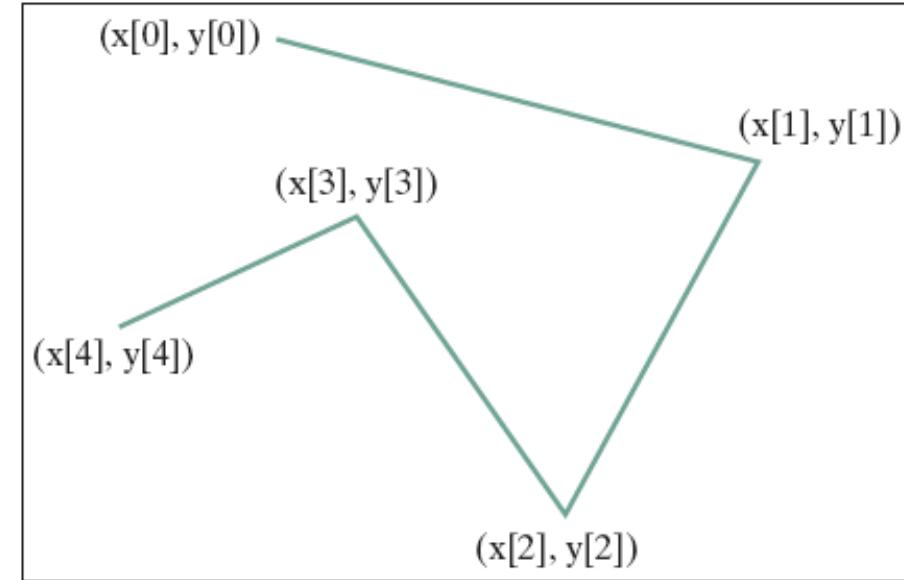
Creates an empty Arc.

Creates an Arc with the specified arguments.

Polygon and Polyline



(a) Polygon



(b) Polyline

The `getter` and `setter` methods for property values and a `getter` for property itself are provided in the class, but omitted in the UML diagram for brevity.

`javafx.scene.shape.Polygon`

```
+Polygon ()  
+Polygon (double... points)  
+getPoints () :  
    ObservableList<Double>
```

Creates an empty polygon.

Creates a polygon with the given points.

Returns a list of double values as x- and y-coordinates of the points.

Event Programming

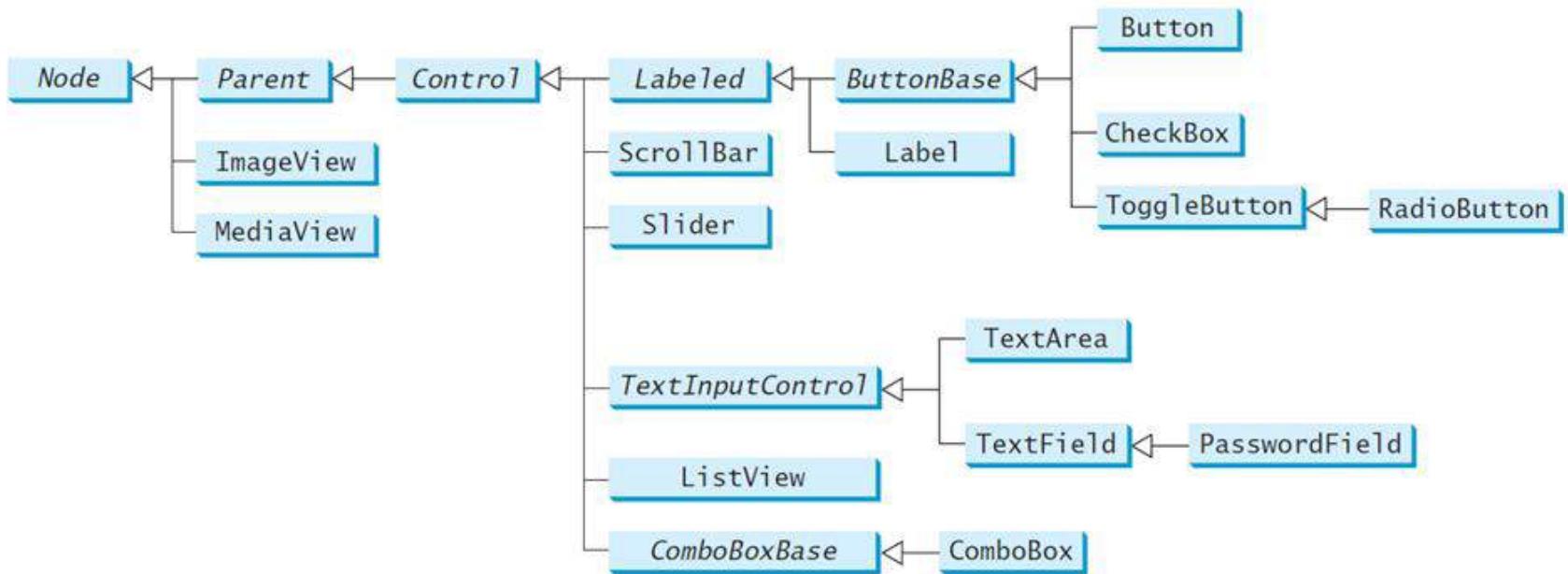
- Procedural programming is executed in procedural/statement order
- In event-driven programming, code is executed upon activation of events
- Operating Systems constantly monitor events
 - Ex: keystrokes, mouse clicks, etc...
- The OS:
 - sorts out these events
 - reports them to the appropriate programs

Where do we come in?

- For each control (button, combo box, etc.):
 - define an event handler
 - construct an instance of event handler
 - tell the control who its event handler is
- Event Handler?
 - code with response to event
 - a.k.a. event listener

Java's Event Handling

- An *event source* is a GUI control
 - JavaFX: **Button**, **ChoiceBox**, **ListView**, etc.



http://docs.oracle.com/javase/8/javafx/user-interface-tutorial/ui_controls.htm

- different types of sources:
 - can detect different types of events
 - can register different types of listeners (handlers)

Java's Event Handling

- When the user interacts with a control (source):
 - an *event object* is constructed
 - the event object is sent to all registered *listener objects*
 - the listener object (handler) responds as you defined it to

Event Listeners (Event Handler)

- Defined by you, the application programmer
 - you customize the response
 - How?
 - Inheritance & Polymorphism
- You define your own listener class
 - implement the appropriate interface
 - define responses in all necessary methods

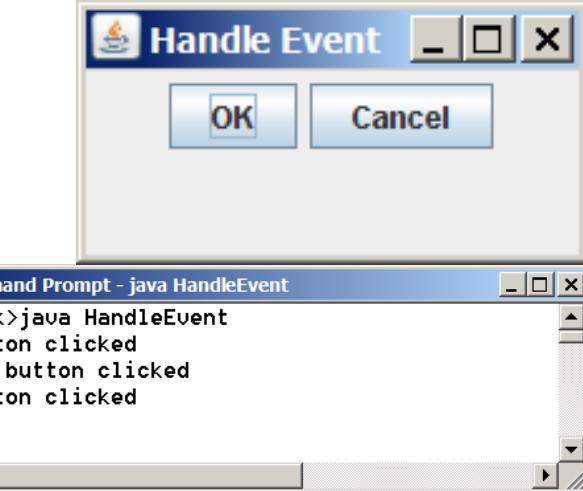
Event Objects

- Contain information about the event
- Like what?
 - location of mouse click
 - event source that was interacted with
 - etc.
- Listeners use them to properly respond
 - different methods inside a listener object can react differently to different types of interactions

```

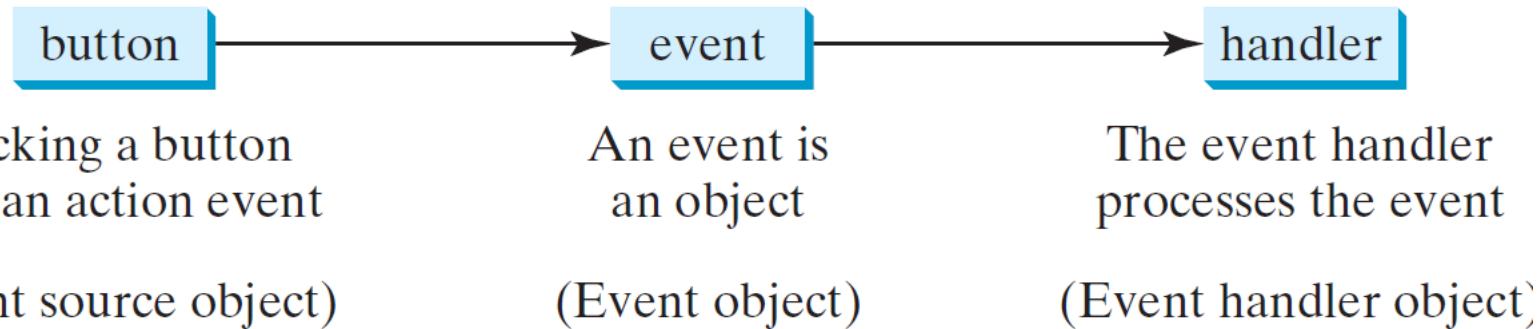
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.layout.HBox;
import javafx.scene.control.Button;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.geometry.Pos;
public class HandleEvent extends Application {
    public void start(Stage primaryStage) {
        HBox pane = new HBox(10);
        Button btOK = new Button("OK");
        Button btCancel = new Button("Cancel");
        OKHandlerClass handler1 = new OKHandlerClass();
        btOK.setOnAction(handler1);
        CancelHandlerClass handler2 = new CancelHandlerClass();
        btCancel.setOnAction(handler2);
        pane.getChildren().addAll(btOK, btCancel);
        Scene scene = new Scene(pane);
        primaryStage.setScene(scene); primaryStage.show();
    }.../*main*/
}
class OKHandlerClass implements EventHandler<ActionEvent> {
    @Override
    public void handle(ActionEvent e) {
        System.out.println("OK button clicked");
    }
}
class CancelHandlerClass implements EventHandler<ActionEvent> {
    @Override
    public void handle(ActionEvent e) {
        System.out.println("Cancel button clicked");
    }
}

```

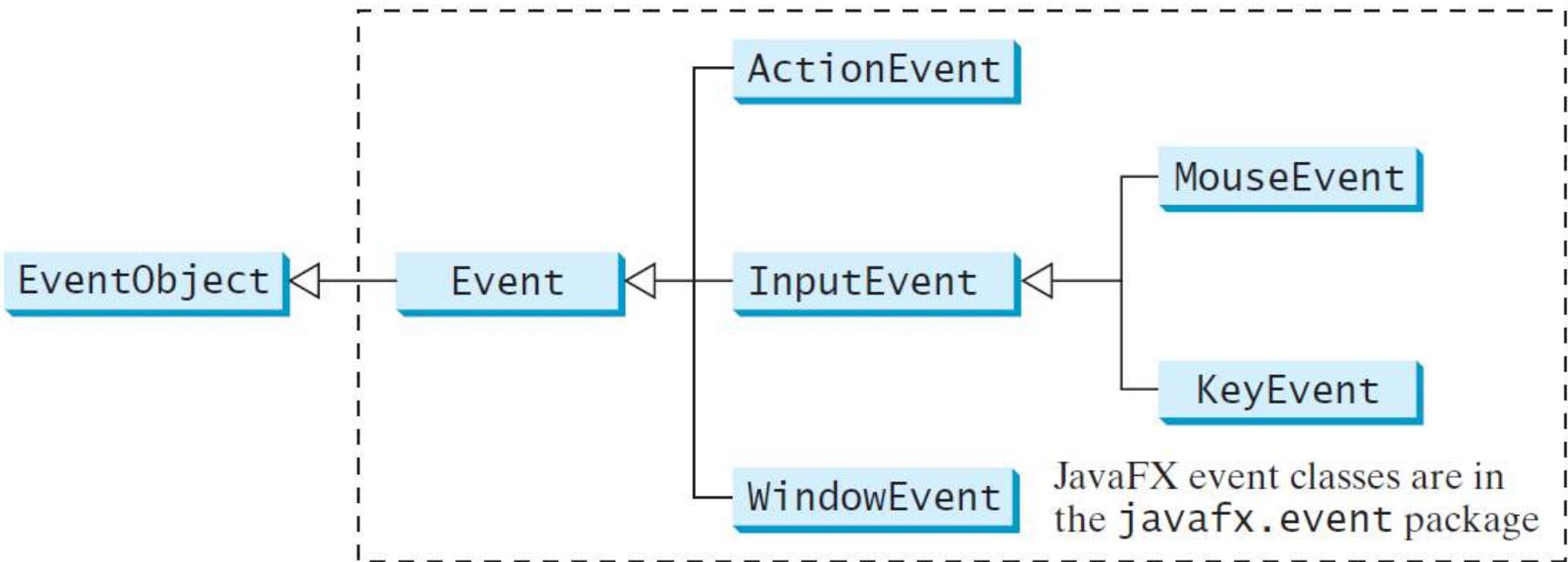


Handling GUI Events

- Source object: button.
 - An event is generated by external user actions such as mouse movements, mouse clicks, or keystrokes.
- An event can be defined as a type of signal to the program that something has happened.
- Listener object contains a method for processing the event.



Event Classes



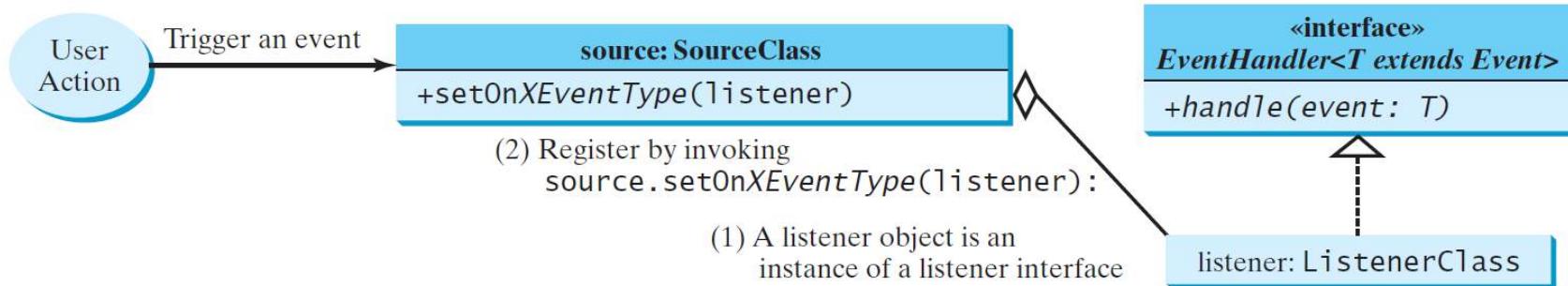
Event Information

- An event object contains whatever properties are pertinent to the event:
 - the *source object* of the event using the **getSource()** instance method in the **EventObject** class.
- The subclasses of **EventObject** deal with special types of events, such as button actions, window events, component events, mouse movements, and keystrokes.

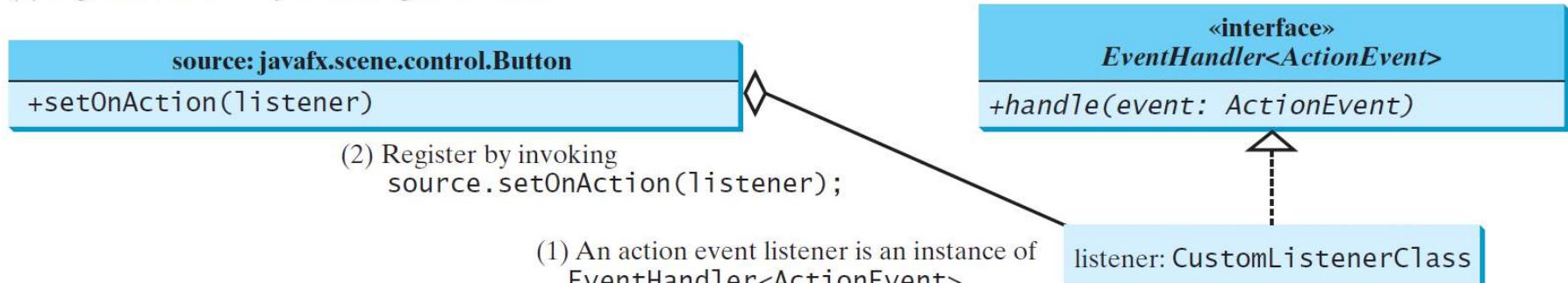
Selected User Actions and Handlers

User Action	Source Object	Event Type Fired	Event Registration Method
Click a button	<code>Button</code>	<code>ActionEvent</code>	<code>setOnAction(EventHandler<ActionEvent>)</code>
Press Enter in a text field	<code>TextField</code>	<code>ActionEvent</code>	<code>setOnAction(EventHandler<ActionEvent>)</code>
Check or uncheck	<code>RadioButton</code>	<code>ActionEvent</code>	<code>setOnAction(EventHandler<ActionEvent>)</code>
Check or uncheck	<code>CheckBox</code>	<code>ActionEvent</code>	<code>setOnAction(EventHandler<ActionEvent>)</code>
Select a new item	<code>ComboBox</code>	<code>ActionEvent</code>	<code>setOnAction(EventHandler<ActionEvent>)</code>
Mouse pressed	<code>Node, Scene</code>	<code>MouseEvent</code>	<code>setOnMousePressed(EventHandler<MouseEvent>)</code>
Mouse released			<code>setOnMouseReleased(EventHandler<MouseEvent>)</code>
Mouse clicked			<code>setOnMouseClicked(EventHandler<MouseEvent>)</code>
Mouse entered			<code>setOnMouseEntered(EventHandler<MouseEvent>)</code>
Mouse exited			<code>setOnMouseExited(EventHandler<MouseEvent>)</code>
Mouse moved			<code>setOnMouseMoved(EventHandler<MouseEvent>)</code>
Mouse dragged			<code>setOnMouseDragged(EventHandler<MouseEvent>)</code>
Key pressed	<code>Node, Scene</code>	<code>KeyEvent</code>	<code>setOnKeyPressed(EventHandler<KeyEvent>)</code>
Key released			<code>setOnKeyReleased(EventHandler<KeyEvent>)</code>
Key typed			<code>setOnKeyTyped(EventHandler<KeyEvent>)</code>

The Delegation Model



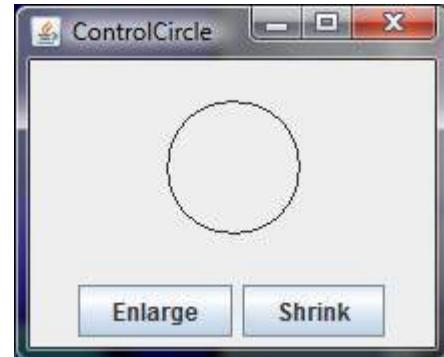
(a) A generic source object with a generic event T



(b) A Button source object with an ActionEvent

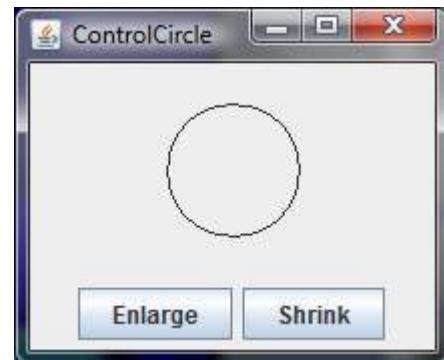
ControlCircle program that uses two buttons to control the size of a circle

```
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.layout.BorderPane;
import javafx.scene.layout.HBox;
import javafx.scene.layout.StackPane;
import javafx.scene.control.Button;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.geometry.Pos;
import javafx.scene.paint.Color;
import javafx.scene.shape.Circle;
public class ControlCircle extends Application {
    private CirclePane circlePane = new CirclePane();
    @Override
    public void start(Stage primaryStage) {
        HBox hBox = new HBox();
        Button btEnlarge = new Button("Enlarge");
        Button btShrink = new Button("Shrink");
        hBox.getChildren().add(btEnlarge);
        hBox.getChildren().add(btShrink);
        btEnlarge.setOnAction(new EnlargeHandler());
        BorderPane borderPane = new BorderPane();
        borderPane.setCenter(circlePane);
        borderPane.setBottom(hBox);
        borderPane.setAlignment(hBox, Pos.CENTER);
        Scene scene = new Scene(borderPane, 200, 150);
        primaryStage.setScene(scene); primaryStage.show();
    }
}
```



ControlCircle program that uses two buttons to control the size of a circle

```
// Inner Class
class EnlargeHandler
    implements EventHandler<ActionEvent> {
    @Override
    public void handle(ActionEvent e) {
        circlePane.enlarge();
    }
}
class CirclePane extends StackPane {
    private Circle circle = new Circle(50);
    public CirclePane() {
        getChildren().add(circle);
        circle.setStroke(Color.BLACK);
        circle.setFill(Color.WHITE);
    }
    public void enlarge() {
        circle.setRadius(circle.getRadius() + 2);
    }
    public void shrink() {
        circle.setRadius(circle.getRadius() > 2
            ? circle.getRadius() - 2 : circle.getRadius());
    }
}
```



Inner Class Listeners

- A listener class is designed specifically to create a listener object for a GUI component (e.g., a button).
- Any object instance of the inner handler class has access to all GUI fields of the outer class.
- It will not be shared by other applications.

Inner Classes

```
public class OuterClass {  
    private int data = 0;  
    OuterClass() {  
        InnerClass y = new InnerClass();  
        y.m2();  
    }  
    public void m1() {  
        data++;  
    }  
    public static void main(String[] args) {  
        OuterClass x = new OuterClass();  
        System.out.println(x.data);  
    }  
    class InnerClass {  
        public void m2() {  
            /* Directly reference data and  
               method defined in outer class */  
            data++;  
            m1();  
        }  
    }  
}
```

- The **InnerClass** is a member of **OuterClass**
 - An inner class can reference the data and methods defined in the outer class in which it nests, so you do not need to pass the reference of the outer class to the constructor of the inner class.
 - An inner class is compiled into a class named **OuterClass\$InnerClass.class**

Inner Classes

- An inner class can be declared **public**, **protected**, or **private** subject to the same visibility rules applied to a member of the class.
- An inner class can be declared **static**:
 - The **static** inner class can be accessed using the outer class name,
 - However, a **static** inner class cannot access nonstatic members of the outer class.

Anonymous Inner Classes

- Inner class listeners can be shortened using anonymous inner classes: inner classes without a name.
 - It combines declaring an inner class and creating an instance of the class in one step.
- An anonymous inner class is declared as follows:

```
new SuperClassName/InterfaceName() {  
    // Implement or override methods in superclass/interface  
    // Other methods if necessary  
}
```

Anonymous Inner Classes

- An anonymous inner class must always extend a superclass or implement an interface, but it **cannot have an explicit extends or implements clause.**
- An anonymous inner class must **implement all the abstract methods** in the superclass or in the interface.
- An anonymous inner class **always uses the no-arg constructor from its superclass to create an instance.**
- If an anonymous inner class implements an interface, the constructor is **Object()**.
- An anonymous inner class is compiled into a class named **OuterClassName\$*n*.class**, where **n** is the count of inner classes.

Anonymous Inner Classes

```
public void start(Stage primaryStage) {  
    // Omitted  
  
    btEnlarge.setOnAction(  
        new EnlargeHandler());  
}  
  
class EnlargeHandler  
    implements EventHandler<ActionEvent> {  
    public void handle(ActionEvent e) {  
        circlePane.enlarge();  
    }  
}
```

(a) Inner class EnlargeListener

```
public void start(Stage primaryStage) {  
    // Omitted  
  
    btEnlarge.setOnAction(  
        new class EnlargeHandler  
            implements EventHandler<ActionEvent>() {  
                public void handle(ActionEvent e) {  
                    circlePane.enlarge();  
                }  
            }  
    );  
}
```

(b) Anonymous inner class

```

import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.layout.HBox;
import javafx.scene.control.Button;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.geometry.Pos;
public class AnonymousHandlerDemo extends Application {
    public void start(Stage primaryStage) {
        HBox hBox = new HBox();
        Button btNew = new Button("New");
        Button btOpen = new Button("Open"); //btSave, btPrint btns.
        hBox.getChildren().addAll(btNew, btOpen);
        // Create and register the handler
        btNew.setOnAction(new EventHandler<ActionEvent>() {
            @Override // Override the handle method
            public void handle(ActionEvent e) {
                System.out.println("Process New");
            }
        });
        btOpen.setOnAction(new EventHandler<ActionEvent>() {
            @Override // Override the handle method
            public void handle(ActionEvent e) {
                System.out.println("Process Open");
            }
        });
    }
}

```



```
Scene scene = new Scene(hBox, 300, 50);  
primaryStage.setTitle("AnonymousHandlerDemo");  
primaryStage.setScene(scene);  
primaryStage.show();  
}  
public static void main(String[] args) {  
    launch(args);  
}  
}
```

Simplifying Event Handling Using Lambda Expressions

- *Lambda expression* is a new feature in Java 8.
 - Predefined functions for the type of the input.
- Lambda expressions can be viewed as an anonymous method with a concise syntax.

```
btEnlarge.setOnAction(  
    new EventHandler<ActionEvent>() {  
        @Override  
        public void handle(ActionEvent e) {  
            // Code for processing event e  
        }  
    } );
```

(a) Anonymous inner class event handler

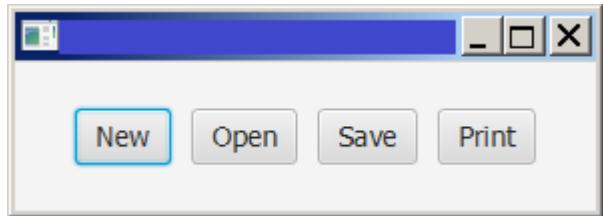
```
btEnlarge.setOnAction(e -> {  
    // Code for processing event e  
});
```

(b) Lambda expression event handler

```

import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.layout.HBox;
import javafx.scene.control.Button;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.geometry.Pos;
public class LambdaHandlerDemo extends Application {
    @Override
    public void start(Stage primaryStage) {
        // Hold two buttons in an HBox
        HBox hBox = new HBox();
        hBox.setSpacing(10);
        hBox.setAlignment(Pos.CENTER);
        Button btNew = new Button("New");
        Button btOpen = new Button("Open");
        Button btSave = new Button("Save");
        Button btPrint = new Button("Print");
        hBox.getChildren().addAll(btNew, btOpen, btSave, btPrint);
        btNew.setOnAction(e -> {System.out.println("Process New");});
        btOpen.setOnAction(e -> {System.out.println("Process Open");});
        btSave.setOnAction(e -> {System.out.println("Process Save");});
        btPrint.setOnAction(e -> {System.out.println("Process Print");});
        Scene scene = new Scene(hBox, 300, 50);
        primaryStage.setScene(scene);
        primaryStage.show();
    }
    public static void main(String[] args) {
        launch(args);  }
}

```



Output:
Process New
Process Open
Process Save
Process Print

Basic Syntax for a Lambda Expression

- The basic syntax for a lambda expression is either:

```
(type1 param1, type2 param2, ...) -> expression
```

or

```
(type1 param1, type2 param2, ...) -> { statements; }
```

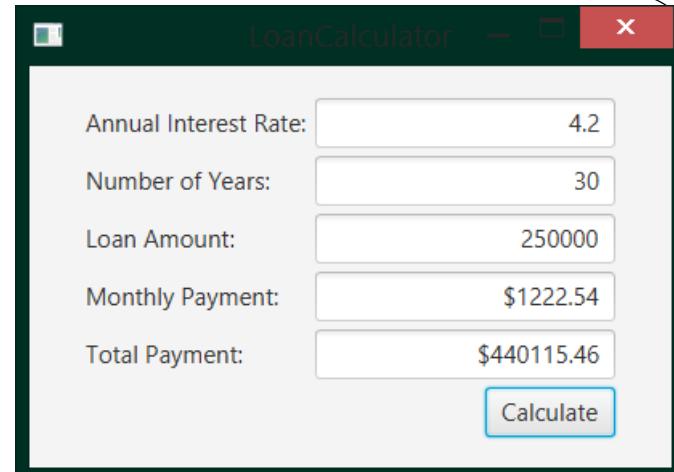
- The data type for a parameter may be explicitly declared or implicitly inferred by the compiler.
- The parentheses can be omitted if there is only one parameter without an explicit data type.

Single Abstract Method Interface (SAM)

- The statements in the lambda expression is all for that method.
- If it contains multiple methods, the compiler will not be able to compile the lambda expression.
- So, for the compiler to understand lambda expressions, the **interface must contain exactly one abstract method**.
- Such an interface is known as a *functional interface*, or a *Single Abstract Method (SAM)* interface.

Loan Calculator

```
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.layout.GridPane;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.control.TextField;
import javafx.geometry.Pos;
import javafx.geometry.HPos;
public class LoanCalculator extends Application {
    private TextField tfAnnualInterestRate = new TextField();
    private TextField tfNumberOfYears = new TextField();
    private TextField tfLoanAmount = new TextField();
    private TextField tfMonthlyPayment = new TextField();
    private TextField tfTotalPayment = new TextField();
    private Button btCalculate = new Button("Calculate");
    @Override
    public void start(Stage primaryStage) {
        // Create UI
        GridPane gridPane = new GridPane();
        gridPane.setHgap(5);
        gridPane.setVgap(5);
        gridPane.add(new Label("Annual Interest Rate:"), 0, 0);
        gridPane.add(tfAnnualInterestRate, 1, 0);
        gridPane.add(new Label("Number of Years:"), 0, 1);
        gridPane.add(tfNumberOfYears, 1, 1);
        gridPane.add(new Label("Loan Amount:"), 0, 2);
        gridPane.add(tfLoanAmount, 1, 2);
        gridPane.add(new Label("Monthly Payment:"), 0, 3);
        gridPane.add(tfMonthlyPayment, 1, 3);
        gridPane.add(new Label("Total Payment:"), 0, 4);
        gridPane.add(tfTotalPayment, 1, 4);
        gridPane.add(btCalculate, 1, 5);
    }
}
```



```
btCalculate.setOnAction(e -> calculateLoanPayment());
Scene scene = new Scene(gridPane, 400, 250);
primaryStage.setScene(scene);
primaryStage.show();
}
private void calculateLoanPayment() {
    // Get values from text fields
    double interest = Double.parseDouble(tfAnnualInterestRate.getText());
    int year = Integer.parseInt(tfNumberOfYears.getText());
    double loanAmount = Double.parseDouble(tfLoanAmount.getText());
    // Create a loan object
    Loan loan = new Loan(interest, year, loanAmount);
    // Display monthly payment and total payment
    tfMonthlyPayment.setText(String.format("%.2f", loan.getMonthlyPayment()));
    tfTotalPayment.setText(String.format("%.2f", loan.getTotalPayment()));
}
public static void main(String[] args) {
    launch(args);
}
}
class Loan implements java.io.Serializable {
    private double annualInterestRate;
    private int numberOfYears;
    private double loanAmount;
    private java.util.Date loanDate;
    public Loan(double annualInterestRate, int numberOfYears, double loanAmount) {
        this.annualInterestRate = annualInterestRate;
        this.numberOfYears = numberOfYears;
        this.loanAmount = loanAmount;
        loanDate = new java.util.Date();
    }
    public double getAnnualInterestRate() {
        return annualInterestRate;
    }
    public void setAnnualInterestRate(double annualInterestRate) {
        this.annualInterestRate = annualInterestRate;
    }
}
```

```
public int getNumberOfYears() {
    return numberOfYears;
}
public void setNumberOfYears(int numberOfYears) {
    this.numberOfYears = numberOfYears;
}
public double getLoanAmount() {
    return loanAmount;
}
public void setLoanAmount(double loanAmount) {
    this.loanAmount = loanAmount;
}
public double getMonthlyPayment() {
    double monthlyInterestRate = annualInterestRate / 1200;
    double monthlyPayment = loanAmount * monthlyInterestRate / (1
        - (Math.pow(1 / (1 + monthlyInterestRate), numberOfYears * 12)));
    return monthlyPayment;
}
public double getTotalPayment() {
    double totalPayment = getMonthlyPayment() * numberOfYears * 12;
    return totalPayment;
}
public java.util.Date getLoanDate() {
    return loanDate;
}
}
```

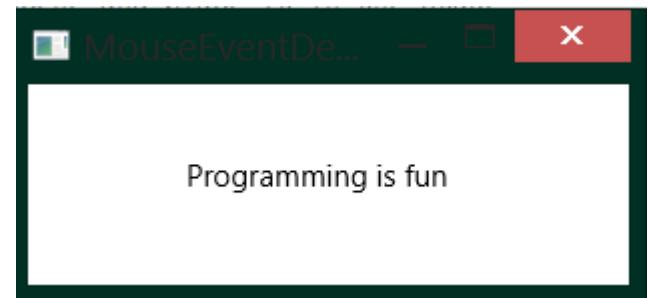
MouseEvent

`javafx.scene.input.MouseEvent`

```
+getButton(): MouseButton  
+getClickCount(): int  
+getX(): double  
+getY(): double  
+getSceneX(): double  
+getSceneY(): double  
+getScreenX(): double  
+getScreenY(): double  
+isAltDown(): boolean  
+isControlDown(): boolean  
+isMetaDown(): boolean  
+isShiftDown(): boolean
```

Indicates which mouse button has been clicked.
Returns the number of mouse clicks associated with this event.
Returns the *x*-coordinate of the mouse point in the event source node.
Returns the *y*-coordinate of the mouse point in the event source node.
Returns the *x*-coordinate of the mouse point in the scene.
Returns the *y*-coordinate of the mouse point in the scene.
Returns the *x*-coordinate of the mouse point in the screen.
Returns the *y*-coordinate of the mouse point in the screen.
Returns true if the **Alt** key is pressed on this event.
Returns true if the **Control** key is pressed on this event.
Returns true if the mouse **Meta** button is pressed on this event.
Returns true if the **Shift** key is pressed on this event.

```
// Move the text with the mouse clicked
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.layout.Pane;
import javafx.scene.text.Text;
public class MouseEventDemo extends Application {
    @Override
    public void start(Stage primaryStage) {
        Pane pane = new Pane();
        Text text = new Text(20, 20, "Programming is fun");
        pane.getChildren().add(text);
        text.setOnMouseDragged(e -> {
            text.setX(e.getX());
            text.setY(e.getY());
        });
        Scene scene = new Scene(pane, 300, 100);
        primaryStage.setTitle("MouseEventDemo");
        primaryStage.setScene(scene);
        primaryStage.show();
    }
    public static void main(String[] args) {
        launch(args);
    }
}
```



The KeyEvent Class

javafx.scene.input.KeyEvent

```
+getCharacter(): String  
+getCode(): KeyCode  
+getText(): String  
+isAltDown(): boolean  
+isControlDown(): boolean  
+isMetaDown(): boolean  
+isShiftDown(): boolean
```

Returns the character associated with the key in this event.
Returns the key code associated with the key in this event.
Returns a string describing the key code.
Returns true if the Alt key is pressed on this event.
Returns true if the Control key is pressed on this event.
Returns true if the mouse Meta button is pressed on this event.
Returns true if the Shift key is pressed on this event.

```

import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.layout.Pane;
import javafx.scene.text.Text;
public class KeyEventDemo extends Application {
    @Override
    public void start(Stage primaryStage) {
        Pane pane = new Pane();
        Text text = new Text(20, 20, "A");
        text.setFocusTraversable(true);
        pane.getChildren().add(text);
        text.setOnKeyPressed(e -> {
            switch (e.getCode()) {
                case DOWN: text.setY(text.getY() + 10); break;
                case UP:   text.setY(text.getY() - 10); break;
                case LEFT: text.setX(text.getX() - 10); break;
                case RIGHT: text.setX(text.getX() + 10); break;
                default:
                    if (Character.isLetterOrDigit(e.getText().charAt(0)))
                        text.setText(e.getText());
            }
        });
        Scene scene = new Scene(pane, 200, 200);
        primaryStage.setTitle("KeyEventDemo");
        primaryStage.setScene(scene);
        primaryStage.show();
    }
    public static void main(String[] args) {
        launch(args);
    }
}

```



The KeyCode Constants

<i>Constant</i>	<i>Description</i>	<i>Constant</i>	<i>Description</i>
HOME	The Home key	CONTROL	The Control key
END	The End key	SHIFT	The Shift key
PAGE_UP	The Page Up key	BACK_SPACE	The Backspace key
PAGE_DOWN	The Page Down key	CAPS	The Caps Lock key
UP	The up-arrow key	NUM_LOCK	The Num Lock key
DOWN	The down-arrow key	ENTER	The Enter key
LEFT	The left-arrow key	UNDEFINED	The keyCode unknown
RIGHT	The right-arrow key	F1 to F12	The function keys from F1 to F12
ESCAPE	The Esc key	0 to 9	The number keys from 0 to 9
TAB	The Tab key	A to Z	The letter keys from A to Z

Control Circle with Mouse and Key

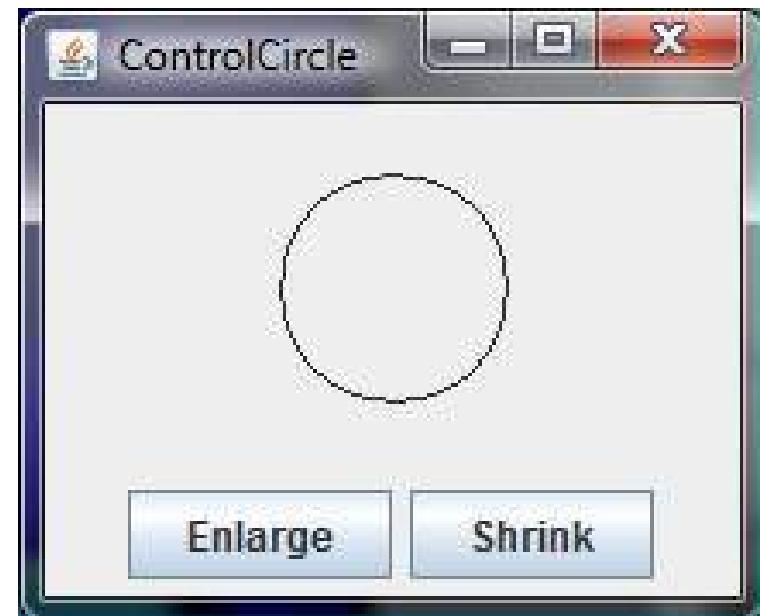
```
import javafx.application.Application;
import javafx.geometry.Pos;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.input.KeyCode;
import javafx.scene.input.MouseButton;
import javafx.scene.layout.HBox;
import javafx.scene.layout.BorderPane;
import javafx.stage.Stage;

public class ControlCircleWithMouseAndKey extends Application {
    private CirclePane circlePane = new CirclePane();

    @Override // Override the start method in the Application class
    public void start(Stage primaryStage) {
        // Hold two buttons in an HBox
        HBox hBox = new HBox();
        hBox.setSpacing(10);
        hBox.setAlignment(Pos.CENTER);
        Button btEnlarge = new Button("Enlarge");
        Button btShrink = new Button("Shrink");
        hBox.getChildren().add(btEnlarge);
        hBox.getChildren().add(btShrink);

        // Create and register the handler
        btEnlarge.setOnAction(e -> circlePane.enlarge());
        btShrink.setOnAction(e -> circlePane.shrink());

        BorderPane borderPane = new BorderPane();
        borderPane.setCenter(circlePane);
        borderPane.setBottom(hBox);
        BorderPane.setAlignment(hBox, Pos.CENTER);
    }
}
```



```
// Create a scene and place it in the stage
Scene scene = new Scene(borderPane, 200, 150);
primaryStage.setTitle("ControlCircle"); // Set the stage title
primaryStage.setScene(scene); // Place the scene in the stage
primaryStage.show(); // Display the stage

circlePane.setOnMouseClicked(e -> {
    if (e.getButton() == MouseButton.PRIMARY) {
        circlePane.enlarge();
    }
    else if (e.getButton() == MouseButton.SECONDARY) {
        circlePane.shrink();
    }
});

scene.setOnKeyPressed(e -> {
    if (e.getCode() == KeyCode.UP) {
        circlePane.enlarge();
    }
    else if (e.getCode() == KeyCode.DOWN) {
        circlePane.shrink();
    }
});

public static void main(String[] args) {
    launch(args);
}
```

Listeners for Observable Objects

- You can add a listener to process a value change in an observable object: an instance of **javafx.beans.Observable**
 - Every binding property is an instance of **Observable**.
 - **Observable** contains the **addListener(InvalidationListener listener)** method for adding a listener.
 - Once the value is changed in the property, a listener is notified.
 - The listener class should implement the **InvalidationListener** interface, which uses the **invalidated(Observable o)** method to handle the property value change.

Listeners for Observable Objects

```
import javafx.beans.InvalidationListener;
import javafx.beans.Observable;
import javafx.beans.property.DoubleProperty;
import javafx.beans.property.SimpleDoubleProperty;
public class ObservablePropertyDemo {
    public static void main(String[] args) {
        DoubleProperty balance = new SimpleDoubleProperty();
        balance.addListener(new InvalidationListener() {
            public void invalidated(Observable ov) {
                System.out.println("The new value is " +
                    balance.doubleValue());
            }
        });
        balance.set(4.5);
    }
}
```

Output:

The new value is 4.5

Animation

- JavaFX provides the **Animation** class with the core functionality for all animations:

javafx.animation.Animation

-autoReverse: BooleanProperty
-cycleCount: IntegerProperty
-rate: DoubleProperty
-status: ReadOnlyObjectProperty
<Animation.Status>

+pause(): void
+play(): void
+stop(): void

The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.

Defines whether the animation reverses direction on alternating cycles.
Defines the number of cycles in this animation.

Defines the speed and direction for this animation.

Read-only property to indicate the status of the animation.

Pauses the animation.

Plays the animation from the current position.

Stops the animation and resets the animation.

PathTransition

javafx.animation.PathTransition

-duration: ObjectProperty<Duration>
-node: ObjectProperty<Node>
-orientation: ObjectProperty
 <PathTransition.OrientationType>
-path: ObjectType<Shape>

+PathTransition()
+PathTransition(duration: Duration,
 path: Shape)
+PathTransition(duration: Duration,
 path: Shape, node: Node)

The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.

The duration of this transition.

The target node of this transition.

The orientation of the node along the path.

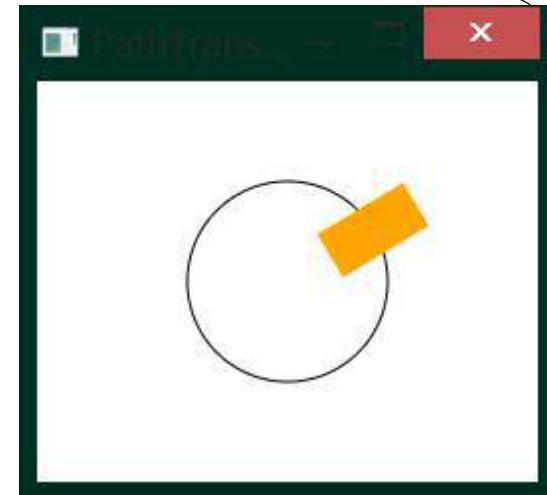
The shape whose outline is used as a path to animate the node move.

Creates an empty PathTransition.

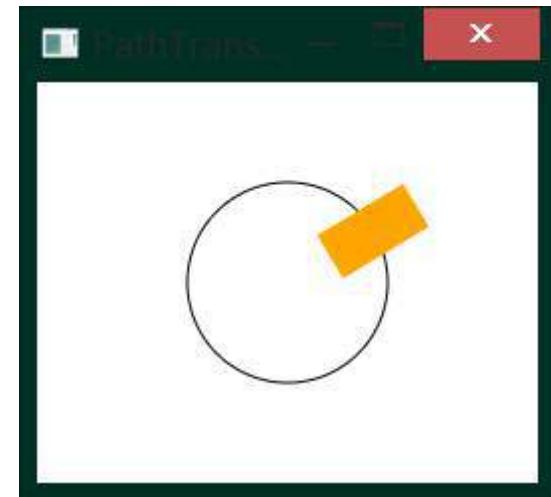
Creates a PathTransition with the specified duration and path.

Creates a PathTransition with the specified duration, path, and node.

```
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.layout.Pane;
import javafx.scene.paint.Color;
import javafx.scene.shape.Rectangle;
import javafx.scene.shape.Circle;
import javafx.animation.PathTransition;
import javafx.animation.Timeline;
import javafx.util.Duration;
public class PathTransitionDemo extends Application {
    @Override
    public void start(Stage primaryStage) {
        Pane pane = new Pane();
        Rectangle rectangle = new Rectangle(0, 0, 25, 50);
        rectangle.setFill(Color.ORANGE);
        Circle circle = new Circle(125, 100, 50);
        circle.setFill(Color.WHITE);
        circle.setStroke(Color.BLACK);
        pane.getChildren().addAll(circle, rectangle);
        // Create a path transition
        PathTransition pt = new PathTransition();
        pt.setDuration(Duration.millis(4000));
        pt.setPath(circle);
        pt.setNode(rectangle);
```



```
pt.setOrientation(  
    PathTransition.OrientationType.  
        ORTHOGONAL_TO_TANGENT);  
pt.setCycleCount(Timeline.INDEFINITE);  
pt.setAutoReverse(true);  
pt.play(); // Start animation  
circle.setOnMousePressed(e -> pt.pause());  
circle.setOnMouseReleased(e -> pt.play());  
Scene scene = new Scene(pane, 250, 200);  
primaryStage.setTitle("PathTransitionDemo");  
primaryStage.setScene(scene);  
primaryStage.show();  
}  
public static void  
    main(String[] args) {  
    launch(args);
```



```
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.layout.Pane;
import javafx.scene.shape.Line;
import javafx.animation.PathTransition;
import javafx.scene.image.ImageView;
import javafx.util.Duration;
public class FlagRisingAnimation extends Application {
    @Override
    public void start(Stage primaryStage) {
        Pane pane = new Pane();
        ImageView imageView = new ImageView("us.jpg");
        pane.getChildren().add(imageView);
        PathTransition pt = new PathTransition(
            Duration.millis(10000),
            new Line(100, 200, 100, 0),
            imageView);
        pt.setCycleCount(5);
        pt.play(); // Start animation
        Scene scene = new Scene(pane, 250, 200);
        primaryStage.setScene(scene); primaryStage.show();
    }
}
```

(c) Paul Fodor and Pearson Inc.



FadeTransition

The **FadeTransition** class animates the change of the opacity in a node over a given time:

javafx.animation.FadeTransition

-duration: ObjectProperty<Duration>
-node: ObjectProperty<Node>
-fromValue: DoubleProperty
-toValue: DoubleProperty
-byValue: DoubleProperty

+FadeTransition()
+FadeTransition(duration: Duration)
+FadeTransition(duration: Duration,
node: Node)

The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.

The duration of this transition.

The target node of this transition.

The start opacity for this animation.

The stop opacity for this animation.

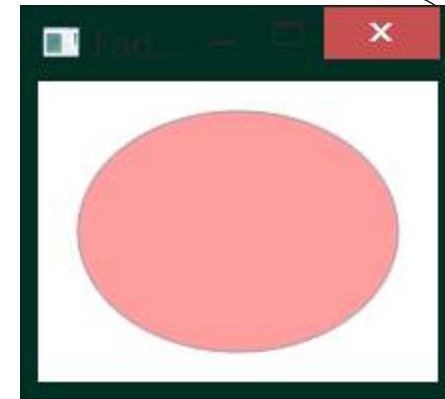
The incremental value on the opacity for this animation.

Creates an empty FadeTransition.

Creates a FadeTransition with the specified duration.

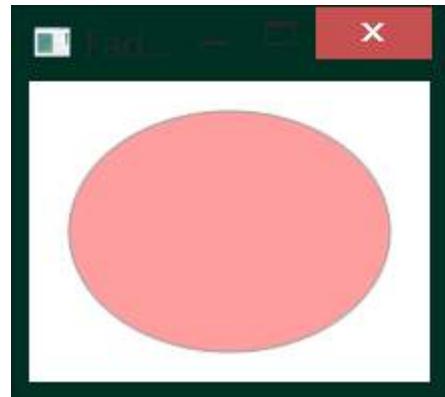
Creates a FadeTransition with the specified duration and node.

```
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.layout.Pane;
import javafx.scene.paint.Color;
import javafx.scene.shape.Ellipse;
import javafx.animation.FadeTransition;
import javafx.animation.Timeline;
import javafx.util.Duration;
public class FadeTransitionDemo extends Application {
    @Override
    public void start(Stage primaryStage) {
        Pane pane = new Pane();
        Ellipse ellipse = new Ellipse(10, 10, 100, 50);
        ellipse.setFill(Color.RED);
        ellipse.setStroke(Color.BLACK);
        ellipse.centerXProperty().bind(pane.widthProperty().divide(2));
        ellipse.centerYProperty().bind(pane.heightProperty().divide(2));
        ellipse.radiusXProperty().bind(pane.widthProperty().multiply(0.4));
        ellipse.radiusYProperty().bind(pane.heightProperty().multiply(0.4));
        pane.getChildren().add(ellipse);
        // Apply a fade transition to ellipse
        FadeTransition ft = new FadeTransition(Duration.millis(3000), ellipse);
        ft.setFromValue(1.0);
        ft.setToValue(0.1);
        ft.setCycleCount(Timeline.INDEFINITE);
        ft.setAutoReverse(true);
        ft.play(); // Start animation
        // Control animation
        ellipse.setOnMousePressed(e -> ft.pause());
        ellipse.setOnMouseReleased(e -> ft.play());
    }
}
```



```
// Create a scene and place it in the stage
Scene scene = new Scene(pane, 200, 150);
primaryStage.setTitle("FadeTransitionDemo"); // Set the stage title
primaryStage.setScene(scene); // Place the scene in the stage
primaryStage.show(); // Display the stage
}

public static void main(String[] args) {
    launch(args);
}
}
```



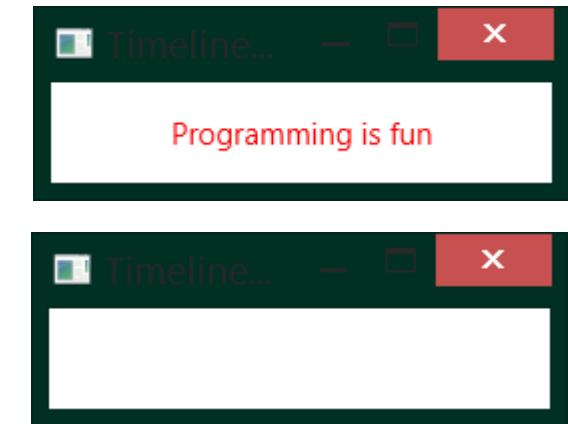
Timeline

- **PathTransition** and **FadeTransition** define specialized animations.
- The **javafx.animation.Timeline** class can be used to program any animation using one or more **javafx.animation.KeyFrames**
 - **KeyFrame** defines target values at a specified point in time for a set of variables that are interpolated along a **Timeline**.
 - Each **KeyFrame** is executed sequentially at a specified time interval.
 - **Timeline** inherits from **Animation**.

```

import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.layout.StackPane;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.scene.paint.Color;
import javafx.scene.text.Text;
import javafx.animation.Animation;
import javafx.animation.KeyFrame;
import javafx.animation.Timeline;
import javafx.util.Duration;
public class TimelineDemo extends Application {
    @Override
    public void start(Stage primaryStage) {
        StackPane pane = new StackPane();
        Text text = new Text(20, 50, "Programming if fun");
        text.setFill(Color.RED);
        pane.getChildren().add(text);
        // Create a handler for changing text
        EventHandler<ActionEvent> eH = e -> {
            if (text.getText().length() != 0) {
                text.setText("");
            } else {
                text.setText("Programming is fun");
            }
        };
        Timeline animation = new Timeline(new KeyFrame(Duration.millis(500), eH));
        animation.setCycleCount(Timeline.INDEFINITE);
        // Start animation
        animation.play();
    }
}

```



```
// Pause and resume animation
text.setOnMouseClicked(e -> {
    if (animation.getStatus() ==
        Animation.Status.PAUSED) {
        animation.play();
    } else {
        animation.pause();
    }
});
Scene scene = new Scene(pane, 250, 50);
primaryStage.setTitle("TimelineDemo");
primaryStage.setScene(scene);
primaryStage.show();
}

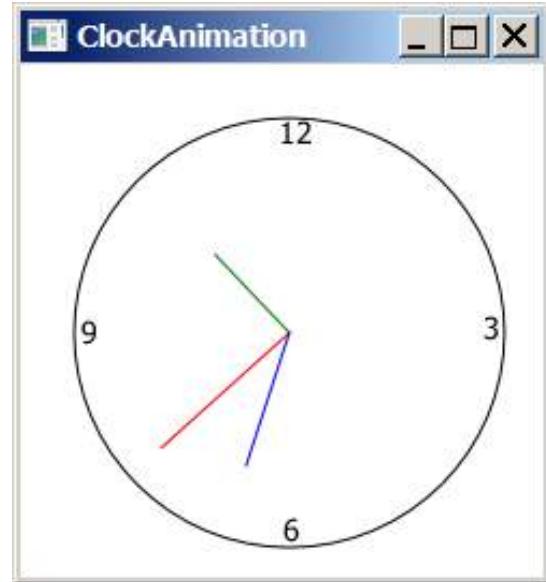
public static void main(String[] args) {
    launch(args);
}
}
```

```

import javafx.application.Application;
import javafx.stage.Stage;
import javafx.animation.KeyFrame;
import javafx.animation.Timeline;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.scene.Scene;
import javafx.util.Duration;
public class ClockAnimation extends Application {
    @Override
    public void start(Stage primaryStage) {
        ClockPane clock = new ClockPane(); // Create a clock
        // Create a handler for animation
        EventHandler<ActionEvent> eventHandler = e -> {
            clock.setCurrentTime(); // Set a new clock time
        };
        // Create an animation for a running clock
        Timeline animation = new Timeline(
            new KeyFrame(Duration.millis(1000), eventHandler));
        animation.setCycleCount(Timeline.INDEFINITE);
        animation.play(); // Start animation
        Scene scene = new Scene(clock, 250, 250);
        primaryStage.setTitle("ClockAnimation");
        primaryStage.setScene(scene);
        primaryStage.show();
    }
    public static void main(String[] args) {
        launch(args);
    }
}

```

Clock Animation



```
// ClockPane:  
import java.util.Calendar;  
import java.util.GregorianCalendar;  
import javafx.scene.layout.Pane;  
import javafx.scene.paint.Color;  
import javafx.scene.shape.Circle;  
import javafx.scene.shape.Line;  
import javafx.scene.text.Text;  
public class ClockPane extends Pane {  
    private int hour;  
    private int minute;  
    private int second;  
    // Clock pane's width and height  
    private double w = 250, h = 250;  
    public ClockPane() {  
        setCurrentTime();  
    }  
    public ClockPane(int hour, int minute, int second) {  
        this.hour = hour;  
        this.minute = minute;  
        this.second = second;  
        paintClock();  
    }  
    public int getHour() {  
        return hour;  
    }  
    public void setHour(int hour) {  
        this.hour = hour;  
        paintClock();  
    }  
}
```

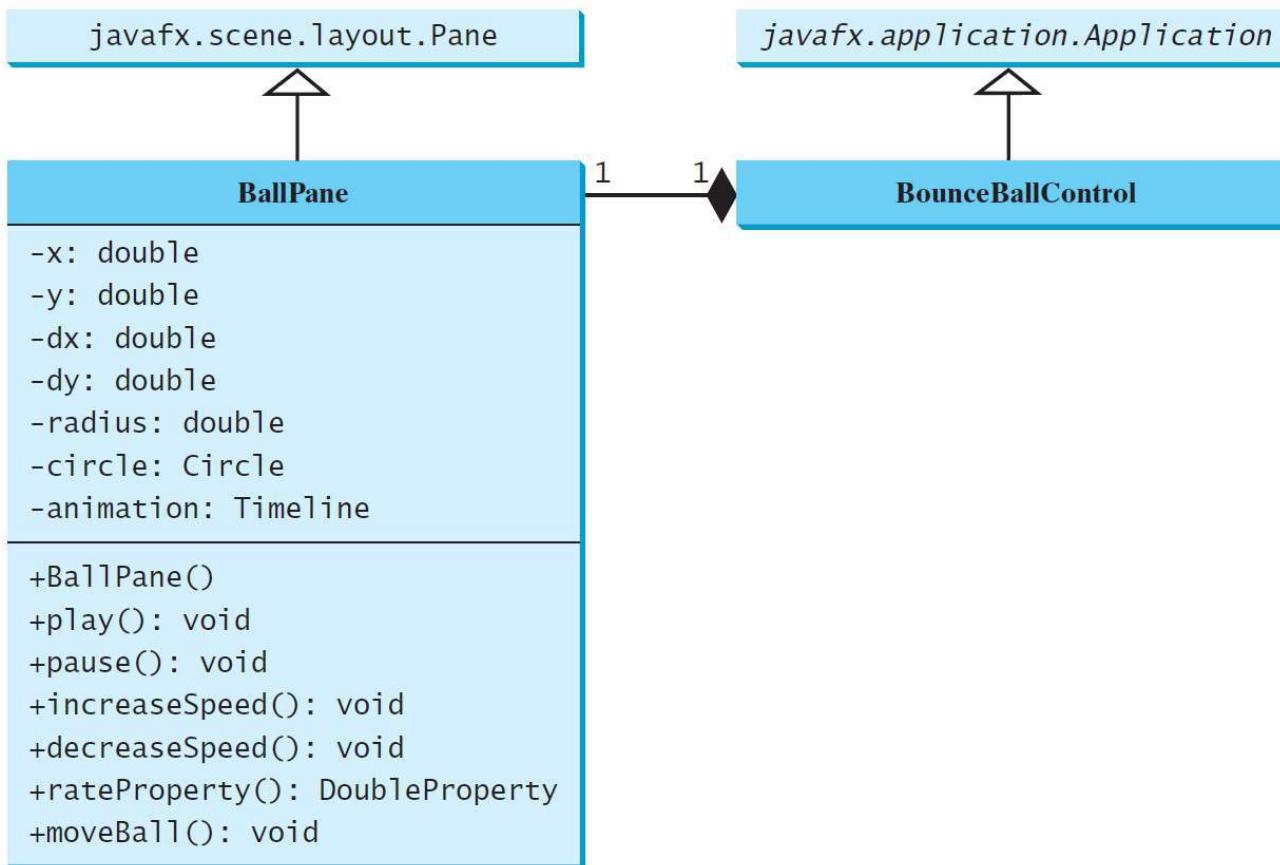
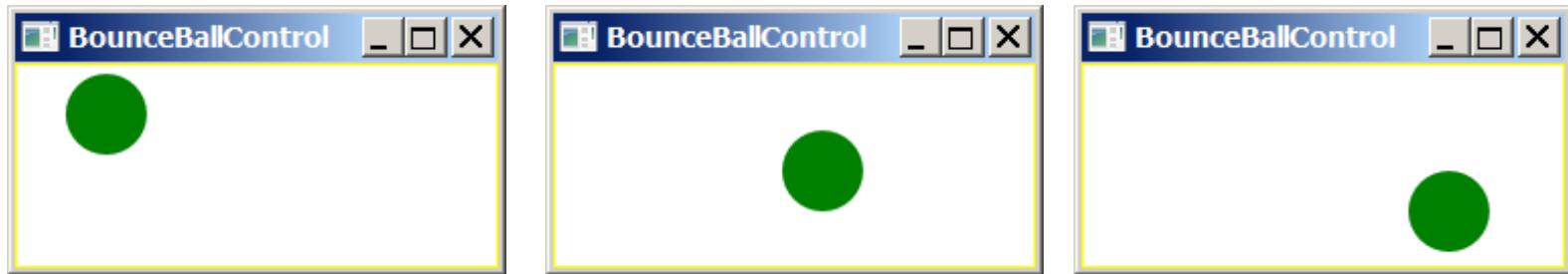
```
public int getMinute() {
    return minute;
}
public void setMinute(int minute) {
    this.minute = minute;
    paintClock();
}
public int getSecond() {
    return second;
}
public void setSecond(int second) {
    this.second = second;
    paintClock();
}
public double getW() {
    return w;
}
public void setW(double w) {
    this.w = w;
    paintClock();
}
public double getH() {
    return h;
}
public void setH(double h) {
    this.h = h;
    paintClock();
}
public void setCurrentTime() {
    Calendar calendar = new GregorianCalendar();
    this.hour = calendar.get(Calendar.HOUR_OF_DAY);
    this.minute = calendar.get(Calendar.MINUTE);
    this.second = calendar.get(Calendar.SECOND);
    paintClock(); // Repaint the clock
}
```

```

private void paintClock() {
    // Initialize clock parameters
    double clockRadius = Math.min(w, h) * 0.8 * 0.5;
    double centerX = w / 2;
    double centerY = h / 2;
    // Draw circle
    Circle circle = new Circle(centerX, centerY, clockRadius);
    circle.setFill(Color.WHITE);
    circle.setStroke(Color.BLACK);
    Text t1 = new Text(centerX - 5, centerY - clockRadius + 12, "12");
    Text t2 = new Text(centerX - clockRadius + 3, centerY + 5, "9");
    Text t3 = new Text(centerX + clockRadius - 10, centerY + 3, "3");
    Text t4 = new Text(centerX - 3, centerY + clockRadius - 3, "6");
    // Draw second hand
    double sLength = clockRadius * 0.8;
    double secondX = centerX + sLength * Math.sin(second * (2 * Math.PI / 60));
    double secondY = centerY - sLength * Math.cos(second * (2 * Math.PI / 60));
    Line sLine = new Line(centerX, centerY, secondX, secondY);
    sLine.setStroke(Color.RED);
    // Draw minute hand
    double mLength = clockRadius * 0.65;
    double xMinute = centerX + mLength * Math.sin(minute * (2 * Math.PI / 60));
    double minuteY = centerY - mLength * Math.cos(minute * (2 * Math.PI / 60));
    Line mLine = new Line(centerX, centerY, xMinute, minuteY);
    mLine.setStroke(Color.BLUE);
    // Draw hour hand
    double hLength = clockRadius * 0.5;
    double hourX = centerX + hLength * Math.sin((hour % 12 + minute / 60.0) * (2 * Math.PI / 12));
    double hourY = centerY - hLength * Math.cos((hour % 12 + minute / 60.0) * (2 * Math.PI / 12));
    Line hLine = new Line(centerX, centerY, hourX, hourY);
    hLine.setStroke(Color.GREEN);
    getChildren().clear();
    getChildren().addAll(circle, t1, t2, t3, t4, sLine, mLine, hLine);
}

```

Bouncing Ball



```
import javafx.scene.layout.Pane;
import javafx.animation.KeyFrame;
import javafx.animation.Timeline;
import javafx.beans.property.DoubleProperty;
import javafx.scene.paint.Color;
import javafx.scene.shape.Circle;
import javafx.util.Duration;
public class BallPane extends Pane {
    public final double radius = 20;
    private double x = radius, y = radius;
    private double dx = 1, dy = 1;
    private Circle circle = new Circle(x, y, radius);
    private Timeline animation;
    public BallPane() {
        circle.setFill(Color.GREEN); // Set ball color
        getChildren().add(circle); // Place a ball into this pane
        // Create an animation for moving the ball
        animation = new Timeline(new KeyFrame(Duration.millis(50), e -> moveBall()));
        animation.setCycleCount(Timeline.INDEFINITE);
        animation.play(); // Start animation
    }
    public void play() {
        animation.play();
    }
    public void pause() {
        animation.pause();
    }
    public void increaseSpeed() {
        animation.setRate(animation.getRate() + 0.1);
    }
    public void decreaseSpeed() {
        animation.setRate(
            animation.getRate() > 0 ? animation.getRate() - 0.1 : 0);
    }
}
```

```
public DoubleProperty rateProperty() {
    return animation.rateProperty();
}

protected void moveBall() {
    // Check boundaries
    if (x < radius || x > getWidth() - radius) {
        dx *= -1; // Change ball move direction
    }
    if (y < radius || y > getHeight() - radius) {
        dy *= -1; // Change ball move direction
    }

    // Adjust ball position
    x += dx;
    y += dy;
    circle.setCenterX(x);
    circle.setCenterY(y);
}
}
```

```
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.input.KeyCode;
public class BounceBallControl extends Application {
    @Override
    public void start(Stage primaryStage) {
        BallPane ballPane = new BallPane(); // Create a ball pane
        // Pause and resume animation
        ballPane.setOnMousePressed(e -> ballPane.pause());
        ballPane.setOnMouseReleased(e -> ballPane.play());
        // Increase and decrease animation
        ballPane.setOnKeyPressed(e -> {
            if (e.getCode() == KeyCode.UP) {
                ballPane.increaseSpeed();
            } else if (e.getCode() == KeyCode.DOWN) {
                ballPane.decreaseSpeed();
            }
        });
        Scene scene = new Scene(ballPane, 250, 150);
        primaryStage.setTitle("BounceBallControl");
        primaryStage.setScene(scene);
        primaryStage.show();
        // Must request focus after the primary stage is displayed
        ballPane.requestFocus();
    }

    public static void main(String[] args) {
        launch(args);
    }
}
```

JavaFX support for mobile devices

- JavaFX has event programming support for mobile devices:

`javafx.scene.inputSwipeEvent,`
`javafx.scene.inputTouchEvent,`
`javafx.scene.inputZoomEvent.`

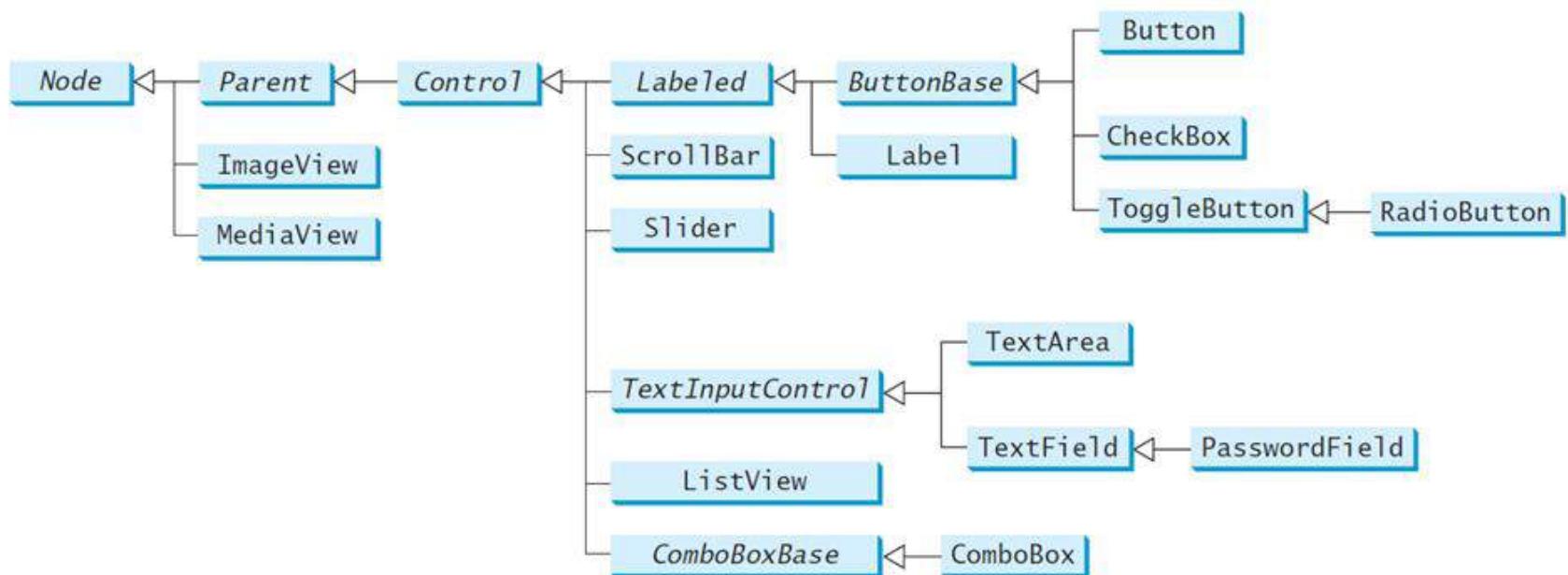
- Example:

<http://docs.oracle.com/javase/8/javafx/events-tutorial/gestureeventsjava.htm>

<http://docs.oracle.com/javase/8/javafx/events-tutorial/toucheventsjava.htm>

Control Nodes

- Input control nodes:



Labeled class

- A label is a display area for a short text, a node, or both
 - It is often used to label other controls (usually text fields)
 - Labels and buttons share many common properties: these common properties are defined in the Labeled class

javafx.scene.control.Labeled

```
-alignment: ObjectProperty<Pos>
-contentDisplay:
    ObjectProperty<ContentDisplay>
-graphic: ObjectProperty<Node>
-graphicTextGap: DoubleProperty
-textFill: ObjectProperty<Paint>
-text: StringProperty
-underline: BooleanProperty
-wrapText: BooleanProperty
```

The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.

Specifies the alignment of the text and node in the labeled.
Specifies the position of the node relative to the text using the constants TOP, BOTTOM, LEFT, and RIGHT defined in ContentDisplay.
A graphic for the labeled.
The gap between the graphic and the text.
The paint used to fill the text.
A text for the labeled.
Whether text should be underlined.
Whether text should be wrapped if the text exceeds the width.

Label class

javafx.scene.control.Labeled



javafx.scene.control.Label

+Label()
+Label(text: String)
+Label(text: String, graphic: Node)

Creates an empty label.

Creates a label with the specified text.

Creates a label with the specified text and graphic.

```

import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.control.ContentDisplay;
import javafx.scene.control.Label;
import javafx.scene.image.Image;
import javafx.scene.image.ImageView;
import javafx.scene.layout.HBox;
import javafx.scene.layout.StackPane;
import javafx.scene.paint.Color;
import javafx.scene.shape.Circle;
import javafx.scene.shape.Rectangle;
import javafx.scene.shape.Ellipse;
public class LabelWithGraphic extends Application {
    @Override
    public void start(Stage primaryStage) {
        ImageView us = new ImageView(new Image("us.jpg"));
        Label lb1 = new Label("US\n50 States", us);
        lb1.setStyle("-fx-border-color: green; -fx-border-width: 2");
        lb1.setContentDisplay(ContentDisplay.BOTTOM);
        lb1.setTextFill(Color.RED);
        Label lb2 = new Label("Circle", new Circle(50, 50, 25));
        lb2.setContentDisplay(ContentDisplay.TOP);
        lb2.setTextFill(Color.ORANGE);
        Label lb3 = new Label("Retangle", new Rectangle(10, 10, 50, 25));
        lb3.setContentDisplay(ContentDisplay.RIGHT);
        Label lb4 = new Label("Ellipse", new Ellipse(50, 50, 50, 25));
        lb4.setContentDisplay(ContentDisplay.LEFT);
    }
}

```



```
Ellipse ellipse = new Ellipse(50, 50, 50, 25);
ellipse.setStroke(Color.GREEN);
ellipse.setFill(Color.WHITE);
StackPane stackPane = new StackPane();
stackPane.getChildren().addAll(ellipse, new Label("JavaFX"));
Label lb5 = new Label("A pane inside a label", stackPane);
lb5.setContentDisplay(ContentDisplay.BOTTOM);

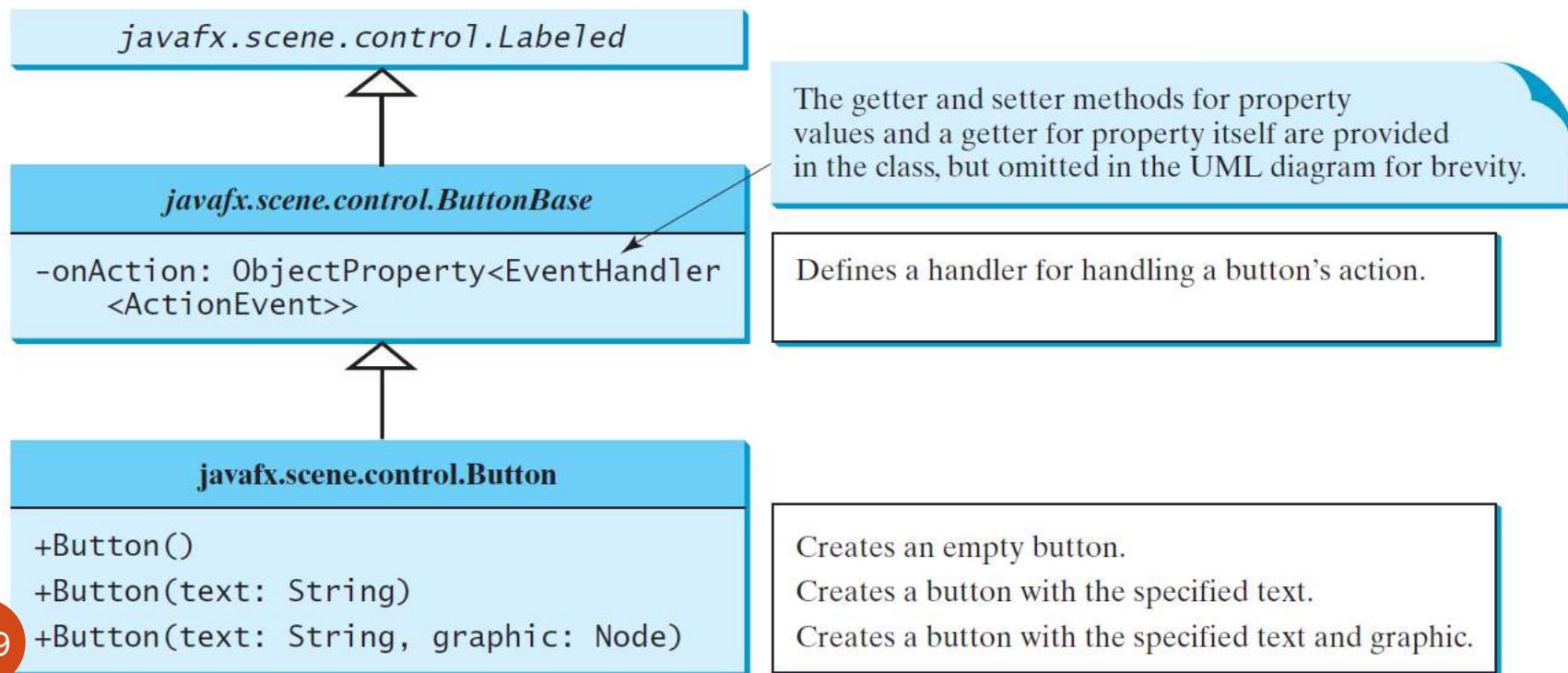
HBox pane = new HBox(20);
pane.getChildren().addAll(lb1, lb2, lb3, lb4, lb5);

Scene scene = new Scene(pane, 700, 150);
primaryStage.setTitle("LabelWithGraphic");
primaryStage.setScene(scene);
primaryStage.show();
}

public static void main(String[] args) {
    launch(args);
}
}
```

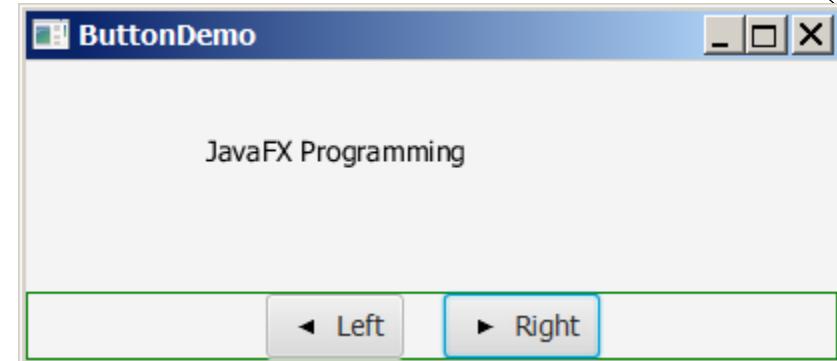
ButtonBase and Button

- A button is a control that triggers an action event when clicked.
- JavaFX provides regular buttons, toggle buttons, check box buttons, and radio buttons.
- The common features of these buttons are defined in `ButtonBase` and `Labeled` classes.



```
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.geometry.Pos;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.image.ImageView;
import javafx.scene.layout.BorderPane;
import javafx.scene.layout.HBox;
import javafx.scene.layout.Pane;
import javafx.scene.text.Text;

public class ButtonDemo extends Application {
    @Override
    public void start(Stage primaryStage) {
        Scene scene = new Scene(getPane(), 450, 200);
        primaryStage.setTitle("ButtonDemo");
        primaryStage.setScene(scene);
        primaryStage.show();
    }
    protected Text text = new Text(50, 50, "JavaFX Programming");
    protected BorderPane getPane() {
        HBox paneForButtons = new HBox(20);
        Button btLeft = new Button("Left", new ImageView("image/left.gif"));
        Button btRight = new Button("Right", new ImageView("image/right.gif"));
        paneForButtons.getChildren().addAll(btLeft, btRight);
        paneForButtons.setAlignment(Pos.CENTER);
        paneForButtons.setStyle("-fx-border-color: green");
        BorderPane pane = new BorderPane();
        pane.setBottom(paneForButtons);
    }
}
```



```
Pane paneForText = new Pane();
paneForText.getChildren().add(text);
pane.setCenter(paneForText);

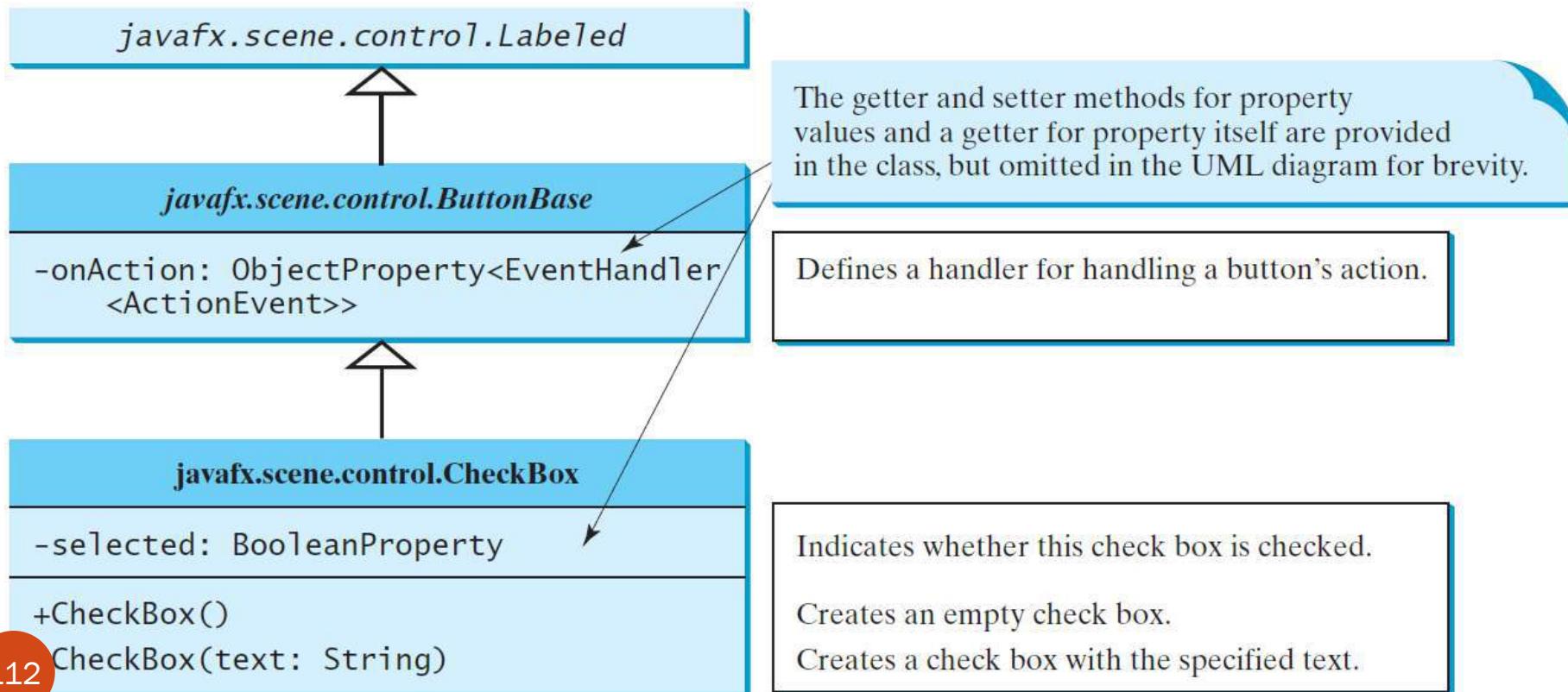
btLeft.setOnAction(e -> text.setX(text.getX() - 10));
btRight.setOnAction(e -> text.setX(text.getX() + 10));

return pane;
}

public static void main(String[] args) {
    launch(args);
}
}
```

CheckBox

- A CheckBox is used for the user to make a selection (square box).
- CheckBox inherits all the properties from ButtonBase and Labeled: onAction, text, graphic, alignment, graphicTextGap, textFill, contentDisplay.



```

import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.geometry.Insets;
import javafx.scene.control.CheckBox;
import javafx.scene.layout.BorderPane;
import javafx.scene.layout.VBox;
import javafx.scene.text.Font;
import javafx.scene.text.FontPosture;
import javafx.scene.text.FontWeight;
public class CheckBoxDemo extends ButtonDemo {
    @Override // Override the getPane() method in the super class
    protected BorderPane getPane() {
        BorderPane pane = super.getPane();

        Font fontBoldItalic = Font.font("Times New Roman",
            FontWeight.BOLD, FontPosture.ITALIC, 20);
        Font fontBold = Font.font("Times New Roman",
            FontWeight.BOLD, FontPosture.REGULAR, 20);
        Font fontItalic = Font.font("Times New Roman",
            FontWeight.NORMAL, FontPosture.ITALIC, 20);
        Font fontNormal = Font.font("Times New Roman",
            FontWeight.NORMAL, FontPosture.REGULAR, 20);

        text.setFont(fontNormal);

        VBox paneForCheckBoxes = new VBox(20);
        paneForCheckBoxes.setPadding(new Insets(5, 5, 5, 5));
        paneForCheckBoxes.setStyle("-fx-border-color: green");

```



```

CheckBox chkBold = new CheckBox("Bold");
CheckBox chkItalic = new CheckBox("Italic");
paneForCheckboxes.getChildren().addAll(chkBOLD, chkITALIC);
pane.setRight(paneForCheckboxes);

EventHandler<ActionEvent> handler = e -> {
    if (chkBold.isSelected() && chkItalic.isSelected()) {
        text.setFont(fontBoldItalic); // Both check boxes checked
    } else if (chkBold.isSelected()) {
        text.setFont(fontBold); // The Bold check box checked
    } else if (chkItalic.isSelected()) {
        text.setFont(fontItalic); // The Italic check box checked
    } else {
        text.setFont(fontNormal); // Both check boxes unchecked
    }
};

chkBold.setOnAction(handler);
chkItalic.setOnAction(handler);

return pane; // Return a new pane
}

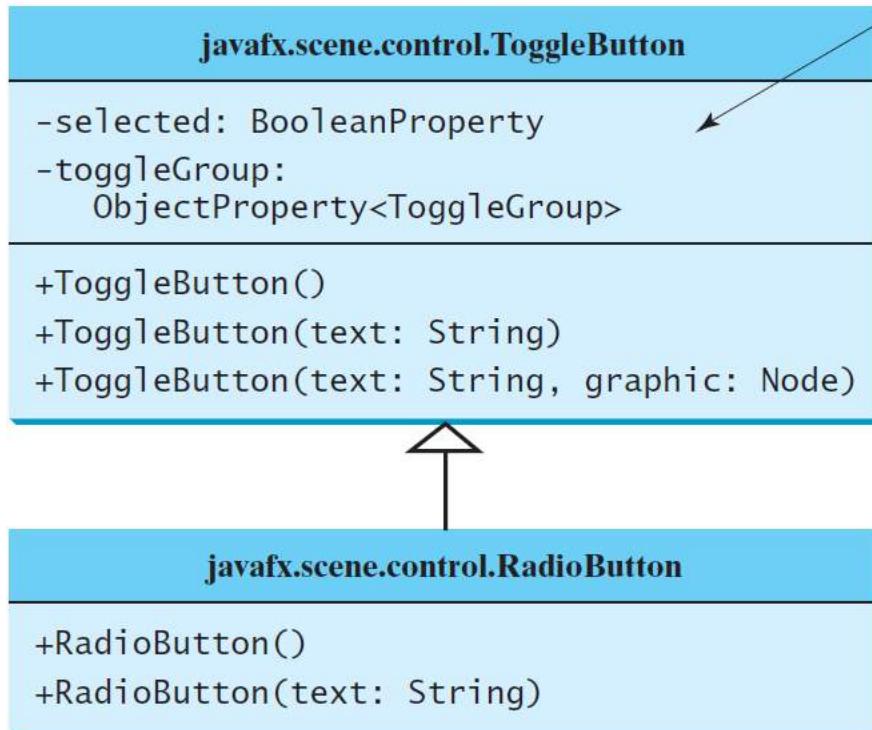
// the start method is inherited from the superclass ButtonDemo

public static void main(String[] args) {
    launch(args);
}

```

RadioButton

- Radio buttons allow to choose a **single** item from a group of choices.
 - Radio buttons display a circle that is either filled (if selected) or blank (if not selected).



The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.

Indicates whether the button is selected.
Specifies the button group to which the button belongs.

Creates an empty toggle button.
Creates a toggle button with the specified text.
Creates a toggle button with the specified text and graphic.

Creates an empty radio button.
Creates a radio button with the specified text.

```

import static javafx.application.Application.launch;
import javafx.geometry.Insets;
import javafx.scene.control.RadioButton;
import javafx.scene.control.ToggleGroup;
import javafx.scene.layout.BorderPane;
import javafx.scene.layout.VBox;
import javafx.scene.paint.Color;

public class RadioButtonDemo extends CheckBoxDemo {
    @Override // Override the getPane() method in the super class
    protected BorderPane getPane() {
        BorderPane pane = super.getPane();
        VBox paneForRadioButtons = new VBox(20);
        paneForRadioButtons.setPadding(new Insets(5, 5, 5, 5));
        paneForRadioButtons.setStyle("-fx-border-color: green");
        RadioButton rbRed = new RadioButton("Red");
        RadioButton rbGreen = new RadioButton("Green");
        RadioButton rbBlue = new RadioButton("Blue");
        paneForRadioButtons.getChildren().addAll(rbRed, rbGreen, rbBlue);
        pane.setLeft(paneForRadioButtons);
        ToggleGroup group = new ToggleGroup();
        rbRed.setToggleGroup(group);
        rbGreen.setToggleGroup(group);
        rbBlue.setToggleGroup(group);

        rbRed.setOnAction(e -> {
            if (rbRed.isSelected()) {
                text.setFill(Color.RED);
            }
        });
    }
}

```



```
rbGreen.setOnAction(e -> {
    if (rbGreen.isSelected()) {
        text.setFill(Color.GREEN);
    }
}) ;

rbBlue.setOnAction(e -> {
    if (rbBlue.isSelected()) {
        text.setFill(Color.BLUE);
    }
}) ;

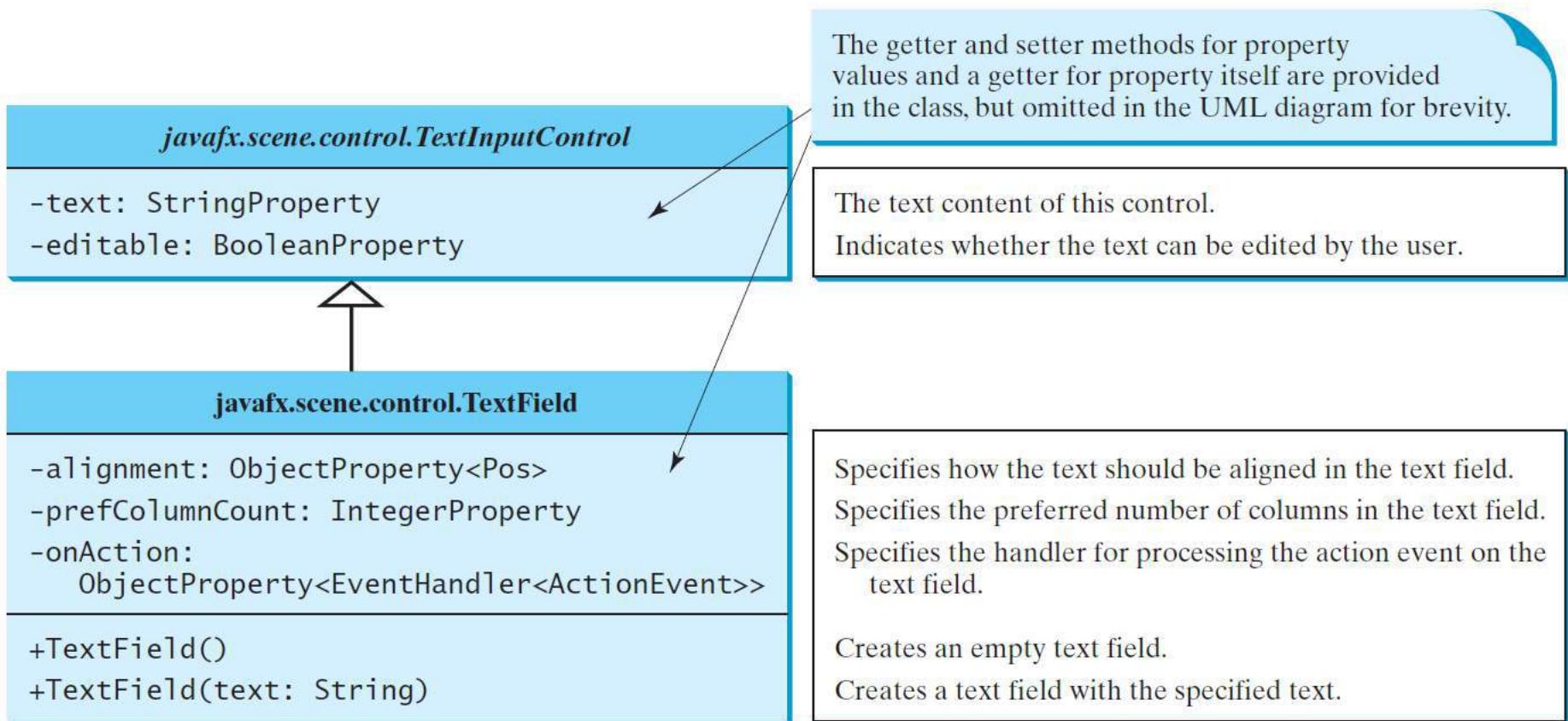
return pane;
}

// the start method is inherited from the superclass ButtonDemo

public static void main(String[] args) {
    launch(args);
}
}
```

TextField

- A text field can be used to enter or display a string. TextField is a subclass of TextInputControl.



```

import static javafx.application.Application.launch;
import javafx.geometry.Insets;
import javafx.geometry.Pos;
import javafx.scene.control.Label;
import javafx.scene.control.TextField;
import javafx.scene.layout.BorderPane;
public class TextFieldDemo extends RadioButtonDemo {
    @Override
    protected BorderPane getPane() {
        BorderPane pane = super.getPane();

        BorderPane paneForTextField = new BorderPane();
        paneForTextField.setPadding(new Insets(5, 5, 5, 5));
        paneForTextField.setStyle("-fx-border-color: green");
        paneForTextField.setLeft(new Label("Enter a new message: "));

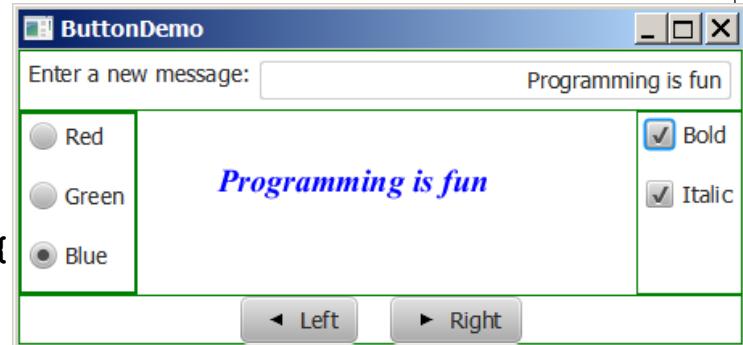
        TextField tf = new TextField();
        tf.setAlignment(Pos.BOTTOM_RIGHT);
        paneForTextField.setCenter(tf);
        pane.setTop(paneForTextField);

        tf.setOnAction(e -> text.setText(tf.getText()));

        return pane;
    }

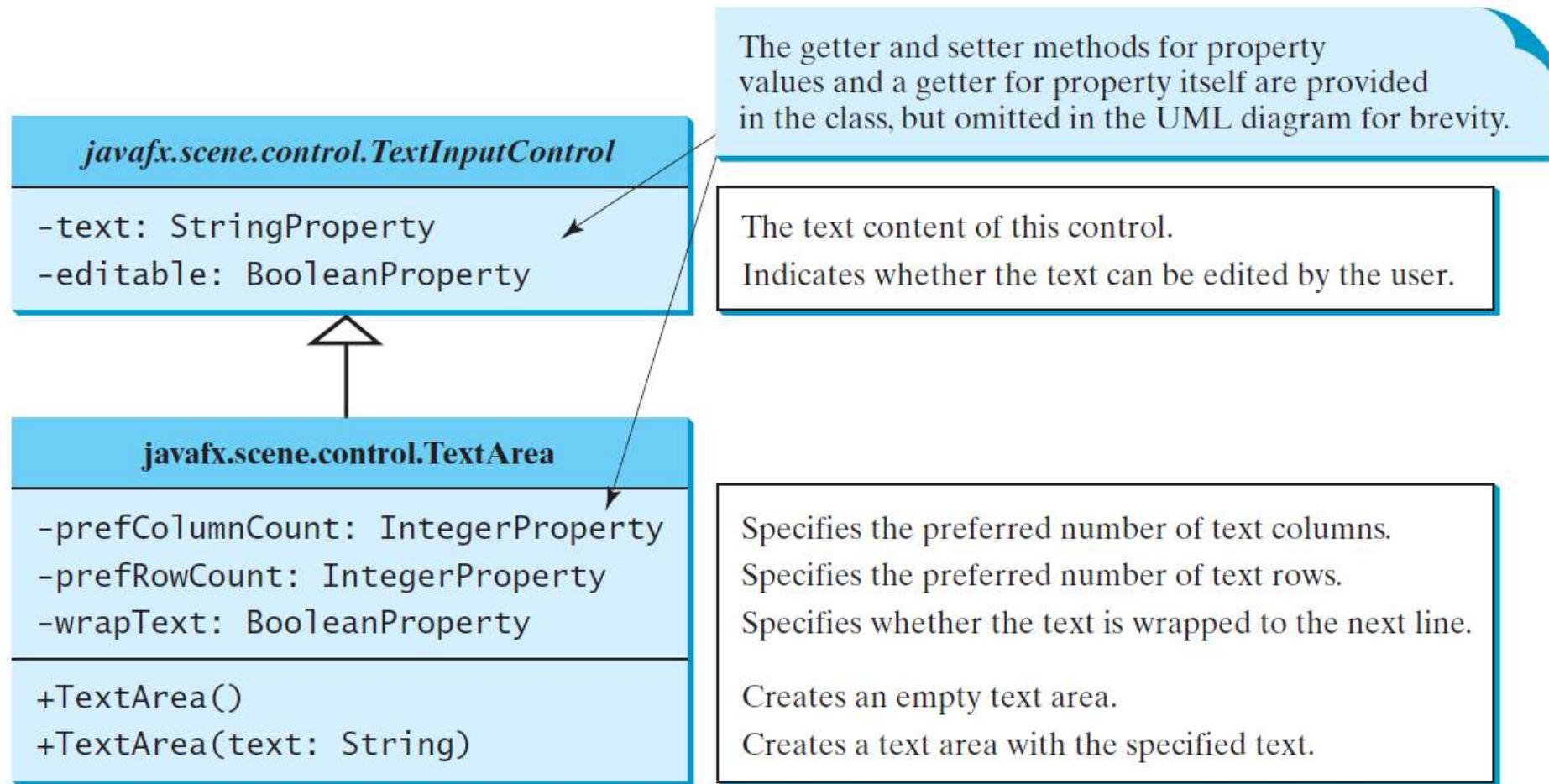
    public static void main(String[] args) {
        launch(args);
    }
}

```



TextArea

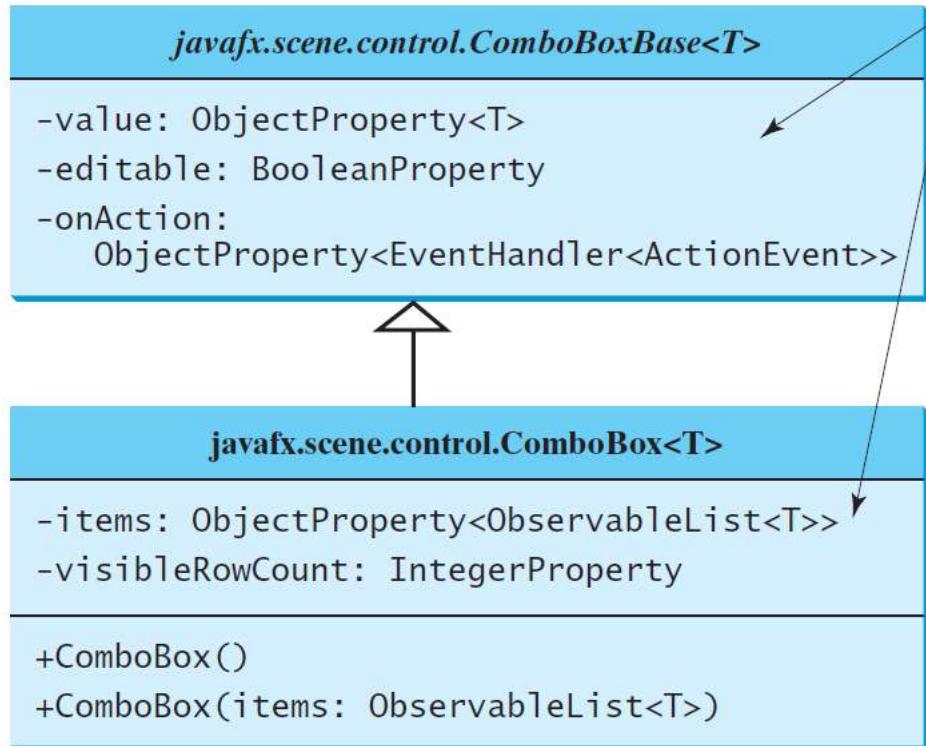
- A TextArea enables the user to enter multiple lines of text.



ComboBox



- A combo box, also known as a choice list or drop-down list, contains a list of items from which the user can choose.



The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.

The value selected in the combo box.
Specifies whether the combo box allows user input.
Specifies the handler for processing the action event.

The items in the combo box popup.
The maximum number of visible rows of the items in the combo box popup.
Creates an empty combo box.
Creates a combo box with the specified items.

ListView

- A list view is a component that performs basically the same function as a combo box, but it enables the user to choose a single value or **multiple values**.

`javafx.scene.control.ListView<T>`

```
-items: ObjectProperty<ObservableList<T>>
-orientation: BooleanProperty
-selectionModel:
    ObjectProperty<MultipleSelectionModel<T>>

+ListView()
+ListView(items: ObservableList<T>)
```

The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.

The items in the list view.

Indicates whether the items are displayed horizontally or vertically in the list view.

Specifies how items are selected. The `SelectionMode` is also used to obtain the selected items.

Creates an empty list view.

Creates a list view with the specified items.

```

import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.control.ListView;
import javafx.scene.control.ScrollPane;
import javafx.scene.control.SelectionMode;
import javafx.scene.image.ImageView;
import javafx.scene.layout.BorderPane;
import javafx.scene.layout.FlowPane;
import javafx.collections.FXCollections;

public class ListViewDemo extends Application {
    // Declare an array of Strings for flag titles
    private String[] flagTitles = {"United States of America", "Canada", "China",
        "Denmark", "France", "Germany", "India"};
    // Declare an ImageView array for the national flags
    private ImageView[] ImageViews = {
        new ImageView("image/us.gif"),
        new ImageView("image/ca.gif"),
        new ImageView("image/china.gif"),
        new ImageView("image/denmark.gif"),
        new ImageView("image/fr.gif"),
        new ImageView("image/germany.gif"),
        new ImageView("image/india.gif")
    };
    @Override
    public void start(Stage primaryStage) {
        ListView<String> lv = new ListView<>(FXCollections
            .observableArrayList(flagTitles));
        lv.setPrefSize(400, 400);
    }
}

```



```
lv.getSelectionModel().setSelectionMode(SelectionMode.MULTIPLE);

// Create a pane to hold image views
FlowPane imagePane = new FlowPane(10, 10);
BorderPane pane = new BorderPane();
pane.setLeft(new ScrollPane(lv));
pane.setCenter(imagePane);

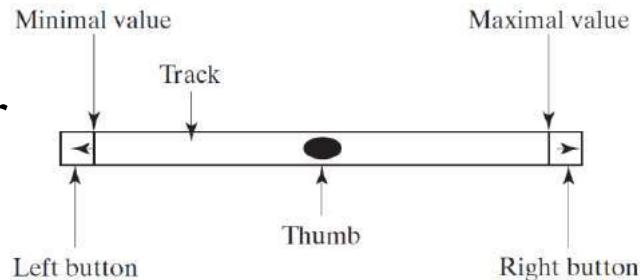
lv.getSelectionModel().selectedItemProperty().addListener(
    ov -> {
    imagePane.getChildren().clear();
    for (Integer i: lv.getSelectionModel().getSelectedIndices()) {
        imagePane.getChildren().add(ImageViews[i]);
    }
});

Scene scene = new Scene(pane, 450, 170);
primaryStage.setTitle("ListViewDemo");
primaryStage.setScene(scene);
primaryStage.show();
}

public static void main(String[] args) {
    launch(args);
}
}
```

ScrollBar

- A scroll bar is a control that enables the user to select from a range of values. The scrollbar appears in two styles: horizontal and vertical.



javafx.scene.control.ScrollBar

-blockIncrement: DoubleProperty	
-max: DoubleProperty	
-min: DoubleProperty	
-unitIncrement: DoubleProperty	
-value: DoubleProperty	The amount to adjust the scroll bar if the track of the bar is clicked (default: 10).
-visibleAmount: DoubleProperty	The maximum value represented by this scroll bar (default: 100).
-orientation: ObjectProperty<Orientation>	The minimum value represented by this scroll bar (default: 0).
+ScrollBar()	The amount to adjust the scroll bar when the <code>increment()</code> and <code>decrement()</code> methods are called (default: 1).
+increment()	Current value of the scroll bar (default: 0).
+decrement()	The width of the scroll bar (default: 15).

The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.

Specifies the orientation of the scroll bar (default: HORIZONTAL).

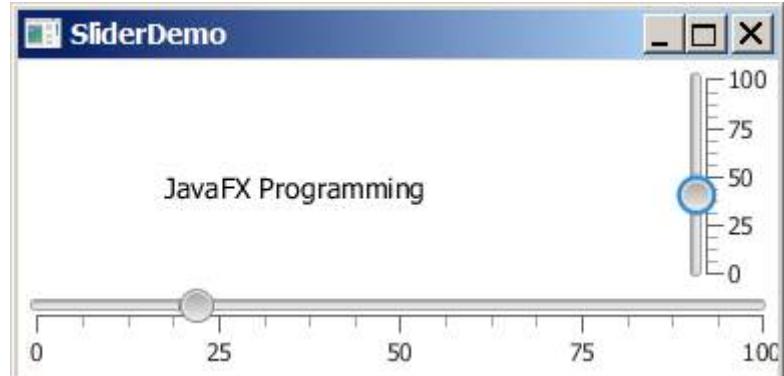
Creates a default horizontal scroll bar.

Increments the value of the scroll bar by `unitIncrement`.

Decrements the value of the scroll bar by `unitIncrement`.

Slider

- Slider is similar to ScrollBar, but Slider has more properties and can appear in many forms.



`javafx.scene.control.Slider`

```
-blockIncrement: DoubleProperty  
-max: DoubleProperty  
-min: DoubleProperty  
-value: DoubleProperty  
-orientation: ObjectProperty<Orientation>  
-majorTickUnit: DoubleProperty  
-minorTickCount: IntegerProperty  
-showTickLabels: BooleanProperty  
-showTickMarks: BooleanProperty  
  
+Slider()  
+Slider(min: double, max: double,  
        value: double)
```

The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.

The amount to adjust the slider if the track of the bar is clicked (default: 10).
The maximum value represented by this slider (default: 100).
The minimum value represented by this slider (default: 0).
Current value of the slider (default: 0).
Specifies the orientation of the slider (default: HORIZONTAL).
The unit distance between major tick marks.
The number of minor ticks to place between two major ticks.
Specifies whether the labels for tick marks are shown.
Specifies whether the tick marks are shown.

Creates a default horizontal slider.
Creates a slider with the specified min, max, and value.

Media

- The Media class is used to obtain the source of a media type.
- The MediaPlayer class is used to play and control the media.
- The MediaView class is used to display video.

javafx.scene.media.Media

```
-duration: ReadOnlyObjectProperty<Duration>
-width: ReadOnlyIntegerProperty
-height: ReadOnlyIntegerProperty

+Media(source: String)
```

The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.

The durations in seconds of the source media.

The width in pixels of the source video.

The height in pixels of the source video.

Creates a **Media** from a URL source.

MediaPlayer

- The MediaPlayer class plays and controls media with properties: `autoPlay`, `currentCount`, `cycleCount`, `mute`, `volume`, and `totalDuration`.

`javafx.scene.media.MediaPlayer`

```
-autoPlay: BooleanProperty  
-currentCount: ReadOnlyIntegerProperty  
-cycleCount: IntegerProperty  
-mute: BooleanProperty  
-volume: DoubleProperty  
-totalDuration:  
    ReadOnlyObjectProperty<Duration>
```

```
+MediaPlayer(media: Media)  
+play(): void  
+pause(): void  
+seek(): void
```

The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.

Specifies whether the playing should start automatically.
The number of completed playback cycles.
Specifies the number of time the media will be played.
Specifies whether the audio is muted.
The volume for the audio.
The amount of time to play the media from start to finish.

Creates a player for a specified media.
Plays the media.
Pauses the media.
Seeks the player to a new playback time.

MediaView

- The MediaView class is a subclass of Node that provides a view of the Media being played by a MediaPlayer.
 - The MediaView class provides the properties for viewing the media.

javafx.scene.media.MediaView

-x: DoubleProperty
-y: DoubleProperty
-mediaPlayer:
 ObjectProperty<MediaPlayer>
-fitWidth: DoubleProperty
-fitHeight: DoubleProperty

+MediaView()
+MediaView(mediaPlayer: MediaPlayer)

The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.

Specifies the current x-coordinate of the media view.

Specifies the current y-coordinate of the media view.

Specifies a media player for the media view.

Specifies the width of the view for the media to fit.

Specifies the height of the view for the media to fit.

Creates an empty media view.

Creates a media view with the specified media player.

Example: Using Media

- This example displays a video in a view: use the play/pause button to play or pause the video and use the rewind button to restart the video, and use the slider to control the volume of the audio.



media: Media

mediaPlayer: MediaPlayer

mediaView: MediaView

```
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.control.Slider;
import javafx.scene.layout.BorderPane;
import javafx.scene.layout.HBox;
import javafx.scene.layout.Region;
import javafx.scene.media.Media;
import javafx.scene.media.MediaPlayer;
import javafx.scene.media.MediaView;
import javafx.geometry.Pos;
import javafx.util.Duration;
public class MediaDemo extends Application {
    private static final String MEDIA_URL = "sample.mp4";
    @Override
    public void start(Stage primaryStage) {
        Media media = new Media(MEDIA_URL);
        MediaPlayer mediaPlayer = new MediaPlayer(media);
        MediaView mediaView = new MediaView(mediaPlayer);
        Button playButton = new Button(">");
        playButton.setOnAction(e -> {
            if (playButton.getText().equals(">")) {
                mediaPlayer.play();
                playButton.setText("||");
            } else {
                mediaPlayer.pause();
                playButton.setText(">");
            }
        });
    }
});
```

```

        Button rewindButton = new Button("<<") ;
        rewindButton.setOnAction(e -> mediaPlayer.seek(Duration.ZERO)) ;
        Slider slVolume = new Slider() ;
        slVolume.setPrefWidth(150) ;
        slVolume.setMaxWidth(Region.USE_PREF_SIZE) ;
        slVolume.setMinWidth(30) ;
        slVolume.setValue(50) ;
        mediaPlayer.volumeProperty().bind(slVolume.valueProperty().divide(100)) ;
        HBox hBox = new HBox(10) ;
        hBox.setAlignment(Pos.CENTER) ;
        hBox.getChildren().addAll(playButton, rewindButton,
                               new Label("Volume"), slVolume) ;
        BorderPane pane = new BorderPane() ;
        pane.setCenter(mediaView) ;
        pane.setBottom(hBox) ;

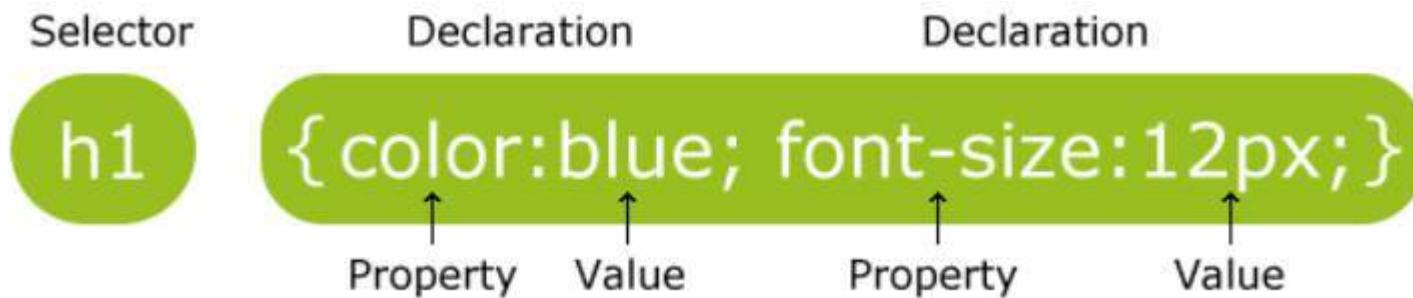
        Scene scene = new Scene(pane, 650, 500) ;
        primaryStage.setTitle("MediaDemo") ;
        primaryStage.setScene(scene) ;
        primaryStage.show() ;
    }

    public static void main(String[] args) {
        launch(args) ;
    }
}

```

CSS

- Cascading Style Sheets (CSS) is a language used for describing the look and formatting of a document.
 - CSS is designed primarily to enable the separation of document content from document presentation (layout, colors, and fonts).
 - It is used to style web pages and user interfaces written in HTML, XHTML, and any kind of XML document.
 - The CSS language specifications are Web standards maintained by the World Wide Web Consortium (W3C).
 - CSS rule set example:



JavaFX CSS

- JavaFX Cascading Style Sheets (CSS) is based on the W3C CSS and allows to customize and develop themes for JavaFX controls and scene graph objects
 - <http://docs.oracle.com/javafx/2/api/javafx/scene/doc-files/cssref.html>
 - JavaFX uses the prefix "-fx-" to define its vendor CSS properties (separate from W3C CSS).
- A style sheet uses the style class or style id to define styles.
 - Multiple style classes can be applied to a single node and a style id to a unique node.
 - The syntax `.styleclass` defines a style class.
 - The syntax `#styleid` defines a style id.

Style Class and Style ID

- `mystyle.css:`

```
.plaincircle {  
    -fx-fill: white;  
    -fx-stroke: black;  
}  
.circleborder {  
    -fx-stroke-width: 5;  
    -fx-stroke-dash-array: 12 2 4 2;  
}  
.border {  
    -fx-border-color: black;  
    -fx-border-width: 5;  
}  
#redcircle {  
    -fx-fill: red;  
    -fx-stroke: red;  
}  
#greencircle {  
    -fx-fill: green;  
    -fx-stroke: green;  
}
```

```
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.layout.HBox;
import javafx.scene.layout.Pane;
import javafx.scene.shape.Circle;
public class StyleSheetDemo extends Application {
    @Override
    public void start(Stage primaryStage) {
        HBox hBox = new HBox(5);
        Scene scene = new Scene(hBox, 300, 250);
        // Load the stylesheet
        scene.getStylesheets().add("mystyle.css");
        Pane panel = new Pane();
        Circle circle1 = new Circle(50, 50, 30);
        Circle circle2 = new Circle(150, 50, 30);
        Circle circle3 = new Circle(100, 100, 30);
        panel.getChildren().addAll(circle1, circle2, circle3);
        panel.getStyleClass().add("border");
        circle1.getStyleClass().add("plaincircle"); // Add a style class
        circle2.getStyleClass().add("plaincircle"); // Add a style class
        circle3.setId("redcircle"); // Add a style id
        Pane pane2 = new Pane();
        Circle circle4 = new Circle(100, 100, 30);
```

```

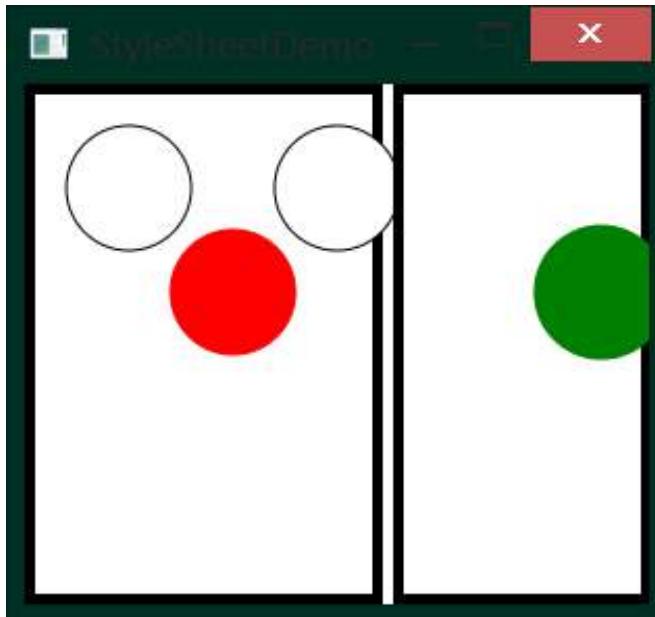
        circle4.getStyleClass().addAll("circleborder", "plainCircle");
        circle4.setId("greencircle"); // Add a style class
        pane2.getChildren().add(circle4);
        pane2.getStyleClass().add("border");

        hBox.getChildren().addAll(panel1, pane2);

        primaryStage.setTitle("StyleSheetDemo");
        primaryStage.setScene(scene);
        primaryStage.show();
    }

    // Launch the program from command-line
    public static void main(String[] args) {
        launch(args);
    }
}

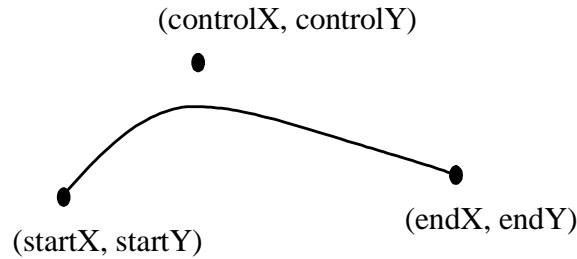
```



QuadCurve

- A quadratic curve is mathematically defined as a quadratic polynomial.

```
QuadCurve(double startX, double startY,  
double controlX, double controlY, double  
endX, double endY)
```



QuadCurve

javafx.scene.shape.QuadCurve
-startX: DoubleProperty
-startY: DoubleProperty
-endX: DoubleProperty
-endY: DoubleProperty
-controlX: DoubleProperty
-controlY: DoubleProperty
+QuadCurve ()
+QuadCurve (startX: double, startY: double, controlX: double, controlY: double, endX: double, endY: double)

The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.

The x-coordinate of the start point (default 0).

The y-coordinate of the start point (default 0)..

The x-coordinate of the end point (default 0)..

The y-coordinate of the end point (default 0)..

The x-coordinate of the control point (default 0)..

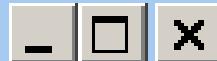
The y-coordinate of the control point (default 0)..

Creates an empty quad curve.

Creates a quad curve with the specified arguments.

Menus

- Menus make selection easier and are widely used in window applications.
 - JavaFX provides five classes that implement menus: MenuBar, Menu, MenuItem, CheckMenuItem, and RadioButtonMenuItem.
- MenuBar is a top-level menu component used to hold the menus.
 - A menu consists of menu items that the user can select (or toggle on or off).
 - A menu item can be an instance of MenuItem, CheckMenuItem, or RadioButtonMenuItem.
 - Menu items can be associated with nodes and keyboard accelerators.

**Operation** [Exit](#)

<u>Add</u>	Ctrl-A
<u>Subtract</u>	Ctrl-S
<u>Multiply</u>	Ctrl-M
<u>Divide</u>	Ctrl-D

Number 1 Number 2 Result

```
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.control.Menu;
import javafx.scene.controlMenuBar;
import javafx.scene.controlMenuItem;
import javafx.scene.control.TextField;
import javafx.scene.input.KeyCombination;
import javafx.scene.layout.HBox;
import javafx.scene.layout.VBox;
import javafx.geometry.Pos;

public class MenuDemo extends Application {
    private TextField tfNumber1 = new TextField();
    private TextField tfNumber2 = new TextField();
    private TextField tfResult = new TextField();
    (c) Paul Fodor and Pearson Inc.
```

```
@Override
public void start(Stage primaryStage) {
    MenuBar menuBar = new MenuBar();

    Menu menuOperation = new Menu("Operation");
    Menu menuExit = new Menu("Exit");
    menuBar.getMenus().addAll(menuOperation, menuExit);

    MenuItem menuItemAdd = new MenuItem("Add");
    MenuItem menuItemSubtract = new MenuItem("Subtract");
    MenuItem menuItemMultiply = new MenuItem("Multiply");
    MenuItem menuItemDivide = new MenuItem("Divide");
    menuOperation.getItems().addAll(menuItemAdd, menuItemSubtract,
        menuItemMultiply, menuItemDivide);

    MenuItem menuItemClose = new MenuItem("Close");
    menuExit.getItems().add(menuItemClose);

    menuItemAdd.setAccelerator(
        KeyCombination.keyCombination("Ctrl+A"));
    menuItemSubtract.setAccelerator(
        KeyCombination.keyCombination("Ctrl+S"));
    menuItemMultiply.setAccelerator(
        KeyCombination.keyCombination("Ctrl+M"));
    menuItemDivide.setAccelerator(
        KeyCombination.keyCombination("Ctrl+D"));

    HBox hBox1 = new HBox(5);
    tfNumber1.setPrefColumnCount(2);
    tfNumber2.setPrefColumnCount(2);
    tfResult.setPrefColumnCount(2);
}
```

```

hBox1.getChildren().addAll(new Label("Number 1:") , tfNumber1,
    new Label("Number 2:") , tfNumber2, new Label("Result:") ,
    tfResult);
hBox1.setAlignment(Pos.CENTER);

HBox hBox2 = new HBox(5);
Button btAdd = new Button("Add");
Button btSubtract = new Button("Subtract");
Button btMultiply = new Button("Multiply");
Button btDivide = new Button("Divide");
hBox2.getChildren().addAll(btAdd, btSubtract, btMultiply, btDivide);
hBox2.setAlignment(Pos.CENTER);

VBox vBox = new VBox(10);
vBox.getChildren().addAll(menuBar, hBox1, hBox2);
Scene scene = new Scene(vBox, 300, 250);
primaryStage.setTitle("MenuDemo"); // Set the window title
primaryStage.setScene(scene); // Place the scene in the window
primaryStage.show(); // Display the window
// Handle menu actions
menuItemAdd.setOnAction(e -> perform('+'));
menuItemSubtract.setOnAction(e -> perform('-'));
menuItemMultiply.setOnAction(e -> perform('*'));
menuItemDivide.setOnAction(e -> perform('/'));
menuItemClose.setOnAction(e -> System.exit(0));
// Handle button actions
btAdd.setOnAction(e -> perform('+'));
btSubtract.setOnAction(e -> perform('-'));
btMultiply.setOnAction(e -> perform('*'));
btDivide.setOnAction(e -> perform('/'));
(c) Paul Fodor and Pearson Inc.
}

```

```
private void perform(char operator) {
    double number1 = Double.parseDouble(tfNumber1.getText());
    double number2 = Double.parseDouble(tfNumber2.getText());

    double result = 0;
    switch (operator) {
        case '+': result = number1 + number2; break;
        case '-': result = number1 - number2; break;
        case '*': result = number1 * number2; break;
        case '/': result = number1 / number2; break;
    }

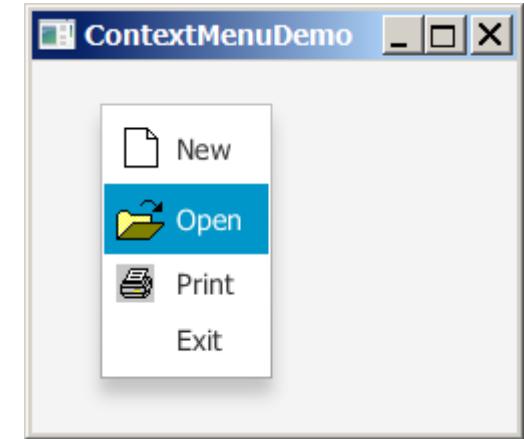
    tfResult.setText(result + "");
};

public static void main(String[] args) {
    launch(args);
}
}
```

Context Menu

- A context menu (also known as a popup menu) is like a regular menu, but does not have a menu bar and can float anywhere on the screen.
- Creating a context menu is similar to creating a regular menu.
 - First, create an instance of ContextMenu, then add MenuItem, CheckMenuItem, and RadioMenuItem to the context menu.

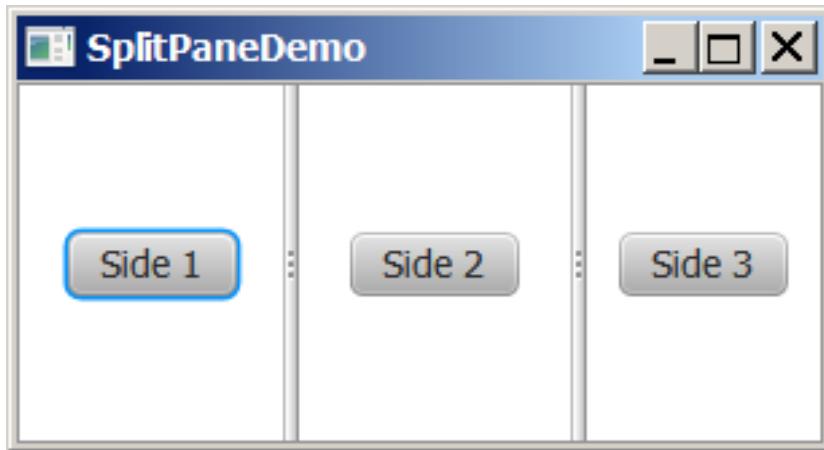
```
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.layout.Pane;
import javafx.scene.control.ContextMenu;
import javafx.scene.control.MenuItem;
import javafx.scene.image.ImageView;
public class ContextMenuDemo extends Application {
    @Override
    public void start(Stage primaryStage) {
        ContextMenu contextMenu = new ContextMenu();
        MenuItem menuItemNew = new MenuItem("New",
            new ImageView("image/new.gif"));
        MenuItem menuItemOpen = new MenuItem("Open",
            new ImageView("image/open.gif"));
        MenuItem menuItemPrint = new MenuItem("Print",
            new ImageView("image/print.gif"));
        MenuItem menuItemExit = new MenuItem("Exit");
        contextMenu.getItems().addAll(menuItemNew, menuItemOpen,
            menuItemPrint, menuItemExit);
        Pane pane = new Pane();
        Scene scene = new Scene(pane, 300, 250);
        primaryStage.setTitle("ContextMenuDemo");
        primaryStage.setScene(scene);
        primaryStage.show();
    }
}
```



```
pane.setOnMousePressed(  
    e -> contextMenu.show(pane, e.getScreenX(), e.getScreenY()));  
  
menuItemNew.setOnAction(e -> System.out.println("New"));  
menuItemOpen.setOnAction(e -> System.out.println("Open"));  
menuItemPrint.setOnAction(e -> System.out.println("Print"));  
menuItemExit.setOnAction(e -> System.exit(0));  
}  
  
public static void main(String[] args) {  
    launch(args);  
}  
}
```

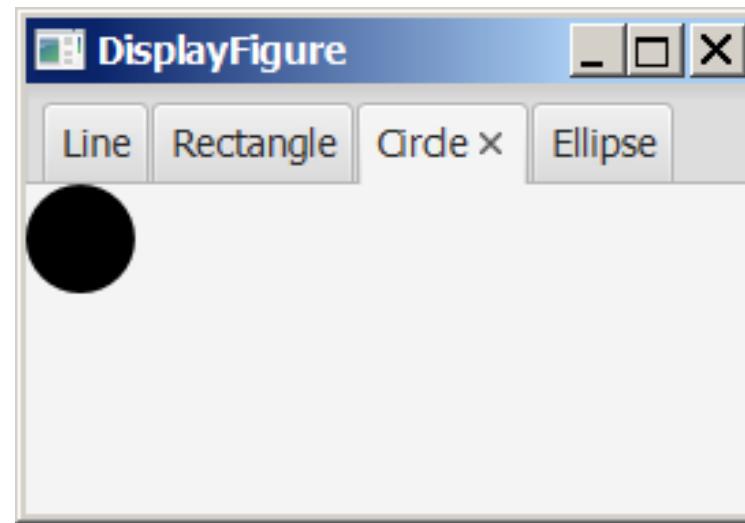
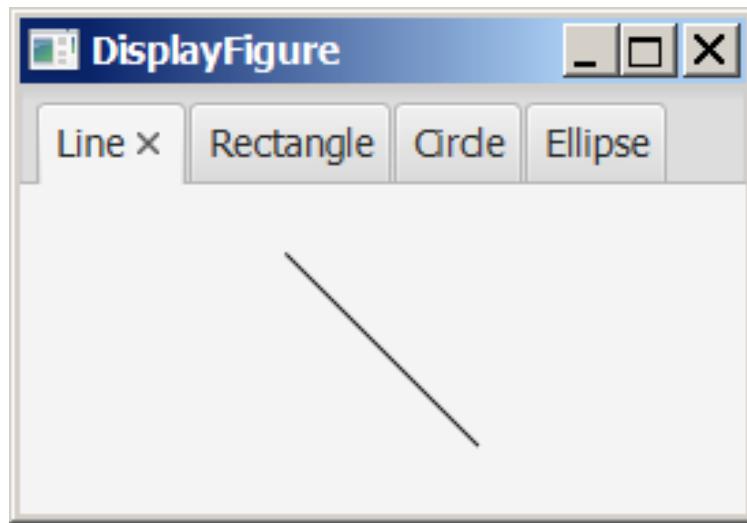
SplitPane

- The SplitPane class can be used to display multiple panes and allow the user to adjust the size of the panes.

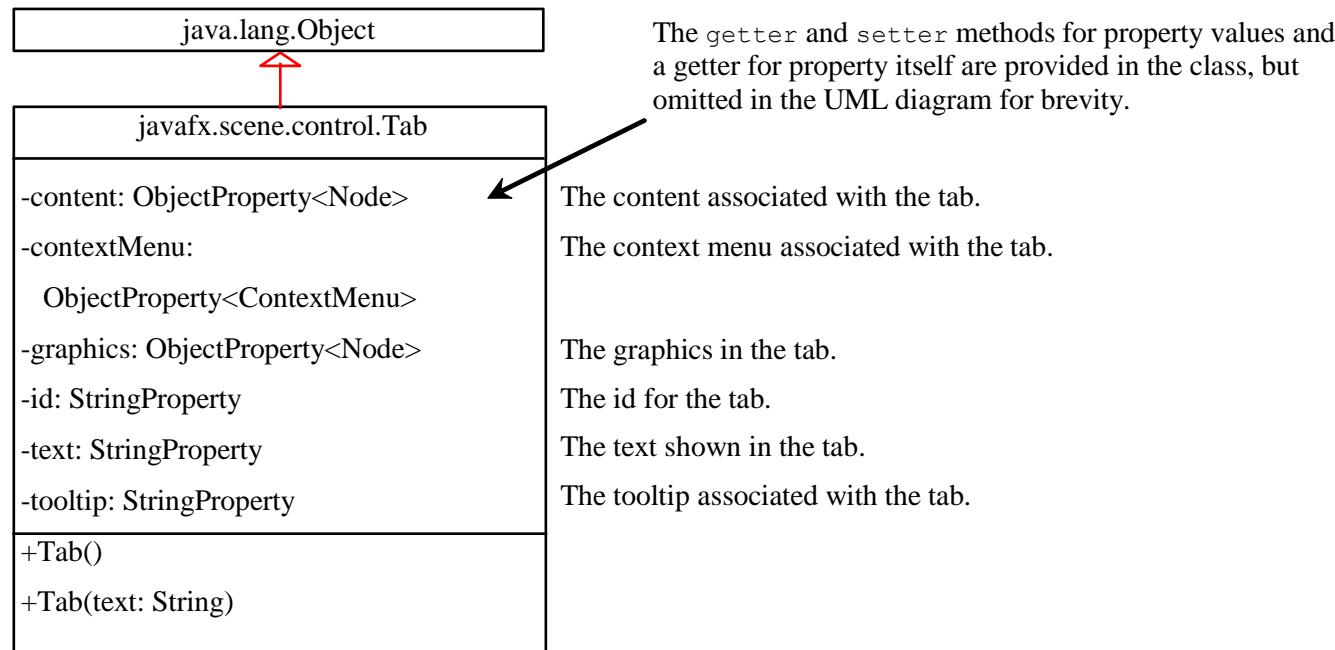
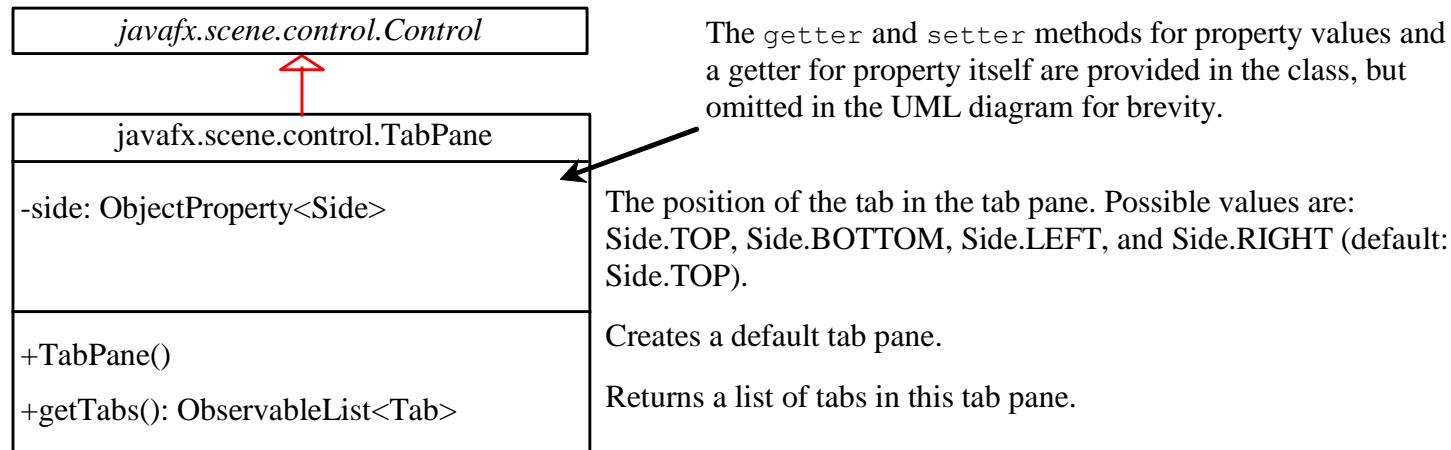


TabPane

- The TabPane class can be used to display multiple panes with tabs.



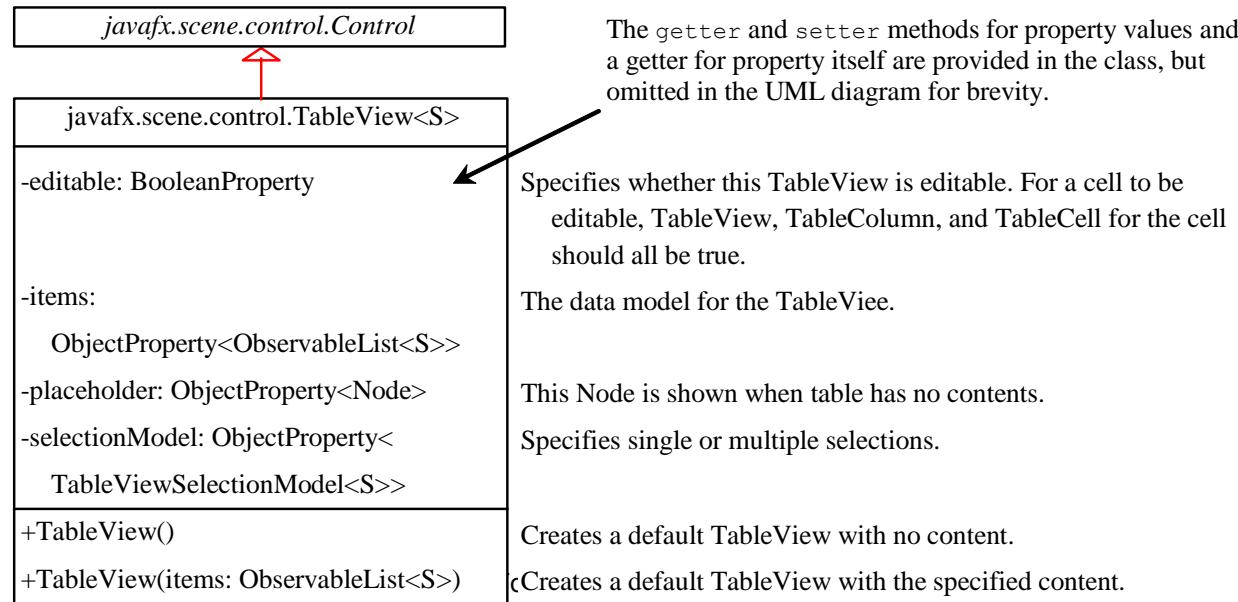
TabPane



TableView

- You can display tables using the TableView class.

Country	Capital	Population (million)	Is Democratic?
USA	Washington DC	280.0	true
Canada	Ottawa	32.0	true
United Kingdom	London	60.0	true
Germany	Berlin	83.0	true
France	Paris	60.0	true



The TableColumn Class

java.lang.Object	The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.
javafx.scene.control.TableColumn<S, T>	
-editable: BooleanProperty	Specifies whether this TableColumn allows editing.
-cellValueFactory:	The cell value factory to specify how to populate all cells within a single column.
ObjectProperty<Callback<TableColumn. CellDataFeatures<S,T>, ObservableValue <T>>>	
-graphic: ObjectProperty<Node>	The graphic for this TableColumn.
-id: StringProperty	The id for this TableColumn.
-resizable: BooleanProperty	Indicates whether the column is resizable.
-sortable: BooleanProperty	Indicates whether the column is sortable.
-text: StringProperty	Text in the table column header.
-style: StringProperty	Specify the CSS style for the column.
-visible: BooleanProperty	Specify whether the column is visible (default: true).
+TableColumn()	Creates a default TableColumn.
+TableColumn(text: String)	Creates a TableView with the specified header text.

FXML

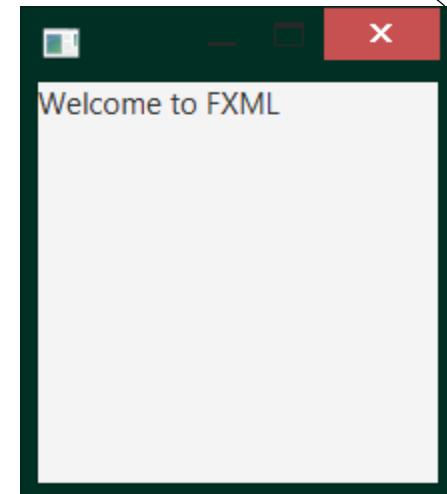
- FXML is a declarative XML-based language created by Oracle Corporation for defining the user interface of a JavaFX 2.0 application.
 - It can be edited and created using the JavaFX Scene Builder 2 (downloaded separately from J2SE)
 - Create a new JavaFX project in Netbeans and you will get 3 files: an FXML file with the UI design, a main application .java file that loads the FXML and a controller for the event handlers for the UI Nodes.

FXML document:

```
<?xml version="1.0" encoding="UTF-8"?>
<?import java.lang.*?>
<?import java.util.*?>
<?import javafx.scene.*?>
<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>

<AnchorPane id="AnchorPane" prefHeight="200" prefWidth="200"
xmlns:fx="http://javafx.com/fxml/1" xmlns="http://javafx.com/javafx/8"
fx:controller="javafxapplication1.FXMLDocumentController">

    <children>
        <FlowPane prefHeight="200.0" prefWidth="200.0">
            <children>
                <Label fx:id="label" minHeight="16" minWidth="69"
text="Welcome to FXML" />
            </children>
        </FlowPane>
    </children>
</AnchorPane>
```



```
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
public class JavaFXApplication5 extends Application {
    @Override
    public void start(Stage stage) throws Exception {
        Parent root = FXMLLoader.load(getClass().getResource("FXMLDocument.fxml"));
        Scene scene = new Scene(root);
        stage.setScene(scene);
        stage.show();
    }
    public static void main(String[] args) {
        launch(args);
    }
}

import java.net.URL;
import java.util.ResourceBundle;
import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.scene.control.Label;
public class FXMLDocumentController implements Initializable {
    @FXML
    private Label label;
    @Override
    public void initialize(URL url, ResourceBundle rb) {
    }
}
```

HTML in JavaFX

- HTML intro.: the Internet Web pages format
- Example: html_sample_01.html

```
<!DOCTYPE html>
<html>
<body>

<h1>My First Heading</h1>
<p>My first paragraph.</p>

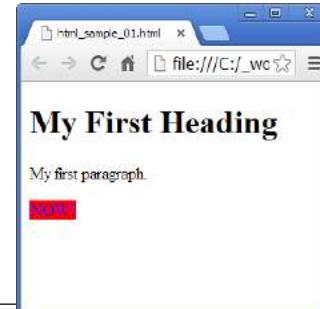
<span style="background-color:red; color:blue" >NOW?</span>

</body>
</html>
```

← This is the HTML tag. Every HTML page has one

← This is a heading

← This is a paragraph



HTML

- HTML is a language for describing web pages.
- HTML stands for **Hyper Text Markup Language**
- HTML is a **markup** language
- A markup language is a set of markup **tags**
- The tags **describe** document content
- HTML documents contain HTML **tags** and plain **text**
- HTML documents are also called **web pages**

HTML

- HTML markup tags are usually called HTML tags
- HTML tags are keywords (tag names) surrounded by **angle brackets** like <html>
- HTML tags normally **come in pairs** like and
- The first tag in a pair is the **start tag**, the second tag is the **end tag**
- The end tag is written like the start tag, with a **forward slash** before the tag name
- Start and end tags are also called **opening tags** and **closing tags**

<tagname>content</tagname>

<p>This is a paragraph.</p>

(c) Paul Fodor and Pearson Inc.

HTML by Examples

- http://www.w3schools.com/html/html_examples.asp
- HTML links:
 - This is a link
 - It appears as: This is a link
- HTML images:
 -
 - It appears as:



JavaFX with HTML

- You can put HTML code in JavaFX:

```
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.layout.StackPane;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.scene.control.Button;
import javafx.scene.web.WebEngine;
import javafx.scene.web.WebView;
public class HTMLDemo extends Application {
    @Override
    public void start(Stage primaryStage) {
        WebView browser = new WebView();
        WebEngine webEngine = browser.getEngine();
        webEngine.loadContent("<html><b><u>T</u>wo</b><br>lines</html>");
        StackPane root = new StackPane();
        root.getChildren().add(browser);
        Scene scene = new Scene(root, 100, 150);
        primaryStage.setTitle("Hello World!");
        primaryStage.setScene(scene);
        primaryStage.show();
    }
    public static void main(String[] args) {
        launch(args);
    }
}
```



```
import org.w3c.dom.Document;
// ... get the document of the engine
Document doc = webEngine.getDocument();
// and the elements
import org.w3c.dom.Element;
... Element el = doc.getElementById("id1");
```

JavaFX with HTML

- You can get the Document only when the synchronized WebEngine had finished loading the page. That is,

```
Document doc = webEngine.getDocument();
```

may be null if the page is not loaded yet.

- Solution: listen to the state of the WebEngine object to know when it is done loading:

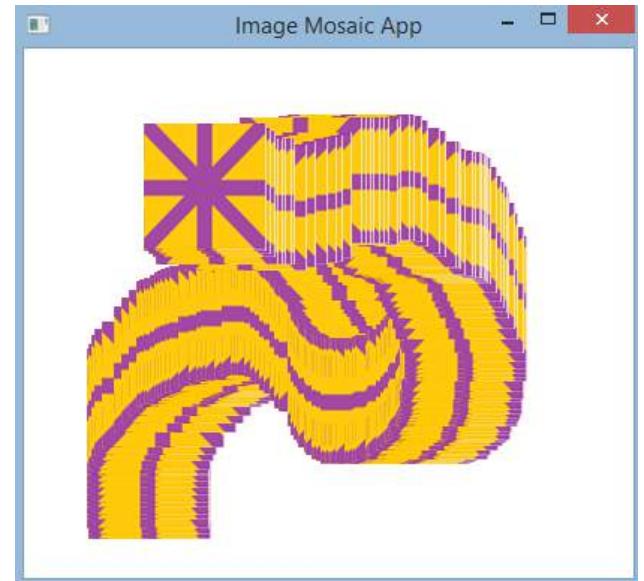
```
engine.getLoadWorker().stateProperty().addListener(  
    (ObservableValue<? extends State> observable,  
     State oldValue, State newValue)  
    -> {  
        if (newValue == State.SUCCEEDED)  
            docManager.setStatsDoc(engine.getDocument());  
    } );
```

javafx.scene.canvas.Canvas

- `javafx.scene.canvas.Canvas` is an image that can be drawn on using a set of graphics commands provided by a `GraphicsContext`.
- `javafx.scene.canvas.GraphicsContext` issues draw calls to a `Canvas` using a buffer:
 - each call pushes the necessary parameters onto the buffer where they will be later rendered onto the image of the `Canvas` node by the rendering thread at the end of a pulse.

```
Canvas canvas = new Canvas(250,250) ;  
GraphicsContext gc =  
    canvas.getGraphicsContext2D() ;  
gc.fillRect(75,75,100,100) ;
```

```
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.canvas.Canvas;
import javafx.scene.canvas.GraphicsContext;
import javafx.geometry.Point2D;
import javafx.geometry.Rectangle2D;
import javafx.scene.Group;
import javafx.scene.image.Image;
import javafx.stage.Screen;
import java.util.ArrayList;
import java.util.Iterator;
public class CanvasDemo extends Application {
    Stage primaryStage;
    Scene scene;
    Canvas canvas;
    GraphicsContext gc;
    Image logo1Image, logo2Image;
    ArrayList<Point2D> logo1Locations, logo2Locations;
    @Override
    public void start(Stage initPrimaryStage) {
        primaryStage = initPrimaryStage;
        initStage();
        initData();
        initGUI();
        initHandlers();
    }
}
```

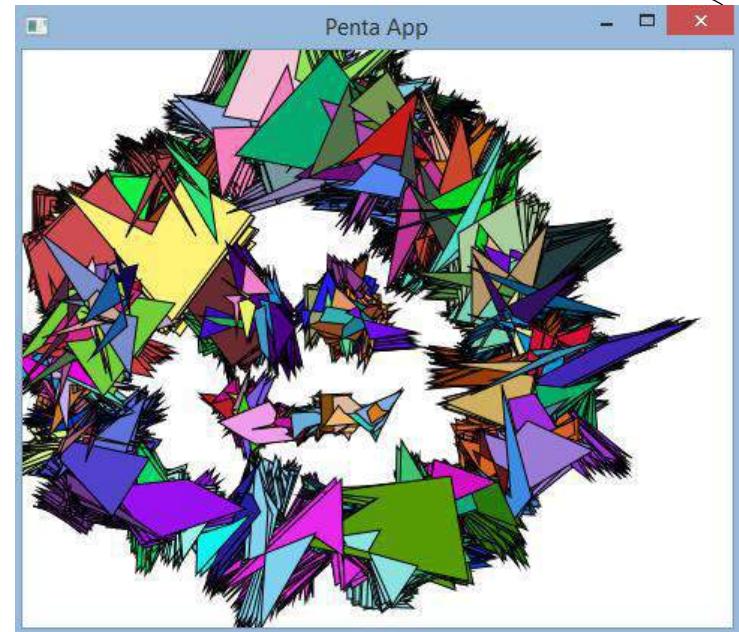


```
public void initStage() {
    Screen screen = Screen.getPrimary();
    Rectangle2D bounds = screen.getVisualBounds();
    primaryStage.setX(bounds.getMinX());
    primaryStage.setY(bounds.getMinY());
    primaryStage.setWidth(bounds.getWidth());
    primaryStage.setHeight(bounds.getHeight());
}
public void initData() {
    logo1Locations = new ArrayList();
    logo2Locations = new ArrayList();
    logo1Image = new Image("file:images/logo1.png");
    logo2Image = new Image("file:images/logo2.png");
}
public void initGUI() {
    canvas = new Canvas();
    gc = canvas.getGraphicsContext2D(); // is graphics destination: monitor
    Group root = new Group();
    root.getChildren().add(canvas);
    scene = new Scene(root);
    primaryStage.setScene(scene);
    primaryStage.show();
    canvas.setWidth(scene.getWidth());
    canvas.setHeight(scene.getHeight());
}
```

```
public void initHandlers() {
    canvas.setOnMouseClicked(mouseEvent -> {
        Point2D point = new Point2D(mouseEvent.getX(), mouseEvent.getY());
        if (!logo1Locations.contains(point)) {
            logo1Locations.add(point);
        }
        draw();
    });
    canvas.setOnMouseDragged(mouseEvent -> {
        Point2D point = new Point2D(mouseEvent.getX(), mouseEvent.getY());
        if (!logo2Locations.contains(point)) {
            logo2Locations.add(point);
        }
        draw();
    });
}
public void draw() {
    Iterator<Point2D> it = logo1Locations.iterator();
    while (it.hasNext()) {
        Point2D p = it.next();
        gc.drawImage(logo1Image, p.getX(), p.getY());
    }
    it = logo2Locations.iterator();
    while (it.hasNext()) {
        Point2D p = it.next();
        gc.drawImage(logo2Image, p.getX(), p.getY());
    }
}
public static void main(String[] args) {
    launch();
}
```

```
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.geometry.Rectangle2D;
import javafx.scene.Group;
import javafx.scene.canvas.Canvas;
import javafx.scene.canvas.GraphicsContext;
import javafx.scene.paint.Color;
import javafx.stage.Screen;
import java.util.ArrayList;
public class PentaApp extends Application {
    private Stage primaryStage;
    private Scene scene;
    private Canvas canvas;
    private GraphicsContext gc;
    private ArrayList<double[]> xPoints;
    private ArrayList<double[]> yPoints;
    private ArrayList<Color> colors;

    @Override
    public void start(Stage initPrimaryStage) {
        primaryStage = initPrimaryStage;
        initStage();
        initData();
        initGUI();
        initHandlers();
    }
}
```



```
public void initStage() {
    primaryStage.setTitle("Penta App");
    Screen screen = Screen.getPrimary(); // is graphics destination: monitor
    Rectangle2D bounds = screen.getVisualBounds();
    primaryStage.setX(bounds.getMinX());
    primaryStage.setY(bounds.getMinY());
    primaryStage.setWidth(bounds.getWidth());
    primaryStage.setHeight(bounds.getHeight());
}
public void initData() {
    xPoints = new ArrayList();
    yPoints = new ArrayList();
    colors = new ArrayList();
}
public void initGUI() {
    canvas = new Canvas();
    gc = canvas.getGraphicsContext2D();
    Group root = new Group();
    root.getChildren().add(canvas);
    scene = new Scene(root);
    primaryStage.setScene(scene);
    primaryStage.show();
    canvas.setWidth(scene.getWidth());
    canvas.setHeight(scene.getHeight());
}
public void initHandlers() {
    canvas.setOnMouseClicked(mouseEvent -> {
        if (mouseEvent.getClickCount() == 2) {
            xPoints.clear();
            yPoints.clear();
        }
    });
}
```

```
    colors.clear();
    gc.clearRect(0, 0, canvas.getWidth(), canvas.getHeight());
}
});

canvas.setOnMouseDragged(mouseEvent -> {
    double x = mouseEvent.getX();
    double y = mouseEvent.getY();
    double[] xs = new double[5];
    double[] ys = new double[5];
    // CENTER
    xs[0] = x;
    ys[0] = y - (int) (Math.random() * 20) - 1;
    // TOP-RIGHT POINT
    xs[1] = x + (int) (Math.random() * 15) + 1;
    ys[1] = y - (int) (Math.random() * 10) - 1;
    // BOTTOM-RIGHT POINT
    xs[2] = x + (int) (Math.random() * 10) + 1;
    ys[2] = y + (int) (Math.random() * 15) + 1;
    // BOTTOM-LEFT POINT
    xs[3] = x - (int) (Math.random() * 10) - 1;
    ys[3] = y + (int) (Math.random() * 15) + 1;
    // TOP-LEFT POINT
    xs[4] = x - (int) (Math.random() * 15) - 1;
    ys[4] = y - (int) (Math.random() * 10) - 1;
    xPoints.add(xs);
    yPoints.add(ys);
    int r = (int) (Math.random() * 256);
    int g = (int) (Math.random() * 256);
    int b = (int) (Math.random() * 256);
    colors.add(Color.rgb(r, g, b));
    PentaApp.this.draw();
});
}
```

```
public void draw() {  
    for (int i = 0; i < xPoints.size(); i++) {  
        double[] xVertices = xPoints.get(i);  
        double[] yVertices = yPoints.get(i);  
        for (int j = 0; j < 5; j++) {  
            xVertices[j] += (int) (Math.random() * 9) - 4;  
            yVertices[j] += (int) (Math.random() * 9) - 4;  
        }  
        Color color = colors.get(i);  
        gc.setFill(color);  
        gc.fillPolygon(xVertices, yVertices, 5);  
        gc.setStroke(Color.BLACK);  
        gc.strokePolygon(xVertices, yVertices, 5);  
    }  
}  
  
public static void main(String[] args) {  
    launch(args);  
}  
}
```

LAB-10

REG NO:19BCN7141

QUESTION:

Write a program to create two threads T1, T2 by implementing runnable interface to perform Matrix addition and multiplication, respectively.

CODE:

```
class chanu implements Runnable{  
    public void run(){  
        int c,d;  
        int a[][] = {{1,1,1},{2,2,2},{3,3,3}};  
        int b[][] = {{1,1,1},{2,2,2},{3,3,3}};  
        int sum[][] = new int[3][3];  
  
        for (c = 0; c < 3; c++)  
            for (d = 0; d < 3; d++)  
                sum[c][d] = a[c][d] + b[c][d];  
  
        System.out.println("Sum of the matrices:");  
  
        for (c = 0; c < 3; c++)  
        {  
            for (d = 0; d < 3; d++)  
                System.out.print(sum[c][d]+\t");  
  
            System.out.println();  
        }  
        int i,j,k;  
        int a1[][]={{1,1,1},{2,2,2},{3,3,3}};  
        int b1[][]={{1,1,1},{2,2,2},{3,3,3}};  
        int c1[][] = new int[3][3];  
        System.out.println("Multiplication of the matrices:");  
        for(i = 0;i < 3;i++)  
        {  
            for(j = 0;j < 3;j++)  
            {  
                c1[i][j] = 0;  
                for(k = 0;k < 3;k++)  
                {  
                    c1[i][j] = c1[i][j]+a1[i][k]*b1[k][j];  
                }  
                System.out.print(c1[i][j]+\t");  
            }  
            System.out.println();  
        }  
    }  
}  
class RunnableMatrices{
```

```
public static void main(String[] args) {  
    chanu c1=new chanu();  
    Thread t=new Thread(c1);  
    chanu c2=new chanu();  
    Thread t1=new Thread(c2);  
    t.start();  
    t1.start();  
}  
}
```

OUTPUT:

```
D:\Javap>javac RunnableMatrices.java  
  
D:\Javap>java RunnableMatrices  
Sum of the matrices:  
2 2 2  
4 4 4  
6 6 6  
Multiplication of the matrices:  
6 6 6  
12 12 12  
18 18 18
```

LAB ASSIGNMENT-9

Reg No:19BCN7141

Q1) Write a generic program to sort array of elements in ascending / lexicographical order.

CODE:

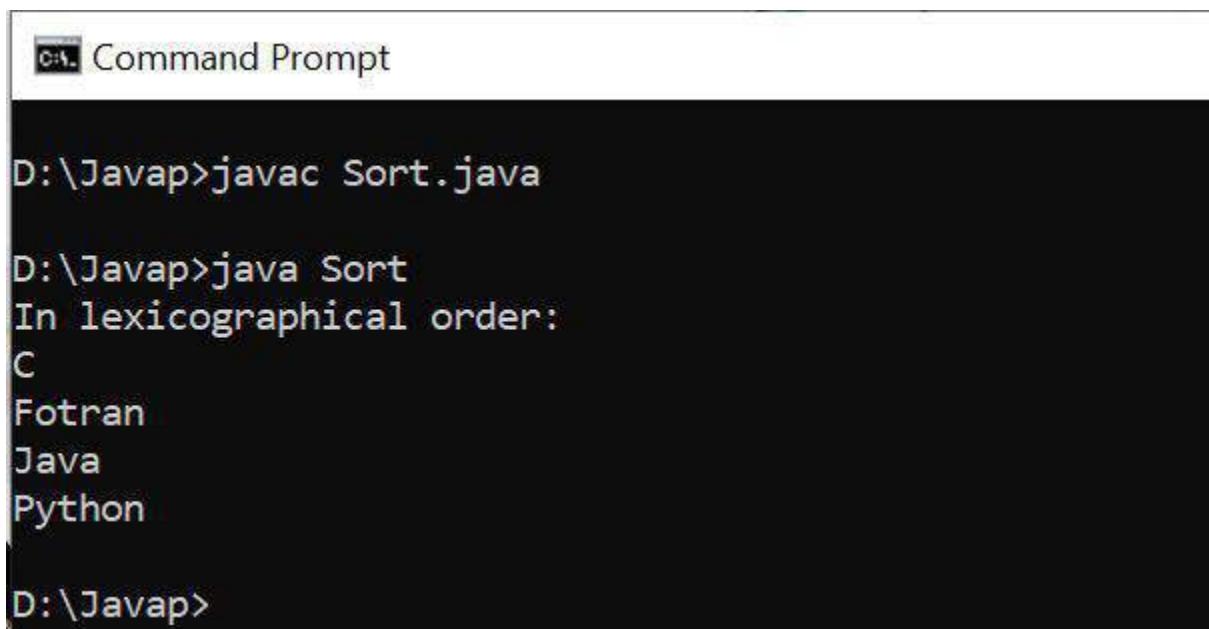
```
public class Sort<S>{

    public static void main(String[] args) {
        String[] words = { "Fotran", "C", "Java", "Python" };

        for(int i = 0; i < 3; ++i) {
            for (int j = i + 1; j < 4; ++j) {
                if (words[i].compareTo(words[j]) > 0) {
                    String temp = words[i];
                    words[i] = words[j];
                    words[j] = temp;
                }
            }
        }

        System.out.println("In lexicographical order:");
        for(int i = 0; i < 4; i++) {
            System.out.println(words[i]);
        }
    }
}
```

OUTPUT:



The screenshot shows a Windows Command Prompt window titled "Command Prompt". The command line shows the path "D:\Javap>" followed by the command "javac Sort.java". The output of the program execution is displayed below the command line, showing the sorted list of words: "In lexicographical order:", followed by "C", "Fotran", "Java", and "Python".

```
D:\Javap>javac Sort.java

D:\Javap>java Sort
In lexicographical order:
C
Fotran
Java
Python

D:\Javap>
```

Q2) Write a program for the following.

- i) Read an integer number (n) using main thread
- ii) Create a user thread T1 using anonymous class to find reverse of number (n).
- iii) Create another user thread T2 using anonymous class to check whether the input number (n) is palindrome or not.

Note: A number is said to be palindrome if $n==reverse(n)$. Example: 121 is a palindrome and 122 is not a palindrome

CODE:

```
import java.util.*;
class chanu extends Thread{
    int a=122;
    int reverse=0;
    public void run() {
        while(a!=0){
            int dig=a%10;
            reverse=reverse*10+dig;
            a/=10;
        }
        System.out.println("Reverse of the number : "+reverse);
        if(reverse==a)
            System.out.println("It is a Palindrome Number");
        else
            System.out.println("It is not a Palindrome");
    }
}

class Test{
    public static void main(String[] args) {
        chanu T=new chanu();
        chanu T1 =new chanu();
        chanu T2=new chanu();
        T.start();
        T1.start();
        T2.start();
    }
}
```

OUTPUT:

Select Command Prompt

```
D:\Javap>javac Test.java

D:\Javap>java Test
Reverse of the number : 221
It is not a Palindrome

D:\Javap>
```

ASSIGNMENT_LAB11

QUESTION:

- Q1) Write a multi-thread program for the following scenario. Create three threads T1, T2, T3 by extending class Thread. (13M)**
- a) T1 should read adjacency matrix of undirected Graph G
 - b) T2 should compute degree matrix of G
 - c) T3 should compute Laplacian matrix (i.e. Degree Matrix - Adjacency matrix)

SOLUTION:

```
import java.util.*;
class Thread1 extends Thread
{
    static Scanner in=new Scanner(System.in);
    static int v;
    static int a[][];
    public void run()
    {
        System.out.println("Enter no.of vertices of graph");
        v=in.nextInt();
        a=new int[v][v];
        System.out.println("Enter the adjacency matrix:");
        for(int i=0;i<v;i++)
        {
            for(int j=0;j<v;j++)
            {
                a[i][j]=in.nextInt();
            }
        }
    }
}
class Thread2 extends Thread
{
    static int count,v;
    static int d[],adj[],a[];
    Thread2()
    {
        this.v=v;
        for(int i=0;i<v;i++)
        {
            for(int j=0;j<v;j++)
            {
                adj[i][j]=a[i][j];
            }
        }
    }
}
```

```

        }
    }
    public void run()
    {
        Thread1 t1=new Thread1();
        t1.start();
        Thread1 t=new Thread1();
        for(int i=0;i<v;i++)
        {
            for(int j=0;j<v;j++)
            {
                d[i][j]=0;
            }
        }

        for(int i=0;i<v;i++)
        {
            count=0;
            for(int j=0;j<v;j++)
            {
                if(adj[i][j]==1)
                {
                    count++;
                }
            }
            d[i][i]=count;
        }
        System.out.println();
        System.out.println("The degree matrix:");
        for(int i=0;i<v;i++)
        {
            for(int j=0;j<v;j++)
            {
                System.out.print(d[i][j]+" ");
            }
            System.out.println();
        }
    }
}

class Thread3 extends Thread
{
    static int l[],adj[],d[],v;
    Thread3()
    {
        this.v=v;
        for(int i=0;i<v;i++)
        {
            for(int j=0;j<v;j++)

```

```

        {
            this.adj[i][j]=adj[i][j];
            this.d[i][j]=d[i][j];
        }
    }

public void run()
{
    Thread2 t2=new Thread2();
    t2.start();
    for(int i=0;i<v;i++)
    {
        for(int j=0;j<v;j++)
        {
            l[i][j]=d[i][j]-adj[i][j];
        }
    }
    System.out.println("Laplacian Matrix:");
    for(int i=0;i<v;i++)
    {
        for(int j=0;j<v;j++)
        {
            System.out.print(l[i][j]+" ");
        }
    }
}
class Ch
{
    public static void main(String args[])
    {
        Thread3 t3=new Thread3();
        t3.start();
    }
}

```

OUTPUT:

```

D:\java_prg>java Ch
Enter no.of vertices of graph
2
Enter the adjacency matrix:
1 2
2 3
D:\java_prg>
D:\java_prg>Ch
Laplacian Matrix:
2 0
0 2

```

QUESTION:

Q2) Write a generic class with bounded type Number. This class should contain one generic type array. Write a constructor to get values of generic array. This class should contain static method called Max which takes one generic array as an argument and returns maximum of all array elements as a double value. Write a main class to demonstrate generic class.

SOLUTION:

```
public class Array<E>{
    public static <E extends Comparable<E>> E Max(E[] list) {
        E max = list[0];
        for (int i = 1; i < list.length; i++) {
            if (list[i].compareTo(max) > 0) {
                max = list[i];
            }
        }
        return max;
    }
    public static void main(String args[])
    {
        Integer[] intArray={23,34,56,78,90};
        System.out.println("MAX ELEMENT IN ARRAY IS:");
        System.out.println(Max(intArray));
    }
}
```

OUTPUT:**Result**

```
$javac Array.java
$java -Xmx128M -Xms16M Array
MAX ELEMENT IN ARRAY IS:
90
```

LAB ASSIGNMENT-9

Reg No:19BCN7141

Q1) Write a generic program to sort array of elements in ascending / lexicographical order.

CODE:

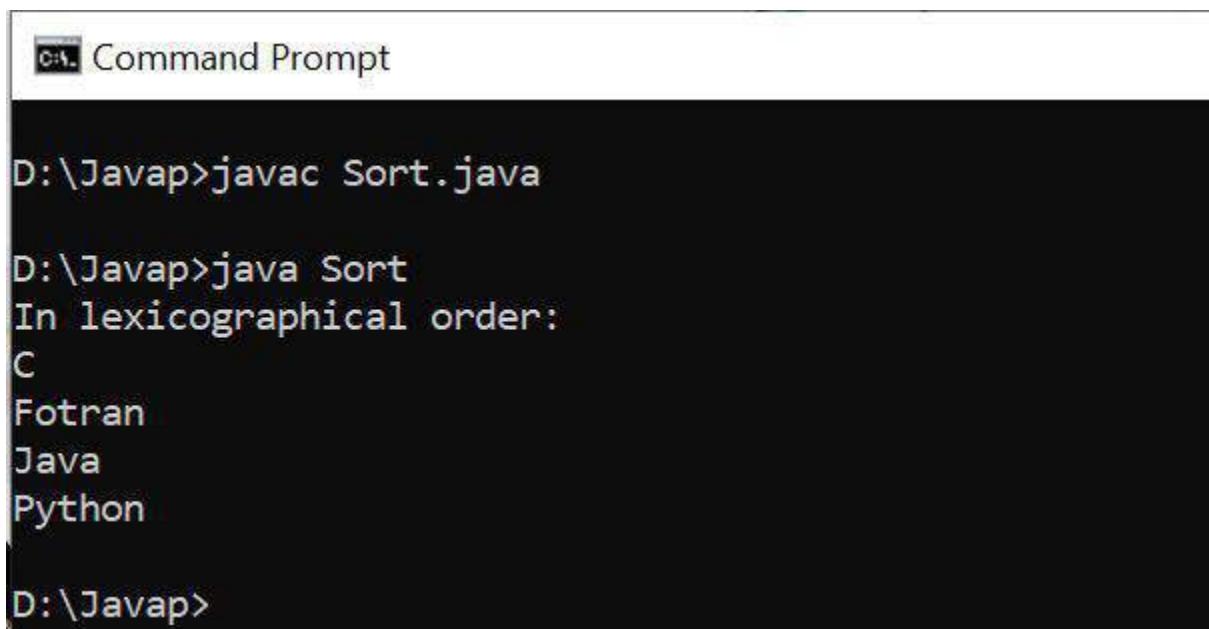
```
public class Sort<S>{

    public static void main(String[] args) {
        String[] words = { "Fotran", "C", "Java", "Python" };

        for(int i = 0; i < 3; ++i) {
            for (int j = i + 1; j < 4; ++j) {
                if (words[i].compareTo(words[j]) > 0) {
                    String temp = words[i];
                    words[i] = words[j];
                    words[j] = temp;
                }
            }
        }

        System.out.println("In lexicographical order:");
        for(int i = 0; i < 4; i++) {
            System.out.println(words[i]);
        }
    }
}
```

OUTPUT:



The screenshot shows a Windows Command Prompt window titled "Command Prompt". The command line shows the path "D:\Javap>" followed by the command "javac Sort.java". The output of the program execution is displayed below, showing the sorted list of words: C, Fotran, Java, Python.

```
D:\Javap>javac Sort.java

D:\Javap>java Sort
In lexicographical order:
C
Fotran
Java
Python

D:\Javap>
```

Q2) Write a program for the following.

- i) Read an integer number (n) using main thread
- ii) Create a user thread T1 using anonymous class to find reverse of number (n).
- iii) Create another user thread T2 using anonymous class to check whether the input number (n) is palindrome or not.

Note: A number is said to be palindrome if $n==reverse(n)$. Example: 121 is a palindrome and 122 is not a palindrome

CODE:

```
import java.util.*;
class chanu extends Thread{
    int a=122;
    int reverse=0;
    public void run() {
        while(a!=0){
            int dig=a%10;
            reverse=reverse*10+dig;
            a/=10;
        }
        System.out.println("Reverse of the number : "+reverse);
        if(reverse==a)
            System.out.println("It is a Palindrome Number");
        else
            System.out.println("It is not a Palindrome");
    }
}

class Test{
    public static void main(String[] args) {
        chanu T=new chanu();
        chanu T1 =new chanu();
        chanu T2=new chanu();
        T.start();
        T1.start();
        T2.start();
    }
}
```

OUTPUT:

Select Command Prompt

```
D:\Javap>javac Test.java

D:\Javap>java Test
Reverse of the number : 221
It is not a Palindrome

D:\Javap>
```

LAB-8

REG NO:19BCN7141

QUESTION:

Write a JAVA program to define a generic method that counts the number of elements in an array T[] that are greater than a specified element elem.

CODE:

```
import java.util.*;
public class Lab8<T extends Comparable<T>> {

    public static < T > void printArray( T[] inputArray ) {
        for(T e : inputArray) {
            System.out.printf("%s ", e);
        }
        System.out.println();
    }

    public static <T extends Number> void countArray(T[] inputArray, int elem) {
        int i = 0;
        for (T e : inputArray) {
            if(elem < e.intValue ())
                i++;
        }
        System.out.println(i);
    }

    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);
        Integer[] intArray = { 1, 2, 3, 4, 5 };
        System.out.println("\nArray contains:");
        printArray(intArray);
        int elem=0;
        System.out.println("Enter the specified number: ");
        elem = sc.nextInt();
        countArray(intArray, elem);
    }
}
```

OUTPUT:

Command Prompt

```
D:\Javap>javac Lab8.java
```

```
D:\Javap>java Lab8
```

```
Array contains:
```

```
1 2 3 4 5
```

```
Enter the specified number:
```

```
4
```

```
1
```

```
D:\Javap>
```

ASSIGNMENT _ LAB8 _ 251120

QUESTION:1

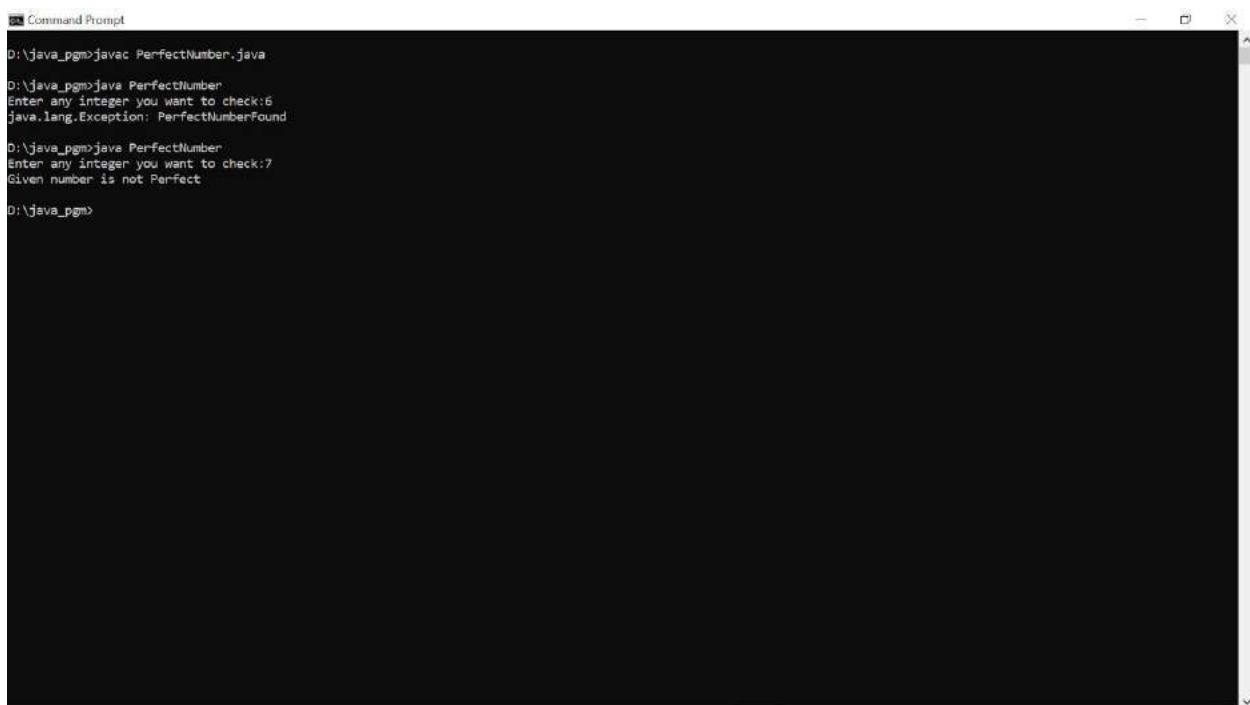
Perfect number is a positive number which sum of all positive divisors excluding that number is equal to that number. For example 6 is perfect number since divisor of 6 are 1, 2 and 3. Sum of its divisor is $1 + 2 + 3 = 6$. Note: 6 is the smallest perfect number. Next perfect number is 28 since $1 + 2 + 4 + 7 + 14 = 28$. Write a program to accept the number from the user and throw a user defined exception PerfectNumberFound if that number is perfect number.

SOLUTION:

```
import java.util.Scanner;
public class PerfectNumber
{
    public static void main(String[] args)
    {
        int n, s= 0;
        Scanner in = new Scanner(System.in);
        System.out.print("Enter any integer you want to check:");
        n = in.nextInt();
        for(int i = 1; i < n; i++)
        {
            if(n % i == 0)
            {
                s= s + i;
            }
        }
        try
        {
            if(s == n)
            {
                throw new Exception("PerfectNumberFound");
            }
        else
```

```
        {
            System.out.println("Given number is not Perfect");
        }
    }
catch(Exception e)
{
    System.out.println(e);
}
}
}
```

OUTPUT:



```
D:\java_pgm>javac PerfectNumber.java
D:\java_pgm>java PerfectNumber
Enter any integer you want to check:6
java.lang.Exception: PerfectNumberFound
D:\java_pgm>java PerfectNumber
Enter any integer you want to check:7
Given number is not Perfect
D:\java_pgm>
```

QUESTION:2

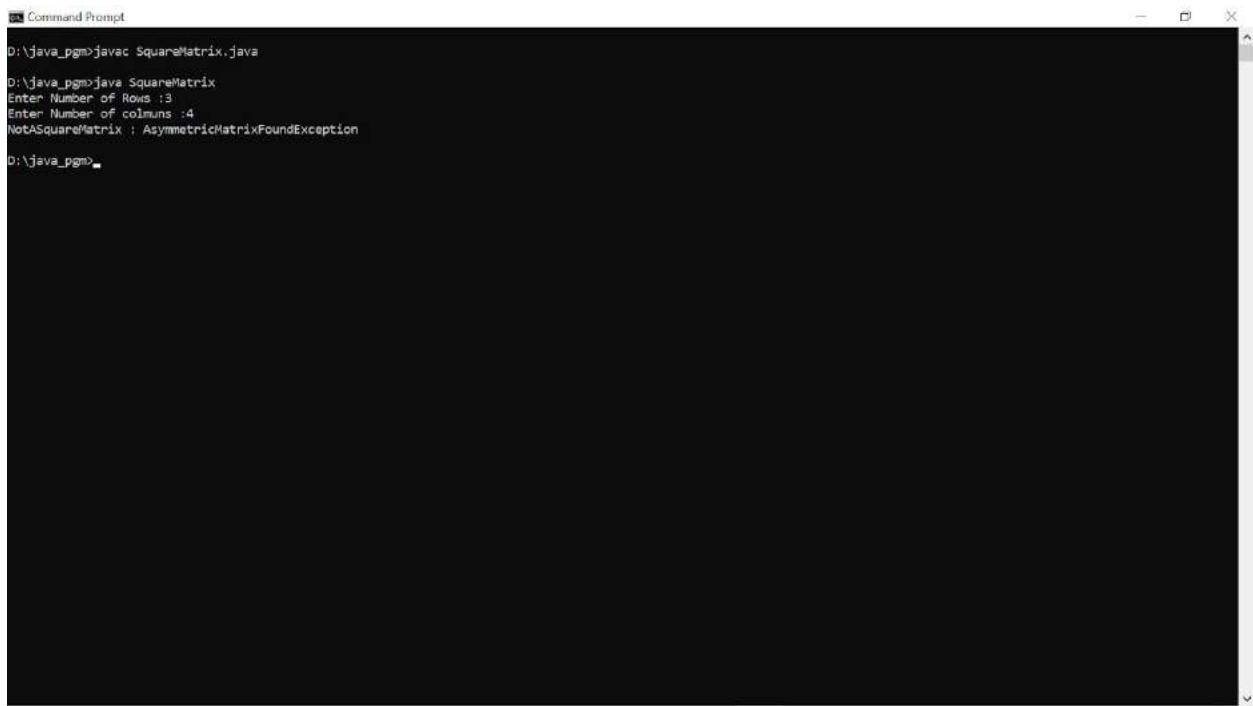
Write a program to read a matrix of size mXn. check wether matrix is square or not. If matrix is not square then through an `NotASquareMatrix` exception. Similarly through an userdefined exception `AsymmetricMatrixFoundException`.

SOLUTION:

```
import java.util.*;
class NotASquareMatrixException extends Exception
{
    String str1;
    NotASquareMatrixException(String str2) {
        str1=str2;
    }
    public String toString(){
        return ("NotASquareMatrix : "+str1) ;
    }
}
class SquareMatrix
{
    public static void main(String args[])
    {
        Scanner in=new Scanner(System.in);
        System.out.print("Enter Number of Rows :");
        int n=in.nextInt();
        System.out.print("Enter Number of colmuns :");
        int m=in.nextInt();
        int a[][]=new int[n][m];
        try
        {
            if(m==n)
            {
                System.out.println("Entered Matrix is Square!");
            }
            else
            {
                throw new
NotASquareMatrixException("AsymmetricMatrixFoundException");
            }
        }
        catch(NotASquareMatrixException e)
        {
            System.out.println(e);
        }
    }
}
```

```
    }  
}
```

OUTPUT:



The screenshot shows a Windows Command Prompt window titled "Command Prompt". The command line shows the following sequence of actions:

```
D:\java_pgm>javac SquareMatrix.java
D:\java_pgm>java SquareMatrix
Enter Number of Rows :3
Enter Number of columns :4
NotASquareMatrix : AsymmetricMatrixFoundException
D:\java_pgm>
```

The user has entered the command `java SquareMatrix`. The application prompts for the number of rows and columns. When the user enters 3 rows and 4 columns, it prints an error message: "NotASquareMatrix : AsymmetricMatrixFoundException".

ASSIGNMENT _ LAB _ 6

**REG.NO:19BCN7141
NAME :K.CHANAKYA**

RAGHAVENDRA

QUESTION : 1

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.

SOLUTION:

```
import java.util.*;
class ExceptionLineTooLong extends Exception
{
}

class Exc
{
    public static void main(String args[])
    {
        try
        {
            Scanner in=new Scanner(System.in);
            System.out.print("Enter the String : ");
            String s=in.nextLine();
            int c=0;
            for(int i=0;i<s.length();i++)
            {
                if(s.charAt(i)!=' ')
                    c++;
            }
            if(c>=80)

```

```

{
    throw new ExceptionLineTooLong();
}
}
catch(ExceptionLineTooLong e)
{
    System.out.print("\nException Caught :");
    System.out.println(e);
}
}
}
}

```

OUTPUT:

```

D:\java_pgm>javac Exc.java
D:\java_pgm>java Exc
Enter the String : qweertyuioplkkjhhhhggsaannmcnjuioppmmnjkkmnnnjm njskkssnjjdjj jjjjjaksallizmmkkslxlllas
Exception Caught :ExceptionLineTooLong
D:\java_pgm>

```

QUESTION :2

Realize a Java class `Matrix` to represent bi-dimensional matrices of real numbers. The class should export the following methods:

A. `Matrix(int n, int m)` : constructor that creates a matrix of size $n \times m$, with all values initially set to 0;

B. Matrix product(Matrix m) : that returns the matrix that is the product of the object and of m, if the two matrices have compatible dimensions, and null otherwise.

C. ExceptionWrongMatrixDimension that is thrown in the method check() if the dimension of the matrix is wrong for the multiplication of matrix.

SOLUTION:

```
class ExceptionWrongMatrixDimension extends Exception
{
}

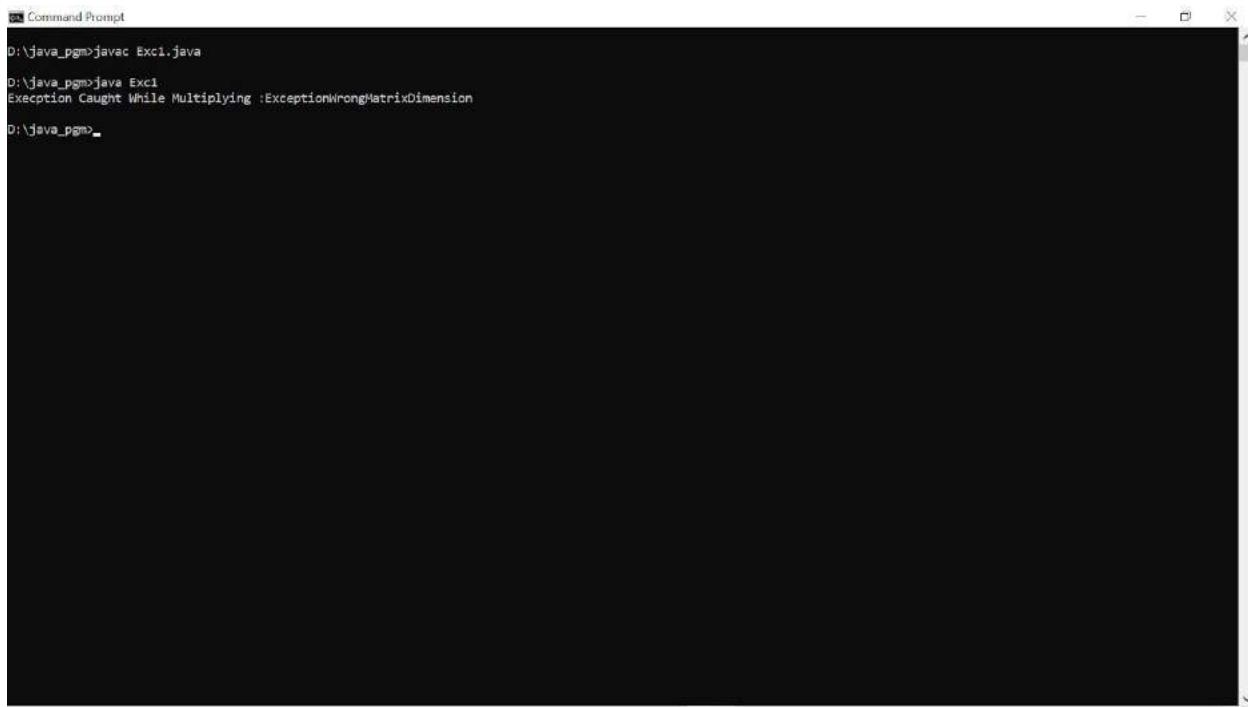
class Matrix
{
    int N;
    int M;
    int[][] a;
    Matrix(int n,int m)
    {
        N=n;
        M=m;
        a=new int[n][m];
    }
    public Matrix(int[][] a)
    {
        M = a.length;
        N = a[0].length;
        this.a = new int[M][N];
        for (int i = 0; i < M; i++)
            for (int j = 0; j < N; j++)
                this.a[i][j] = a[i][j];
    }
    public Matrix matrixProduct(Matrix B)
    {
        Matrix A = this;
        try
        {
            if(A.N!=B.M)
                throw new ExceptionWrongMatrixDimension();
        }
    }
}
```

```

catch(ExceptionWrongMatrixDimension e)
{
    System.out.print("Exception Caught While Multiplying :");
    System.out.println(e);
}
Matrix C = new Matrix(A.M, B.N);
for (int i = 0; i < C.M; i++)
    for (int j = 0; j < C.N; j++)
        for (int k = 0; k < A.N; k++)
            C.a[i][j] += (A.a[i][k] * B.a[k][j]);
return C;
}
public void show()
{
    for (int i = 0; i < M; i++) {
        for (int j = 0; j < N; j++)
            System.out.print(""+a[i][j]);
        System.out.println();
    }
}
}
class Exc1
{
    public static void main(String args[])
    {
        int a[][]={{1,2,3},{4,5,6},{7,8,9}};
        Matrix d=new Matrix(a);
        Matrix c=new Matrix(3,4);
        d.matrixProduct(c);
    }
}

```

OUTPUT:



```
D:\java_pgm>javac Exc1.java
D:\java_pgm>java Exc1
Exception Caught While Multiplying :ExceptionWrongMatrixDimension
D:\java_pgm>
```

ASSIGNMENT _ LAB _ 5

QUESTION _ 1:

A particular application may require you to keep track of fish and other inhabitants of the ocean. You might have a Shark class, a Tuna class, a Porpoise class. Any behaviours that are common to several classes of animals should be specified in a superclass and inherited by its subclasses. For instance, all animals eat. So you could have an Animal superclass that declares a method for eating. Animal should be an abstract class, since any concrete animal you create will be of a specific animal subclass. There are two kinds of animals, those that move around in the ocean and those that do not. It makes sense to abstract the behaviour of swimming for a subclass of Animals. So you could have an abstract subclass of the Animal class. Note that the Swimmer class does not have to code the eat method, since Swimmer is abstract. But any concrete class (i.e., a class that you want to create an instance of) that extends Swimmer must implement both eat and swim. Implement the java program for this scenario with the appropriate data members and member functions.

ANSWER:

```
abstract class Animal
{
    public abstract void eat (Object ob);
}

abstract class Swimmer extends Animal
{
    public abstract void swim();
}

class Shark extends Swimmer
{
    public void swim()
    {
        System.out.println ("shark is swimming");
    }

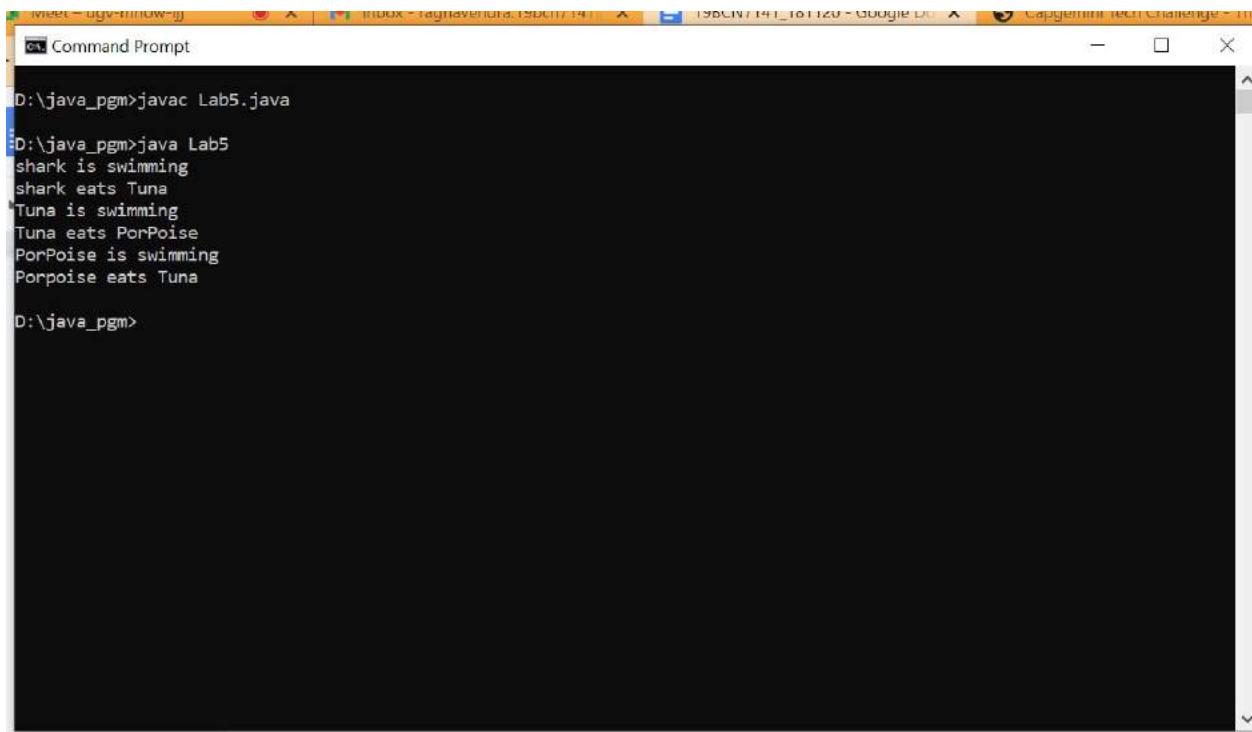
    public void eat (Object ob)
    {
        System.out.println ("shark eats " + ob.toString());
    }

    public String toString()
    {
```

```
        return "shark";
    }
}
class Tuna extends Swimmer
{
    public void swim()
    {
        System.out.println ("Tuna is swimming");
    }
    public void eat (Object ob)
    {
        System.out.println ("Tuna eats " + ob.toString());
    }
    public String toString()
    {
        return "Tuna";
    }
}
class PorPoise extends Swimmer
{
    public void swim()
    {
        System.out.println ("PorPoise is swimming");
    }
    public void eat (Object ob)
    {
        System.out.println ("Porpoise eats " + ob.toString());
    }
    public String toString()
    {
        return "PorPoise";
    }
}
class Lab5
{
    public static void main(String args[])
    {
        Shark ob1=new Shark();
        Tuna ob2=new Tuna();
        PorPoise ob=new PorPoise();
        ob1.swim();
        ob1.eat(ob2);
        ob2.swim();
        ob2.eat(ob);
    }
}
```

```
        ob.swim();
        ob.eat(ob2);
    }
}
```

OUTPUT:



A screenshot of a Windows Command Prompt window titled "Command Prompt". The window shows the command "D:\java_pgm>javac Lab5.java" being run, followed by the program's output: "shark is swimming", "shark eats Tuna", "Tuna is swimming", "Tuna eats PorPoise", "PorPoise is swimming", and "Porpoise eats Tuna". The window has a standard Windows title bar with icons for minimize, maximize, and close.

```
D:\java_pgm>javac Lab5.java
D:\java_pgm>java Lab5
shark is swimming
shark eats Tuna
Tuna is swimming
Tuna eats PorPoise
PorPoise is swimming
Porpoise eats Tuna
D:\java_pgm>
```

QUESTION_2:

You are asked to write a discount system for a beauty saloon, which provides services and sells beauty products. It offers 3 types of memberships: Premium, Gold and Silver. Premium, gold and silver members receive a discount of 20%, 15%, and 10%, respectively, for all services provided. Customers without membership receive no discount. All members receive a flat 10% discount on products purchased. Your system shall consist of three classes: Customer, Discount and Visit. The visit class computes the total bill of purchases X of products and Y of services. Implement the program for above scenario and demonstrate it at least for 3 different cases. Note: The class Discount contains only static variables and methods. Refer the diagram for the implementation.

ANSWER:

```
import java.util.Date;
class MemberType {
public final static String PREMIUM = "PREMIUM";
public final static String GOLD = "GOLD";
public final static String SILVER = "SILVER";
}
class DiscountRate {
private static double serviceDiscountPremium = 0.2;
private static double serviceDiscountGold = 0.15;
private static double serviceDiscountSilver = 0.1;
private static double productDiscountPremium = 0.1;
private static double productDiscountGold = 0.1;
private static double productDiscountSilver = 0.1;
public static double getServiceDiscountRate(String type) {
if (type == null || type.isEmpty()) {
return 0;
}
if (type.equalsIgnoreCase(MemberType.PREMIUM)) {
return serviceDiscountPremium;
} else if (type.equalsIgnoreCase(MemberType.GOLD)) {
return serviceDiscountGold;
} else if (type.equalsIgnoreCase(MemberType.SILVER)) {
return serviceDiscountSilver;
}
return 0;
}
public static double getProductDiscountRate(String type) {

if (type == null || type.isEmpty()) {
return 0;
}

if (type.equalsIgnoreCase(MemberType.PREMIUM)) {
return productDiscountPremium;
} else if (type.equalsIgnoreCase(MemberType.GOLD)) {
return productDiscountGold;
} else if (type.equalsIgnoreCase(MemberType.SILVER)) {
return productDiscountSilver;
}
return 0;
}
}
```

```
class Customer {

    private String name;

    private boolean member;
    private String memberType;
    public Customer(String name) {
        super();
        this.name = name;
    }
    public String getName() {
        return name;
    }
    public boolean isMember() {
        return member;
    }
    public void setMember(boolean member) {
        this.member = member;
    }
    public String getMemberType() {
        return memberType;
    }
    public void setMemberType(String memberType) {
        this.memberType = memberType;
    }
    public String toString() {
        return "\nCustomer [name=" + name + ", member=" + member + ",
               memberType=" + memberType + "]";
    }
}

class Visit {
    private Customer customer;
    private Date date;
    private double serviceExpense;
    private double productExpense;
    public Visit(Customer customer, Date date) {
        super();
        this.customer = customer;
        this.date = date;
    }
    public String getName() {
        return this.customer.getName();
    }
    public double getServiceExpense() {
```

```
        return serviceExpense;
    }
    public void setServiceExpense(double serviceExpense) {
        this.serviceExpense = serviceExpense;
    }
    public double getProductExpense() {
        return productExpense;
    }
    public void setProductExpense(double productExpense) {
        this.productExpense = productExpense;
    }
    public double getTotalExpense() {
        double serviceDiscount =
        DiscountRate.getServiceDiscountRate(this.customer.getMemberType());
        double productDiscount =
        DiscountRate.getProductDiscountRate(this.customer.getMemberType());
        double serviceDiscountAmt = this.serviceExpense * serviceDiscount;
        double productdiscountAmt = this.productExpense * productDiscount;
        return (this.serviceExpense - serviceDiscountAmt) +
        (this.productExpense - productdiscountAmt);
    }

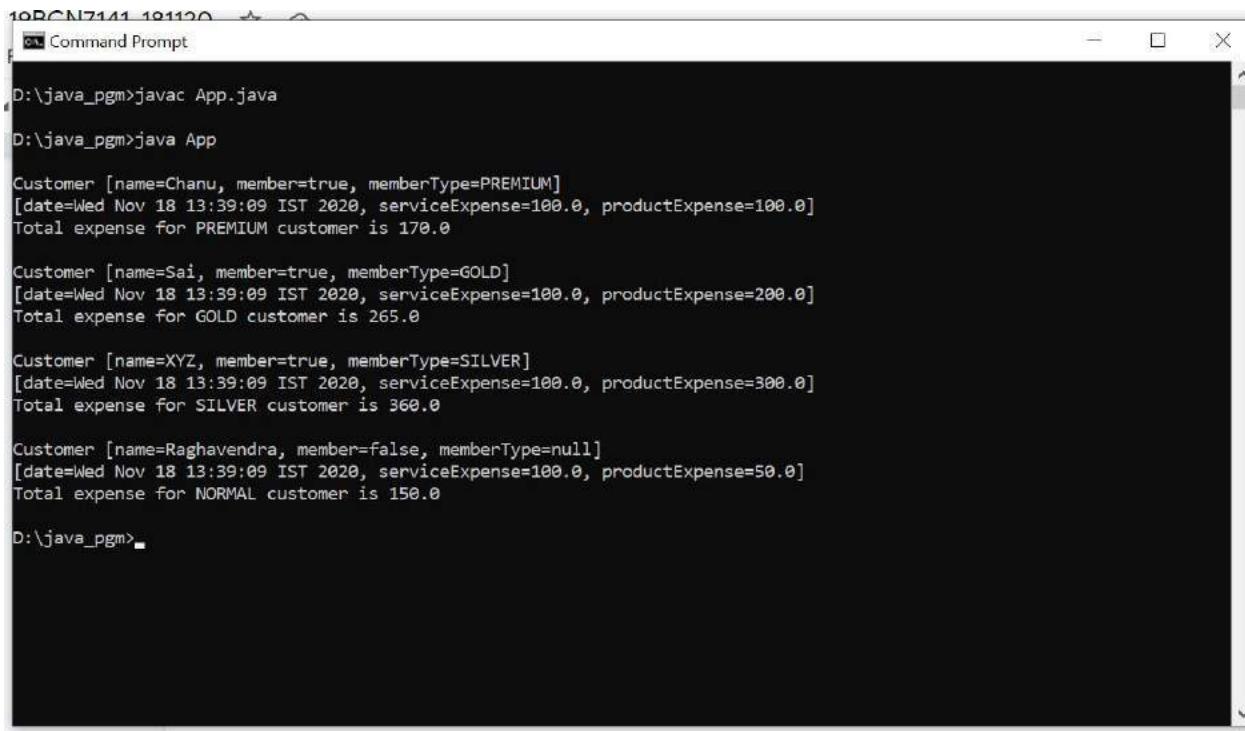
    public String toString() {
        return "[date=" + date + ", serviceExpense=" + serviceExpense +
        ", productExpense=" + productExpense + "]";
    }
}

public class App {

    public static void main(String[] args) {
        Customer premiumCusto = new Customer("Chanu");
        premiumCusto.setMember(true);
        premiumCusto.setMemberType(MemberType.PREMIUM);
        System.out.println(premiumCusto);
        Visit pcVisit = new Visit(premiumCusto, new Date());
        pcVisit.setServiceExpense(100);
        pcVisit.setProductExpense(100);
        double totalExpense = pcVisit.getTotalExpense();
        System.out.println(pcVisit);
        System.out.println("Total expense for PREMIUM customer is " +
        totalExpense);
        Customer goldCusto = new Customer("Sai");
        goldCusto.setMember(true);
        goldCusto.setMemberType(MemberType.GOLD);
    }
}
```

```
System.out.println(goldCusto);
Visit gcVisit = new Visit(goldCusto, new Date());
gcVisit.setServiceExpense(100);
gcVisit.setProductExpense(200);
System.out.println(gcVisit);
totalExpense = gcVisit.getTotalExpense();
System.out.println("Total expense for GOLD customer is " +
totalExpense);
Customer silverCusto = new Customer("Arundhati");
silverCusto.setMember(true);
silverCusto.setMemberType(MemberType.SILVER);
System.out.println(silverCusto);
Visit scVisit = new Visit(silverCusto, new Date());
scVisit.setServiceExpense(100);
scVisit.setProductExpense(300);
System.out.println(scVisit);
totalExpense = scVisit.getTotalExpense();
System.out.println("Total expense for SILVER customer is " +
totalExpense);
Customer normalCusto = new Customer("Raghavendra");
System.out.println(normalCusto);
Visit ncVisit = new Visit(normalCusto, new Date());
ncVisit.setServiceExpense(100);
ncVisit.setProductExpense(50);
System.out.println(ncVisit);
totalExpense = ncVisit.getTotalExpense();
System.out.println("Total expense for NORMAL customer is " +
totalExpense);
}
}
```

OUTPUT:



D:\java_pgm>javac App.java
D:\java_pgm>java App
Customer [name=Chanu, member=true, memberType=PREMIUM]
[date=Wed Nov 18 13:39:09 IST 2020, serviceExpense=100.0, productExpense=100.0]
Total expense for PREMIUM customer is 170.0

Customer [name=Sai, member=true, memberType=GOLD]
[date=Wed Nov 18 13:39:09 IST 2020, serviceExpense=100.0, productExpense=200.0]
Total expense for GOLD customer is 265.0

Customer [name=XYZ, member=true, memberType=SILVER]
[date=Wed Nov 18 13:39:09 IST 2020, serviceExpense=100.0, productExpense=300.0]
Total expense for SILVER customer is 360.0

Customer [name=Raghavendra, member=false, memberType=null]
[date=Wed Nov 18 13:39:09 IST 2020, serviceExpense=100.0, productExpense=50.0]
Total expense for NORMAL customer is 150.0

D:\java_pgm>

ASSIGNMENT_LAB_6

QUESTION :

Write a java program to implement the concept of inheritance where the subclass StaffMember inherits the super class Department and implements the interface Publication. Print the details of department along with their faculty details and their publication. Consider 15 faculty members are in the dept.

Department - class

Data members: Department Name, HOD Name, total students, no of sections
Methods: Constructor, showDepartmentDetails()

StaffMember -class

Data members: Staff Member Name, staff id, Staff Qualification, designation,experience (all are private)

Methods: Constructor, showStaffDetail()

Publication-interface

journalcount, projectcount, patterncount
public void show_publication_detail().

SOLUTION:

```
class Department
{
    String D_Name;
    String H_Name;
    int No_S;
    int No_Se;
    Department(String D_Name, String H_Name, int No_S, int No_Se)
    {
        this.D_Name=D_Name;
        this.H_Name=H_Name;
        this.No_S=No_S;
        this.No_Se=No_Se;
    }
    public void showDepartmentDetails()
    {
        System.out.println("The Department Name is :" + D_Name + "\nHOD Name
is :" + H_Name + "\nTotal Students :" + No_S + "\nNumber of Sections :" + No_Se);
    }
}
```

```
}

interface Publication
{
    int j_c=3;
    int Po_c=4;
    int pa_c=56;
    public void showPublicationDetails();
}

class StaffMember extends Department implements Publication
{
    private String st_name;
    private double st_id;
    private String st_q;
    private String st_d;
    private int st_e;
    StaffMember(String st_name,double st_id,String st_q,String st_d,int st_e)
    {
        super("", "", 0, 0);
        this.st_name = st_name;
        this.st_id = st_id;
        this.st_q = st_q;
        this.st_d = st_d;
        this.st_e = st_e;
    }
    public void showStaffDetail()
    {
        System.out.println("The Staff Name is :" + st_name + "\nStaff id is "
                + st_id + "\nStaff Qualification :" + st_q + "\nStaff Designation :" + st_d + "\nStaff
Experience :" + st_e);
    }
    public void showPublicationDetails()
    {
        System.out.println(st_name + " has published book \n having Journal
                :" + j_c + "\n Project Count :" + Po_c + "\n Pattern Count :" + pa_c);
    }
}
class Lab6
{
    public static void main(String args[])
    {
```

```
Department a=new Department("CSE","Chanu",15,4);
a.showDepartmentDetails();
StaffMember b=new
StaffMember("Sai",9493567,"M.B.A","Software Engineer",4);
b.showStaffDetail();
b.showPublicationDetails();
StaffMember c=new
StaffMember("Polu",9493567,"B.tech","Software Engineer",8);
c.showStaffDetail();
c.showPublicationDetails();
StaffMember d=new
StaffMember("Kolli",9493567,"M.com","Software Engineer",5);
d.showStaffDetail();
d.showPublicationDetails();
StaffMember e=new
StaffMember("Umesh",9493567,"B.ed","Software Engineer",2);
e.showStaffDetail();
e.showPublicationDetails();
StaffMember f=new
StaffMember("Kohli",9493567,"M.B.A","Software Engineer",6);
f.showStaffDetail();
f.showPublicationDetails();
StaffMember g=new
StaffMember("Dravid",9493567,"M.A","Software Engineer",7);
g.showStaffDetail();
g.showPublicationDetails();
StaffMember h=new
StaffMember("Rahul",9493567,"M.ed","Software Engineer",3);
h.showStaffDetail();
h.showPublicationDetails();
StaffMember i=new
StaffMember("Jampa",9493567,"M.B.A","Software Engineer",1);
i.showStaffDetail();
i.showPublicationDetails();
StaffMember j=new
StaffMember("Anshuka",9493567,"M.tech","Software Engineer",6);
j.showStaffDetail();
j.showPublicationDetails();
StaffMember k=new
StaffMember("Raghavendra",5693567,"M.B.A","Software Engineer",9);
```

```
k.showStaffDetail();
k.showPublicationDetails();
StaffMember l=new
StaffMember("Bhumrah",9493567,"B.tech","Software Engineer",7);
l.showStaffDetail();
l.showPublicationDetails();
StaffMember m=new
StaffMember("Shami",8493567,"M.B.A","Software Engineer",4);
m.showStaffDetail();
m.showPublicationDetails();
StaffMember n=new
StaffMember("Chahal",7493578,"M.B.A","Software Engineer",5);
n.showStaffDetail();
n.showPublicationDetails();
StaffMember o=new
StaffMember("Kuldeep",6493567,"M.B.A","Software Engineer",2);
o.showStaffDetail();
o.showPublicationDetails();
StaffMember p=new
StaffMember("Pant",5493567,"M.B.A","Data Analist",4);
p.showStaffDetail();
p.showPublicationDetails();
}
}
```

OUTPUT:

```
ps Command Prompt
D:\Java_pgm>javac Lab6.java
D:\Java_pgm>java Lab6
The Department Name is :CSE
HOD Name is :Chetu
Total Students :15
Number of Sections :4
The Staff Name is :Sei
Staff id is :9493567.0
Staff Qualification :M.B.A
Staff Designation :Software Engineer
Staff Experience :4
Sei has published book
having Journal :3
Project Count :4
Pattern Count :56
The Staff Name is :Polu
Staff id is :9493567.0
Staff Qualification :B.tech
Staff Designation :Software Engineer
Staff Experience :8
Polu has published book
having Journal :3
Project Count :4
Pattern Count :56
The Staff Name is :Kolli
Staff id is :9493567.0
Staff Qualification :M.com
Staff Designation :Software Engineer
Staff Experience :5
Kolli has published book
having Journal :3
Project Count :4
Pattern Count :56
The Staff Name is :Umesh
Staff id is :9493567.0
Staff Qualification :B.ed
Staff Designation :Software Engineer
Staff Experience :2
Umesh has published book
having Journal :3
Project Count :4
Pattern Count :56
```

```
ps Command Prompt
The Staff Name is :Raghavendra
Staff id is :5693567.0
Staff Qualification :M.B.A
Staff Designation :Software Engineer
Staff Experience :9
Raghavendra has published book
having Journal :3
Project Count :4
Pattern Count :56
The Staff Name is :Bhumrah
Staff id is :9493567.0
Staff Qualification :B.tech
Staff Designation :Software Engineer
Staff Experience :7
Bhumrah has published book
having Journal :3
Project Count :4
Pattern Count :56
The Staff Name is :Shami
Staff id is :8493567.0
Staff Qualification :M.B.A
Staff Designation :Software Engineer
Staff Experience :4
Shami has published book
having Journal :3
Project Count :4
Pattern Count :56
The Staff Name is :Chahal
Staff id is :7493578.0
Staff Qualification :M.B.A
Staff Designation :Software Engineer
Staff Experience :5
Chahal has published book
having Journal :3
Project Count :4
Pattern Count :56
The Staff Name is :Kuldeep
Staff id is :6493567.0
Staff Qualification :M.B.A
Staff Designation :Software Engineer
Staff Experience :2
Kuldeep has published book
having Journal :3
```

ASSIGNMENT-4

REG.NO:19BCN7141

QUESTION:1

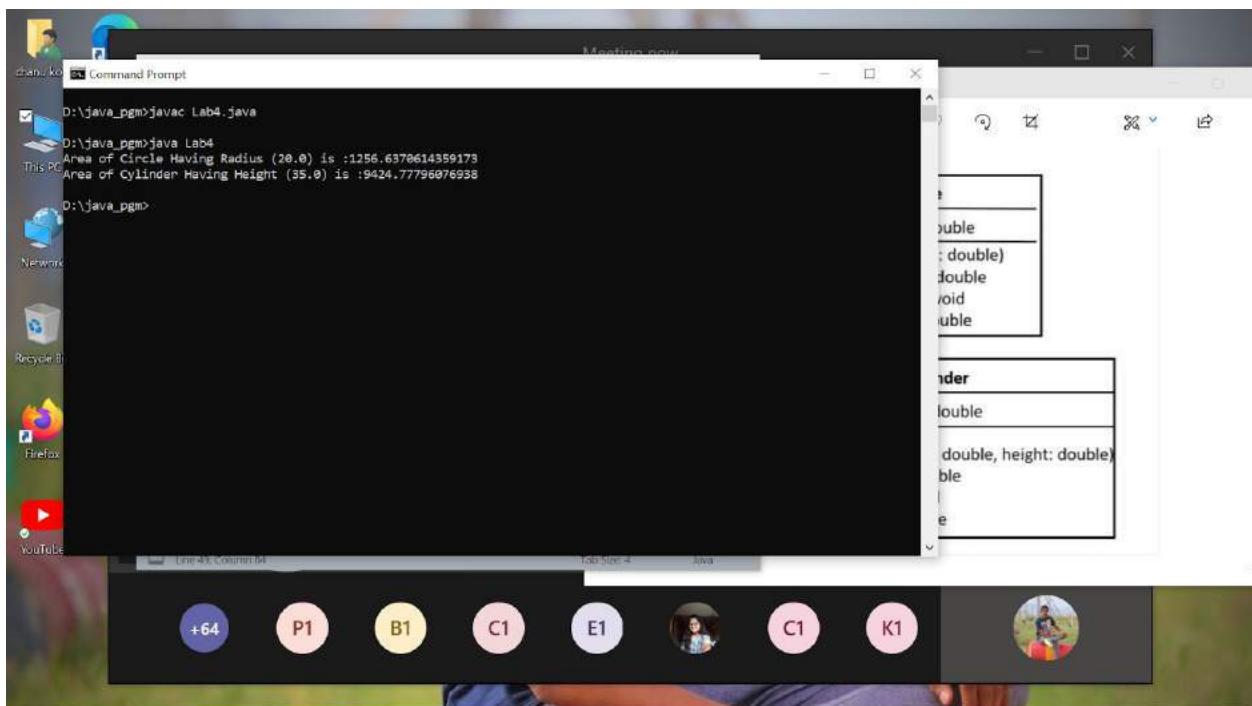
```
import java.util.*;
class Cricle
{
    double radius;
    Cricle(double radius)
    {
        this.radius=radius;
    }
    public double getRadius()
    {
        return radius;
    }
    public void setRadius(double radius)
    {
        this.radius=radius;
    }
    public double getArea()
    {
        return Math.PI*radius*radius;
    }
}
class Cylinder extends Cricle
{
    double height;
    Cylinder(double radius,double height)
    {
        super(radius);
        this.height=height;
    }
    public double getHeight()
    {
        return height;
    }
    public void setHeight(double height)
    {
        this.height=height;
    }
    public double getArea()
```

```

    {
        return 2*Math.PI*radius*(height+radius);
    }
}
class Lab4{
    public static void main(String args[])
    {
        Cricle a = new Cricle(20.0);
        Cylinder c = new Cylinder(25.0,35.0);
        System.out.println("Area of Circle Having Radius (" +a.getRadius() +") is
:" +a.getArea());
        System.out.println("Area of Cylinder Having Height (" +c.getHeight() +") is
:" +c.getArea());
    }
}

```

OUTPUT:



QUESTION-2:

```
import java.util.Scanner;

public class ATM {

    public static Scanner kbd = new Scanner(System.in);
    public static String checkID(String cardNum, String pwd)
    {
        String result = "error";
        String a = "4567-7654 2522 4250";

        if (cardNum.equals(a.substring(0, a.indexOf(" "))) &&
            pwd.equals(a.substring(a.indexOf(" ") + 1, a.lastIndexOf(" "))))
            return result = a.substring(a.lastIndexOf(" ") + 1);

        return result;
    }

    public static int menu()
    {
        int menuChoice;
        do
        {
            System.out.print("\nPlease Choose From the Following Options:"
                + "\n 1. Display Balance \n 2. Deposit"
                + "\n 3. Withdraw\n 4. Log Out\n\n");

            menuChoice = kbd.nextInt();

            if (menuChoice < 1 || menuChoice > 4){
                System.out.println("error");
            }
        }while (menuChoice < 1 || menuChoice > 4);

        return menuChoice;
    }
}
```

```
}

public static void displayBalance(double x)
{
    System.out.printf("\nYour Current Balance is $%.2f\n", x);
}

public static double deposit(double x, double y)
{
    double depositAmt = y, currentBal = x;
    double newBalance = depositAmt + currentBal;

    System.out.printf("Your New Balance is $%.2f\n", newBalance);

    return newBalance;
}

public static double withdraw(double x, double y)
{
    double withdrawAmt = y, currentBal = x, newBalance;

    newBalance = currentBal - withdrawAmt;
    System.out.printf("Your New Balance is %.2f\n", newBalance);

    return newBalance;
}

public static void main(String[] args) {

    String accNum, pass, origBal = "error";
    int count = 0, menuOption = 0;
    double depositAmt = 0, withdrawAmt = 0, currentBal=0;
    boolean foundNonDigit;
    do{
        foundNonDigit = false;
        System.out.println("Please Enter Your Card Number: ");
        accNum = kbd.next();
```

```
System.out.println("Enter Your PIN: ");
pass = kbd.next();

origBal = checkID(accNum, pass);

count++;

if (count >= 3 && origBal.equals("error")){
    System.out.print("Maximum Login Attempts Reached.");
    System.exit(0);
}
if (!(origBal.equals("error"))){
    System.out.println("\nYour New Balance is: $ "+ origBal);
}
else
    System.out.println(origBal);

}while(origBal.equals("error"));

currentBal=Double.parseDouble(origBal);
while (menuOption != 4)
{
    menuOption=menu();
    switch (menuOption)
    {
        case 1:
            displayBalance(currentBal);
            break;
        case 2:
            System.out.print("\nEnter Amount You Wish to Deposit: $ ");
            depositAmt = kbd.nextDouble();
            currentBal=deposit(depositAmt, currentBal);
            break;
        case 3:
            System.out.print("\nEnter Amount You Wish to Withdrawl: $ ");
```

```
withdrawAmt = kbd.nextDouble();

while(withdrawAmt>currentBal){
    System.out.print("ERROR: TRANSACTION UNSUCCESSFUL " +
"INSUFFICIENT FUNDS!! "
    + "PLEASE ENTER A DIFFERENT AMOUNT: $");
    withdrawAmt = kbd.nextDouble();
}

currentBal = withdraw(currentBal, withdrawAmt);
break;
case 4:
    System.out.print("\nTransaction successful Thank For Using My
ATM. Have a Nice Day. Good-Bye!");
    System.exit(0);
break;
}
}
}
```

OUTPUT:

D:\java_pgm\ATM.java (java_pgm) - Sublime Text (UNREGISTERED)

File Edit Selection Find Goto Tools Project Preferences Help

FOLDE Command Prompt

```
/* D:\java_pgm>javac ATM.java
/* D:\java_pgm>java ATM
Please Enter Your Card Number:
4567-7654 2522 4298
Enter Your PIN:
Your New Balance is: $ 4250

Please Choose From the Following Options:
1. Display Balance
2. Deposit
3. Withdraw
4. Log Out
error
Please Choose From the Following Options:
1. Display Balance
2. Deposit
3. Withdraw
4. Log Out
1
Your Current Balance is $4250.00

Please Choose From the Following Options:
1. Display Balance
25
+ " \n 3. Withdraw\n 4. Log Out\n\n");
Main2.java 26
MainClass.java 27
menuChoice = kbd.nextInt();
DedArray.java 28
if (menuChoice < 1 || menuChoice > 4){
Output.java 29
System.out.println("error");
Overload.java 30
}
mainmenu.java 31

```

D:\java_pgm\ATM.java [java_pgm] - Sublime Text (UNREGISTERED)

File Edit Selection Find Goto Tools Project Preferences Help

FOLDE Command Prompt

```
// Enter Amount You Wish to Deposit: $ 4000
// Your New Balance is $8250.00

// Please Choose From the Following Options:
// 1. Display Balance
// 2. Deposit
// 3. Withdraw
// 4. Log Out

// 3

// Enter Amount You Wish to Withdrawl: $ 2000
// Your New Balance is 6250.00

// Please Choose From the Following Options:
// 1. Display Balance
// 2. Deposit
// 3. Withdraw
// 4. Log Out

// 4

// Transaction successful Thank For Using My ATM. Have a Nice Day. Good-Bye!
D:\java_pgm>cls
```

Line 29, Column 6 Spaces: 4 Java

LAB_3 ASSIGNMENT

QUESTION:

Create an interface GeometricFigure with two abstract methods area and setDimensions. Create each of the following classes.

- a. Create Ellipse as a subclass of GeometricFigure. Ellipse has two attributes a and b, both of type int for major and minor axes.
- b. Create Rectangle as a subclass of GeometricFigure. Rectangle has two attributes a and b, both of type int for length and width.
- c. Create Square as a subclass of Rectangle and call super class methods to accomplish the tasks.

Write a main class to test above classes.

SOLUTION:

```
import java.util.*;  
interface GeometricFigure  
{  
    public abstract double Area();  
    public abstract void setDimensions(int a,int b);  
}  
class Ellipse implements GeometricFigure  
{  
    int a;  
    int b;  
    public void setDimensions(int a,int b)  
    {
```

```
this.a=a;
this.b=b;
System.out.println("The Major axis of Ellipse is :" +a+ "\nThe
Minor axis of Ellipse is :" +b);
}
public double Area()
{
    return Math.PI*a*b;
}
}
class Rectangle implements GeometricFigure
{
    int length;
    int width;
    public void setDimensions(int len,int wid)
    {
        length=len;
        width=wid;
        System.out.println("The Length of Rectangle is
:" +length+ "\nThe Width of Rectangle is :" +width);
    }
    public double Area()
    {
        return length*width;
    }
}
class Square extends Rectangle
{
    int s;
    int h;
    public void setDimensions(int s,int s1)
```

```
{  
    this.s=s;  
    h=s1;  
}  
public double Area()  
{  
    return s*s;  
}  
}  
  
class Mail  
{  
    public static void main(String args[])  
    {  
        Square obj=new Square();  
        obj.setDimensions(10,12);  
        System.out.println("Area of Square is :" +obj.Area());  
        Rectangle ob=new Rectangle();  
        ob.setDimensions(12,32);  
        System.out.println("Area of Rectangle is :" +ob.Area());  
        Ellipse ob1=new Ellipse();  
        ob1.setDimensions(23,45);  
        System.out.println("Area of Ellipse is :" +ob1.Area());  
    }  
}
```

OUTPUT:

The screenshot shows a Sublime Text window with multiple tabs open. The current tab is 'Mail.java'. The code in 'Mail.java' is as follows:

```
1 import java.util.*;
2 interface GeometricFigure
3 {
4     public abstract double Area();
5     public abstract void setDimensions(int a,int b);
6 }
7 class E { Command Prompt
8 {
9     in:D:\java_pgm>javac Mail.java
10    in
11    pu:D:\java_pgm>java Mail
12    { Area of Square is :100.0
13      The Length of Rectangle is :12
14      The Width of Rectangle is :32
15      Area of Rectangle is :384.0
16      pu The Major axis of Ellipse is :23
17      { The Minor axis of Ellipse is :45
18        Area of Ellipse is :3251.548396465436
19      }
20    }
21 } D:\java_pgm>
22 class R
23 {
24     int
25     int
26     pub
27     {
28
29
30     }
31     pub
32     {
33
34     }
35     }
36   }
37 class S
38 {
39     int
40     int h;
41     public void setDimensions(int s,int si)
42 }
```

The command prompt window shows the execution of the Java code. It outputs the area of a square (100.0), the dimensions of a rectangle (length 12, width 32, area 384.0), and the area of an ellipse (3251.548396465436).

LAB ASSINGEMENT_1

QUESTION:

Write a static method to convert a String into corresponding telephone number. This static method should use following rules for substituting letters or symbols with digits.

If current character is an uppercase vowel, then replace with 2 and for lower vowel replace with 3. Similarly, for uppercase consonants replace with 4 and for lowercase consonants replace with 5. If any special symbol is seen, then it must be replaced with its corresponding position (Consider that starting position is 1).

SOLUTION:

```
import java.util.Scanner;
class Lab1
{
    static void convert(char str[])
    {
        int N = str.length;

        for (int i = 0; i < N; i++)
        { int count=0;
        if (str[i] == 'A' || str[i] == 'E' ||
            str[i] == 'I' || str[i] == 'O' ||
            str[i] == 'U')
        {
            System.out.print(2);
        }
        else if (str[i] == 'a' || str[i] == 'e' ||
            str[i] == 'i' || str[i] == 'o' ||
            str[i] == 'u')
        {
            System.out.print(3);
        }
        else if (str[i] >= 'A' && str[i] <= 'Z')
        {
            System.out.print(4);
        }
    }
}
```

```

        else if (str[i] >= 'a' && str[i] <= 'z')
    {
        System.out.print(5);
    }
else if (str[i] >= '0' && str[i] <= '9') {
    System.out.print(str[i]);
}
else{
    System.out.print(i+1);
}
}
}

public static void main(String []args)
{
    Scanner in =new Scanner(System.in);
    System.out.print("Enter the String To Convert :");
    String s = in.nextLine();
    System.out.print("The Converted String is .");
    convet(s.toCharArray());
}
}

```

OUTPUT:

```

D:\java_pgm>javac Lab1.java
D:\java_pgm>java Lab1
Enter the String To Convert :Chenu@123
The Converted String is :453536123
D:\java_pgm>

```

LAB ASSINGEMENT_2

QUESTION.1:

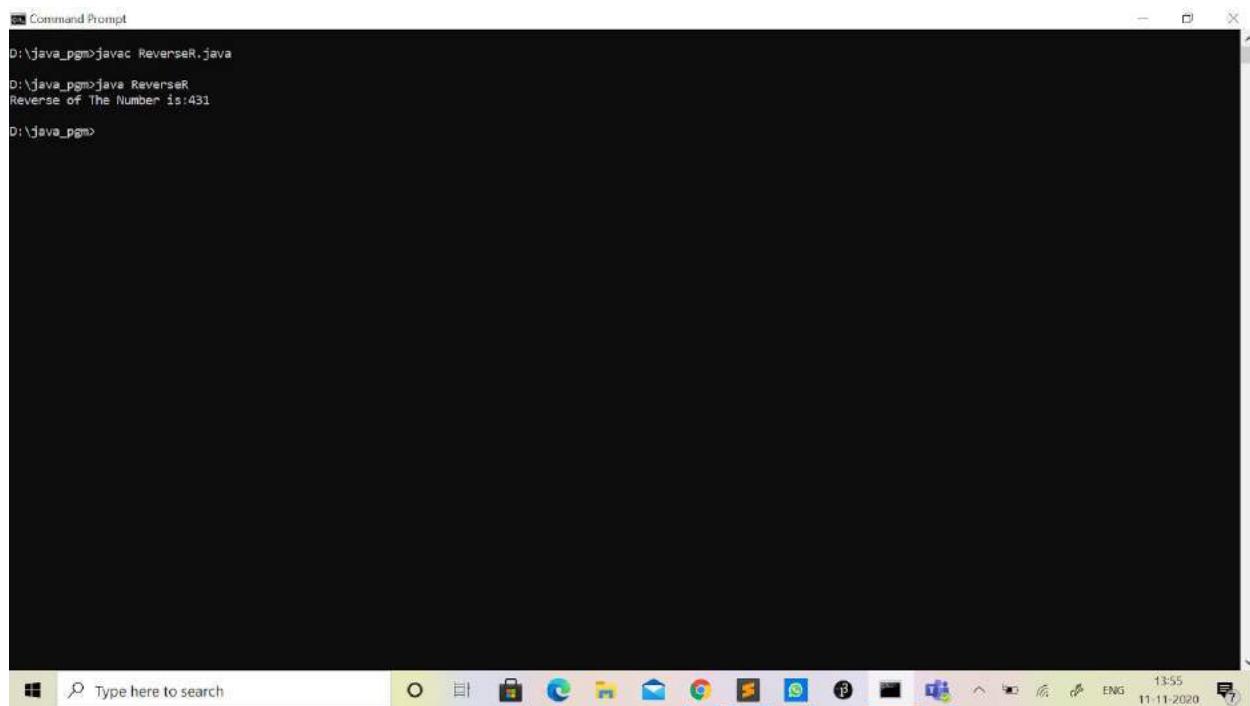
Create a class Reverse with one data member called num of type int and a static method find() to return reverse of num. Create another class ReverseR from Reverse which overrides method find() and this override method uses recursion to find and returns reverse of data member of its super class.

SOLUTION:

```
class Reverse
{
    static int n1;
    public static int find(int n)
    {
        int sum=0;
        while(n>0)
        {
            int s=n%10;
            sum=sum*10+s;
            n=n/10;
        }
        return sum;
    }
}
class ReverseR extends Reverse
{
    public static int find(int n)
    {
        int s=0;
        while(n!=0)
        {
            int r=n%10;
            s=s*10+r;
            n=n/10;
            find(n);
        }
        return s;
    }
}
```

```
public static void main(String args[])
{
    ReverseR t=new ReverseR();
    t.n1=134;
    System.out.println("Reverse of The Number is:"+t.find(t.n1));
}
}
```

OUTPUT:



The screenshot shows a Windows Command Prompt window titled "Command Prompt". The window contains the following text:

```
D:\java_pgm>javac ReverseR.java
D:\java_pgm>java ReverseR
Reverse of The Number is:431
D:\java_pgm>
```

The window has a standard Windows title bar and taskbar at the bottom. The taskbar includes icons for File Explorer, Task View, Start, Edge, Mail, Google Chrome, File Explorer, and others. The system tray shows the date and time as "11-11-2020 13:55" and the language as "ENG".

QUESTION_2:

A two-dimensional array with equal number of rows and columns filled with distinct integers is a magic square if the sum of the elements in each row, in each column, and in the two diagonals have the same value. Write a program to test whether or not a two-dimensional array is a magic square. Assume that user is supposed to enter numbers row by row. The program must do the following:

- a. Prompt user for enough number of integers
- b. Verify that the integers are distinct
- c. Test whether or not it is a magic square

SOLUTION:

```
import java.util.Scanner;
public class MagicSquare {
    public static void main(String[] args) {
        int i,j;
        int sum_row, sum_col, sum_diagonal = 0, sum = 0;
        boolean magic=true;
        int[][] square = new int[3][3];
        Scanner in = new Scanner(System.in);

        //Read number for each cell
        System.out.println("Enter the elements : ");
        for ( i=0; i<3; i++)
            for ( j=0; j<3; j++)
                square[i][j] = in.nextInt();

        //Display square
        System.out.println("the square matrix is:");
        for (i=0; i<3; i++)
        {
            for ( j=0; j<3; j++)
                System.out.print(square[i][j] + " ");
            System.out.println();
        }

        //Calculate sum of the first row
        for ( j=0; j<3; j++)
```

```

sum += square[0][j];

//Calculate sum of 2nd and 3rd row
for ( i=1; i<3; i++) {
    sum_row = 0;
    for (j=0; j<3; j++)
        sum_row += square[i][j];
    if (sum_row != sum) {
        magic = false;
        break;
    }
}

//Calculate sum of each column
if (magic) {
    for (j=0; j<3; j++) {
        sum_col = 0;
        for ( i=0; i<3; i++)
            sum_col += square[i][j];
        if (sum_col!= sum) {
            magic = false;
            break;
        }
    }
}

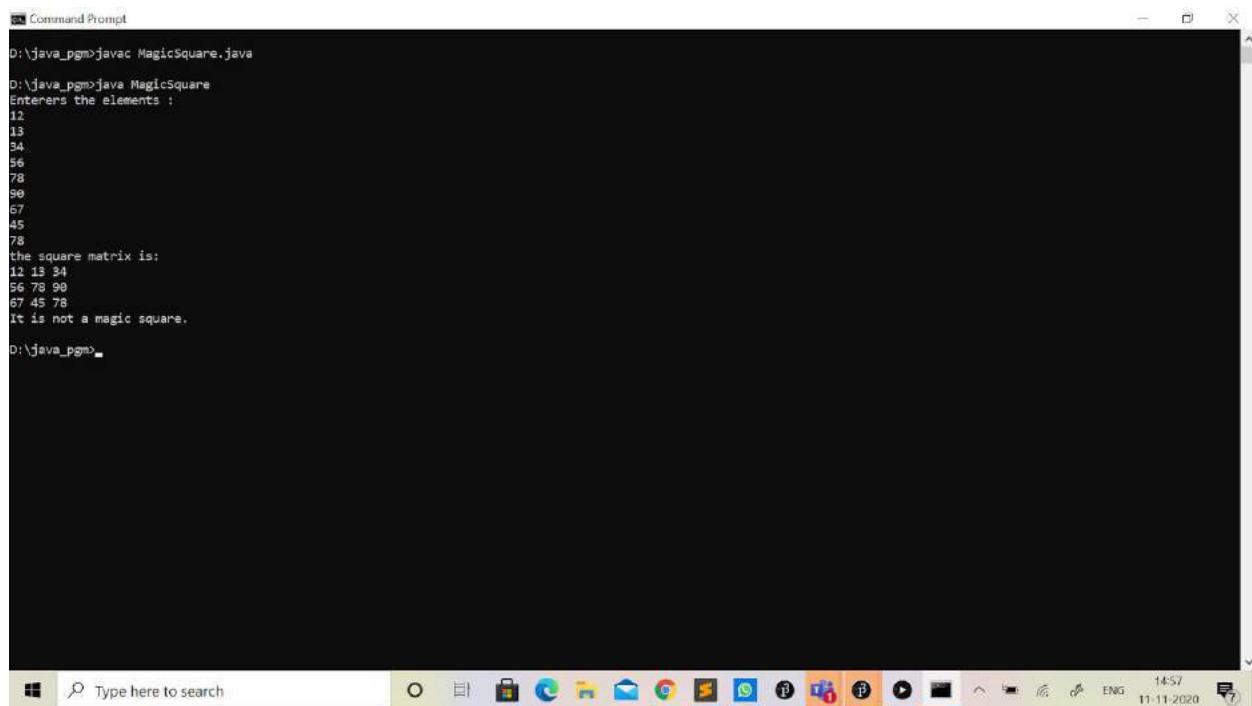
//Calculate sum of first diagonal
if (magic) {
    for ( i=0; i<3; i++)
        for ( j=0; j<3; j++)
            if ( i== j)
                sum_diagonal += square[i][j];
    if (sum_diagonal != sum) {
        magic = false;
    }
}

//Calculate sum of second diagonal
if (magic) {
    sum_diagonal = 0;
    for ( i=0; i<3; i++)
        for (j=0; j<3; j++)
            if ((i+j) == 2)
                sum_diagonal += square[i][j];
}

```

```
if (sum_diagonal != sum) {  
    magic = false;  
}  
}  
  
//Display result  
if (magic)  
    System.out.println("It magic square!");  
else  
    System.out.println("It is not a magic square.");  
}  
}
```

OUTPUT:



The screenshot shows a Windows Command Prompt window titled "Command Prompt". The command `D:\java_pgm>javac MagicSquare.java` is entered, followed by `D:\java_pgm>java MagicSquare`. The program prompts the user to "Enter the elements :". The user inputs the matrix elements: 12, 13, 34, 56, 78, 90, 67, 45, 78. The program then displays the matrix and concludes that it is not a magic square.

```
D:\java_pgm>javac MagicSquare.java  
D:\java_pgm>java MagicSquare  
Enter the elements :  
12  
13  
34  
56  
78  
90  
67  
45  
78  
the square matrix is:  
12 13 34  
56 78 90  
67 45 78  
It is not a magic square.  
D:\java_pgm>
```

NAME: THOTA GURUTHEJA REDDY

REGNO: 19BCD7034

LAB:OOP LAB 1

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($m_2+m_3+m_1/3$)

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();

```

NAME: THOTA GURUTHEJA REDDY
REGNO: 19BCD7034
LAB:OOP LAB 1

    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();
        }
    }
}

```

NAME: THOTA GURUTHEJA REDDY

REGNO: 19BCD7034

LAB:OOP LAB 1

}

OUTPUT:

```
Enter Details of student
Enter name
Guru
Enter registration number
19bcd7034
Enter Department
cse
Enter address
vijayawada
Enter Year
2020
Marks in subject 1
88
Marks in subject 2
89
Marks in subject 3
87
Enter Details of student
Enter name
Bhuvanesh
Enter registration number
19bcd7088
Enter Department
cse
Enter address
guntur
Enter Year
2020
Marks in subject 1
85
Marks in subject 2
83
Marks in subject 3
86
Enter Details of student
Enter name
Ramesh
Enter registration number
19bcd7124
Enter Department
cse
Enter address
kadapa
Enter Year
}2020
```

NAME: THOTA GURUTHEJA REDDY

REGNO: 19BCD7034

LAB:OOP LAB 1

```
Enter address
kadapa
Enter Year
2020
Marks in subject 1
78
Marks in subject 2
89
Marks in subject 3
92
Enter Details of student
Enter name
Manish
Enter registration number
19bcd7132
Enter Department
cse
Enter address
kurnool
Enter Year
2020
Marks in subject 1
78
Marks in subject 2
86
Marks in subject 3
82
Enter Details of student
Enter name
Ujwal
Enter registration number
19bcd7325
Enter Department
cse
Enter address
nellore
Enter Year
2020
Marks in subject 1
83
Marks in subject 2
87
Marks in subject 3
81
```

NAME: THOTA GURUTHEJA REDDY

REGNO: 19BCD7034

LAB:OOP LAB 1

Details of the student are:

Name of the Student is Guru

Registration number of Student is 19bcd7034

Department of Student is cse

year 2020

Address of Student vijayawada

Marks of the Student are 88,89,87

Percentage of Student is 88.0

Details of the student are:

Name of the Student is Bhuvanesh

Registration number of Student is 19bcd7088

Department of Student is cse

year 2020

Address of Student guntur

Marks of the Student are 85,83,86

Percentage of Student is 84.0

Details of the student are:

Name of the Student is Ramesh

Registration number of Student is 19bcd7124

Department of Student is cse

year 2020

Address of Student kadapa

Marks of the Student are 78,89,92

Percentage of Student is 86.0

Details of the student are:

Name of the Student is Manish

Registration number of Student is 19bcd7132

Department of Student is cse

year 2020

Address of Student kurnool

Marks of the Student are 78,86,82

Percentage of Student is 82.0

Details of the student are:

Name of the Student is Ujwal

Registration number of Student is 19bcd7325

Department of Student is cse

year 2020

Address of Student nellore

Marks of the Student are 83,87,81

Percentage of Student is 83.0

NAME: THOTA GURUTHEJA REDDY

REGNO: 19BCD7034

LAB:OOP LAB 1

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;
        }
    }
}
```

NAME: THOTA GURUTHEJA REDDY
REGNO: 19BCD7034
LAB:OOP LAB 1

```
    }  
  
    if(temp==sum)  
  
        System.out.println("palindrome number \n Door opened");  
  
    else  
  
        System.out.println("not palindrome \n Door not opened");  
  
    }  
}
```

OUTPUT:

```
5  
103  
200  
303  
400  
503  
palindrome number  
Door opened  
  
...Program finished with exit code 0  
Press ENTER to exit console.█
```

NAME: THOTA GURUTHEJA REDDY

REGNO: 19BCD7034

LAB:OOP LAB 1

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;
```

NAME: THOTA GURUTHEJA REDDY
REGNO: 19BCD7034
LAB:OOP LAB 1

```
        }  
    }  
}  
  
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:

```
Number of strings=4  
Enter Strings  
glasses  
spoons  
chairs  
gloves  
Strings in ascending order are: chairs,glasses,gloves,spoons  
  
...Program finished with exit code 0  
Press ENTER to exit console.[]
```

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();
}
}
```

```
Enter radius of the circle
6
Area of the circle is: 113.03999999999999
Circumference of the circle is: 37.68
```

```
...Program finished with exit code 0
Press ENTER to exit console.[]
```

```
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();
}
}
```

```
Is there 3 electives yes or no
yes
Enter Name
Gurutheja
Enter registration number
197034
Enter address
vijayawada
Enter sub_mark1
98
Enter sub_mark2
89
Enter sub_mark3
95
Enter elec1
86
Enter elec2
83
Enter elec3
84
Is there 3 electives yes or no
yes
Enter Name
Bhuvanesh
Enter registration number
197088
Enter address
Agiripalli
Enter sub_mark1
92
Enter sub_mark2
95
Enter sub_mark3
81
Enter elec1
82
Enter elec2
84
Enter elec3
75
Is there 3 electives yes or no
no
```

```
Enter Name
Ramesh
Enter registration number
197125
Enter address
Guntur
Enter sub_mark1
75
Enter sub_mark2
85
Enter sub_mark3
76
Enter elec1
15
Enter elec2
45
Details of the student are:
Name: Gurutheja
Registration number: 197125
Department: CSE
Address: vijayawada
Grade of the student is B
Details of the student are:
Name: Bhuvanesh
Registration number: 197125
Department: CSE
Address: Agiripalli
Grade of the student is B
Details of the student are:
Name: Ramesh
Registration number: 197125
Department: CSE
Address: Guntur
Grade of the student is E

...Program finished with exit code 0
Press ENTER to exit console.■
```

1.

```
import java.util.Scanner;

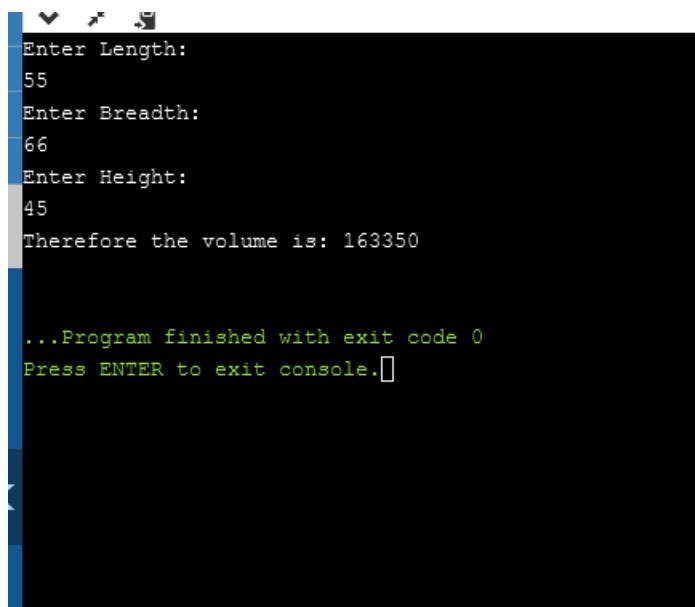
class Rectangle{
    int length;
    int breadth;
    public Rectangle(int length,int breadth){
        this.length=length;
        this.breadth=breadth;
    }
}

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:



```
Enter Length:  
55  
Enter Breadth:  
66  
Enter Height:  
45  
Therefore the volume is: 163350  
  
...Program finished with exit code 0  
Press ENTER to exit console.[]
```

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);  
}  
}
```

```
Enter name of the student  
GuruTheja  
Enter email  
vitap.ac.in  
Enter gender  
M  
Enter Department  
CSE  
Details of the student are:  
Name: GuruTheja  
Gender: M  
Email: GuruTheja@vitap.ac.in  
Department is: CSE  
  
...Program finished with exit code 0  
Press ENTER to exit console.[]
```

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();
}

}

}
```

```
Enter number of staff members:  
2  
Enter Department details  
cse  
john  
55  
3  
Enter Staff Details  
satish  
23232  
mtech  
proffessor  
6years  
Enter Department details  
Ece  
ramesh  
35  
2
```

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:

```
Enter name of the student
GuruTheja
Enter Registration number
19bcd7034
Enter type of Degree
Undergraduate
Enter years
3
Name of the Student: GuruTheja
Registration number is: 19bcd7034
Degree: Undergraduate
Years: 3
I am an Undergraduate

...Program finished with exit code 0
Press ENTER to exit console.[]
```

2. Create a class Shapes with a method “calculatearea()” to find the area of a circle (πr^2), square(a^2), rectangle($w \times h$) 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);
    }
}
```

```
Enter radius:  
5  
Enter height:  
2  
Enter width:  
6  
Enter length:  
3  
Enter base length:  
7  
Area of circle is: 78.5  
Area of square is: 9.0  
Area of rectangle is: 12.0  
Area of triangle is: 7.0  
  
...Program finished with exit code 0  
Press ENTER to exit console.█
```

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[]={ };
    }
}
```

```
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:

```
Enter integer  
6  
Enter double  
4.3  
Enter String  
Guru  
Enter number of elements in array  
5  
88  
45  
62  
71  
94  
Sum of numbers is 12  
Sum of numbers is 10.3  
Sum of numbers is 10.3  
Sum of numbers is 8.6  
Sum of strings is GuruGuru  
Sum of numbers in array is 360  
  
...Program finished with exit code 0  
Press ENTER to exit console.[]
```

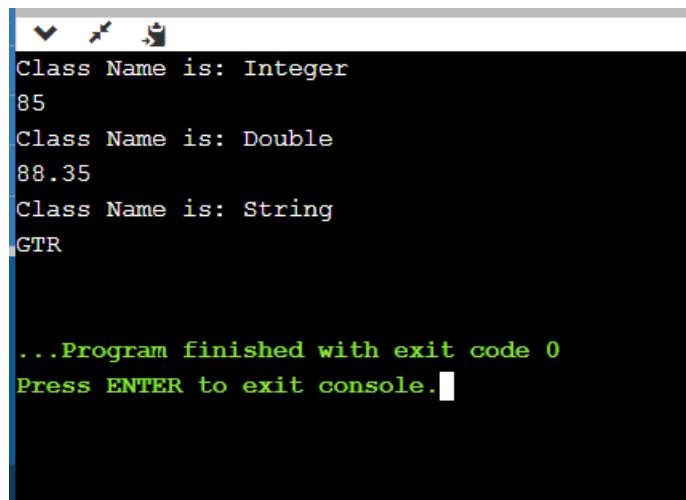
1. Write a java generic program to create class that takes one type parameter to print the different data type value and its class name.

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:



A screenshot of a Java console window. The window has a dark background and light-colored text. It displays the following output:

```
Class Name is: Integer
85
Class Name is: Double
88.35
Class Name is: String
GTR

...Program finished with exit code 0
Press ENTER to exit console.■
```

2. 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));
        }
    }
}
```

```
Enter the two numbers:
88
96
Maximum of given two numbers is:
96

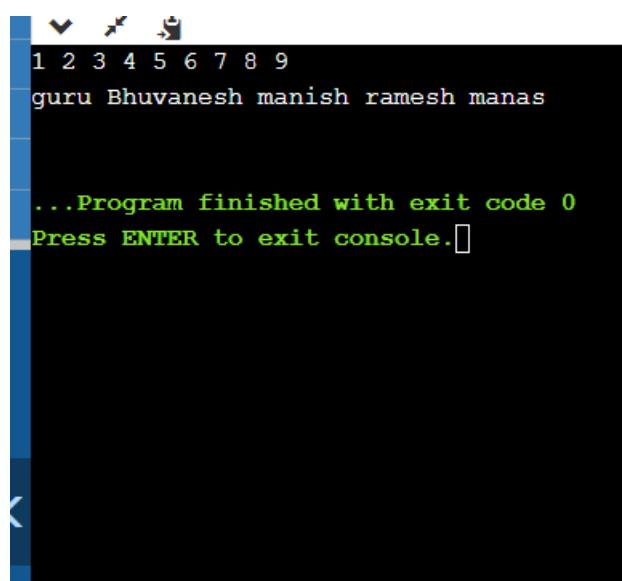
...Program finished with exit code 0
Press ENTER to exit console.□
```

OutPut:

3. 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);
    }
}
```



```
1 2 3 4 5 6 7 8 9
guru Bhuvanesh manish ramesh manas

...Program finished with exit code 0
Press ENTER to exit console.[]
```

Output:

4. 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:

```
Frequencies of each element in array are:  
1 1 1 1 2 2 1 1  
Number of even numbers in array is: 6  
  
...Program finished with exit code 0  
Press ENTER to exit console.[]
```

1. Write a java program to demonstrate ArithmeticException, NullPointerException, ArrayIndexOutOfBoundsException and NumberFormatException 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:

```
Exception in thread "main" java.lang.ArithmetricException: / by zero
    at Main.method1(Main.java:4)
    at Main.main(Main.java:22)
```

```
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 3
    at Main.method2(Main.java:10)
    at Main.main(Main.java:23)
```

```
Exception in thread "main" java.lang.NullPointerException
    at Main.method3(Main.java:15)
    at Main.main(Main.java:23)
```

```
Exception in thread "main" java.lang.NumberFormatException: For input string: "hello"
    at java.lang.NumberFormatException.forInputString(NumberFormatException.java:65)
    at java.lang.Integer.parseInt(Integer.java:580)
    at java.lang.Integer.parseInt(Integer.java:615)
    at Main.method4(Main.java:19)
    at Main.main(Main.java:24)
```

```
...Program finished with exit code 1
```

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(ArithmetricException e)

{

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

}

finally

{

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

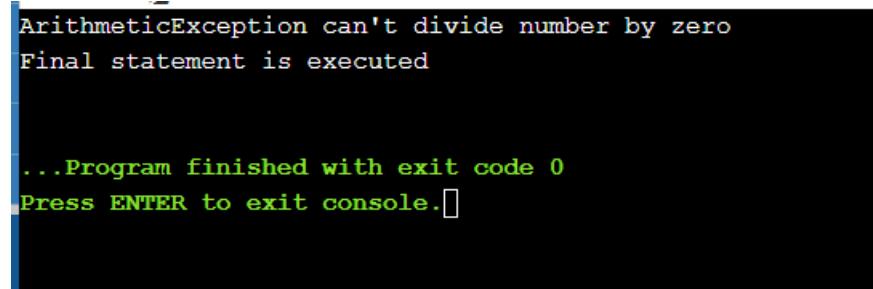
}

}

}

```

OutPut:



A screenshot of a terminal window displaying the output of a Java program. The output is as follows:

```

ArithmetricException can't divide number by zero
Final statement is executed

...Program finished with exit code 0
Press ENTER to exit console. []

```

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 voting");  
            }  
        }  
        catch(Exception e)  
        {  
            System.out.println("Exception "+e);  
        }  
    }  
}
```

Output:

```
Enter the age of voter  
13  
Age of the voter is less than 18, voting not allowed  
Exception check18  
  
...Program finished with exit code 0  
Press ENTER to exit console.
```

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);
}
}
```

```
Gurutheja
Enter Registration number:
19bcd7034
Enter mark1:
98
Enter mark2:
86
Enter details of Student-2
Enter Name :
Bhuvanesh
Enter Registration number:
19bcd7088
Enter mark1:
89
Enter mark2:
94
Student-1 Details:
Name Gurutheja
Registration number 19bcd7034
mark1 98
mark2 86
Student-2 Details:
Name Bhuvanesh
Registration number 19bcd7088
mark1 89
mark2 94

Student S1 has got High Average

...Program finished with exit code 0
Press ENTER to exit console.[]
```

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:

```
Enter number of rows in matrix1
2
Enter number of columns in matrix1
2
Enter elements of matrix1
1
2
3
4
Enter number of rows in matrix2
2
Enter number of columns in matrix2
3
Enter elements of matrix2
5
6
7
8
9
10
Matrix addition is not possible
MatrixAdditionNotPossible
```

```
...Program finished with exit code 0
Press ENTER to exit console.[]
```

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:

```
Name Gurutheja  
Registration number 197034  
Course CSE-DA  
part1 23  
part2 32  
Core 35  
Total 90  
NOT ELIGIBLE  
NotEligibleException  
  
...Program finished with exit code 0  
Press ENTER to exit console.[]
```

```
Name Gurutheja  
Registration number 197034  
Course CSE-DA  
part1 45  
part2 32  
Core 65  
Total 142  
WAITING LIST  
WaitingListException  
  
...Program finished with exit code 0  
Press ENTER to exit console.[]
```

```
Name Gurutheja
Registration number 197034
Course CSE-DA
part1 45
part2 42
Core 75
Total 162
ELIGIBLE

...Program finished with exit code 0
Press ENTER to exit console.[]
```

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:

```
Enter the String
asdfghjkl;'zxcvbnm,./qwertyuiop[]asdfghjkl;'zxcvbnm,./qwertyuiop[]asdfghjkl;
'zxcvbnm,./asdfghjkl;'qwertyuiop[]qwertyuiop[]qwertyuiop[]asdfghjkl;'zxcvbnm
,.qwertyuiop[]asdfghjkl;zxcvbnm,.1234567890-
ExceptionLineTooLong

...Program finished with exit code 0
Press ENTER to exit console.[]
```

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:

```
Numbers both Fibonacci and prime below 100 are:  
1  
2  
3  
5  
13  
89  
[Finished in 3.2s]
```

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);
    }
}

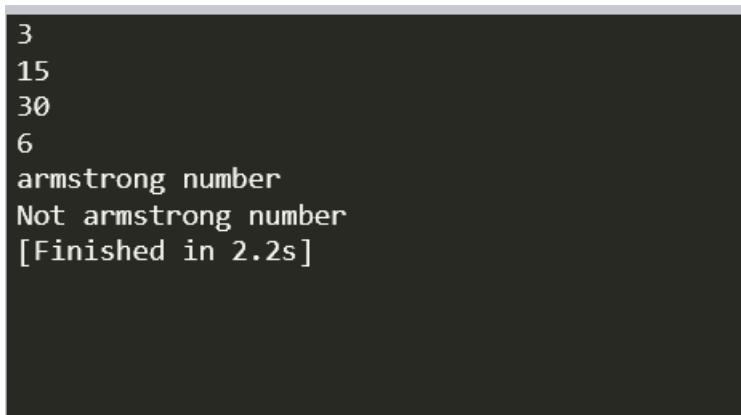
class Thread2 extends A{
    int g;
    Thread2(int g){
        this.g=g;
    }
}
```

```
public void run(){
    super.armstrong(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:



```
3
15
30
6
armstrong number
Not armstrong number
[Finished in 2.2s]
```

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:

```
Enter the size of matrix  
3  
Enter elements in matrix  
4  
5  
6  
78  
9  
6  
5  
4  
2  
Total frequency of odd numbers is 3  
Total frequency of even numbers is 6
```

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:

```
1
2
3
1
2
3
1
2
3
...
Program finished with exit code 0
Press ENTER to exit console.
```

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
3
212
1859
846
632
935
591
307
1911
368
1127
8791
Sum is odd
2
2
1436
1094
1355
1208
1213
90
658
781
1398
1211
10446
Sum is even

...Program finished with exit code 0
Press ENTER to exit console.□
```

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 $0 \leq k \leq 25$.

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("Decrypted message " + mDecryption(c, key));
}
}

```

Output:

```
Enter the Message to encrypt
gurukillher
Enter the Key for encryption
5
Encrypted message lzwzpnqqmjw
Deprecated message gurukillher

...Program finished with exit code 0
Press ENTER to exit console.□
```

b.

```
public class Main extends Thread
{
    public static String m = "gdhrzfnnncanx";
    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:

```
Message for key 1 is fcgqyemmbz mw
Message for key 0 is gdhrzfnn canx
Message for key 2 is ebfpxd llaylv
Message for key 4 is czdnvbjjywjt
Message for key 5 is bycmuaiixvis
Message for key 3 is daeowckkzxku
Message for key 6 is axbltz hhw uhr
Message for key 7 is zwaksyggvtgq
Message for key 8 is yvzjrxffusfp
Message for key 9 is xuyiqweetreo
Message for key 10 is wtxhpvddsqdn
Message for key 11 is vswgouccrpcm
Message for key 12 is urvfntbbqobl
Message for key 13 is tquemsaapnak
Message for key 14 is sptdlrz zomzj
Message for key 15 is rosckqyynlyi
Message for key 16 is qnrbjp xxm kxh
Message for key 18 is olpz hnvvkivf
Message for key 17 is pmqaiowwljwg
Message for key 19 is nkoygmuujhue
Message for key 20 is m jnxflttigtd
Message for key 21 is limweksshfsc
Message for key 22 is kh1vdjrrgerb
Message for key 23 is jgkuciqqfdqa
Message for key 24 is ifjtbh ppecpz
Message for key 25 is heisagoodboy
```

```
...Program finished with exit code 0
Press ENTER to exit console.□
```

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:

```
Hi  
Hello  
Tomorrow do you have exam  
Yes  
Have you prepared?  
ya I am preparing  
All the best!  
Thank you  
  
...Program finished with exit code 0  
Press ENTER to exit console.█
```

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:

```
PRODUCER produced 1  
CONSUMER consumed 1  
PRODUCER produced 2  
CONSUMER consumed 2  
PRODUCER produced 3  
CONSUMER consumed 3  
PRODUCER produced 4  
CONSUMER consumed 4  
PRODUCER produced 5  
CONSUMER consumed 5
```

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.out.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 if (s1.equals("*"))
                te = (Double.parseDouble(s0) * Double.parseDouble(s2));
            l.setText(te + "");
        }
    }
}

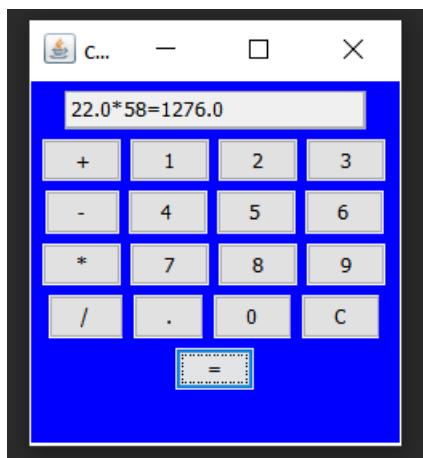
```

```

        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(JFrame.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:



Details of Student

Name	RegNo	Address	Dept	CGPA	
GuruTheja	19BCD7034	Vijayawada	CSE	9.8	▲
Bhuvanesh	19BCD7088	Agiripalli	CSE	9.8	▼



"C:\Program Files\Java\jdk-9.0.4\bin\java.exe" "-javaagent:C:\Program Files\

, "CSE", "9.
"};

ROLLBAR_AL
BAR_ALWAYS

);

Name: Thota Gurutheja Reddy

Reg No: 19BCD7034

Lab 12

1. Create your own menu and display it using javaFX

Ans:

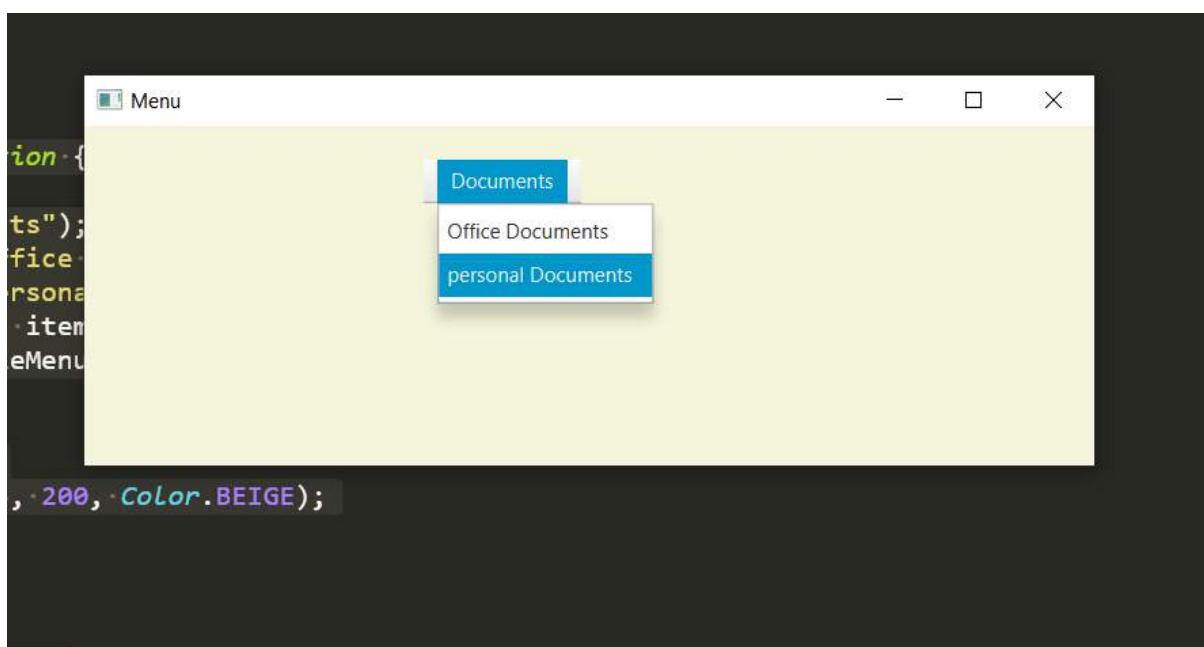
```
import javafx.application.Application;  
import javafx.scene.Group;  
import javafx.scene.Scene;  
import javafx.scene.control.Menu;  
import javafx.scene.control.MenuBar;  
import javafx.scene.control.MenuItem;  
import javafx.scene.paint.Color;  
import javafx.stage.Stage;  
  
public class MenuBar_1 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();

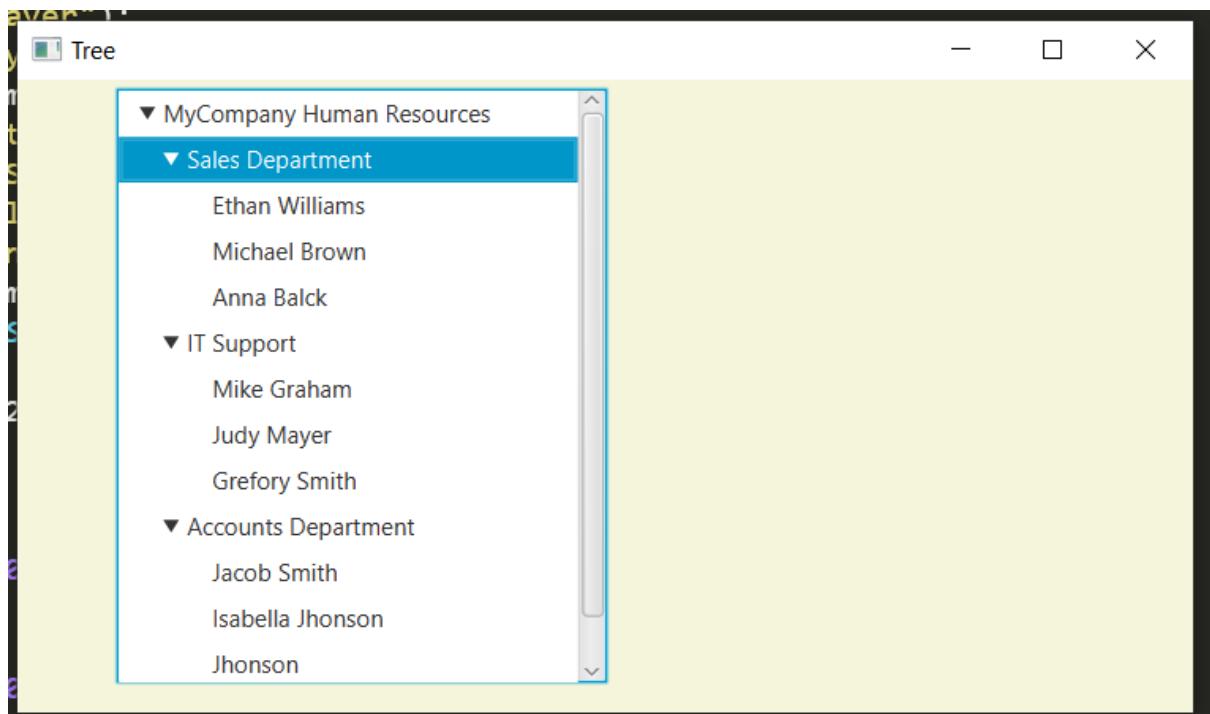
}

public static void main(String args[]){

    launch(args);

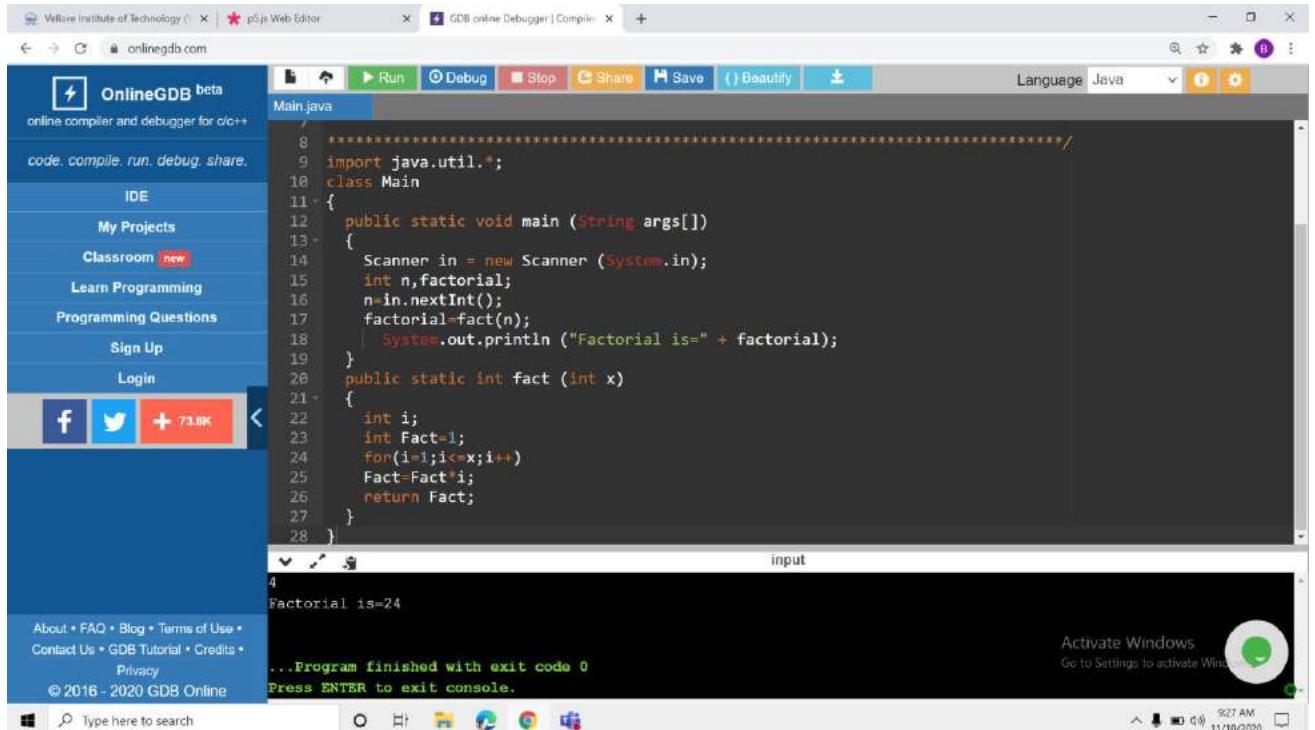
}

}
```



Object oriented programming lab 1

- 1) Write a Program to find Factorial of a given integer using Recursive Method call



The screenshot shows the OnlineGDB beta IDE interface. On the left, there's a sidebar with links for My Projects, Classroom (new), Learn Programming, Programming Questions, Sign Up, and Login. Below the sidebar are social media sharing icons for Facebook, Twitter, and LinkedIn, with a count of 73.8K. The main workspace is titled "Main.java" and contains the following Java code:

```
8 ****
9 import java.util.*;
10 Class Main
11 {
12     public static void main (String args[])
13     {
14         Scanner in = new Scanner (System.in);
15         int n,factorial;
16         n=in.nextInt();
17         factorial=fact(n);
18         System.out.println ("Factorial is=" + factorial);
19     }
20     public static int fact (int x)
21     {
22         int i;
23         int Fact=1;
24         for(i=1;i<=x;i++)
25             Fact=Fact*i;
26         return Fact;
27     }
28 }
```

The output window below the code shows the result of running the program with input '4':

```
4
Factorial is=24
...Program finished with exit code 0
Press ENTER to exit console.
```

- 2) Write separate methods for Summation, Subtraction, Multiplication and Division. Now take a mathematical expression as input from user and evaluate it to find the value.

Vellore Institute of Technology | pjs Web Editor | GDB online Debugger | Compiler | +

onlinedb.com

OnlineGDB beta
online compiler and debugger for c/c++
code. compile. run. debug. share.

IDE
My Projects
Classroom new
Learn Programming
Programming Questions
Sign Up
Login

[f](#) [t](#) [+ 73.8K](#)

Main.java

```
9 import java.util.*;
10 public class Main {
11
12 public static void main(String[] args) {
13 // TODO Auto-generated method stub
14 Scanner in = new Scanner(System.in);
15 System.out.print("Enter the expression : ");
16 String s = in.nextLine();
17 String exp[] = {"", ""};
18 int str_ind=0;
19 char c=' ';
20 for(int i=0;i<s.length();i++) {
21 if(s.charAt(i)=='+'||s.charAt(i)=='-'||s.charAt(i)=='*'||s.charAt(i)=='/') {
22 c=s.charAt(i);
23 str_ind++;
24 }
25 else {
26 exp[str_ind]=exp[str_ind]+s.charAt(i);
27 }
28 }
29 if(c=='*')
```

input

Enter the expression : 9*6

Result is :54

...Program finished with exit code 0
Press ENTER to exit console.

Activate Windows
Go to Settings to activate Windows 10

Type here to search

10:12 AM 11/10/2020

Vellore Institute of Technology | pjs Web Editor | GDB online Debugger | Compiler | +

onlinedb.com

OnlineGDB beta
online compiler and debugger for c/c++
code. compile. run. debug. share.

IDE
My Projects
Classroom new
Learn Programming
Programming Questions
Sign Up
Login

[f](#) [t](#) [+ 73.8K](#)

Main.java

```
29 if(c=='+') {
30 System.out.println("Result is :" + sum(exp[0],exp[1]));
31 else if(c=='-') {
32 System.out.println("Result is :" + sub(exp[0],exp[1]));
33 else if(c=='*') {
34 System.out.println("Result is :" + mul(exp[0],exp[1]));
35 else {
36 System.out.println("Result is :" + div(exp[0],exp[1]));
37 }
38 static int sum(String x,String y) {
39 int X = Integer.parseInt(x);
40 int Y = Integer.parseInt(y);
41
42 return X+Y;
43 }
44 static int sub(String x,String y) {
45 int X = Integer.parseInt(x);
46 int Y = Integer.parseInt(y);
47 return X-Y;
48 }
49 static int mul(String x,String y) {
```

input

Enter the expression : 9*6

Result is :54

...Program finished with exit code 0
Press ENTER to exit console.

Activate Windows
Go to Settings to activate Windows 10

Type here to search

10:12 AM 11/10/2020

The screenshot shows a Java application running on OnlineGDB beta. The code in Main.java performs arithmetic operations on strings representing integers:

```

Main.java
39 int X = Integer.parseInt(x);
40 int Y = Integer.parseInt(y);
41
42 return X*Y;
43 }
44 static int sub(String x,String y) {
45 int X = Integer.parseInt(x);
46 int Y = Integer.parseInt(y);
47 return X-Y;
48 }
49 static int mul(String x,String y) {
50 int X = Integer.parseInt(x);
51 int Y = Integer.parseInt(y);
52 return X*Y;
53 }
54 static double div(String x,String y) {
55 int X = Integer.parseInt(x);
56 int Y = Integer.parseInt(y);
57 return X/Y;
58 }
59 }

```

The input provided was "9*6", and the result was "54". The output window shows the program finished with exit code 0.

- 3) Write a methods to 1) find the substring from a given string of user given length, 2) Make all the characters of a string to upper case, 3) Make all the characters of a string to lower case. User should provide the string and chose option from these.

The screenshot shows a Java application running on GDB online Debugger (Compiler). The code in Main.java provides three options for string manipulation:

```

Main.java
11 import java.util.*;
12 public class Main {
13
14     public static void main(String[] args) {
15         // TODO Auto-generated method stub
16
17         Scanner in = new Scanner(System.in);
18         System.out.print("Enter the String : ");
19         String s=in.nextLine();
20         System.out.println("Enter the choice\n1-SubString\n2- To lower case\n3-To Upper case");
21         int n = in.nextInt();
22         switch(n) {
23             case 1:{
24                 System.out.println("Enter the indexes");
25                 int o = in.nextInt();
26                 int e = in.nextInt();
27                 subString(s,e,o);
28             }
29             break;

```

The input provided was "Vit Amaravati" and "1-SubString". The output window shows the program finished with exit code 0.

GDB online Debugger | Compiler x +

onlinedb.com

Main.java

```
29         break;
30     case 2:{
31         System.out.println(toUp(s));
32     }
33     break;
34     case 3:{
35         System.out.println(tosp(s));
36     }
37     break;
38     default :
39         System.out.println("Choose from the above");
40     }
41 }
42 public static String subString(String s, int o, int e) {
43     String h="";
44     for(int i=o;i<e;i++)
45         h=h+s.charAt(i);
46     return h;
47 }
```

input

Enter the String : Vit Amaravati
Enter the choice
1-SubString

Activate Windows
Go to Settings to activate Windows.

GDB online Debugger | Compiler x +

onlinedb.com

Main.java

```
47 }
48 public static String tosp(String s) {
49     String h="";
50     for(int i=0;i<s.length();i++) {
51         char c =s.charAt(i);
52         if(c>='A'&&c<='Z') {
53             c=(char)(c+32);
54             h+=c;
55         }
56         else
57             h+=c;
58     }
59     return h;
60 }
61 public static String toUp(string s) {
62     String h="";
63     for(int i=0;i<s.length();i++) {
64         char c =s.charAt(i);
65         if(c>='a'&&c<='z') {
```

input

Enter the String : Vit Amaravati
Enter the choice
1-SubString

Activate Windows
Go to Settings to activate Windows.

The screenshot shows a Java code editor with the following code:

```

59     return h;
60   }
61   public static String toUp(String s) {
62     String h="";
63     for(int i=0;i<s.length();i++) {
64       char c =s.charAt(i);
65       if(c>='a'&&c<='z') {
66         c=(char)(c-32);
67         h+=c;
68       } else
69         h+=c;
70     }
71   }
72   return h;
73 }
74 }
75 }

```

Below the code, there is an 'input' field containing:

```

Enter the String : Vit Amaravati
Enter the choice
1-SubString
2-To lower case
3-To Upper case
2
VIT AMARAVATI

```

4) Write a method for reversing a string and another method for reversing a sentence using the reversal of string. User should give a sentence as input.

Ex. Input: That person is good. Output: doog si nosrep tahT.

The screenshot shows a Java code editor with the following code:

```

Main.java
6  Code, Compile, Run and Debug online from anywhere in world.
7
8 ****
9 import java.util.*;
10 public class Main {
11
12   public static void main(String[] args) {
13     // TODO Auto-generated method stub
14     Scanner in = new Scanner(System.in);
15     System.out.print("Enter the String : ");
16     String s = in.nextLine();
17     System.out.println("Reverse the String :" +rev(s));
18   }
19   public static String rev(String s) {
20     String h="";
21     for(int i=s.length()-1;i>=0;i--) {
22       h+=s.charAt(i);
23     }
24     return h;
25   }
26
27 }

```

Below the code, there is an 'input' field containing:

```

Enter the String : That person is good
Reverse the String :doog si nosrep tahT

```

At the bottom of the console, it says:

```

...Program finished with exit code 0
Press ENTER to exit console.

```

Object oriented programming lab 2

1) write a Program to find a given integer is prime or not using static Method one time private Method another

```
1 import java.util.Scanner;
2 public class Main {
3     static int i = 2, digit, sum;
4     public static int prime(int n) {
5         if (n <= 1)
6             return 1;
7         else
8             i++;
9         if (i < n)
10            prime();
11         else
12            return -1;
13     }
14     public static void main(String args[])
15     {
16         int n;
17         Scanner input = new Scanner(System.in);
18         System.out.println("Enter a number:");
19         n = input.nextInt();
20         ans = prime(n);
21         if (ans == 1)
22             System.out.println("It is a prime number");
23         else
24             System.out.println("It is not a prime number");
25     }
26 }
```

Execute Mode, Version, Inputs & Arguments
JDK 11.0.4 Interactive Stdin Inputs
CommandLine Arguments 29
Execute

Result
CPU Time: 0.25 sec(s), Memory: 24220 kilobyte(s)
Enter a number:
It is a prime number

Waiting for recompilation... Go to Settings to activate Windows

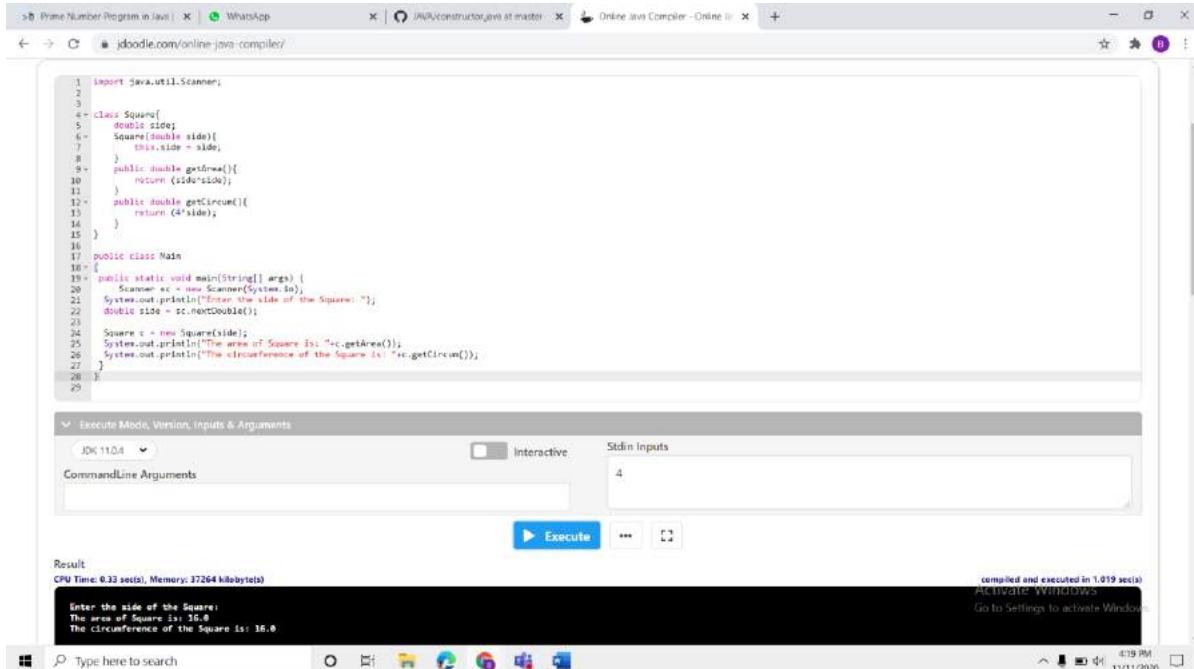
```
1 import java.util.Scanner;
2 public class Main {
3     public static void main(String[] args) {
4         Scanner in = new Scanner(System.in);
5         Main ob = new Main();
6         System.out.print("Enter the number to verify : ");
7         int n = in.nextInt();
8         if (isPrime(n))
9             System.out.println(n+" is a prime number");
10        else
11            System.out.println(n+" is not a prime number");
12    }
13    final boolean isPrime(int n) {
14        if(n==2)
15            return true;
16        for(int i=2;i<n/2;i++)
17            if(n % i == 0)
18                return false;
19        }
20        return true;
21    }
22 }
23 }
24 return false;
25 }
26 }
```

Execute Mode, Version, Inputs & Arguments
JDK 11.0.4 Interactive Stdin Inputs
CommandLine Arguments 29
Execute

Result
CPU Time: 0.31 sec(s), Memory: 35012 kilobyte(s) compiled and executed in 1.013 sec(s)
Enter the number to verify : 29 is not a prime number

Microsoft 365 Share memories across devices Microsoft 365
Buy Now →
Activate Windows Go to Settings to activate Windows

2) Write a Program to find area and circumference of shapes (like triangle, square, and rectangle) using constructors.



```

1 import java.util.Scanner;
2
3 class Square{
4     double side;
5     Square(double side){
6         this.side = side;
7     }
8     public double getArea(){
9         return (side*side);
10    }
11    public double getPerimeter(){
12        return (4*side);
13    }
14 }
15
16 public class Main
17 {
18     public static void main(String[] args) {
19         Scanner sc = new Scanner(System.in);
20         System.out.println("Enter the side of the Square: ");
21         double side = sc.nextDouble();
22
23         Square s = new Square(side);
24         System.out.println("The area of Square is: "+s.getArea());
25         System.out.println("The circumference of the Square is: "+s.getPerimeter());
26     }
27 }
28

```

Execute Mode, Version, Inputs & Arguments

JDK 11.0.4 Interactive Stdin Inputs
4

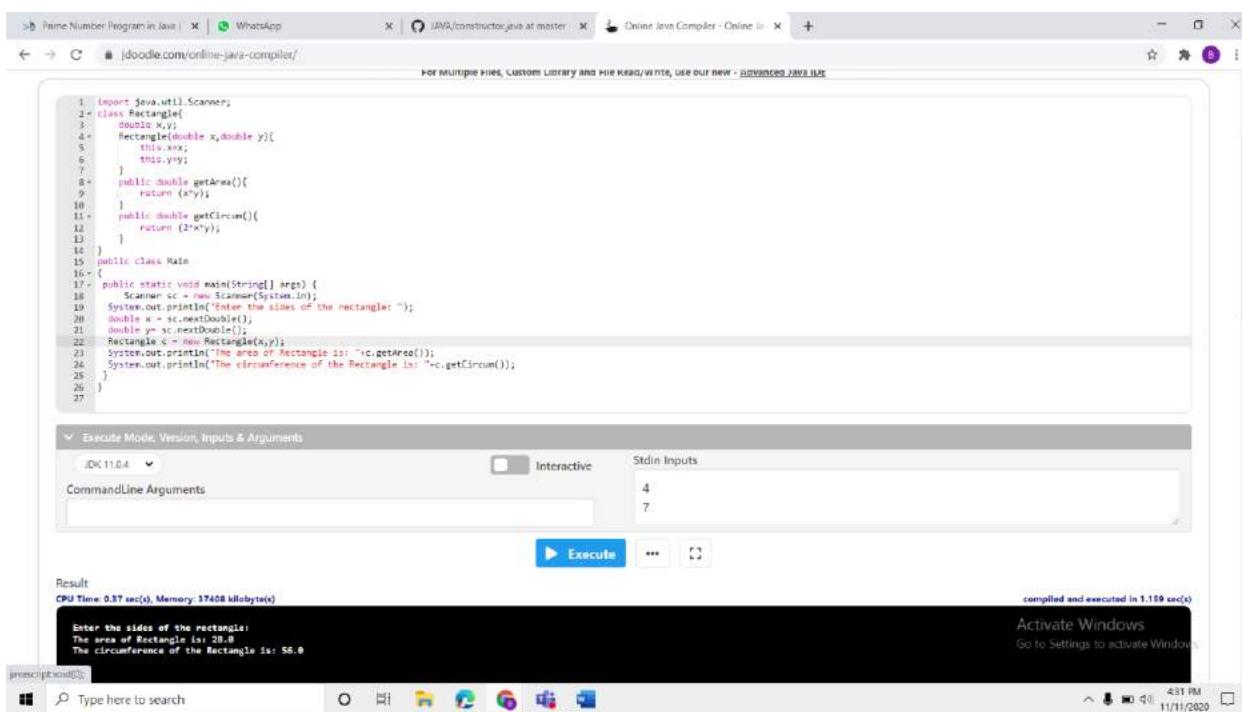
Result
CPU Time: 0.33 sec(s), Memory: 37264 kilobyte(s)

```

Enter the side of the Square:
The area of Square is: 36.0
The circumference of the Square is: 16.0

```

compiled and executed in 1.019 sec(s)
Activate Windows
Go to Settings to activate Windows



```

1 import java.util.Scanner;
2 class Rectangle{
3     double x,y;
4     Rectangle(double x,double y){
5         this.x=x;
6         this.y=y;
7     }
8     public double getArea(){
9         return (x*y);
10    }
11    public double getPerimeter(){
12        return (2*x+2*y);
13    }
14 }
15
16 public class Main
17 {
18     public static void main(String[] args) {
19         Scanner sc = new Scanner(System.in);
20         System.out.println("Enter the sides of the rectangle: ");
21         double x = sc.nextDouble();
22         double y = sc.nextDouble();
23         Rectangle c = new Rectangle(x,y);
24         System.out.println("The area of Rectangle is: "+c.getArea());
25         System.out.println("The circumference of the Rectangle is: "+c.getPerimeter());
26     }
27 }
28

```

Execute Mode, Version, Inputs & Arguments

JDK 11.0.4 Interactive Stdin Inputs
4
7

Result
CPU Time: 0.37 sec(s), Memory: 37408 kilobyte(s)

```

Enter the sides of the rectangle:
The area of Rectangle is: 28.0
The circumference of the Rectangle is: 56.0

```

compiled and executed in 1.159 sec(s)
Activate Windows
Go to Settings to activate Windows

The screenshot shows a Java code editor and a terminal window. The code is a Java program that defines a Triangle class with methods to calculate area and circumference. It also includes a main method to read side lengths from standard input and print the results. The terminal window shows the output of running the program with inputs 3, 4, and 5, displaying the area as 6.0 and the circumference as 12.0.

```
1 import java.util.Scanner;
2
3
4 class Triangle{
5     double a,b,c;
6     Triangle(double a,double b,double c){
7         this.a=a;
8         this.b=b;
9         this.c=c;
10    }
11    public double getArea(){
12        return ((a*b)/2);
13    }
14    public double getCircum(){
15        return (a+b+c);
16    }
17 }
18 public class Main{
19
20 {
21    public static void main(String[] args) {
22        Scanner sc = new Scanner(System.in);
23        System.out.print("Enter the sides of the Triangle: ");
24        double a = sc.nextDouble();
25        double b = sc.nextDouble();
26        double c=sc.nextDouble();
27
28        Triangle x = new Triangle(a,b,c);
29        System.out.println("The area of Triangle is: "+x.getArea());
30        System.out.println("The circumference of the Triangle is: "+x.getCircum());
31    }
32 }
```

Execute Mode, Version, Inputs & Arguments

Stdin Inputs

JDK 11.0.4 Interactive

3
4
5

CommandLine Arguments

Execute

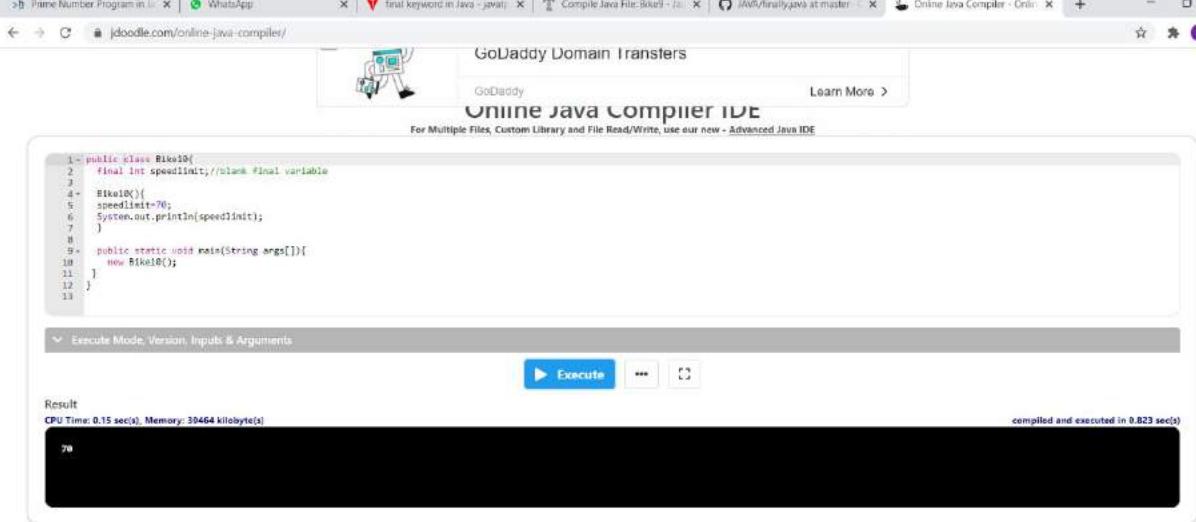
Result

CPU Time: 0.32 sec(s). Memory: 37232 kilobyte(s) compiled and executed in 1.073 sec(s)

Enter the sides of the Triangle:
The area of Triangle is: 6.0
The circumference of the Triangle is: 12.0

Microsoft 365 Access your world LEARN MORE → Activate Windows Go to Settings to activate Windows. 5:00 PM 11/11/2020

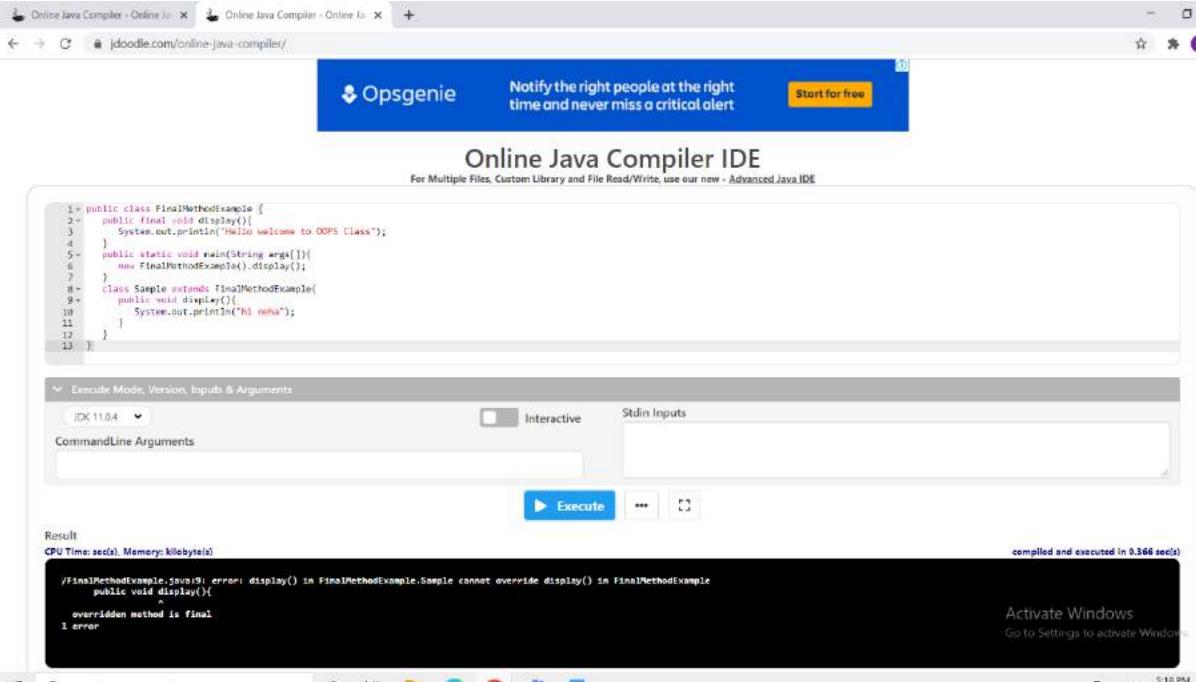
3) Write a program to show the use of final variable and final Method



```
1- public class Bike10{
2-     final int speedInit;/blank final variable
3-
4-     Bike10(){
5-         speedInit=70;
6-         System.out.println(speedInit);
7-     }
8-
9-     public static void main(String args[]){
10-         new Bike10();
11-     }
12- }
```

Result
CPU Time: 0.15 sec(s), Memory: 30464 kilobyte(s)

compiled and executed in 0.823 sec(s)



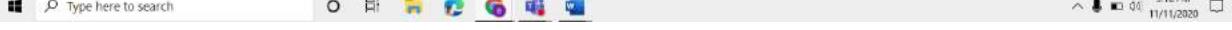
```
1- public class FinalMethodExample {
2-     public final void display(){
3-         System.out.println("Hello welcome to OOPS Class");
4-     }
5-     public static void main(String args[]){
6-         new FinalMethodExample().display();
7-     }
8-     class Sample extends FinalMethodExample{
9-         public void display(){
10-             System.out.println("Hi reha");
11-         }
12-     }
13- }
```

Result
CPU Time: sec(s), Memory: 0 kilobyte(s)

/FinalMethodExample.java:9: error: display() in FinalMethodExample.Sample cannot override display() in FinalMethodExample
public void display()
 ^
 overridden method is final
1 error

compiled and executed in 0.366 sec(s)

Activate Windows
Go to Settings to activate Windows.



4) Write a program to find out all vowels in one constructor if input is string and find the sum of all digits in another constructor if the input is number.

The screenshot shows a web-based Java compiler interface. The code area contains a Java program to calculate the sum of digits in a string and count vowels. The result section shows the output of the program, which includes the input string "abcd", the number of vowels (0), and the sum of digits (4). The compiler also displays performance metrics like CPU time and memory usage.

```
1 import java.util.Scanner;
2 public class Main {
3     int sum = 0;
4     char ch;
5     int n;
6     for(int i=0; i<number.length(); i++) {
7         ch = number.charAt(i);
8         if(Character.isDigit(ch)) {
9             n = Character.getNumericValue(ch);
10            sum += n;
11        }
12    }
13    return sum;
14 } public static void main(String[] args) {
15     Scanner in = new Scanner(System.in);
16     System.out.print("Input the string: ");
17     String str = in.nextLine();
18     System.out.print("Number of Vowels in the string: " + count_Vowels(str)+"\n");
19     int count = 0;
20     Scanner scan = new Scanner(System.in);
21     //str = scan.nextLine();
22     sum = sumOfDigits(str);
23     System.out.println("The sum of "
24     + " digits in the string "+str
25     + " is "+sum);
26     scan.close();
27 }
28 public static int count_Vowels(String str)
29 {
30     int count = 0;
31     for (int i = 0; i < str.length(); i++)
32     {
33         if ((str.charAt(i) == 'a' || str.charAt(i) == 'e' || str.charAt(i) == 'i'
34             || str.charAt(i) == 'o' || str.charAt(i) == 'u'))
35         {
36             count++;
37         }
38     }
39     return count;
40 }
```

Execute Mode, Version, Inputs & Arguments

Execute

Result

CPU Time: 0.30 sec(s), Memory: 37832 Kilobyte(s)

Input the string:abcd
Number of Vowels in the string: 0
The sum of digits in the string 4 = 4

compiled and executed in 0.937 sec(s)
Activate Windows
Go to Settings to activate Windows.

Object oriented programming lab3

- 1) Write classes with class variables, methods and use inheritance to implement the below situations. Make objects to connect classes. Write print functions in all methods to show the sequence the method calling.

```
import java.util.Scanner;
class Vitap
{
    void getaadmission(){
        System.out.println("welcome to VIT amaravati");
        System.out.println("you have to options to join either mtech or
btech");

    }
    class courses extends Vitap{
        void btech()
        {
            System.out.println("The courses available in B.tech are: \n1.Electronics
engineering\n2.Computer science engineering.\n3.Mechanical engineering.");
            System.out.println("The B.Tech is an eight semester program");
        }

        void mtech()
        {
            System.out.println("The courses available in B.tech are:
\n1.Electronics engineering\n2.Computer science engineering.\n3.Mechanical
engineering.");
            System.out.println("The B.Tech is a four semester program");
        }
    }
    class universityfaculties extends courses
    {

        void universityfaculty()
```

```
{  
    System.out.println("there are Three grades of faculties in  
vitap:\n1.Senior grade professors\n2.assistant professors\n3.research  
scholars");  
  
}  
void stafflevels()  
{  
    System.out.println("The staff grades follows as A>B>C");  
    System.out.println("Where A is senior grade professosr B is  
assistant level professors C is research scholars");  
}  
}  
public class Main{  
    public static void main(String[] args)  
    {  
        Scanner sc=new Scanner(System.in);  
  
        courses student1=new courses();  
        student1.getadmission();  
        System.out.println("enter the program you want to  
join\n1.Btech\n2.Mtech");  
        int n=sc.nextInt();  
  
        if(n==1){  
            student1.btech();  
        }  
        else if(n==2){  
            student1.mtech();  
        }  
        universtiyfaculties student_1=new universtiyfaculties();  
        student_1.universtiyfaculty();  
        student_1.stafflevels();  
    }  
}
```

```
C:\java>java Main
welcome to VIT amaravati
you have to options to join either mtech or btech
enter the program you want to join
1.Btech
2.Mtech
1
The courses available in B.tech are:
1.Electronics engineering
2.Computer science engineering.
3.Mechanical engineering.
The B.Tech is an eight semester program
there are Three grades of faculties in vitap:
1.Senior grade professors
2.assistant professors
3.research scholars
The staff grades follows as A>B>C
Where A is senior grade professors B is assistant level professors C is research scholars
```

- 2) . Vehicles may be two wheeler, three wheeler or more than three wheelers. They have different way to drive, different purpose, different mechanical structure, different sizes etc

Code:

```
package vehicle;
class Vehicle
{
    public static void main(String arg[])
    {
        Twowheeler t1;
        Threewheeler th1;
        Fourwheeler f1;
        t1=new Twowheeler("TN74 12345", 1,2);
        th1=new Threewheeler("TN74 54321", 4,3);
        f1=new Fourwheeler("TN34 45677",5,4);
        t1.display();
        th1.display();
        f1.display();
    }
}
```

```
class Vehicledemo
{
    String regno;
    int model;
    Vehicledemo(String r, int m)
    {
        regno=r;
        model=m;
    }
    void display()
    {
        System.out.println("Registration no: "+regno);
        System.out.println("Model no: "+model);
    }
}

class Twowheeler extends Vehicledemo
{
    int noofwheel;
    Twowheeler(String r,int m,int n)
    {
        super(r,m);
        noofwheel=n;
    }
    void display()
    {
        System.out.println("Two wheeler tvs");
        super.display();
        System.out.println("No. of wheel : " +noofwheel);
    }
}

class Threewheeler extends Vehicledemo
{
    int noofleaf;
```

```
Threewheeler(String r,int m,int n)
{
    super(r,m);
    noofleaf=n;
}
void display()
{
    System.out.println("Three wheeler auto");
    super.display();
    System.out.println("No. of leaf:" +noofleaf);
}
}

class Fourwheeler extends Vehicledemo
{
    int noofleaf;
    Fourwheeler(String r,int m,int n)
    {
        super(r,m);
        noofleaf=n;
    }
    @Override
    void display()
    {
        System.out.println("Four wheeler car");
        super.display();
        System.out.println("No. of leaf:" +noofleaf);
    }
}
```

```
C:\java>javac notepad3.java
```

```
C:\java>java Vehicle
Two wheeler tvs
Registration no: TN74 12345
Model no: 1
No. of wheel : 2
Three wheeler auto
Registration no: TN74 54321
Model no: 4
No. of leaf:3
Four wheeler car
Registration no: TN34 45677
Model no: 5
No. of leaf:4
```

Object oriented programming lab 4

Slot L5

Prof. Dr. Sandipan Maiti

1) Write classes with class variables, methods and use overloading, overriding to implement the below situations. Make objects to connect classes. Write print functions in all methods to show the sequence the method calling.

1. A person is doing business of raw woods, his education is very less so he can do only addition of profit to the original cost of raw wood. Now his son is doing the same business but he is much educated. He sales the raw wood according to customer required sizes and able to calculate much variant price. He also started discount mechanism in this business. He also started making furniture and sale them in high profit with discounted scheme.

Code:

```
package javaapplication4;  
import java.util.Scanner;  
class Wood{  
    Wood(){  
  
    }  
    Wood(Wood p, double cp){  
        length=p.length;  
        width=p.width;  
        cpsa=cp;  
    }  
    int width;
```

```
int length;
double cpsa;
int getLength() {
    return length;
}
int getWidth() {
    return width;
}
double getcost(double cost,double area) {
    return cost*area;
}
int getArea(int length,int width) {
    return length*width;
}
}
class Father{
    double cost;
    double profit;
    double total;
    void setTotal(double cost,double profit) {
        total=cost+profit;
    }
}
```

```
double getTotal() {  
    return total;  
}  
}  
  
class Son extends Father{  
    int p;//disount %  
  
    void setOrder(Wood obj, double cpsa, int p) {  
        Wood w = new Wood(obj,cpsa);  
        cost=w.getcost(cpsa, w.getArea(w.length,w.width));  
        setTotal(cost,p);  
    }  
    void setTotal(double cost, int p) {  
        total = cost-cost*p/100;  
    }  
}  
  
public class JavaApplication4 {  
  
    public static void main(String[] args) {  
        // TODO Auto-generated method stub  
        Scanner in = new Scanner(System.in);
```

```
System.out.print("Enter the length of the wood req :");
Wood w = new Wood();
Son s = new Son();
w.length = in.nextInt();
System.out.print("Enter the length of the wood req :");
w.width=in.nextInt();
System.out.print("Is discount applicable if yes ?% :");
s.p=in.nextInt();
System.out.println("Your order is being placed....");
s.setOrder(w, 5.6, s.p);
System.out.println("Total price :" +s.cost);
s.setTotal(s.cost, s.p);
System.out.println("Total price to be paid : " +s.total);

}

Output:
```



```
Notifications | Output - JavaApplication4 (ru... ×
ant -f C:\\Users\\DELL\\Downloads\\JavaApplication4 -Dnb.internal.action.name=run run
init:
Deleting: C:\\Users\\DELL\\Downloads\\JavaApplication4\\build\\built-jar.properties
deps-jar:
Updating property file: C:\\Users\\DELL\\Downloads\\JavaApplication4\\build\\built-jar.properties
compile:
run:
Enter the size :60
Enter the size :49
Is discount applicable if yes ?% :25
order confirmed
cost price :16464.0
the cost of the total price is : 12348.0
BUILD SUCCESSFUL (total time: 33 seconds)
```

2) Flight is one way to transportation has some difference from Bus, small cars, trucks and ships. These all transporters have manufacturing company. There are some agency who will purchase these transporters and use them according to their wish to transport goods or persons may be in / out the country / states etc. This process need money to run it successfully. The cost of transportation depends of the various points.

Code:

```
package javaapplication1;

import java.util.*;

class Vehicle{

    int fare;
    double dist;

    Vehicle(int fare, int dist){

        this.fare=fare;
```

```
        this.dist=dist;  
    }  
  
    double getBill(){  
        return fare*dist;  
    }  
}  
  
class Cargo extends Vehicle{  
    int items;  
    Cargo(int fare, int distance,int items){  
        super(fare,distance);  
        this.items=items;  
    }  
    @Override  
    double getBill() {  
        return super.getBill()*items;  
    }  
}  
  
class Car extends Vehicle{  
    int passengers=1;  
    Car(int fare,int dist, int passengers){  
        super(fare,dist);  
        this.passengers=passengers;
```

```
    }

    @Override

    double getBill() {

        return passengers*super.getBill();

    }

}

class Aeroplane extends Cargo{

    int passengers;

    Aeroplane(int fare,int dist,int items,int passengers){

        super(fare,dist,items);

        this.passengers=passengers;

    }

    @Override

    double getBill() {

        return ((super.getBill()/2)+passengers*fare*dist);

    }

}

public class JavaApplication1 {

    public static void main(String[] args) {

        Scanner in = new Scanner(System.in);

    }

}
```

```
int passengers,fare,dist,goods=0;
System.out.print("Journey type (t/g):");
String s = in.next();
if(s.equalsIgnoreCase("t")) {
    System.out.print("How many members :");
    passengers=in.nextInt();
    System.out.print("Estimated distance to destination ");
    dist=in.nextInt();
    if(passengers<=4) {
        Car c= new Car(2000,dist,passengers);
        System.out.print("Total bill is :" +c.getBill());
    }
    else {
        Aeroplane a = new
Aeroplane(20000,dist,0,passengers);
        System.out.print("Total bill is :" +a.getBill());
    }
}
else {
    System.out.print("Enter total number of goods ");
    goods=in.nextInt();
    System.out.print("Estimated distance to destination ");
}
```

```

        dist=in.nextInt();

        System.out.print("Any supporting staff :");

        passengers=in.nextInt();

        Aeroplane a = new
Aeroplane(60000,dist,goods,passengers);

        System.out.print("Total bill is :" +a.getBill());

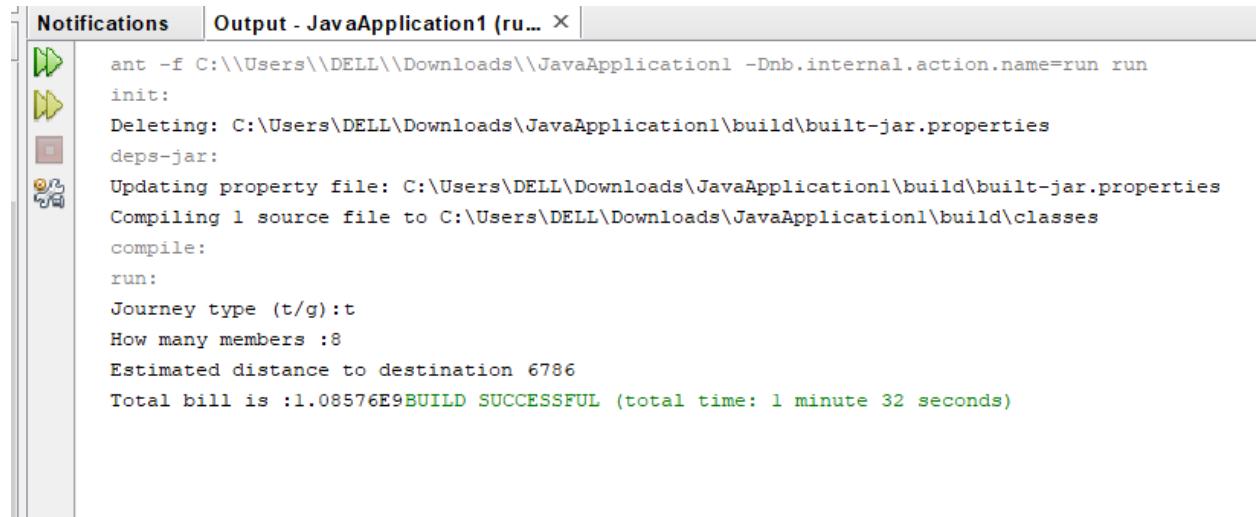
    }

}

}

```

Output:



The screenshot shows the Eclipse IDE interface with the 'Notifications' and 'Output' tabs selected. The 'Output' tab displays the following build log:

```

Notifications | Output - JavaApplication1 (ru... X |
[green arrow] ant -f C:\\\\Users\\\\DELL\\\\Downloads\\\\JavaApplication1 -Dnb.internal.action.name=run run
init:
Deleting: C:\\Users\\DELL\\Downloads\\JavaApplication1\\build\\built-jar.properties
deps-jar:
Updating property file: C:\\Users\\DELL\\Downloads\\JavaApplication1\\build\\built-jar.properties
Compiling 1 source file to C:\\Users\\DELL\\Downloads\\JavaApplication1\\build\\classes
compile:
run:
Journey type (t/g):t
How many members :8
Estimated distance to destination 6786
Total bill is :1.08576E9BUILD SUCCESSFUL (total time: 1 minute 32 seconds)

```

Object oriented programming lab6

Reg No: 19bce7097

Solt L5

- 1) Write programs to show the use of each access specifier as shown in below table.

Code :

```
public class Trail {  
    //The specifiers are applied to variables  
    // to show the visibility  
    public static String publicvar="This is public variable";  
    protected static String protectedvar="This is protected  
    variable";  
    static String defaultvar="This is default variable";  
    private static String privatevar="This is default variable";  
    public static void main(String[] args) {  
        System.out.println("=====Within same class & same  
        package =====");  
        System.out.println(publicvar+"\n"+privatevar+  
        "\n"+defaultvar+"\n"+privatevar);  
    }  
}  
class Trail2 extends Trail{  
    public static void main(String[] args) {  
        System.out.println("=====Within sub class & same  
        package =====");  
        System.out.println(publicvar+"\n"+protectedvar+  
        "\n"+privatevar);  
    }  
}
```

```
"\n"+defaultvar+"\n"/+privatevar/)  
;//Error inside comments  
}  
}  
  
import LAB6A.Trail;  
class Trail2 extends Trail{  
public static void main(String[] args) {  
System.out.println("=====Within same class & same  
package =====");  
System.out.println(publicvar+"\n"+protectedvar/*+  
"\n"+defaultvar+"\n"+t.privatevar*/);  
//only public protected are  
}  
}  
  
public class Test {  
public static void main(String[] args) {  
System.out.println("=====Within non derived class &  
different package  
=====");  
System.out.println(Trail.publicvar+"\n/*+t.privatevar+  
"\n"+t.defaultvar+"\n"+t.privatevar*/);  
//Only public variable is visible  
}  
}
```

2) 2. Write program to show the exception for the situations given.

- A) Arithmetic exception.
- B) Null pointer exception.
- C) Array index out of bound exception.
- D) Number format exception.
- E) Class not found except

Code:

```
class Test
{
    public static void main(String[] args)
    {
        System.out.println("1");
        System.out.println("2");
        System.out.println("3");
        System.out.println("4");
        System.out.println(100/0); // arithmetic exception
        // these will not print and flow of print has disturbed
        System.out.println("5");
        System.out.println("6");
```

```
        System.out.println("7");
        System.out.println("8");
        System.out.println("9");
    }
}
```

Code2:

```
class Test1
{
    public static void main(String args[])
    {
        try{
            String str=null;
            System.out.println (str.length());
        }
        catch(NullPointerException e){
            System.out.println("NullPointerException..");
        }
    }
}
```

Code 3:

```
class Test2
{
    public static void main(String args[])
    {
        try{
            int a[] = new int[4];
            //Array has only 4 elements
            a[5] = 12;
        }
        catch(ArrayIndexOutOfBoundsException e){
            System.out.println ("ArrayIndexOutOfBoundsException");
        }
    }
}
```

Code 4:

```
class Test3
{
    public static void main(String args[])
    {
        try{
```

```
int num=Integer.parseInt ("neha") ;  
System.out.println(num);  
}catch(NumberFormatException e){  
    System.out.println("Number format exception  
occurred");  
}  
}  
}
```

Code 5:

```
class Test {  
  
    public static void main(String[] args) {  
  
        try {
```

```
// the forname method in class class looks for  
the mentioned class
```

```
Class.forName("The Class do not Exist");
```

```
}
```

```
catch (ClassNotFoundException e)
```

```
{
```

```
e.printStackTrace();
```

```
}
```

```
}
```

```
}
```

Output

```
C:\java>javac arthimatic.java

C:\java>java Test
1
2
3
4
Exception in thread "main" java.lang.ArithmetricException: / by zero
        at Test.main(arthimatic.java:11)

C:\java>javac nullpoint.java

C:\java>java Test1
NullPointerException..

C:\java>javac arrayindex.java

C:\java>java Test2
ArrayIndexOutOfBoundsException

C:\java>javac numberformat.java

C:\java>java Test3
Number format exception occurred
```

```
C:\java>javac classnotfound.java

C:\java>java Test4
Error: Could not find or load main class Test4
Caused by: java.lang.ClassNotFoundException: Test4

C:\java>_
```

3) Write a program to sort all strings given by user using an in build generic data structure in java. This program will sort the strings first based on length of it. Sort based on characters from left, if length is equal.

Code :

```
package trail;

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

        Scanner ob=new Scanner(System.in);
        System.out.println("enter the sentence");
        String s=ob.nextLine();
        ArrayList<String> A = new ArrayList<String>();

        for(String word : s.split(" ")) {
            A.add(word);
        }
        System.out.println("String List:"+A);
        Collections.sort(A);
        System.out.println("\nSorted String List "+A);

        System.out.println("\nThe lengths of the Strings: " + lengths(A));
        System.out.println("\nEnter the value to search");
        String v=ob.nextLine();
        search(v,A);
    }

    public static <T> void search(T v,ArrayList<String> A) {
```

```
T c=v;

String c1=c.toString();

int j=0;

for (String i : A) {

    if(i.contains(c1)) {

        j++;

    }

}

if(j>0) {

    System.out.println("\nvalue "+c+" is present in the array list");

}

else {

    System.out.println("\nvalue "+c+" is not present in the array

list");

}

}

public static ArrayList<Integer> lengths(ArrayList<String> A) {

ArrayList<Integer> lengthList = new ArrayList<Integer>();

for (String s : A)

    lengthList.add(s.length());

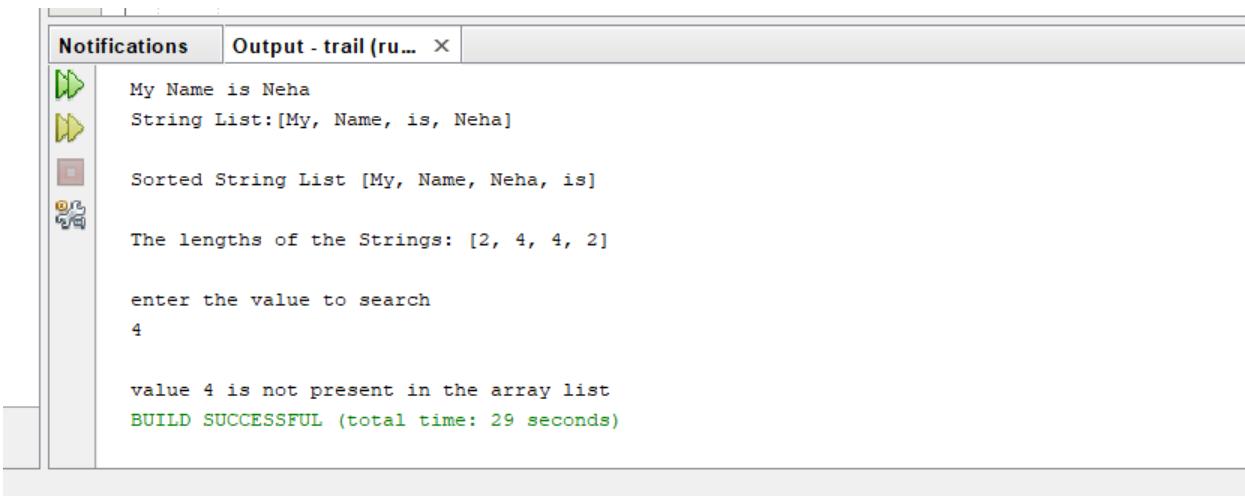


return lengthList;

}
```

}

Output:



The screenshot shows an IDE's output window titled "Output - trail (ru... X)". The window displays the following text output from a Java application:

```
My Name is Neha
String List:[My, Name, is, Neha]

Sorted String List [My, Name, Neha, is]
The lengths of the Strings: [2, 4, 4, 2]

enter the value to search
4

value 4 is not present in the array list
BUILD SUCCESSFUL (total time: 29 seconds)
```

1)

```
1 import java.util.*;
2 import java.io.*;
3 public class Main {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6         try {
7             System.out.println("Trying to read file not present in the directory");
8             File myObj = new File("filename.txt");
9             Scanner myReader = new Scanner(myObj);
10            while (myReader.hasNextLine()) {
11                String data = myReader.nextLine();
12                System.out.println(data);
13            }
14            myReader.close();
15        } catch (FileNotFoundException e) {
16            System.out.println("An error occurred: " +e);
17        }
18        try{
19            System.out.print("Enter divisor: ");
20            int divisor = sc.nextInt();
21            int quotient = 100/divisor;
22            System.out.println(quotient);
23        }catch(Exception e){
24            System.out.println(e);
25        }
26
27        String ptr = null;
28        try{
29            System.out.println("\nSetting string's value as null and comparing...");
30            if (ptr.equals("Hello world"))
31                System.out.print("Same");
32            else
33                System.out.print("Not Same");
34        }catch(NullPointerException e) {
35            System.out.println(e);
36        }
37
38        int[] a =new int[5];
39        try{
40            System.out.println("\nSetting value to index 6 in array of size 5...");
41            a[6]=69420;
```

input

```
Number format exception as String contains decimal but is parsed into integer..
java.lang.NumberFormatException: For input string: "123.33"

class loader is not able to find the class
java.lang.ClassNotFoundException: myPackage.example.Sample
```

2)

```
 1 import java.util.*;
2 public class Main{
3     public static void main(String[] args){
4         Scanner sc=new Scanner(System.in);
5         System.out.println("enter the phone number");
6         long number=sc.nextLong();
7         int length = String.valueOf(number).length();
8         System.out.println(length);
9         int firstdigit=FirstDigit(number);
10        int twodigits=FirsttwoDigits(number);
11        try{
12            if(length<10){
13                throw new LessDigitsException("required atleast 10 digits");
14            }
15        }catch(LessDigitsException exp){
16            System.out.println("exception caught"+exp);
17        }
18        try{
19            if(length==11 && firstdigit!=0){
20                throw new FirstdigitNonZeroException("if the number is 11 digits first digit must be 0");
21            }
22        }catch(FirstdigitNonZeroException exp){
23            System.out.println("exception caught"+exp);
24        }
25        try{
26            if(length==12 && twodigits!=91){
27                throw new FirsttwoDigitsException("if the entered number is 12 digits first two digits must be 91");
28            }
29        }catch(FirsttwoDigitsException exp){
30            System.out.println("exception caught"+exp);
31        }
32        try{
33            if(length>=13){
34                throw new ToolongException("the number should be less than 13 digits");
35            }
36        }catch(ToolongException exp){
37            System.out.println("exception caught"+exp);
38        }
39    }
40 }
41
```

enter the phone number
8374646757
10
...Program finished with exit code 0

3)

```
1 import java.io.File;
2 import java.util.*;
3 public class Main{
4
5 static void checkForValidCharacter(String m) throws Exception{
6     for(int i=0;i<m.length();i++) {
7         if (!(((int)m.charAt(i)>=48)&&((int)m.charAt(i)<=57))){
8             throw new Exception();
9         }
10    }
11 }
12 static void length(String m) throws Exception {
13     if(!(m.length()>=10 && m.length()<=12))
14         throw new Exception();
15 }
16 static String trimAccording(String m) throws Exception {
17     if(m.length()==10)
18         return m;
19     else if (m.length()==11) {
20         if(m.charAt(0)!='0')
21             throw new Exception();
22         else return m.substring(1,10);
23     }else {
24         if(!m.substring(0,2).equals("91"))
25             throw new Exception();
26         else return m.substring(2,12);
27     }
28 }
29
30 }
31 public static void main(String[] args) {
32     // TODO Auto-generated method stub
33     Scanner in = new Scanner(System.in);
34
35     while(true) {
36         try {
37             System.out.print("Enter the Mobile ");
38             String m =in.nextLine();
39             checkForValidCharacter(m);
40             length(m);
41             m=trimAccording(m);
42             System.out.println(" Valid number");
43             break;
44         }
45     }
46 }
```

Enter the Mobile 8375656757

input

Valid number

Object oriented programming lab 8

SLOT : L5

19BCE7097

CODE:

```
import java.util.*;
import java.io.*;

class WordLength extends Thread{
    File file;
    WordLength(File file){
        setFile(file);
    }
    void setFile(File file) {
        this.file =file;
    }
    public void run() {
        System.out.println("WordLength thread was initiaed");
        Scanner sc;
        String sent;
        String word="";
        try {
            sc = new Scanner(file);
            while (sc.hasNextLine()) {
                sent="";
                sent=sc.nextLine();
                for(int i=0;i<sent.length();i++) {
                    if(sent.charAt(i)==' '| |sent.charAt(i)=='.') {
                        if(word.length()>=3)
```

```

        System.out.print(word+" ");
        word="";
    }
    else {
        word+=sent.charAt(i);
    }
}

}

} catch (FileNotFoundException e) {
    System.out.println(e);
}

}

}

class Vowel extends Thread{
    File file;
    Vowel(File file){
        setFile(file);
    }
    void setFile(File file) {
        this.file = file;
    }
    void checkVowel(String word) {
        int count=0;
        for(int i=0;i<word.length();i++) {

            if(word.charAt(i)=='a'||word.charAt(i)=='e'||word.charAt(i)=='i'||word.charAt(i)=='o'||word.ch
arAt(i)=='u')

```

```

        count++;

    else
if(word.charAt(i)=='A' | |word.charAt(i)=='E' | |word.charAt(i)=='I' | |word.charAt(i)=='O' | |word.charAt(i)='U')

        count++;

    }

if(count>1)

    System.out.println(word +" "+count);

}

public void run() {

    System.out.println("Vowel thread was initiaed");

    Scanner sc;

    String sent;

    String word="";

    try {

        sc = new Scanner(file);

        while (sc.hasNextLine()) {

            sent="";

            sent=sc.nextLine();

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

                if(sent.charAt(i)==' '| |sent.charAt(i)=='.') {

                    word="";

                }

                else {

                    word+=sent.charAt(i);

                }

            }

        }

    }

}

```

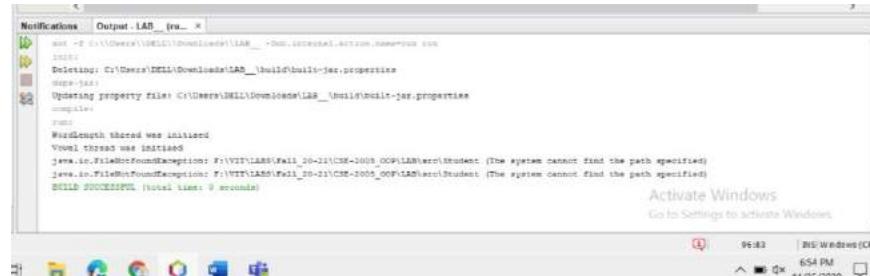
```

        } catch (FileNotFoundException e) {
            System.out.println(e);
        }
    }

    public class LAB__ {

        public static void main(String[] args) {
            File file;
            file = new File("F:\\VIT\\LABS\\Fall_20-21\\CSE-2005_OOP\\LAB\\src\\Student");
            WordLength w = new WordLength(file);
            w.start();
            Vowel v = new Vowel(file);
            v.start();
        }
    }
}

```



CODE2:

```

package lab_8_2;

import java.util.*;
import java.io.File;

class Average extends Thread{
    File file;

```

```

Average(File file){
    setFile(file);
}

void setFile(File file) {
    this.file=file;
}

public void run() {
    System.out.println("Average Thread is initiated");

    Scanner sc;
    String []str;
    double avg1=0,avg2=0;

    try {
        sc=new Scanner(file);
        while (sc.hasNextLine()) {
            str=sc.nextLine().split(" ");
            avg1=(Integer.parseInt(str[0])+Integer.parseInt(str[1]))/2;
            avg2=(Integer.parseInt(str[1])+Integer.parseInt(str[2]))/2;
            System.out.println(avg2-avg1);
        }
    }

    catch(Exception e) {

    }
}

class Prime extends Thread{
    File file;
    Prime(File file){

```

```

        setFile(file);

    }

void setFile(File file) {
    this.file=file;
}

public void run() {
    System.out.println("Prime Thread is initiated");

    Scanner sc;
    String []str;
    boolean b=true;
    try {
        sc=new Scanner(file);
        while (sc.hasNextLine()) {
            str=sc.nextLine().split(" ");
            for(int i=0;i<str.length;i++) {
                int n=Integer.parseInt(str[i]);
                for(int i1=0;i1<n/2;i1++) {
                    if(n%i1==0) {
                        b=false;
                        break;
                    }
                }
                if(b)
                    System.out.println(n+" prime");
                else
                    System.out.println(n+" Not a prime");
            }
        }
    }
}

```

```
        }

    }

}

}

public class LAB_8_2 {

    public static void main(String[] args) {

        File file;

        file = new File("F:\\VIT\\LABS\\Fall_20-21\\CSE-2005_OOP\\LAB\\src\\Student");

        Average a = new Average(file);

        a.start();

        Prime p = new Prime(file);

        p.start();

    }

}
```

OUTPUT:

```
Notifications Output LAB_0_2 [run... X

[INFO] 
[INFO] at org.jenkinsci.plugins.workflow.steps.core.ExecutorStepExecution$1.run(ExecutorStepExecution.java:102)
[INFO] at java.util.concurrent.Executors$RunnableAdapter.call(Executors.java:510)
[INFO] at java.util.concurrent.FutureTask.run(FutureTask.java:266)
[INFO] at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149)
[INFO] at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624)
[INFO] at java.lang.Thread.run(Thread.java:748)
[INFO] 
[INFO] Deleting: C:\Users\DELL\Downloads\LAB_0_2\build\build->jet.properties
[INFO] Deleting: C:\Users\DELL\Downloads\LAB_0_2\build\build->jet.properties
[INFO] Updating property file: C:\Users\DELL\Downloads\LAB_0_2\build\build->jet.properties
[INFO] 
[INFO] BUILD SUCCESSFUL (total time: 0 seconds)
```

Object oriented programming 10

Code :

```
package mythread1;  
import java.util.*;  
import java.io.*;  
class S1 extends Thread{  
    synchronized boolean isPrime(int n) {  
        switch (n) {  
            case 0, 1 -> {  
                return false;  
            }  
            case 2 -> {  
                return true;  
            }  
            default -> {  
                for(int i=3;i<n/2;i++)  
                    if(n%i==0)  
                        return false;  
            }  
        }  
    }  
}
```

```
return true;  
}  
  
@Override  
public void run() {  
int n=10;//temperory  
System.out.println("S1 start's");  
Random r = new Random();  
ArrayList<Integer> a = new ArrayList<>();  
  
//-----  
for(int i=0;i<n;i++) {  
a.add(r.nextInt(1000-10)+10);  
}  
try {  
String f1 = null;  
try (FileWriter A = new FileWriter(f1)) {  
for(int i:a) {  
A.write(i+" ");  
}  
}  
}  
}
```

```
        catch(IOException e) {  
    }  
  
}  
}  
  
class S2 extends S1 {  
    @Override  
    public void run() {  
        int x=10;//temperory  
        System.out.println("S2 start's");  
        Random r = new Random();  
        ArrayList<Integer> a = new ArrayList<>();  
  
        //-----  
        for(int i=0;i<x;i++) {  
            a.add(r.nextInt(2000-50)+50);  
        }  
        try {  
            String f1 = null;  
            try (FileWriter A = new FileWriter(f1)) {
```



```
//-----
for(int i=0;i<4;i++) {
    a.add(r.nextInt(50-1)+1);
}

try {
    try (FileWriter A = new FileWriter(f1)) {
        for(int i:a) {
            A.write(i+" ");
        }
    }
    catch(IOException e) {

    }
}

public class Mythread1 {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        S1 a = new S1();
        S2 b = new S2();
    }
}
```

```

M c = new M();

try {
    a.start();
    a.join();
    b.start();
    b.join();
    c.start();
    c.join();
}

catch(InterruptedException e) {

}

}

```

}

}

}

Output :

```

Notifications Output - Mythread1 [run]
BUILD 1 C:\Users\DELL\Downloads\Mythread1\build\bin\Mythread1.jar-<null>
Deleting: C:\Users\DELL\Downloads\Mythread1\build\bin\Mythread1.jar.properties
delete-<null>
Updating property file: C:\Users\DELL\Downloads\Mythread1\build\bin\Mythread1.jar.properties
Compiling 1 source file to C:\Users\DELL\Downloads\Mythread1\build\classes
compile:
run:
 01 start's
  02 start's
  03 start's
BUILD SUCCESSFUL (total time: 1 second)
  
```

Object oriented programming lab 11

19BCE7097

Slot :L5

Code :

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

class MyFrame
    extends JFrame
    implements ActionListener {

    // Components of the Form
    private Container c;
    private JLabel title;
    private JLabel name;
    private JTextField tname;
    private JLabel mno;
    private JTextField tmno;
```

```
private JLabel gender;  
private JRadioButton male;  
private JRadioButton female;  
private ButtonGroup gengp;  
private JLabel dob;  
private JComboBox date;  
private JComboBox month;  
private JComboBox year;  
private JLabel add;  
private JTextArea tadd;  
private JCheckBox term;  
private JButton sub;  
private JButton reset;  
private JTextArea tout;  
private JLabel res;  
private JTextArea resadd;  
  
private String dates[]  
= { "1", "2", "3", "4", "5",  
  "6", "7", "8", "9", "10",
```

```
"11", "12", "13", "14", "15",
"16", "17", "18", "19", "20",
"21", "22", "23", "24", "25",
"26", "27", "28", "29", "30",
"31" };

private String months[]
= { "Jan", "feb", "Mar", "Apr",
"May", "Jun", "July", "Aug",
"Sep", "Oct", "Nov", "Dec" };

private String years[]
= { "1995", "1996", "1997", "1998",
"1999", "2000", "2001", "2002",
"2003", "2004", "2005", "2006",
"2007", "2008", "2009", "2010",
"2011", "2012", "2013", "2014",
"2015", "2016", "2017", "2018",
"2019" };

// constructor, to initialize the components
// with default values.
```

```
public MyFrame()
{
    setTitle("Registration Form");
    setBounds(300, 90, 900, 600);
    setDefaultCloseOperation(EXIT_ON_CLOSE);
    setResizable(false);

    c = getContentPane();
    c.setLayout(null);

    title = new JLabel("Registration Form");
    title.setFont(new Font("Arial", Font.PLAIN, 30));
    title.setSize(300, 30);
    title.setLocation(300, 30);
    c.add(title);

    name = new JLabel("Name");
    name.setFont(new Font("Arial", Font.PLAIN, 20));
    name.setSize(100, 20);
    name.setLocation(100, 100);
```

```
c.add(name);

tname = new JTextField();
tname.setFont(new Font("Arial", Font.PLAIN, 15));
tname.setSize(190, 20);
tname.setLocation(200, 100);
c.add(tname);

mno = new JLabel("Mobile");
mno.setFont(new Font("Arial", Font.PLAIN, 20));
mno.setSize(100, 20);
mno.setLocation(100, 150);
c.add(mno);

tmno = new JTextField();
tmno.setFont(new Font("Arial", Font.PLAIN, 15));
tmno.setSize(150, 20);
tmno.setLocation(200, 150);
c.add(tmno);
```

```
gender = new JLabel("Gender");
gender.setFont(new Font("Arial", Font.PLAIN, 20));
gender.setSize(100, 20);
gender.setLocation(100, 200);
c.add(gender);
```

```
male = new JRadioButton("Male");
male.setFont(new Font("Arial", Font.PLAIN, 15));
male.setSelected(true);
male.setSize(75, 20);
male.setLocation(200, 200);
c.add(male);
```

```
female = new JRadioButton("Female");
female.setFont(new Font("Arial", Font.PLAIN, 15));
female.setSelected(false);
female.setSize(80, 20);
female.setLocation(275, 200);
c.add(female);
```

```
gengp = new ButtonGroup();
gengp.add(male);
gengp.add(female);

dob = new JLabel("DOB");
dob.setFont(new Font("Arial", Font.PLAIN, 20));
dob.setSize(100, 20);
dob.setLocation(100, 250);
c.add(dob);

date = new JComboBox(dates);
date.setFont(new Font("Arial", Font.PLAIN, 15));
date.setSize(50, 20);
date.setLocation(200, 250);
c.add(date);

month = new JComboBox(months);
month.setFont(new Font("Arial", Font.PLAIN, 15));
month.setSize(60, 20);
month.setLocation(250, 250);
```

```
c.add(month);

year = new JComboBox(years);
year.setFont(new Font("Arial", Font.PLAIN, 15));
year.setSize(60, 20);
year.setLocation(320, 250);
c.add(year);

add = new JLabel("Address");
add.setFont(new Font("Arial", Font.PLAIN, 20));
add.setSize(100, 20);
add.setLocation(100, 300);
c.add(add);

tadd = new JTextArea();
tadd.setFont(new Font("Arial", Font.PLAIN, 15));
tadd.setSize(200, 75);
tadd.setLocation(200, 300);
tadd.setLineWrap(true);
c.add(tadd);
```

```
term = new JCheckBox("Accept Terms And Conditions.");
term.setFont(new Font("Arial", Font.PLAIN, 15));
term.setSize(250, 20);
term.setLocation(150, 400);
c.add(term);

sub = new JButton("Submit");
sub.setFont(new Font("Arial", Font.PLAIN, 15));
sub.setSize(100, 20);
sub.setLocation(150, 450);
sub.addActionListener(this);
c.add(sub);

reset = new JButton("Reset");
reset.setFont(new Font("Arial", Font.PLAIN, 15));
reset.setSize(100, 20);
reset.setLocation(270, 450);
reset.addActionListener(this);
c.add(reset);
```

```
tout = new JTextArea();
tout.setFont(new Font("Arial", Font.PLAIN, 15));
tout.setSize(300, 400);
tout.setLocation(500, 100);
tout.setLineWrap(true);
tout.setEditable(false);
c.add(tout);
```

```
res = new JLabel("");
res.setFont(new Font("Arial", Font.PLAIN, 20));
res.setSize(500, 25);
res.setLocation(100, 500);
c.add(res);
```

```
resadd = new JTextArea();
resadd.setFont(new Font("Arial", Font.PLAIN, 15));
resadd.setSize(200, 75);
resadd.setLocation(580, 175);
resadd.setLineWrap(true);
```

```
c.add(resadd);

setVisible(true);

}

// method actionPerformed()
// to get the action performed
// by the user and act accordingly
public void actionPerformed(ActionEvent e)
{
    if (e.getSource() == sub) {
        if (term.isSelected()) {
            String data1;
            String data
                = "Name : "
                + tname.getText() + "\n"
                + "Mobile : "
                + tmno.getText() + "\n";
            if (male.isSelected())
                data1 = "Gender : Male"
        }
    }
}
```

```
        + "\n";
    else
        data1 = "Gender : Female"
        + "\n";
String data2
= "DOB : "
+ (String)date.getSelectedItem()
+ "/" + (String)month.getSelectedItem()
+ "/" + (String)year.getSelectedItem()
+ "\n";
String data3 = "Address : " + tadd.getText();
tout.setText(data + data1 + data2 + data3);
tout.setEditable(false);
res.setText("Registration Successfully..");
}
else {
    tout.setText("");
    resadd.setText("");
    res.setText("Please accept the"
```

```
        + " terms & conditions..");

    }

}

else if (e.getSource() == reset) {

    String def = "";

    tname.setText(def);

    tadd.setText(def);

    tmno.setText(def);

    res.setText(def);

    tout.setText(def);

    term.setSelected(false);

    date.setSelectedIndex(0);

    month.setSelectedIndex(0);

    year.setSelectedIndex(0);

    resadd.setText(def);

}

}

}
```

```

// Driver Code

class Registration {

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

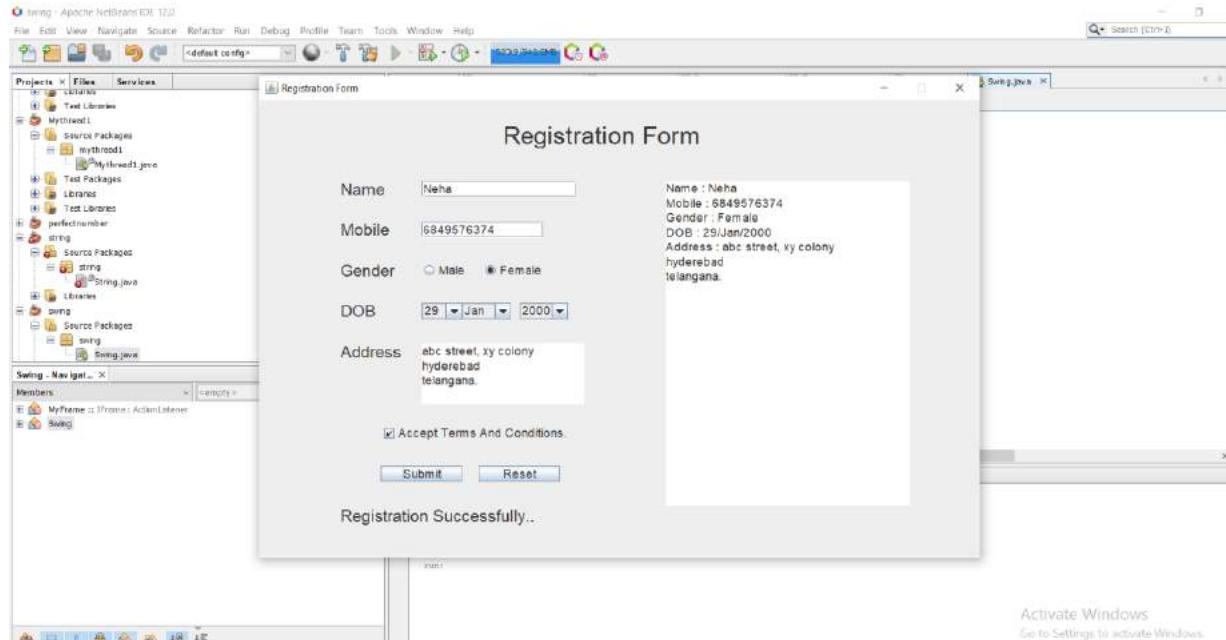
    {
        MyFrame f = new MyFrame();

    }

}

```

Output :



```

1 import java.util.*;
2 public class Holiday
3 {
4     private String name;
5     private int day;
6     private String month;
7
8     Holiday(String n, int d, String m)
9     {
10         name = n;
11         day = d;
12         month = m;
13     }
14     static boolean inSameMonth(Holiday h, Holiday h1)
15     {
16         if (h.month == h1.month)
17         {
18             return true;
19         }
20         else
21         {
22             return false;
23         }
24     }
25     static double avgDate(Holiday[] h)
26     {
27         int sum = 0;
28         for (int i = 0; i < h.length; i++)
29         {
30             sum = sum + h[i].day;
31         }
32         int avg;
33         avg = (sum / (h.length));
34         return (double) avg;
35     }
36     public static void main(String[] args)
37     {
38
39         Scanner sc = new Scanner(System.in);
40         Holiday obj = new Holiday("Independence day", 15, "may");
41         Holiday obj1 = new Holiday("Diwali", 14, "may");
42         System.out.println("Checking Test cases");
43         System.out.println(inSameMonth(obj, obj1));
44         System.out.println("Enter the size of the array");
45         int n = sc.nextInt();
46         Holiday h[] = new Holiday[n];
47         for (int i = 0; i < n; i++)
48         {
49             System.out.println("Enter the Holiday date " + (i + 1));
50             int day = sc.nextInt();
51             h[i] = new Holiday("Name", day, "Month");
52         }
53         System.out.println("The average of Holiday dates is " + avgDate(h));
54         System.out.println();
55         System.out.println("All Test cases passed");
56     }
57 }
58
59

```

Lab assignment-1

Name:B.Naveen sai

Regno:19BCD7015

Slot:L5

Output

```
Command Prompt
Microsoft Windows [Version 10.0.18363.1139]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\91817>cd Desktop

C:\Users\91817\Desktop>cd javap

C:\Users\91817\Desktop\javap>javac Holiday.java

C:\Users\91817\Desktop\javap>java Holiday
Checking Test cases
true
Enter the size of the array
3
Enter the Holiday date 1
10
Enter the Holiday date 2
20
Enter the Holiday date 3
30
The average of Holiday dates is 20.0

All Test cases passed

C:\Users\91817\Desktop\javap>
```

The screenshot shows a dual-pane interface of Notepad++. On the left, the code for `Holiday.java` is displayed:

```
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
```

On the right, a Command Prompt window shows the execution of the Java program:

```
Microsoft Windows [Version 10.0.18363.1139]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\91817>cd Desktop

C:\Users\91817\Desktop>cd javap

C:\Users\91817\Desktop\javap>javac Holiday.java

C:\Users\91817\Desktop\javap>java Holiday
Checking Test cases
true
Enter the size of the array
3
Enter the Holiday date 1
10
Enter the Holiday date 2
20
Enter the Holiday date 3
30
The average of Holiday dates is 20.0

All Test cases passed

C:\Users\91817\Desktop\javap>
```

CSE2005 Lab Assignment-2

Name:B.Naveen Sai

Reg no:19BCD7015

Slot:L5

```
class Author
{
    String FName;
    String LName;
    public Author(String FName,String LName)
    {
        FName=this.FName;
        LName=this.LName;
    }
    public void setFName(String FirstName)
    {
        this.FName=FirstName;
    }
    public void setLName(String LastName)
    {
        this.LName=LastName;
    }
    public String getFName()
    {
        return FName;
    }
    public String getLName()
```

```
{  
    return FName;  
}  
  
public String toString()  
{  
    String s;  
    s="Author name is : "+FName;  
    return s;  
}  
}  
  
class Book  
{  
    String title;  
    String author;  
    double price;  
  
    public Book(String title,String author,double price)  
    {  
        this.title=title;  
        this.author=author;  
        this.price=price;  
    }  
  
    public void setTitle(String title)  
    {  
        this.title=title;  
    }  
  
    public void setAuthor(String author)  
    {
```

```
this.author=author;
}

public void setPrice(double price)
{
    this.price=price;
}

public String getTitle()
{
    return title;
}

public String getAuthor()
{
    return author;
}

public String toString()
{
    String s;
    s=" Book Details : "+title+"\n Author Name is :" +author+"\n Price is:"+price;
    return s;
}

class BookGroup{
    public static void main(String[] args)
    {
        Book obj=new Book("Object Oriented Programming Systems","HerbertSchildert ",1000);
        System.out.println(obj);
    }
}
```

OUTPUT

```
Command Prompt
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\91817>cd Desktop

C:\Users\91817\Desktop>cd javap

C:\Users\91817\Desktop\javap>javac 19BCD7015_Lab2.java

C:\Users\91817\Desktop\javap>java BookGroup
Book Details : Object Oriented Programming Systems
Author Name is :HerbertSchildert
Price is:1000.0

C:\Users\91817\Desktop\javap>
```

The screenshot shows an IDE interface with a code editor and a terminal window. The code editor displays Java code for a `BookGroup` class. The terminal window shows the execution of the code, including the compilation of `19BCD7015_Lab2.java` and the execution of `java BookGroup`, which outputs the book's details.

```
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
19BCD7015_Lab2.java
54
55     public String getTitle()
56     {
57         return title;
58     }
59     public String getAuthor()
60     {
61         return author;
62     }
63     public String toString()
64     {
65         String s;
66         s=" Book Details : "+titl
67         return s;
68     }
69 }
70 class BookGroup{
71     public static void main(
72     {
73         Book obj=new Book("Object
74         System.out.println(obj);
```

```
Microsoft Windows [Version 10.0.18363.1139]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\91817>cd Desktop
C:\Users\91817\Desktop>cd javap
C:\Users\91817\Desktop\javap>javac 19BCD7015_Lab2.java
C:\Users\91817\Desktop\javap>java BookGroup
Book Details : Object Oriented Programming Systems
Author Name is :HerbertSchildert
Price is:1000.0
C:\Users\91817\Desktop\javap>
```

CSE2005 Lab Assignment-3

Name:B.Naveen Sai

Reg no:19BCD7015

Slot:L5

```
import java.util.*;
class Sem1
{
    int oops,maths,english,ece;
    double avg;
    Sem1(){
        Scanner in=new Scanner(System.in);
        System.out.println("Enter the marks of oops subject :");
        this.oops = in.nextInt();
        System.out.println("Enter the marks of maths subject :");
        this.maths = in.nextInt();
        System.out.println("Enter the marks of english subject :");
        this.english = in.nextInt();
        System.out.println("Enter the marks of ece subject :");
        this.ece = in.nextInt();
        avg=(oops+maths+english+ece)/(double)4;
    }
}
class Sem2 extends Sem1{
    int oops1,maths1,english1,ece1;
    double avg1;
```

```
    Sem2(){  
        Scanner in=new Scanner(System.in);  
        System.out.println("Enter the marks of oops1 subject :");  
        this.oops1 = in.nextInt();  
        System.out.println("Enter the marks of maths1 subject :");  
        this.maths1 = in.nextInt();  
        System.out.println("Enter the marks of english1 subject :");  
        this.english1 = in.nextInt();  
  
        System.out.println("Enter the marks of ece1 subject :");  
        this.ece1 = in.nextInt();  
        avg1=(oops1+maths1+english1+ece1)/(double)4;  
    }  
}  
  
class Sem3 extends Sem2  
{  
    int oops2,maths2,english2,ece2;  
    double avg2;  
    Sem3(){  
        Scanner in=new Scanner(System.in);  
        System.out.println("Enter the marks of oops2 subject :");  
        this.oops2 = in.nextInt();  
        System.out.println("Enter the marks of maths2 subject :");  
        this.maths2 = in.nextInt();  
        System.out.println("Enter the marks of english2 subject :");  
        this.english2 = in.nextInt();  
        System.out.println("Enter the marks of ece2 subject :");  
    }  
}
```

```
this.ece2 = in.nextInt();
avg2=(oops2+maths2+english2+ece2)/(double)4;
}
}

class Sem4 extends Sem3{
int oops3,maths3,english3,ece3;
double avg3,totalavg;
public Sem4(){
Scanner in=new Scanner(System.in);
System.out.println("Enter the marks of oops3 subject :");
this.oops3 = in.nextInt();
System.out.println("Enter the marks of maths3 subject :");
this.maths3 = in.nextInt();
System.out.println("Enter the marks of english3 subject :");
this.english3 = in.nextInt();
System.out.println("Enter the marks of ece3 subject :");
this.ece3 = in.nextInt();
avg3=(oops3+maths3+english3+ece3)/(double)4;
totalavg=(avg+avg1+avg2+avg3)/(double)4;
}
public static void main(String[] args) {
Sem4 d = new Sem4();
System.out.println("Average of Sem1 : "+d.avg);
System.out.println("Average of Sem2 : "+d.avg1);
System.out.println("Average of Sem3 : "+d.avg2);
System.out.println("Average of Sem4 : "+d.avg3);
System.out.println("Total Average all Semesters : "+d.totalavg);
```

```
}
```

```
}
```

OUTPUT

```
C:\Select Command Prompt
C:\Users\91817\Desktop\javap>javac pk3.java
C:\Users\91817\Desktop\javap>java Sem4
Enter the marks of oops subject :
45
Enter the marks of maths subject :
46
Enter the marks of english subject :
47
Enter the marks of ece subject :
48
Enter the marks of oops1 subject :
49
Enter the marks of maths1 subject :
50
Enter the marks of english1 subject :
49
Enter the marks of ece1 subject :
48
Enter the marks of oops2 subject :
47
Enter the marks of maths2 subject :
45
Enter the marks of english2 subject :
46
Enter the marks of ece2 subject :
43
Enter the marks of oops3 subject :
42
Enter the marks of maths3 subject :
41
Enter the marks of english3 subject :
40
Enter the marks of ece3 subject :
48
Average of Sem1 : 46.5
Average of Sem2 : 49.0
Average of Sem3 : 45.25
Average of Sem4 : 42.75
Total Average all Semesters : 45.875
```

CSE2005 LAB-4

Name:B.Naveen sai

Regno:19BCD7015

Slot:L5

```
import java.util.*;  
abstract class Shape  
{  
    String shapeName;  
    public abstract double area();  
    public abstract String toString();  
}  
class Sphere extends Shape  
{  
    double radius;  
    double area;  
    Sphere(double r)  
    {  
        radius=r;  
    }  
    public double area()  
    {  
        return 4*3.14*radius*radius;  
    }  
    public String toString()  
    {  
        return "Radius is : "+radius;
```

```
}

}

class Rectangle extends Shape {

    double length;
    double width;

    Rectangle(double l, double w)

    {
        length= l;
        width= w;
    }

    public double area()

    {
        return length * width;
    }

    public String toString()

    {
        return "Length is : "+length +" Width is : "+width;
    }
}

class Cylinder extends Shape

{
    double height;
    double radius;

    Cylinder(double h, double r)

    {
        radius = r;
        height = h;
    }
}
```

```
public double area()
{
    return 3.14*radius*radius*height;
}

public String toString()
{
    return "Radius is : "+radius+ " height is : "+height;
}

}

class Paint
{
    double coverage;

    Paint(double c)
    {
        coverage = c;
    }

    public String toString()
    {
        String c="";
        c+="coverage";
        return c;
    }
}

public double amount(Shape s)
{
    System.out.println ("Computing amount for " + s);
    return 0;
}
```

```
public class PaintThings {  
  
    public static void main(String[] args)  
    {  
        Scanner sc=new Scanner(System.in);  
        System.out.println("Enter the coverage value");  
        final double COVERAGE = sc.nextInt();  
        Paint paint = new Paint(COVERAGE);  
        Rectangle deck=new Rectangle(20,35);  
        Sphere bigBall=new Sphere(15);  
        Cylinder tank=new Cylinder(30, 10);;  
  
        double deckAmount, ballAmount, tankAmount;  
        ballAmount = bigBall.area() / COVERAGE;  
        deckAmount = deck.area() / COVERAGE;  
        tankAmount = tank.area() / COVERAGE;  
        paint.amount(bigBall);  
        paint.amount(deck);  
        paint.amount(tank);  
        System.out.println("The amount of paint need for Sphere is : "+ballAmount);  
        System.out.println("The amount of paint need for Rectangle is : "+deckAmount);  
        System.out.println("The amount of paint need for Cylinder is: "+tankAmount);  
    }  
}
```

OUTPUT

```
cmd Command Prompt

C:\Users\91817\Desktop\javap>javac PaintThings.java

C:\Users\91817\Desktop\javap>java PaintThings
Enter the coverage value
30
Computing amount for Radius is : 15.0
Computing amount for Length is : 20.0 Width is : 35.0
Computing amount for Radius is : 10.0 height is : 30.0
The amount of paint need for Sphere is : 94.2
The amount of paint need for Rectangle is : 23.33333333333332
The amount of paint need for Cylinder is: 314.0

C:\Users\91817\Desktop\javap>
```

CSE2005 LAB-5

Name:B.Naveen sai

Regno:19BCD7015

Slot:L5

```
class Generics<S,A,I>
{
    S obj1;
    A obj2;
    I obj3;
    Generics(S obj1,A obj2,I obj3)
    {
        this.obj1=obj1;
        this.obj2=obj2;
        this.obj3=obj3;
    }
    void print()
    {
        System.out.print(obj1+" ");
        System.out.print(obj2+" ");
        System.out.print(obj3+" ");
        System.out.println();
    }
}
```

```
class GenericMethods

{
    public static void main(String args[])
    {
        Generics<Integer,Integer,String> t1=new
            Generics<Integer,Integer,String>(18,10861,"Virat");
        Generics<String,Integer,Integer> t2=new
            Generics<String,Integer,Integer>("Sachin",10,18426);
        Generics<Integer,String,Integer> t3=new
            Generics<Integer,String,Integer>(7,"Dhoni",10534);
        Generics<Integer,Integer,String> t4=new
            Generics<Integer,Integer,String>(10201,333,"Gayle");
        Generics<String,Integer,Integer> t5=new
            Generics<String,Integer,Integer>("ABD",9577,17);

        t1.print();
        t2.print();
        t3.print();
        t4.print();
        t5.print();
    }
}
```

OUTPUT

```
Command Prompt
Microsoft Windows [Version 10.0.18363.1198]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\91817>cd Desktop

C:\Users\91817\Desktop>cd javap

C:\Users\91817\Desktop\javap>javac Main.java

C:\Users\91817\Desktop\javap>java GenericMethods
18 10861 Virat
Sachin 10 18426
7 Dhoni 10534
10201 333 Gayle
ABD 9577 17

C:\Users\91817\Desktop\javap>
```

CSE2005 Lab assignment-6

Name:B.Naveen sai

Reg no:19BCD7015

Slot:L5

```
import java.util.Scanner;  
class Lab6  
{  
    int children,sweets;  
    Lab6(int c,int s)  
    {  
        children=c;  
        sweets=s;  
    }  
}  
  
class Main  
{  
    public static void main(String args[])  
    {  
        Scanner sc =new Scanner(System.in);  
        System.out.println("Enter the no of children");  
        int c=sc.nextInt();  
        System.out.println("Enter the no of Sweets");  
    }  
}
```

```
int s=sc.nextInt();

Lab6 obj = new Lab6(0,9);

try
{
if(obj.children==obj.sweets)

{
System.out.println("There are equal no of sweets and children");

}

}

catch(Exception e)

{
System.out.println(e);

}

try
{
if(obj.children==0

){

throw new Exception();

}

}

catch (Exception e)

{

System.out.println("We got an exception because zero no of Children");

System.out.println(e);

}
```

```
}
```

```
}
```

```
}
```

OUTPUT

```
Command Prompt
Microsoft Windows [Version 10.0.18363.1198]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\91817>cd Desktop

C:\Users\91817\Desktop>cd javap

C:\Users\91817\Desktop\javap>javac Main.java

C:\Users\91817\Desktop\javap>java Main
Enter the no of children
0
Enter the no of Sweets
35
We got an exception
java.lang.Exception

C:\Users\91817\Desktop\javap>
```

CSE2005 Lab Assignment-7

Name:B.Naveen sai

Reg no:19BCD7015

Slot:L5

```
import java.util.*;  
class Accounts{  
    public static void main(String args[]){  
        Scanner sc=new Scanner(System.in);  
        double total_balance,excess_amount=0;  
        double account_balance=15000;  
        System.out.println("Your account balance is : "+account_balance);  
        System.out.println("Please enter the amount to add in your balance : ");  
        double deposit_balance=sc.nextDouble();  
        try{  
            if(account_balance+deposit_balance>20000){  
                total_balance=account_balance+deposit_balance;  
                excess_amount=total_balance-20000;  
                throw new Exception();  
            }  
            else{  
                System.out.println("Total account balance : "+(account_balance+deposit_balance));  
            }  
        }  
        catch(Exception e){
```

```
        System.out.println("Your account balance is more than 20k now,so creating FD  
of amount : "+excess_amount);  
    }  
}  
}
```

OUTPUT

Test run-1

```
C:\ Command Prompt  
Microsoft Windows [Version 10.0.18363.1198]  
(c) 2019 Microsoft Corporation. All rights reserved.  
  
C:\Users\91817>cd Desktop  
  
C:\Users\91817\Desktop>cd javap  
  
C:\Users\91817\Desktop\javap>javac Main.java  
  
C:\Users\91817\Desktop\javap>java Accounts  
Your account balance is : 15000.0  
Please enter the amount to add in your balance :  
2000  
Total account balance : 17000.0
```

Test run-2

```
C:\ Command Prompt  
C:\Users\91817\Desktop\javap>javac Main.java  
  
C:\Users\91817\Desktop\javap>java Accounts  
Your account balance is : 15000.0  
Please enter the amount to add in your balance :  
6000  
Your account balance is more than 20k now,so creating FD of amount : 1000.0  
  
C:\Users\91817\Desktop\javap>
```

CSE2005 Lab Assignment-8

Name:B.Naveen sai

Reg no:19BCD7015

Slot:L5

```
import java.util.*;  
class NewThread implements Runnable  
{  
    String name;  
    Thread t;  
    public NewThread(String name) {  
        this.name = name;  
        t = new Thread(this,name); System.out.println("New Thread :" + t);  
    }  
    public void run()  
    {  
        try  
        {  
            for (int i=1;i<=10;i++)  
            {  
                System.out.println(this.name + " : " + i);  
                Thread.sleep(500);  
            }  
        }  
    }  
}
```

```
        catch (InterruptedException e) {  
            System.out.println(this.name + " Interrupted!");  
        }  
        System.out.println(this.name+" is Exiting!");  
    }  
}  
  
class Test  
{  
    public static void main(String[] args)  
    {  
        Thread t1 = new Thread(new NewThread("First"));  
        t1.setPriority(Thread.MIN_PRIORITY);  
  
        Thread t2 = new Thread(new NewThread("Second"));  
        t2.setPriority(t1.getPriority());  
  
        Thread t3 = new Thread(new NewThread("Third"));  
        t3.setPriority(Thread.MAX_PRIORITY);  
  
        t1.start();  
        t2.start();  
        t3.start();  
    }  
}
```

OUTPUT:

```
cmd Command Prompt
C:\Users\91817>cd Desktop

C:\Users\91817\Desktop>cd javap

C:\Users\91817\Desktop\javap>javac Main.java

C:\Users\91817\Desktop\javap>java Test
New Thread :Thread[First,5,main]
New Thread :Thread[Second,5,main]
New Thread :Thread[Third,5,main]
Third : 1
Second : 1
First : 1
Second : 2
First : 2
Third : 2
Second : 3
First : 3
Third : 3
Second : 4
First : 4
Third : 4
Second : 5
First : 5
Third : 5
```

```
Second : 6
First : 6
Third : 6
Second : 7
First : 7
Third : 7
Second : 8
First : 8
Third : 8
Second : 9
First : 9
Third : 9
Second : 10
First : 10
Third : 10
Second is Exiting!
First is Exiting!
Third is Exiting!
```

```
C:\Users\91817\Desktop\javap>
```

CSE2005 Lab Assignment-9

Name:B.Naveen sai

Reg no:19BCD7015

Slot:L5

```
import java.util.*;
public class Main extends Thread {
    static int n;
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the number till which you want to print primes and
composites: ");
        int n = sc.nextInt();
        ThreadA a = new ThreadA(n);
        ThreadB b = new ThreadB(n);
        a.start();
        b.start();
        try {
            a.join();
            b.join();
        } catch (InterruptedException e) {
            System.out.println("Interrupted");
        }
    }
}
class ThreadA extends Thread {
```

```
int num;

ThreadA(int n) {
    num = n;
}

public void run() {
    System.out.println("From Thread A ");
    prime();
    System.out.println("Exiting from Thread A ...");
}

synchronized void prime() {
    int flag = 0;

    System.out.println("Prime Numbers before " + num + " are ");

    for (int i = 1; i<= num; i++) {
        if (i == 1 || i == 0)
            continue;
        flag = 1;
        for (int j = 2; j<= i / 2; ++j) {
            if (i % j == 0) {
                flag = 0;
                break;
            }
        }
        if (flag == 1)
            System.out.println(i);
    }
}

class ThreadB extends Thread {
```

```
int num;  
ThreadB(int n) {  
    num = n;  
}  
public void run() {  
    System.out.println("From Thread B ");  
    composite();  
    System.out.println("Exiting from Thread B ...");  
}  
synchronized void composite() {  
    int flag = 0;  
    System.out.println("Composite Numbers before " + num + " are ");  
    for (int i = 1; i<= num; i++) {  
        if (i == 1 || i == 0)  
            continue;  
        flag = 1;  
        for (int j = 2; j<= i / 2; ++j) {  
            if (i % j == 0) {  
                flag = 0;  
                break;  
            }  
        }  
        if (flag == 0)  
            System.out.println(i);  
    }  
}
```

OUTPUT

```
ca Command Prompt
C:\Users\91817\Desktop\javap>javac Main.java
C:\Users\91817\Desktop\javap>java Main
Enter the number till which you want to print primes and composites:
35
From Thread A
From Thread B
Prime Numbers before 35 are
Composite Numbers before 35 are
2 3 5 7 11 13 4
17 6
19 8
23 9
29 10
31 12
14
15
16
18
20
21
22
24

25
Exiting from Thread A ...
26
27
28
```

```
ca Command Prompt
6
19
8
23
9
29
31
Exiting from Thread A ...
10
12
14
15
16
18
20
21
22
24
25
26
27
28
30
32
33
34
35
Exiting from Thread B ...
C:\Users\91817\Desktop\javap>
```

CSE2005 Lab Assignment-10

Name:B.Naveen Sai

Reg no:19BCD7015

Slot:L5

```
public class Test1 {  
    public static void main(String[] args)  
        throws InterruptedException  
{  
    final Customer p = new Customer();  
    Thread t1 = new Thread(new Runnable() {  
        public void run()  
        {  
            try {  
                p.deposit();  
            }  
            catch (InterruptedException e) {  
                e.printStackTrace();  
            }  
        }  
    });  
    Thread t2 = new Thread(new Runnable() {  
        public void run()  
        {  
            try {  
                p.withdraw();  
            }  
            catch (InterruptedException e) {  
                e.printStackTrace();  
            }  
        }  
    });  
    t1.start();  
    t2.start();  
    t1.join();  
    t2.join();  
}
```

```
        e.printStackTrace();
    }
});

Thread t3 = new Thread(new Runnable() {
    public void run()
    {
        try {
            p.deposit();
        }
        catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
});

t1.start();
t2.start();
}

public static class Customer {
    public void deposit() throws InterruptedException
    {
        synchronized(this){
            System.out.println("Available Balance: 10000");
            System.out.println("About to withdraw: 15000");
            System.out.println("Insufficient Balance - Need to Deposit.");
            System.out.println("About to deposit: 10000");
            System.out.println("Available Balance: 10000");
            System.out.println("Transaction completed");
        }
    }
}
```

```
}

public void withdraw() throws InterruptedException
{
    synchronized(this){
        System.out.println(" ");
        System.out.println("Detected amount: 15000");
        System.out.println("Balance amount: 5000");
        System.out.println("About to deposit: 10000");
        System.out.println("Available Balance: 10000");
        System.out.println("Transaction completed");
    }
}

}

}
```

OUTPUT

```
Command Prompt
Microsoft Windows [Version 10.0.18363.1198]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\91817>cd Desktop

C:\Users\91817\Desktop>cd javap

C:\Users\91817\Desktop\javap>javac Test1.java

C:\Users\91817\Desktop\javap>java Test1
Available Balance: 10000
About to withdraw: 15000
Insufficient Balance - Need to Deposit.
About to deposit: 10000
Available Balance: 10000
Transaction completed

Detected amount: 15000
Balance amount: 5000
About to deposit: 10000
Available Balance: 10000
Transaction completed

C:\Users\91817\Desktop\javap>
```

CSE2005 Lab Assignment-11

Name:B.Naveen sai

Reg no:19BCD7015

Slot:L5

Source code

```
import javax.swing.*;
import javax.swing.border.*;
import java.awt.*;
import java.awt.event.*;
class PizzaCalculator extends JFrame implements ActionListener
{
    JPanel westPanel = new JPanel();
    JPanel eastPanel = new JPanel();
    JPanel centerPanel = new JPanel();
    JPanel northPanel = new JPanel();
    JPanel southPanel = new JPanel();
    JRadioButton Small = new JRadioButton("Small");
    JRadioButton Medium = new JRadioButton("Medium");
    JRadioButton Large = new JRadioButton("Large");
    JCheckBox Plain = new JCheckBox("Plain");
    JCheckBox Sausage = new JCheckBox("Sausage");
    JCheckBox Mushroom = new JCheckBox("Mushroom");
    JCheckBox Pepperoni = new JCheckBox("Pepperoni");
    JButton Calculate = new JButton("Calculate");
    JButton Exit = new JButton("Exit");
    JLabel Total = new JLabel("Price: ");
```

```
JTextField TotalResult = new JTextField(8);
int TopQty = 0;
double Size = 0;
double TopCost = 0;
double TotalCost = 0;
PizzaCalculator()
{
    northPanel.setLayout(new GridLayout());
    northPanel.setBorder(new TitledBorder("Size:"));
    Small.addActionListener(this);
    Medium.addActionListener(this);
    Large.addActionListener(this);
    Calculate.addActionListener(this);
    ButtonGroup pizzaSize = new ButtonGroup();
    pizzaSize.add(Small);
    pizzaSize.add(Medium);
    pizzaSize.add(Large);
    northPanel.add(Small);
    northPanel.add(Medium);
    northPanel.add(Large);
    centerPanel.setLayout(new GridLayout(5, 3));
    centerPanel.setBorder(new TitledBorder("Toppings:"));
    centerPanel.add(Plain);
    centerPanel.add(Sausage);
    centerPanel.add(Mushroom);
    centerPanel.add(Pepperoni);
    southPanel.add(Total);
    southPanel.add(TotalResult);
```

```
southPanel.add(Calculate);
southPanel.add(Exit);
setLayout(new BorderLayout());
add(northPanel, BorderLayout.NORTH);
add(centerPanel, BorderLayout.CENTER);
add(southPanel, BorderLayout.SOUTH);
}

public void actionPerformed(ActionEvent e)
{
if (e.getSource() == Small)
setSizeS();
else if (e.getSource() == Medium)
setSizeM();
else if (e.getSource() == Large)
setSizeL();
setTopCost();
if (e.getSource() == Calculate)
calculate(Size, TopCost);
}

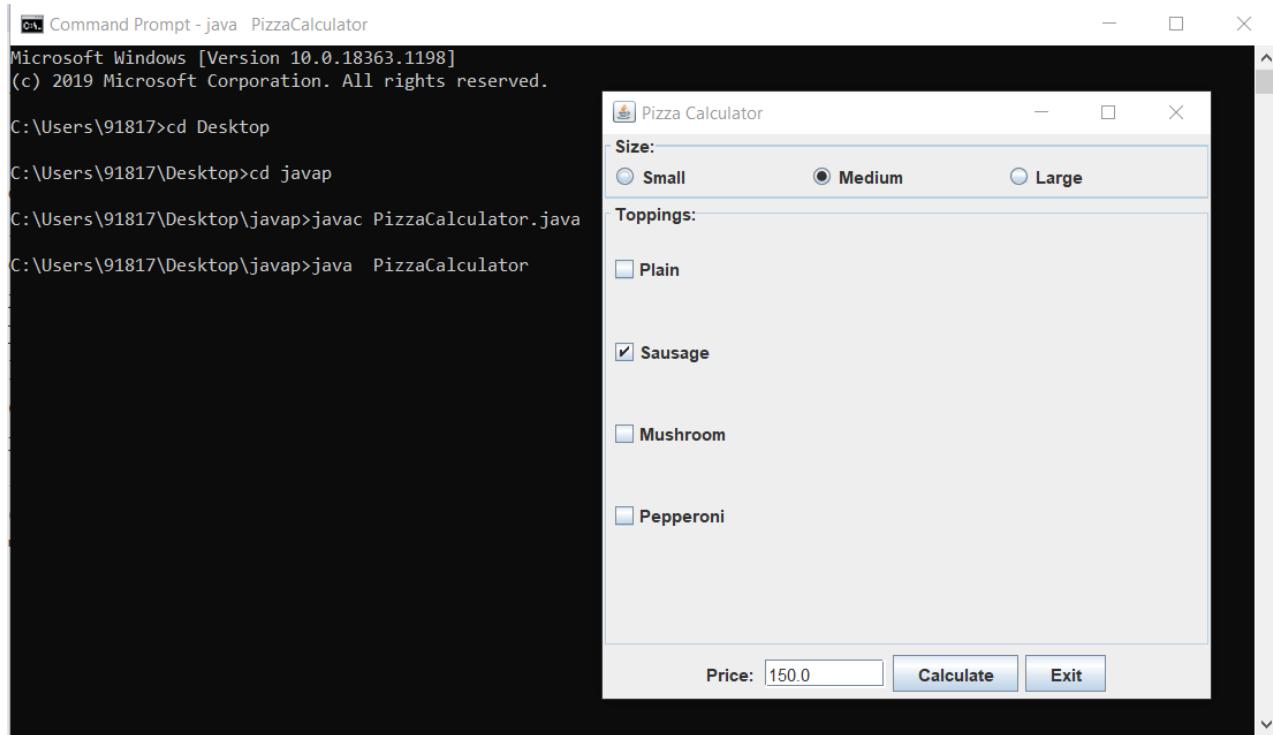
public double setSizeS()
{
return Size = 100;
}

public double setSizeM()
{
return Size = 130;
}

public double setSizeL()
```

```
{  
    return Size = 160;  
}  
  
public double setTopCost()  
{ if ( Plain.isSelected())  
    TopCost = 0;  
    else if (Sausage.isSelected())  
    TopCost = 20;  
    else if (Mushroom.isSelected())  
    TopCost = 20;  
    else if (Pepperoni.isSelected())  
    TopCost = 20;  
    return TopCost;  
}  
  
public void calculate(double Size, double TopCost)  
{  
    TotalResult.setText(String.valueOf(Size + TopCost));  
}  
  
public static void main(String[] args)  
{  
    PizzaCalculator PizzaCalc = new PizzaCalculator();  
    PizzaCalc.setTitle("Pizza Calculator");  
    PizzaCalc.setLocationRelativeTo(null);  
    PizzaCalc.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
    PizzaCalc.setSize(300, 350);  
    PizzaCalc.setVisible(true);  
}  
}
```

OUTPUT



CSE2005 Lab Assignment-12

Name:B.Naveen Sai

Reg no:19BCD7015

Slot:L5

Source Code

```
import javafx.application.Application;  
import javafx.scene.Scene;  
import javafx.scene.control.Button;  
import javafx.scene.layout.*;  
import javafx.event.ActionEvent;  
import javafx.event.EventHandler;  
import javafx.scene.control.*;  
import javafx.stage.Stage;  
import javafx.scene.control.Alert.AlertType;  
import java.time.LocalDate;  
  
public class Lab12_19BCD7015 extends Application {  
  
    public void start(Stage s)  
    {  
  
        s.setTitle("Demonstrate Menus");  
        Menu m = new Menu("File");  
        MenuItem m1 = new MenuItem("Open");  
        MenuItem m2 = new MenuItem("Close");  
        MenuItem m3 = new MenuItem("Save");  
    }  
}
```

```
MenuItem m4 = new MenuItem("Exit");

    m.getItems().add(m1);
    m.getItems().add(m2);
    m.getItems().add(m3);

m.getItems().add(m4);

Menu n = new Menu("Options");
MenuItem helpmenu1=new MenuItem("op");
n.getItems().add(helpmenu1);
Menu l = new Menu("Help");
MenuItem helpmenu2=new MenuItem("hlp");
l.getItems().add(helpmenu2);

MenuBar mb = new MenuBar();

mb.getMenus().addAll(m,n,l);

VBox vb = new VBox(mb);

Scene sc = new Scene(vb, 500, 300);

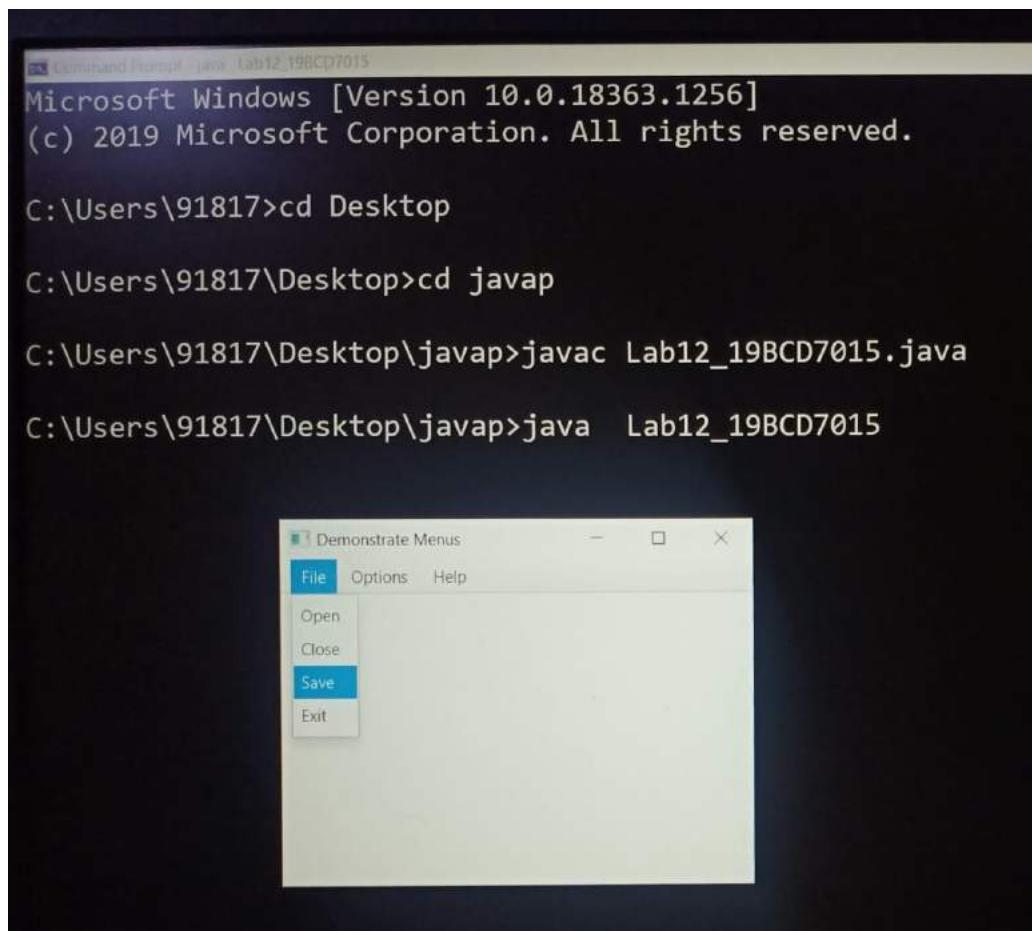
s.setScene(sc);

s.show();
}
```

```
public static void main(String args[])
{
    launch(args);
}

}
```

OUTPUT



The image shows two screenshots demonstrating Java application development.

The top screenshot is a Windows Command Prompt window titled "Command Prompt - Java Lab12_19BCD7015". It displays the following command-line session:

```
Microsoft Windows [Version 10.0.18363.1256]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\91817>cd Desktop

C:\Users\91817\Desktop>cd javap

C:\Users\91817\Desktop\javap>javac Lab12_19BCD7015.java

C:\Users\91817\Desktop\javap>java Lab12_19BCD7015
```

The bottom screenshot shows a Java application window titled "Demonstrate Menus". The window has a menu bar with "File", "Options", and "Help" items. A "File" submenu is open, showing "Open", "Close", "Save" (which is highlighted), and "Exit" options.