1Q. Create a class called Complex for performing arithmetic with complex numbers having real and imaginary part . Write a program to test your class. Use floating-point variables to represent the private data of the class. Provide a constructor that enables an object of this class to be initialized when it's declared. Provide a no-argument constructor with default values in case no initializers are provided. Provide public methods that perform the following operations: addition, subtraction and print complex numbers

```
/*
 * Create a class called Complex
 * for performing arithmetic with complex numbers having real and
 * imaginary part .
 * Write a program to test your class.
 * Use floating-point variables to represent the private data of the
 *  class.  Provide a constructor that enables an object of this
 *  class to be initialized when it's declared.
 *  Provide a no-argument constructor with default values
 *  in case no initializers are provided.
 *  Provide public methods that perform the following operations:
 * addition, subtraction and print complex numbers
 */
class complex1
{
float x,y;
complex1()
{
      x=0;y=0;
}
complex1(float x,float y)
{
      this.x=x;
      this.y=y;
}
complex1 add(complex1 c1,complex1 c2)
{
complex1 c3=new complex1();
c3.x=c1.x+c2.x;
c3.y=c1.y+c2.y;
      return c3;

}
complex1 sub(complex1 c1,complex1 c2)
{
complex1 c3=new complex1();
c3.x=c1.x-c2.x;
c3.y=c1.y-c2.y;
      return c3;

}
void display()
{
      System.out.println("Complex number :"+x+" + i "+y);
}
```

```
}

public class Q1
{
public static void main(String[] args)
      {
            complex1 c1,c2,c3;
            c1=new complex1(2,3);
            c2=new complex1(5,6);
            c3=new complex1();
            c3=c3.add(c1, c2);
            c1.display();
            c2.display();
            c3.display();
      }

}
```

2Q. Create a class Rectangle with attributes length and width, each of which defaults to 1. Provide methods that calculate the rectangle's perimeter and area. It has set and get methods for both length and width. The set methods should verify that length and width are each floating-point numbers larger than 0.0 and less than 20.0. Write a program to test class Rectangle.

```
/*
 * Create a class Rectangle with attributes length and width,
 * each of which defaults to 1.
 * Provide methods that calculate the rectangle's perimeter and area.
 * It has set and get methods for both length and width.
 * The set methods should verify that length and width are
 * each floating-point numbers
 * larger than 0.0 and less than 20.0.
 * Write a program to test class Rectangle
 */
class Rectangle1
{
      float length,width;
      Rectangle1()
      {
      length=width=1.0f;
      }
      void setlength(float length)
      {
            if(length>=0.0 && length<=20.0)
                  this.length=length;
      }
      void setwidth(float width)
      {
            if(width>=0.0 && width<=20.0)
                  this.width=width;
      }
      float getlength()
      {
            return length;
```

```
	}
	float getwidth()
	{
		return width;
	}
	void area()
	{
		float a=length * width;
		System.out.println("Area of Rectangle :"+a);
	}
	void perimeter()
	{
		float p = 2*(length+width);
		System.out.println("Perimeter :"+p);
	}
}
public class Q2 {

	public static void main(String[] args) {
		Rectangle1 r1=new Rectangle1();
		r1.setlength(4.5f); // f used for float variable
		r1.setwidth(7.8f);
		r1.area();
		r1.perimeter();

	}

}
```

3Q Create a class called as time having hours ,minutes and seconds as data members . Identify suitable methods to input hours , minutes , seconds . Compute addition and difference of two-time objects . these methods to return time objects to main method .

```
package internalprgs;
/*
 * Create a class called as time having hours ,minutes and
 * seconds as data members . Identify suitable methods to input hours ,
 * minutes , seconds . Compute addition and difference of
 * two-time objects .
 * these methods to return time objects to main method .
 */
class time1
{
	int hr,min,sec;
	time1()
	{
		hr=min=sec=0;
	}
	void setTime(int hr,int min,int sec)
	{
		this.hr=hr;
		this.min=min;
		this.sec=sec;
	}
```

```
	time1 addTime(time1 t1,time1 t2)
	{
		time1 t3=new time1();

		t3.sec= t1.sec + t2.sec;
		t3.min=t1.min + t2.min + (t3.sec/60);
		t3.hr = t1.hr +t2.hr + (t3.min/60);
		t3.min %=60;
		t3.sec %=60;

		return t3;

	}
	time1 subTime(time1 t1, time1 t2)
	{
		time1 t3=new time1();
		t3.sec= Math.abs(t1.sec - t2.sec);
		t3.min=Math.abs(t1.min - t2.min - (t3.sec/60));
		t3.hr = Math.abs(t1.hr - t2.hr - (t3.min/60));
		t3.min %=60;
		t3.sec %=60;
		return t3;
	}
	void showTime(time1 t1)
	{
		System.out.println("hr : min : sec ---> "+t1.hr+" :
"+t1.min+" : "+t1.sec);
	}
}

public class Q3 {

	public static void main(String[] args)
	{
	time1 t1=new time1();
	time1 t2=new time1();
	time1 t3=new time1();
t1.setTime(2, 40,10);
t2.setTime(2, 40,10);
t3=t3.addTime(t1, t2);
t3.showTime(t3);

	}

}
```

4Q. **Create a package called as SBI . Now create a class on the name Account with suitable data members and methods . in different location create a class SavingsAccount , LoanAccount and inherit data from Account class created in SBI Package . Write all the steps included in this process**

```
package internalprgs;
import sbi.*;
import java.util.*;
class savings extends Account
```

```
{
	void getamt()
	{
		readAcc();
		System.out.println("Enter Savings balance amount :");
		Scanner sc=new Scanner(System.in);
		amt=sc.nextFloat();
		printAcc();
		System.out.println("Savings amount Balance :"+amt);


	}
}
class loan extends Account
{
	void getamt()
	{

		System.out.println("Enter Loan balance amount :");
		Scanner sc=new Scanner(System.in);
		amt=sc.nextFloat();

		System.out.println("Loan amount Balance :"+amt);


	}
}
public class Q4 {

	public static void main(String[] args) {
		savings s=new savings();
		s.getamt();
		loan l=new loan();
		l.getamt();

	}

}

package sbi;
import java.util.*;
public class Account {
	public int accid;
	public String accname;
	public float amt;
	public Account()
	{
		accid=0;
		accname=null;
	}
	public void readAcc()
	{
		Scanner sc=new Scanner(System.in);
		System.out.println("Enter Account name and id : ");
```

```
            accname=sc.nextLine();
            accid=sc.nextInt();

      }
      public void printAcc()
      {
            System.out.println("Account id "+accid);
            System.out.println("Account Holder Name "+accname);

      }


}
```

5Q Create a class called Employee that includes three instance variables—a first name (type String), a last name (type String) and a monthly salary (double). Provide a constructor that initializes the three instance variables. Provide a set and a get method for each instance variable. If the monthly salary is not positive, do not set its value. Write a test app named EmployeeTest that demonstrates class Employee's capabilities. Create two Employee objects and display each object's yearly salary. Then give each Employee a 10% raise and display each Employee's yearly salary again.

```
package internalprgs;
/*Create a class called Employee that includes three instance
 * variables—a first name (type String), a last name (type String)
 *  and a monthly salary (double).
 *  Provide a constructor that initializes the three instance variables.
 *  Provide a set and a get method for each instance variable.
 *  If the monthly salary is not positive, do not set its value.
 *  Write a test app named EmployeeTest that demonstrates class
 *  Employee's capabilities. Create two Employee objects and display
 *  each object's yearly salary.
 *  Then give each Employee a 10% raise and display each Employee's
 *  yearly salary again.
 *
 *
 */
class Employee
{
      String FirstName,LastName;
      double Salary;
      Employee(String FirstName,String LastName,double Salary)
      {
            this.FirstName=FirstName;
            this.LastName=LastName;
            this.Salary=Salary;
      }
      public String getFirstName() {
            return FirstName;
      }
      public void setFirstName(String firstName) {
```

```
            FirstName = firstName;
      }
      public String getLastName() {
            return LastName;
      }
      public void setLastName(String lastName) {
            LastName = lastName;
      }
      public double getSalary() {
            return Salary;
      }
      public void setSalary(double salary) {
            if(Salary>0)
                  Salary = salary;
      }
      public void raiseSalary()
      {
            Salary=Salary+ (0.1) * Salary;
      }
      public void AnnualSalary()
      {
            System.out.println("Annual Salary : "+Salary*12);
      }

}
public class Q5 {

      public static void main(String[] args) {
            Employee e1=new Employee("Raghu ","Rama ",25000);
            Employee e2=new Employee("Raghava ","Madhav ",45000);
            e1.AnnualSalary();e2.AnnualSalary();
            System.out.println("Salaries after raise : ");
            e1.raiseSalary();e2.raiseSalary();
            e1.AnnualSalary();e2.AnnualSalary();



      }

}
```

6Q Create a class called Invoice that a hardware store might use to represent an invoice for an item sold at the store. An Invoice should include four pieces of information as instance variables—a part number (type String), a part description (type String), a quantity of the item being purchased (type int) and a price per item (double). Your class should have a constructor that initializes the four instance variables. Provide a set and a get method for each instance variable. In addition, provide a method named getInvoiceAmount that calculates the invoice amount (i.e., multiplies the quantity by the price per item), then returns the amount as a double value. If the quantity is not positive, it should be set to 0. If the price per item is not positive, it should be set to 0.0. Write a test app named InvoiceTest that demonstrates class Invoice's capabilities.

```
package internalprgs;
/*
 * Create a class called Invoice that a hardware store might use to
 * represent an invoice for an item sold at the store.
 * An Invoice should include four pieces of information as
 * instance variables—a part number (type String), a part description
 *  (type String), a quantity of the item being purchased (type int)
 *  and a price per item (double).
 *  Your class should have a constructor that initializes
 *  the four instance variables.
 *  Provide a set and a get method for each instance variable.
 *  In addition, provide a method named getInvoiceAmount that calculates
 *   the invoice amount
 *   (i.e., multiplies the quantity by the price per item),
 *   then returns the amount as a double value.
 *   If the quantity is not positive, it should be set to 0.
 *   If the price per item is not positive, it should be set to 0.0.
 *   Write a test app named InvoiceTest that demonstrates class Invoice's
 *   capabilities.
 *
 */
class Invoice
{
      String PartNo,Desc;
      int qty;
      double price;
      Invoice(String PartNo,String Desc,int qty,double price)
      {
      this.PartNo=PartNo;
      this.Desc=Desc;
      this.qty=qty;
      this.price=price;
      }
      public String getPartNo() {
            return PartNo;
      }
      public void setPartNo(String partNo) {
            PartNo = partNo;
      }
      public String getDesc() {
            return Desc;
      }
      public void setDesc(String desc) {
            Desc = desc;
      }
      public int getQty() {
            return qty;
      }
      public void setQty(int qty) {

            this.qty = (qty<0)? 0 :qty;
      }
      public double getPrice() {
            return price;
```

```
	}
	public void setPrice(double price) {
		this.price = (price<0) ?0 :price;
	}
	double getInvoiceAmount()
	{
		return (qty*price);
	}
	void display()
	{
		System.out.println("Part No : "+PartNo);
		System.out.println("Part Description  : "+Desc);
		System.out.println("Quantity : "+qty);
		System.out.println("Price :"+price);
	}
}
//Q6.java can be renamed as InvoiceTest.java
// and Q6 class should be renamed as InvoiceTest
public class Q6 {

	public static void main(String[] args) {
		Invoice i1=new Invoice("B101","Two wheeler brake
wire",10,300);
		i1.display();
		System.out.println("Invoice Amount :"+i1.getInvoiceAmount());

	}

}
```

7Q. Write an application that uses String method regionMatches to compare two strings input by the user. The application should input the number of characters to be compared and the starting index of the comparison. The application should state whether the strings are equal. Ignore the case of the characters when performing the comparison.

```
package internalprgs;
import java.util.*;
public class Q7 {

	public static void main(String[] args) {
		String str1 ,str2 ;
		Scanner sc=new Scanner(System.in);
		System.out.println("Enter First String : ");
		str1=sc.nextLine();
		System.out.println("Enter Second String : ");
		str2=sc.nextLine();
	  System.out.print("Return Value :" );
	  // numbers in region matches are the postions
	  // they can be changed
	  System.out.println(str1.regionMatches(true, 2, str2, 0, 3));

	}
```

```
}
```
8Q. Create an Exception for a bean business if quantity is greater than 5 throw QuantityException and inform the customer that Quantity should not exceed 5 . Create another exception to throw InvalidAmountException if the amount is <500 when payment done at billing systems

```
package internalprgs;
/*
 * Create an Exception for a bean business if quantity is greater than 5
 * throw QuantityException and inform the customer that Quantity should
not
 * exceed 5 .
 * Create another exception to throw InvalidAmountException
 * if the amount is <500 when payment done at billing systems */

@SuppressWarnings("serial")
class QuantityException extends Exception
{
 int qty;
 QuantityException(int q)
 {
       qty=q;
 }
 public String toString()
 {
       return "QuantityException "+qty;
 }
}
class InvalidAmountException extends Exception
{
      int amt;
      InvalidAmountException(int a)
      {
            amt=a;
      }
      public String toString()
      {
            return "InvalidAmountException "+amt;
      }
}
public class Q8 {
      static void methodQE(int q) throws QuantityException
      {
      int maxqty=5;
       if(q>maxqty)
            throw new QuantityException(q);
      else
            System.out.println("Quantity :"+q);
      }
      static void methodIAE(int a) throws InvalidAmountException
      {
            int threshold=500;
            if(a<threshold)
                  throw new InvalidAmountException(a);
```

```
            else
                  System.out.println("Amount : "+a);

      }
      public static void main(String[] args) {
            try
            {
            //    methodQE(10); runs when we remove comment
                  methodIAE(1000);
                  methodQE(3);
            methodIAE(250);
            }catch(QuantityException qe)
            {
                  System.out.println(qe);
            }
            catch(InvalidAmountException iae)
            {
                  System.out.println(iae);

            }

      }

}
```

9Q. Create an AWT Application to read a line of text and display the number of words , characters , vowels and consonants .Use proper components wherever required .Layout or bounds can be used while placing the components

```
package internalprgs;
import java.awt.*;
import java.awt.event.*;
class counts extends Frame implements ActionListener
{
      TextField t1;
      Button b1;
      Label l1;
      Label l2;
      counts()
      {
            setTitle("Count Words lines and Paragraphs");
            setVisible(true);
            pack();
            l1=new Label("Enter a line of text : ");
            t1=new TextField(20);
            b1=new Button("Count words characters  vowels and consonants
");
            l2=new Label("   ");
            l2.setSize(50,20);
            add(l1); add(t1);      add(b1);    add(l2);
            setLayout(new FlowLayout());
            b1.addActionListener(this);
            addWindowListener(new WindowAdapter() {
                  public void windowClosing(WindowEvent we)
```

```
                {
                      System.exit(0);
                }
            });


      }

      public void actionPerformed(ActionEvent ae)
      {
               int vCount = 0, cCount = 0,wCount=1;
               String str=t1.getText();
               int i;
               str = str.toLowerCase();
               String text;

            if(ae.getSource()==b1)
            {
                  for(i = 0; i < str.length(); i++)
                  {
             if(str.charAt(i) == 'a' || str.charAt(i) == 'e' ||
str.charAt(i) == 'i' || str.charAt(i) == 'o' || str.charAt(i) == 'u')
                        vCount++;
              else if(str.charAt(i) >= 'a' && str.charAt(i)<='z')
                              cCount++;
              else if(str.charAt(i)==' ')
                        wCount++;
               }
                  text="Characters : "+i+" , Words :"+wCount+" , Vowels
:"+cCount+" , Consonants :"+cCount;
                  l2.setText(text);
            }
      }
}


class Q9
{
public static void main(String args[])
{
new counts();
}
}
```

10Q. Create an Applet / AWT application to compute addition , subtraction and product of two numbers .Use Buttons TextField and Labels wherever required . Layout or bounds can be used while placing the components .

```
package internalprgs;
import java.awt.*;
import java.awt.event.*;
class basiccalci extends Frame implements ActionListener
{
```

```
	Button b1,b2,b3;
	Label l1,l2,l3;
	TextField t1,t2,t3;
	basiccalci()
	{
		setTitle("Simple calculator");
		setLayout(new FlowLayout());
		setSize(300,300);
		setVisible(true);

		l1=new Label("First Number :");
		l2=new Label("Second Number :");
		l3=new Label("Result :");

		b1=new Button("+");
		b2=new Button("SUB ");
		b3=new Button("PROD ");
		t1=new TextField(10);
		t2=new TextField(10);
		t3=new TextField(10);
		add(l1);add(t1);
		add(l2);add(t2);
		add(l3);add(t3);
		add(b1);add(b2);add(b3);
		b1.addActionListener(this);
		b2.addActionListener(this);
		b3.addActionListener(this);
		addWindowListener(new WindowAdapter() {
			public void windowClosing(WindowEvent we)
			{
				System.exit(0);
			}
		});
	}
public void actionPerformed(ActionEvent ae)
	{
		int x=Integer.parseInt(t1.getText());
		int y=Integer.parseInt(t2.getText());
if(ae.getSource()==b1)
	{

	int z=x+y;
	t3.setText(z+""); // t3.setText(z.toString());
	}
	if(ae.getSource()==b2)
	{
	int z=x-y;
	t3.setText(z+"");
	}
	if(ae.getSource()==b3)
	{
	int z=x*y;
	t3.setText(z+"");
	}
```

```
	}
}

public class Q10 {
	public static void main(String[] args) {
		new  basiccalci();
	}

}
```

11Q. Create class SavingsAccount. Use a static variable annualInterestRate to store the annual interest rate for all account holders. Each object of the class contains a private instance variable savingsBalance indicating the amount the saver currently has on deposit. Provide method calculateMonthlyInterest to calculate the monthly interest by multiplying the savingsBalance by annualInterestRate divided by 12–this interest should be added to savingsBalance. Provide a static method modifyInterestRate that sets the annualInterestRate to a new value. Write a program to test class SavingsAccount. Instantiate two savingsAccount objects, saver1 and saver2, with balances of $2000.00 and $3000.00, respectively. Set annualInterestRate to 4%, then calculate the monthly interest for each of 12 months and print the new balances for both savers. Next, set the annualInterestRate to 5%, calculate the next month's interest and print the new balances for both savers.

```
package internalprgs;

 class SAccount {
public static double annualInterestRate;
private double savingsBalance;
public SAccount() {
annualInterestRate = 0.0;
savingsBalance = 0.0;
}
public SAccount(double intRate, double savBal) {
annualInterestRate = intRate;
savingsBalance = savBal;
}
public double calculateMonthlyInterest() {
double intRate = (savingsBalance * annualInterestRate/12);
savingsBalance = savingsBalance + intRate;
return intRate;
}
public static void modifyInterestRate(double newInteresRate) {
annualInterestRate = newInteresRate;
}
public void setSavingsBalance(double newBal) {
savingsBalance = newBal;
}
public double getSavingsBalance() {
return savingsBalance;
}
public double getAnnualInterestRate() {
return annualInterestRate;
}
```

```
}
//filename: SavingsAccountTest.java
// SavingsAccount testing class with the main() method
public class Q11 {
public static void main(String[] args) {
SAccount saver1 = new SAccount();
SAccount saver2 = new SAccount();
saver1.setSavingsBalance(2000.00);
saver2.setSavingsBalance(3000.00);
SAccount.modifyInterestRate(0.04);
saver1.calculateMonthlyInterest();
saver2.calculateMonthlyInterest();
System.out.printf("New Balance for
Saver1=%f\n",saver1.getSavingsBalance());
System.out.printf("New Balance for
Saver2=%f\n",saver2.getSavingsBalance());
SAccount.modifyInterestRate(0.05);
saver1.calculateMonthlyInterest();
saver2.calculateMonthlyInterest();
System.out.printf("New Balance for
Saver1=%f\n",saver1.getSavingsBalance());
System.out.printf("New Balance for
Saver2=%f\n",saver2.getSavingsBalance());
}
}
```

12Q. Create an abstract class called as shape . Define a method called as area in it .Define the method in inherited classes such as triangle (compute area using a b c sides) , Rectangle and Square .

```
package internalprgs;

/*
 * Create an abstract class called as shape . Define
 *  a method called as area in it .Define the method in inherited classes
 * such as triangle (compute area using a b c sides) ,
 *  Rectangle and Square */
 abstract class shape
 {
       public abstract void area();
 }
 class rectangle extends shape
 {
       public void area()
       {
       int l=10,b=20;
       System.out.println("Area = "+(l*b));
       }
 }
 class triangle extends shape
 {
       public void area()
       {
```

```
            int a=10,b=20,c=30;
            float s=(float) ((a+b+c)/2.0);
            // use float to get accurate results
            System.out.println("Area = "+(s*(s-a)*(s-b)*(s-c)));
       }
 }
 class square extends shape
 {
       public void area()
       {
             int s=10;
             System.out.println("Area ="+(s*s));
       }
 }
class Q12
{
      public static void main(String[] args)
      {
 rectangle r=new rectangle();
 r.area();
 square s=new square();
 s.area();
 triangle t=new triangle();
 t.area();


      }

}
```

13. Create a Thread to illustrate ThreadPriority , Join , Alive methods . The logic part to include is printing even numbers .

```
package internalprgs;




class NewThread2 implements Runnable
{
      String name; // name of thread
      Thread t;
      NewThread2(String threadname)
            {
                  name = threadname;
                  t = new Thread(this, name);
                  t.setPriority(4);
                  System.out.println("New thread: " + t);
                  t.start(); // Start the thread
            }

public void run()
{
      try
```

```
	{
		for(int i = 2; i<=20; i+=2)
		{
			System.out.println(name + ": " + i);
			Thread.sleep(1000);
		}
	} catch (InterruptedException e)
		{
		System.out.println(name + " interrupted.");
		}
	System.out.println(name + " exiting.");
	}
}

class Q13
{

	public static void main(String args[])
	{
			NewThread2 ob1 = new NewThread2("One");
			NewThread2 ob2 = new NewThread2("Two");
			NewThread2 ob3 = new NewThread2("Three");


			System.out.println("Thread One is alive: "
+ ob1.t.isAlive());

			System.out.println("Thread Two is alive: "
+ ob2.t.isAlive());
			System.out.println("Thread Three is alive: "
+ ob3.t.isAlive());
//		wait for threads to finish
			try {
				System.out.println("Waiting for threads to
finish.");
				ob1.t.join();
				ob2.t.join();
				ob3.t.join();
				} catch (InterruptedException e)
					{
					System.out.println("Main thread
Interrupted");
					}

System.out.println("Thread One is alive: "
+ ob1.t.isAlive());
			System.out.println("Thread Two is alive: "
+ ob2.t.isAlive());
			System.out.println("Thread Three is alive: "
+ ob3.t.isAlive());
	System.out.println("Main thread exiting.");
}
}
```

14Q Create a Student class with proper attributes , Interface Examination having getScore method signature . Now design a child class called ProgressReport that inherits Student class and implements getScore method . Include suitable methods and attributes wherever necessary

```
// Interface Examination
interface Examination {
    double getScore();
}

// Parent class Student
class Student {
    private String name;
    private int age;
    private String studentId;

    public Student(String name, int age, String studentId) {
        this.name = name;
        this.age = age;
        this.studentId = studentId;
    }

    public String getName() {
        return name;
    }

    public int getAge() {
        return age;
    }

    public String getStudentId() {
        return studentId;
    }

    public String getDetails() {
        return "Name: " + name + ", Age: " + age + ", Student ID: " + studentId;
    }
}

// Child class ProgressReport inheriting Student and implementing Examination
class ProgressReport extends Student implements Examination {
    private double mathScore;
    private double scienceScore;
    private double englishScore;

    public ProgressReport(String name, int age, String studentId, double
mathScore, double scienceScore, double englishScore) {
        super(name, age, studentId);
        this.mathScore = mathScore;
        this.scienceScore = scienceScore;
        this.englishScore = englishScore;
    }
```

```
    @Override
    public double getScore() {
        return (mathScore + scienceScore + englishScore) / 3;
    }

    public double getMathScore() {
        return mathScore;
    }

    public double getScienceScore() {
        return scienceScore;
    }

    public double getEnglishScore() {
        return englishScore;
    }

    public String getReport() {
        return getDetails() + "\nScores - Math: " + mathScore + ", Science: " +
scienceScore + ", English: " + englishScore +
                "\nAverage Score: " + getScore();
    }

    public static void main(String[] args) {
        ProgressReport student1 = new ProgressReport("Alice", 20, "S12345", 85,
90, 78);
        System.out.println(student1.getReport());
    }
}
```

15Q Create an Applet application to write your name using any color or your interest using MouseListener or MouseMotionListener Interface .

```
package internalprgs;
import java.applet.*;
import java.awt.*;
import java.awt.event.*;
/*
 * <applet code="Q15" width=300 height=300>
 * </applet>
 */
public class Q15 extends Applet implements MouseMotionListener
{
public void init()
{

      addMouseMotionListener(this);

}
public void paint(Graphics g)
{
```

```
}
public void mouseDragged(MouseEvent e) {
    Graphics g=getGraphics();
    g.setColor(Color.BLUE);
    g.fillOval(e.getX(),e.getY(),20,20);
}
public void mouseMoved(MouseEvent e) {}

}
```