Problem 1

Develop a Java program to find the type of roots of a quadratic equation. Use object-oriented methodology.

```
class QuadraticEquation
{
  private int a;
  private int b;
  private int c;

  public QuadraticEquation(int x, int y, int z)
  {
    a = x;
    b = y;
    c = z;
  }

  public void printTypeOfRoots()
  {
    int d = b*b - 4*a*c;
    if(d == 0)
    {
      System.out.println("real and identical roots");
    }
```

Output.

Real & identical roots.

```
        if(d>0)
        {
          System.out.println("real and distinct roots");
        }
        if(d<0)
        {
          System.out.println("imaginary roots");
        }
    }
}

class QuadraticEquationDemo
{
    public static void main(String args[])
    {
        QuadraticEquation q=new QuadraticEquation(2,4,2);
        q.printTypeOfRoots();
    }
}
```

Problem 2

There are FIVE students in a class. Develop a Java program that will output roll no, total marks and average marks of each student. Use object-oriented methodology.

```
class Student
{
  private int roll no;
  private int mark1;
  private int mark2;
  private int mark 3;

  public Student (int r, int m1, int m2, int m3)
  {
    rollno = r;
    mark1 = m1;
    mark 2 = m2;
    mark 3 = m3;
  }

  public int getRollNo()
  {
    return rollno;
  }
```

Output

1	60	20.0
2	90	30.0
3	105	35.0
4	120	40.0
5	135	45.0

```
public int total()
{
    return mark1+mark2+mark3;
}

public double average()
{
    return (mark1+mark2+mark3)/3;
}
}

class StudentDemo
{
    public static void main(String args[])
    {
        Student[] s = new Student[5];
        s[0] = new Student(1,20,20,20);
        s[1] = new Student(2,30,30,30);
        s[2] = new Student(3,35,35,35);
        s[3] = new Student(4,40,40,40);
        s[4] = new Student(5,45,45,45);
```

```
for(int i=0; i<5; i++)
 {
    System.out.print(s[i].getRollNo() + "  ");
    System.out.print(s[i].total() + "  ");
    System.out.println(s[i].average());
 }
}
}
```

Problem 3

Develop a Java program that can perform the following operations

1. Multiply a complex number by another complex no
2. Multiply a complex number by a scalar.

```
class ComplexNo
{
   private int real;
   private int img;

   public ComplexNo(int real, int img)
   {
      this.real = real;
      this.img = img;
   }

   public void print()
   {
      System.out.println(real + "+i" + img);
   }
```

Output

First complex no:

10 + i20

Second complex no:

24 + i30

Multiplication product:

-360 + i780

Scalar multiplication product:

100 + i200

```
public ComplexNo add(ComplexNo c)
{
    int real = this.real + c.real;
    int img = this.img + c.img;
    return new ComplexNo(real, img);
}

public ComplexNo multiply(ComplexNo d)
{
    int real = (this.real * d.real) - (this.img * d.img);
    int img = (this.real * d.img) + (d.real * this.img);
    ComplexNo result = new ComplexNo(real, img);
    return result;
}

public ComplexNo multiply(int y)
{
    int real = y * this.real;
    int img = y * this.img;
    ComplexNo result = new ComplexNo(real, img);
    return result;
}
}
```

```
class ComplexNoMultiply
{
  public static void main(String args[])
  {
     ComplexNo d1 = new ComplexNo (10,20);
     System.out.println(" First complex no : ");
     d1.print();
     ComplexNo d2 = new ComplexNo(34,30);
     System.out.println(" Second complex no : ");
     d2.print();
     ComplexNo d3 = d1.multiply(d2);
     System.out.println(" Multiplication product : ");
     d3.print();
     ComplexNo d4 = d1.multiply(10);
     System.out.println("Scalar multiplication product : ");
     d4.print();
  }
}
```

Problem 4

Create a class BankAccount having the following specification
attributes - account no, balance
method - credit, debit, print
Create another class SavingsAccount through inheritance.
It should add the functionality for computing interest.

```
class BankAccount
{
    protected int AccountNo;
    protected int Balance;

    public BankAccount(int AccountNo, int Balance)
    {
        this.AccountNo = AccountNo;
        this.Balance = Balance;
    }

    public void Print()
    {
        System.out.println("Account Number :" + AccountNo);
        System.out.println("Available Balance :" + Balance);
    }
```

Output

Account Number : 12334
Available Balance : 50000
amount of 20000 is credited
Account Number : 12334
Available Balance : 70,000
Simple interest of available balance of 70000 is 1400
amount of 2000 is debited
Account Number : 12334
Available Balance : 68000
Simple interest for available balance of 68000 is 1360

```
public int Credit(int amount)
{
  System.out.println("amount of" + amount + " is credited");
  Balance = Balance + amount;
  Print();
  return Balance;
}

public int Debit(int amount)
{
  System.out.println("amount of" + amount + "is debited");
  Balance = Balance - amount;
  Print();
  return Balance;
}
}

class SavingsAccount extends BankAccount
{
  private final static int Rate = 2;
  public SavingsAccount(int AccountNo, int Balance)
  {
    super(AccountNo, Balance);
  }
```

```
    public void SimpleInterest(int Balance)
    {
        int Si = Balance * Rate / 100;
        System.out.println("Simple interest for available balance
                            of" + Balance + " is" + Si);
    }
}

class Banking
{
    public static void main(String args[])
    {
        SavingsAccount a = new SavingsAccount(12334, 50000);
        a.Print();
        int balance1 = a.Credit(20000);
        a.SimpleInterest(balance1);
        int balance2 = a.Debit(2000);
        a.SimpleInterest(balance2);
    }
}
```

Problem 5

1. Linear Search

```
import java.util.Scanner;
class Linearsearch
{
   public static void main(String args[])
   {
      int []a = new int[10];
      Scanner b= new Scanner(System.in);
      System.out.println("Enter the size :");
      int n=b.nextInt();
      System.out.println("Enter array elements :");
      for(int i=0;i<n;i++)
      {
         a[i] = b.nextInt();
      }
      System.out.println("Enter the element to be searched :");
      int p=b.nextInt();
      int flag=0;
      for(int i=0;i<n;i++)
      {
```

Output:

```
Enter the size :
5
Enter array elements :
1
2
3
4
5
Enter the element to be searched :
3
  3 is found.
```

```
        if(a[i]==p)
        {
            flag=1;
            break;
        }
    }
    if(flag==1)
    {
        System.out.println(p+" is found");
    }
    else
    {
        System.out.println(p+" is not found");
    }
  }
}
```

2. Binary Search

```
import java.util.Scanner;
class Binarysearch
{
   public static void main(String args[])
   {
      Scanner c = new Scanner(System.in);
      System.out.println("Enter the size :");
      int n= c.nextInt();
      int first = 0;
      int last = n-1;
      int mid;
       int[]a = new int[n];
      System.out.println("Enter the elements :");
      for(int i=0; i<n; i++)
      {
         a[i] = c.nextInt();
      }
      System.out.println("Enter the element to be searched :");
      int p= c.nextInt();
      int flag = 0;
```

Output

```
Enter the size :
4
Enter the elements :
12
20
25
43
51
Enter the element to be searched :
43
  43 is found.
```

```
while (first <= last)
{
    mid = (first + last) / 2;
    if (a[mid] == p)
    {
        flag = 1;
        System.out.println (p + " is found");
        break;
    }
    else if (a[mid] < p)
    {
        last = mid - 1;
    }
    else
    {
        first = mid + 1;
    }
}
if (flag == 0)
    System.out.println (p + " is not found");
}
}
```

Problem 6

1. Selection Sort

```
import java.util.Scanner;
class Selection Sort
{
   public static void main(String args[])
   {
      Scanner s= new Scanner (System.in);
      System.out.println("Enter the size : ");
      int n = s.nextInt();
      int []a = new int[n];
      System.out.println("Enter the array elements :");
      for(int i=0; i<n; i++)
      {
        a[i]= s.nextInt();
      }
      int i,j;
      for(int i=0; i<n; i++)
      {
        for (j=i+1; j<n; j++)
```

Output

```
Enter the size :
5
Enter the array elements :
49
12
43
39
20
Array after sorting :
12
20
39
43
49
```

```
{
    if(a[i] >a[j])
    {
        int temp;
        temp = a[i];
        a[i] = a[j];
        a[j] = temp;
    }
  }
}
System.out.println("Array after sorting:");
for(i=0; i<n; i++)
{
    System.out.println(a[i]);
}
}
}
```

2. Bubble Sort

```
import java.util.Scanner;
class BubbleSort
{
  public static void main(String args[])
  {
    Scanner s = new Scanner (System.in);
    System.out.println ("Enter the size :");
    int n = s.nextInt();
    int []a = new int[n];
    System.out.println ("Enter the array elements :");
    for(int i=0; i<n; i++)
    {
      a[i] = s.nextInt();
    }
    int i,j;
    for(i=0; i<n; i++)
    {
      int flag = 0;
      for (j=0; j<n-1; j++)
      {
```

Output

```
Enter the size :
5
Enter the array elements :
34
59
41
20
43
Array after sorting :
20
34
41
43
59
```

```
            if(a[j] > a[j+1])
            {
                int temp;
                temp = a[j];
                a[j]= a[j+1];
                a[j+1] = temp;
                flag = 1;
            }
        }
        if(flag==0)
            break;
    }
    System.out.println("Array after sorting:");
    for(i=0; i<n; i++)
    {
        System.out.println(a[i]);
    }
  }
}
```