1.Write a java program to take input as a command line argument. Your name, course, university roll no and semester. Display the information.
Name : University Roll No : Course
Semester:

```
public class CommandLineInput {
   public static void main(String[] args) {
      if (args.length< 4) {
System.out.println("Please provide all the required arguments: Name, University Roll Number, Course, Semester");
         return;
      }
      String name = args[0];
      int universityRollNo = Integer.parseInt(args[1]);
      String course = args[2];
      int semester = Integer.parseInt(args[3]);
System.out.println("Name: " + name);
System.out.println("University Roll No: " + universityRollNo);
System.out.println("Course: " + course);
System.out.println("Semester: " + semester);
   }
}
```

OUTPUT:

```
PS F:\New folder> cd "f:\New folder\" ; if ($?) { javac CommandLineInput.java } ; if ($?) { java CommandLineInp
ut john 232434354 Btech 3  }
Name: john
University Roll No: 232434354
Course: Btech
Semester: 3
PS F:\New folder>
```

2.Concepts of Java Control statements, Conditional statements , loops and iterations, Wrapper classes, Scanner Class:

Using the switch statement, write a menu-driven program to calculate the maturity amount of a bank deposit.
The user is given the following options:

(i) Term Deposit

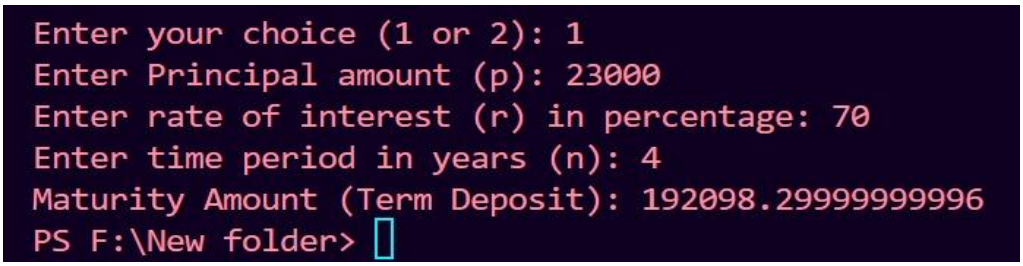
(ii) Recurring Deposit

For option (i) accept Principal (p), rate of interest (r) and time period in years(n). Calculate and output the maturity amount (a) receivable using the formulaa= p[1 + r /100]n.
For option (ii) accept monthly installment (p), rate of interest (r) and time period in months (n). Calculate and output the maturity amount (a) receivable using the formula = p*n+p*n(n+1)/2*r/100*1/12. For an incorrect option, an appropriate error message should be displayed.
[Use Scanner Class to take input]

```
import java.util.Scanner;

public class BankDepositCalculator {

   public static void main(String[] args) {

      Scanner scanner = new Scanner(System.in);

System.out.println("Select the type of deposit:");

System.out.println("1. Term Deposit");

System.out.println("2. Recurring Deposit");

System.out.print("Enter your choice (1 or 2): ");

      int choice = scanner.nextInt();

      switch (choice) {

         case 1:

System.out.print("Enter Principal amount (p): ");

            double principal = scanner.nextDouble();

System.out.print("Enter rate of interest (r) in percentage: ");

            double rate = scanner.nextDouble();

System.out.print("Enter time period in years (n): ");

            double time = scanner.nextDouble();

            double maturityAmountTermDeposit = calculateTermDepositMaturity(principal,
rate, time);
```

```
System.out.println("Maturity Amount (Term Deposit): " + maturityAmountTermDeposit);
            break;
        case 2:
System.out.print("Enter Monthly installment amount (p): ");
            double installment = scanner.nextDouble();
System.out.print("Enter rate of interest (r) in percentage: ");
            rate = scanner.nextDouble();
System.out.print("Enter time period in months (n): ");
            time = scanner.nextDouble();
            double maturityAmountRecurringDeposit =
calculateRecurringDepositMaturity(installment, rate, time);
System.out.println("Maturity Amount (Recurring Deposit): " +
maturityAmountRecurringDeposit);
            break;
        default:
System.out.println("Invalid choice. Please choose 1 or 2.");
    }
scanner.close();
  }
  private static double calculateTermDepositMaturity(double principal, double rate, double
time) {
      return principal * Math.pow((1 + rate / 100), time);
  }
  private static double calculateRecurringDepositMaturity(double installment, double rate,
double time) {
      return installment * time + installment * time * (time + 1) / 2 * rate / 100 * 1 / 12;
  }
}
}
```

OUTPUT:



3. Program to find if the given numbers are Friendly pair or not (Amicable ornot). Friendly Pair are two or more numbers with a common abundance.

Input & Output format:

- **Input consists of 2 integers.**
- **The first integer corresponds to number 1 and the second integer corresponds to number 2.**

If it is a Friendly Pair display Friendly Pair or displays Not Friendly Pair.

For example,6 and 28 are Friendly Pair.
(Sum of divisors of 6)/6 = (Sum of divisors of 28)/28.
Steps to check whether the given numbers are friendly pair or not

- **Input the numbers num1 and num2.**
- **Initialize sum1 = sum2 = 0.**
- **sum1 = sum of all divisors of num1.**
- **sum2 = sum of all divisors of num2.**
- **If (sum1 == num1) and (sum2 == num2), then print "Abundant Numbers".**
- **Else, print "Not Abundant Numbers".**

Program to check whether the given numbers are friendly pair or not.

```
import java.util.Scanner;


public class FriendlyPairChecker {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
System.out.print("Enter number 1: ");
      int num1 = scanner.nextInt();
System.out.print("Enter number 2: ");
      int num2 = scanner.nextInt();


      int sum1 = sumOfDivisors(num1);
      int sum2 = sumOfDivisors(num2);
      if (sum1 == num1 && sum2 == num2) {
System.out.println("Friendly Pair");
      } else {
```

```
System.out.println("Not Friendly Pair");
        }
scanner.close();
    }
    private static int sumOfDivisors(int num) {
        int sum = 1; // 1 is always a divisor
        for (int i = 2; i<= Math.sqrt(num); i++) {
            if (num % i == 0) {
                if (i == (num / i)) {
                    sum += i;
                } else {
                    sum += (i + num / i);
                }
            }
        }
        return sum;
    }
}
```

OUTPUT:

4. Program to replace all 0's with 1 in a given integer. Given an integer as an input,all the 0's in the number has to be replaced with 1.

For example, consider the following number Input: 102405
Output: 112415
Input: 56004
Output: 56114
Steps to replace all 0's with 1 in a given integer

- **Input the integer from the user.**
- **Traverse the integer digit by digit.**
- **If a '0' is encountered, replace it by '1' and Print the integer.**

```
import java.util.Scanner;

public class ReplaceZerosWithOnes {

   public static void main(String[] args) {

      Scanner scanner = new Scanner(System.in);

System.out.print("Enter an integer: ");

      int number = scanner.nextInt();

      String numberString = Integer.toString(number);

      StringBuilder result = new StringBuilder();

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

         char digit = numberString.charAt(i);

         if (digit == '0') {

result.append('1');

         } else {

result.append(digit);

         }

      }

System.out.println("Output: " + result);

scanner.close();

   }

}
```

OUTPUT:

```
PS F:\New folder> cd "f:\New folder\" ; if ($?) { javac ReplaceZerosWithOnes.java } ; if ($?) { java ReplaceZer
osWithOnes }
Enter an integer: 102405
Output: 112415
PS F:\New folder>
```

5. Array in Java:

Printing an array into Zigzag fashion. Suppose youwere given an array of integers, and you are told to sort the integers in a zigzag pattern. In general, ina zigzag pattern, the first integer is less than the second

integer, which is greater than the third integer, which is less than the fourth integer, and so on. Hence, the converted array should be in the form of e1 < e2 > e3 < e4 > e5 < e6. Test cases:

Input 1:7

4 3 7 8 6 2 1

Output 1:

3 7 4 8 2 6 1

Input 2:

4

1 4 3 2

Output 2:

1 4 2 3

```
import java.util.Arrays;

import java.util.Scanner;


public class ZigzagArray {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

System.out.print("Enter the size of the array: ");

        int size = scanner.nextInt();

System.out.println("Enter the array elements:");

int[] array = new int[size];

        for (int i = 0; i< size; i++) {
```

```
            array[i] = scanner.nextInt();
        }
Arrays.sort(array);
        for (int i = 1; i< size - 1; i += 2) {
            int temp = array[i];
            array[i] = array[i + 1];
array[i + 1] = temp;
        }
System.out.println("Output:");
        for (int i = 0; i< size; i++) {
System.out.print(array[i] + " ");
        }
scanner.close();
    }
}
```

OUTPUT:

6. The problem to rearrange positive and negative numbers in an array .

Method: This approach moves all negative numbers to the beginning and positive numbers to the end butchanges the order of appearance of the elements of the array.

Steps:

1. **Declare an array and input the array elements.**
2. **Start traversing the array and if the current element is negative, swap the current elementwith the first positive element and continue traversing until all the elements have been encountered.**
3. **Print the rearranged array.**

Test case:

- **Input: 1 -1 2 -2 3 -3**
- **Output: -1 -2 -3 1 3 2**

```
import java.util.Scanner;


public class RearrangePositiveNegative {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
System.out.print("Enter the size of the array: ");
      int size = scanner.nextInt();
System.out.println("Enter the array elements:");
int[] array = new int[size];
      for (int i = 0; i< size; i++) {
         array[i] = scanner.nextInt();
      }
      int positiveIndex = 0;
      for (int i = 0; i< size; i++) {
         if (array[i] < 0) {
```

```
                int temp = array[i];
                array[i] = array[positiveIndex];
                array[positiveIndex] = temp;
        positiveIndex++;
            }
        }
        System.out.println("Output:");
        for (int i = 0; i< size; i++) {
        System.out.print(array[i] + " ");
        }

        scanner.close();
    }
}
```

OUTPUT:

7.Program to find the saddle point coordinates in a given matrix. A saddle point is an element of the matrix,which is the minimum element in its row and the

maximum in its column.

For example, consider the matrix given belowMat [3][3] 1 2 3

4 5 6

7 8 9

Here, 7 is the saddle point because it is the minimum element in its row and maximum element in its column.

Steps to find the saddle point coordinates in a givenmatrix.

1. **Input the matrix from the user.**
2. **Use two loops, one for traversing the row andthe other for traversing the column.**
3. **If the current element is the minimum element inits row and maximum element in its column, then return its coordinates.**

Else, continue traversing.

```
import java.util.Scanner;


public class SaddlePoint {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
System.out.print("Enter the number of rows: ");
      int rows = scanner.nextInt();
System.out.print("Enter the number of columns: ");
      int columns = scanner.nextInt();
int[][] matrix = new int[rows][columns];
System.out.println("Enter the matrix elements:");
```

```
        for (int i = 0; i< rows; i++) {
            for (int j = 0; j < columns; j++) {
                matrix[i][j] = scanner.nextInt();
            }
        }
        int saddlePointRow = -1;
        int saddlePointColumn = -1;
        for (int i = 0; i< rows; i++) {
            for (int j = 0; j < columns; j++) {
                int currentElement = matrix[i][j];
booleanisSaddlePoint = true;

                for (int k = 0; k < columns; k++) {
                    if (matrix[i][k] <currentElement) {
isSaddlePoint = false;
                        break;
                    }
                }
                for (int k = 0; k < rows; k++) {
                    if (matrix[k][j] >currentElement) {
isSaddlePoint = false;
                        break;
                    }
                }
                if (isSaddlePoint) {
saddlePointRow = i;
saddlePointColumn = j;
                    break;
                }
            }
            if (saddlePointRow != -1 &&saddlePointColumn != -1) {
                break;
```

```
            }
        }
        if (saddlePointRow != -1 &&saddlePointColumn != -1) {
System.out.println("Saddle point found at: (" + saddlePointRow + ", " +
saddlePointColumn + ")");
        } else {
System.out.println("No saddle point found.");
        }


scanner.close();
    }
}
```

OUTPUT:

8. String Handling in Java (using String and String Buffer class):

Program to find all the patterns of 0(1+)0 in the given string. Given a string containing 0's and 1's, find the total number of 0(1+)0 patterns in the string and output it.

0(1+)0 - There should be at least one '1' between the two '0's.Forexample, consider the following string.
Input:01101111010

Output:3
Explanation: 01101111010- count=1

```
public class ZeroOnePattern {
   public static void main(String[] args) {
      String input = "01101111010";
      int count = countZeroOnePatterns(input);
System.out.println("Total number of 0(1+)0 patterns: " + count);
   }
   public static int countZeroOnePatterns(String str) {
      int count = 0;
      int index = 0;
      while (index <str.length())
         index = str.indexOf('0', index);
         if (index == -1) {
            break;
         }
         int oneIndex = str.indexOf('1', index);
         if (oneIndex != -1 &&oneIndex<str.length() - 1 &&str.charAt(oneIndex + 1) == '0') {
            count++;
         }
         index++;
```

```
    }

    return count;

}}
```

OUTPUT:

9. Write a java program to delete vowels from given string using String Buffer class.

```
public class DeleteVowels {
   public static void main(String[] args) {
      String input = "Hello, World!";
      String result = deleteVowels(input);
System.out.println("Original String: " + input);
System.out.println("String after deleting vowels: " + result);
   }
public static String deleteVowels(String str) {
StringBuffersb = new StringBuffer(str);
      for (int i = 0; i<sb.length(); i++) {
         char ch = sb.charAt(i);
         if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u' ||
ch == 'A' || ch == 'E' || ch == 'I' || ch == 'O' || ch == 'U') {
sb.deleteCharAt(i);
i--;
         }
      }
      return sb.toString();
   }
}
```

OUTPUT:

```
PS F:\New folder> cd "f:\New folder\" ; if ($?) { javac DeleteVowels.java } ; if ($?) { java DeleteVowels }
Original String: Hello, World!
String after deleting vowels: Hll, Wrld!
PS F:\New folder>
```

10. Class definition, creating objects and constructors:

Write a java program to create a class named 'Bank ' with the following data members:

- **Name of depositor**
- **Address of depositor**
- **Account Number**
- **Balance in account**

Class 'Bank' has a method for each of the following:

1. **Generate a unique account number for each depositor.**
2. **For first depositor, account number will be 1001, for second depositorit will be 1002 and so on**
3. **Display information and balance of depositor**
4. **Deposit more amount in balance of any depositor**
5. **Withdraw some amount from balance deposited.**

After creating the class, do the following operations.

1. **Enter the information (name, address, account number, balance) of the depositors. Number of depositors is to be entered by the user.**
2. **Print the information of any depositor.**
3. **Add some amount to the account of any depositor and then displayfinal information of that depositor.**
4. **Remove some amount from the account of any. depositor and then display final information of that depositor.**
5. **Change the address of any depositor and then display the finalinformation of that depositor.**
6. **Randomly repeat these processes for some other bank accounts.**

Change address of depositorpublic class Bank {

```
  private static int nextAccountNumber = 1001;


  private String nameOfDepositor;

  private String addressOfDepositor;

  private int accountNumber;

  private double balance;

  public Bank(String name, String address, double initialBalance) {
```

```
this.nameOfDepositor = name;
this.addressOfDepositor = address;
this.accountNumber = nextAccountNumber++;
this.balance = initialBalance;
    }
    public void displayInfoAndBalance() {
System.out.println("Name: " + nameOfDepositor);
System.out.println("Address: " + addressOfDepositor);
System.out.println("Account Number: " + accountNumber);
System.out.println("Balance: $" + balance);
    }
    public void deposit(double amount) {
        balance += amount;
System.out.println("$" + amount + " deposited successfully.");
    }
    public void withdraw(double amount) {
        if (balance >= amount) {
            balance -= amount;
System.out.println("$" + amount + " withdrawn successfully.");
        } else {
System.out.println("Insufficient balance.");
        }
    }
    public void changeAddress(String newAddress) {
addressOfDepositor = newAddress;
System.out.println("Address changed successfully.");
    }

    public static void main(String[] args) {
        Bank depositor1 = new Bank("John Doe", "123 Main St", 1000);
        depositor1.displayInfoAndBalance();
```

```
        depositor1.deposit(500);
        depositor1.displayInfoAndBalance();

        depositor1.withdraw(200);
        depositor1.displayInfoAndBalance();

        depositor1.changeAddress("456 Elm St");
        depositor1.displayInfoAndBalance();

        Bank depositor2 = new Bank("Alice Smith", "789 Oak St", 2000);
        depositor2.displayInfoAndBalance();
    }
}
```

OUTPUT:

```
PS F:\New folder> cd "f:\New folder\" ; if
Name: John Doe
Address: 123 Main St
Account Number: 1001
Balance: $1000.0
$500.0 deposited successfully.
Name: John Doe
Address: 123 Main St
Account Number: 1001
Balance: $1500.0
$200.0 withdrawn successfully.
Name: John Doe
Address: 123 Main St
Account Number: 1001
Balance: $1300.0
Address changed successfully.
Name: John Doe
Address: 456 Elm St
Account Number: 1001
Balance: $1300.0
```

11. Define a class Word Example having the following description:Data members/instance variables:

private String str data: to store a sentence.

Parameterized Constructor WordExample(String) : Accept asentence

which may be terminated by either'.', '? 'or'!' only. The words may be

separated bymore than one blank space and are in UPPER CASE.

Member Methods:

void count Word(): Find the number of words beginning andending with a vowel.

void place Word(): Place the words which begin and end with a vowel at thebeginning, followed by the remaining words as they occur in the sentence

```
public class WordExample {
   private String strdata;
   public WordExample(String sentence) {
this.strdata = sentence;
   }
   public void countWord() {
String[] words = strdata.split("\\s+");
      int count = 0;
for (String word : words) {
         if (word.length() > 0 &&isVowel(word.charAt(0))
&&isVowel(word.charAt(word.length() - 1))) {
            count++;
         }
      }
System.out.println("Number of words beginning and ending with a vowel: " + count);
   }
```

```
   public void placeWord() {
String[] words = strdata.split("\\s+");
      StringBuilder result = new StringBuilder();
      for (String word : words) {
         if (word.length() > 0 &&isVowel(word.charAt(0))
&&isVowel(word.charAt(word.length() - 1))) {
result.insert(0, word + " ");
         }
      }
      for (String word : words) {
         if (!(word.length() > 0 &&isVowel(word.charAt(0))
&&isVowel(word.charAt(word.length() - 1)))) {
result.append(word).append(" ");
         }
      }
System.out.println("Resulting sentence: " + result.toString());
   }
   private booleanisVowel(char ch) {
      return "AEIOU".indexOf(Character.toUpperCase(ch)) != -1;
   }
   public static void main(String[] args) {
WordExample example = new WordExample("THE QUICK BROWN FOX JUMPS OVER THE
LAZY DOG.");
example.countWord();
example.placeWord();
   }
}
```

OUTPUT:

```
PS F:\New folder> cd "f:\New folder\" ; if ($?) { javac WordExample.java } ; if ($?) { java WordExample }
Number of words beginning and ending with a vowel: 0
Resulting sentence: THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG.
PS F:\New folder>
```

12. Method overloading (Compile time Polymorphism):Write a Java program to create a class called Array Demo and overload array Func () function.

void array Func (int [], int) To find all pairs of elements in anArray whose sum is equal to a given number :

Array numbers= [4, 6, 5, -10, 8, 5, 20], target=10
Output :

Pairs of elements whose sum is 10 are :4 + 6 = 10
5 + 5 = 10

-10 + 20 = 10

void array Func(int A[], int p, int B[], int q) Given two sorted arrays Aand B of size p and q, Overload method array Func() to merge elements of A with B by maintaining the sorted order i.e. fill A with first p smallest elements and fill B with remaining elements.

Example:Input :

int[] A = { 1, 5, 6, 7, 8, 10 }

int[] B = { 2, 4, 9 }

Output:

Sorted Arrays:

A: [1, 2, 4, 5, 6, 7]

B: [8, 9, 10]

(Use Compile time Polymorphism Method Overloading)

```
import java.util.Arrays;

public class ArrayDemo {

   public void arrayFunc(int[] numbers, int target) {

System.out.println("Pairs of elements whose sum is " + target + " are:");

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

         for (int j = i + 1; j <numbers.length; j++) {

            if (numbers[i] + numbers[j] == target) {
```

```
System.out.println(numbers[i] + " + " + numbers[j] + " = " + target);
            }
         }
      }
   }
   public void arrayFunc(int[] A, int p, int[] B, int q) {
int[] mergedArray = new int[p + q];
      int i = 0, j = 0, k = 0;
      while (i< p && j < q) {
         if (A[i] < B[j]) {
mergedArray[k++] = A[i++];
         } else {
mergedArray[k++] = B[j++];
         }
      }
      while (i< p) {
mergedArray[k++] = A[i++];
      }
      while (j < q) {
mergedArray[k++] = B[j++];
      }
System.arraycopy(mergedArray, 0, A, 0, p);
System.arraycopy(mergedArray, p, B, 0, q);
Arrays.sort(A);
Arrays.sort(B);
System.out.println("Sorted Arrays:");
System.out.println("A: " + Arrays.toString(A));
System.out.println("B: " + Arrays.toString(B));
   }
   public static void main(String[] args) {
ArrayDemo demo = new ArrayDemo();
int[] numbers = {4, 6, 5, -10, 8, 5, 20};
```

```
        int target = 10;
demo.arrayFunc(numbers, target);
int[] A = {1, 5, 6, 7, 8, 10};
int[] B = {2, 4, 9};
demo.arrayFunc(A, A.length, B, B.length);
    }
}
```

OUTPUT:

```
C:\Users\asus>cd Desktop

C:\Users\asus\Desktop>javac ArrayDemo.java

C:\Users\asus\Desktop>java ArrayDemo
Pairs of elements whose sum is 10 are:
4 + 6 = 10
5 + 5 = 10
-10 + 20 = 10
Sorted Arrays:
A: [1, 2, 4, 5, 6, 7]
B: [8, 9, 10]
```

13. Method overriding (Runtime Polymorphism), Abstract class and Abstractmethod, Inheritance, interfaces:

Write a java program to calculate the area of a rectangle, a square and a circle.Create an abstract class 'Shape' with three abstract methods namely rectangleArea() taking two parameters, squareArea() and circleArea() taking one parameter each.

Now create another class 'Area' containing all the three methods rectangleArea(), squareArea() and circleArea() for printing the area of rectangle, square and circle respectively. Create an object of class Area and callall the three methods.

(Use Runtime Polymorphism).

```
abstract class Shape {
   abstract double rectangleArea(double length, double width);
   abstract double squareArea(double side);
   abstract double circleArea(double radius);
}
class Area extends Shape {
   double rectangleArea(double length, double width) {
      return length * width;
   }
   double squareArea(double side) {
      return side * side;
   }
   double circleArea(double radius) {
      return Math.PI * radius * radius;
   }
}
```

```
public class Main {
    public static void main(String[] args) {
        Area area = new Area();
        double rectangleArea = area.rectangleArea(5, 3);
System.out.println("Area of rectangle: " + rectangleArea);
        double squareArea = area.squareArea(4);
System.out.println("Area of square: " + squareArea);
        double circleArea = area.circleArea(2.5);
System.out.println("Area of circle: " + circleArea);
    }
}
```

OUTPUT:

14. Write a java program to implement abstract class andabstract method with following details:

Create a abstract Base Class TemperatureData members:
double temp;
Method members:

void setTempData(double) abstact void changeTemp()

Sub Class Fahrenheit (subclass of Temperature) Data members:double

ctemp;

method member:

Override abstract method changeTemp() to convert Fahrenheit temperature intodegree Celsius by using formula C=5/9*(F-32) and display converted temperature

Sub Class Celsius (subclass of Temperature)

Data member:

double ftemp;

Method member:

Override abstract method changeTemp() to convert degree Celsius into Fahrenheit temperature by using formula F=9/5*c+32 and display converted temperature.

```
abstract class Temperature {
   double temp;
   abstract void setTempData(double temp);
   abstract void changeTemp();
}
class Fahrenheit extends Temperature {
   double ctemp;
```

```
   void setTempData(double temp) {
this.ctemp = temp;
   }
   void changeTemp() {
      double celsius = (5.0 / 9.0) * (ctemp - 32);
System.out.println("Temperature in Celsius: " + celsius);
   }
}
class Celsius extends Temperature {
   double ftemp;
   void setTempData(double temp) {
this.ftemp = temp;
   }
   void changeTemp() {
      double fahrenheit = (9.0 / 5.0) * ftemp + 32;
System.out.println("Temperature in Fahrenheit: " + fahrenheit);
   }
}

public class Main {
   public static void main(String[] args) {
      Fahrenheit fahrenheit = new Fahrenheit();
      Celsius celsius = new Celsius();
fahrenheit.setTempData(98.6);
celsius.setTempData(37);
System.out.println("Fahrenheit to Celsius conversion:");
fahrenheit.changeTemp();
System.out.println("\nCelsius to Fahrenheit conversion:");
celsius.changeTemp();
   }
}
```

OUTPUT:

```
PS F:\New folder> cd "f:\New folder\" ; if ($?) { javac Main2.java } ; if ($?) { java Main2 }
Fahrenheit to Celsius conversion:
Temperature in Celsius: 37.0

Celsius to Fahrenheit conversion:
Temperature in Fahrenheit: 98.60000000000001
PS F:\New folder>
```

15. Write a java program to create an interface that consists of a method to display volume () as an abstract method and redefine this method in the derived classes to suit their requirements.
Create classes called Cone, Hemisphere and Cylinder that implements the interface.
Using these three classes, design a program that will accept dimensions of a cone, Cylinderr and hemisphere interactively and display the volumes.

Volume of cone = (1/3)πr2h Volume of hemisphere = (2/3)πr3Volume of cylinder = πr2h

```
import java.util.Scanner;
interface Volume {
   double calculateVolume();
}
class Cone implements Volume {
   private double radius;
   private double height;
   public Cone(double radius, double height) {
this.radius = radius;
this.height = height;
   }
   public double calculateVolume() {
      return (1.0 / 3.0) * Math.PI * radius * radius * height;
   }
}
class Hemisphere implements Volume {
```

```
    private double radius;
    public Hemisphere(double radius) {
this.radius = radius;
    }
    public double calculateVolume() {
        return (2.0 / 3.0) * Math.PI * Math.pow(radius, 3);
    }
}

class Cylinder implements Volume {
    private double radius;
    private double height;
    public Cylinder(double radius, double height) {
this.radius = radius;
this.height = height;
    }
    public double calculateVolume() {
        return Math.PI * radius * radius * height;
    }
}
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
System.out.print("Enter radius of Cone: ");
        double coneRadius = scanner.nextDouble();
System.out.print("Enter height of Cone: ");
        double coneHeight = scanner.nextDouble();
        Cone cone = new Cone(coneRadius, coneHeight);
System.out.println("Volume of Cone: " + cone.calculateVolume());
System.out.print("\nEnter radius of Hemisphere: ");
        double hemisphereRadius = scanner.nextDouble();
        Hemisphere hemisphere = new Hemisphere(hemisphereRadius);
```

```
System.out.println("Volume of Hemisphere: " + hemisphere.calculateVolume());

System.out.print("\nEnter radius of Cylinder: ");
        double cylinderRadius = scanner.nextDouble();
System.out.print("Enter height of Cylinder: ");
        double cylinderHeight = scanner.nextDouble();
        Cylinder cylinder = new Cylinder(cylinderRadius, cylinderHeight);
System.out.println("Volume of Cylinder: " + cylinder.calculateVolume());
scanner.close();
    }
}
```

OUTPUT: