Practical No : 07

Write a java program that prompts the user for an integer and then print out all the prime numbers up to that Integer:

```
import java.util.Scanner;

class PrintPrime {
   int n;

   void getData() {
       Scanner sc = new Scanner(System.in);
       System.out.println("Enter Values: ");
       n = sc.nextInt();
   }

   void showData() {
       System.out.print("Prime numbers up to " + n + ":
");
       for (int i = 2; i <= n; i++) {
           int count = 0;
           for (int j = 2; j * j <= i; j++) {
               if (i % j == 0) {
                   count++;
                   break;
               }
           }
           if (count == 0) {
               System.out.print(i + " ");
           }
       }
       System.out.println();
```

```
    }

    public static void main(String[] args) {
        PrintPrime call = new PrintPrime();
        call.getData();
        call.showData();
    }
}
```

Output:

Enter Values:
10
Prime numbers up to 10: 2 3 5 7

Practical No : 09

Write a java program for sorting a given list of names in ascending order.

```
import java.util.*;
import java.lang.*;
public class SortNames {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter the number of names: ");
        int n = scanner.nextInt();
        scanner.nextLine();

        String names[] = new String[n];
```

```
        System.out.println("Enter the names:");
        for (int i = 0; i < n; i++) {
            names[i] = scanner.nextLine();
        }

        for (int i = 0; i < n; i++) {
            for (int j = i + 1; j < n; j++) {
                if (names[i].compareTo(names[j]) > 0) {
                    String temp = names[i];
                    names[i] = names[j];
                    names[j] = temp;
                }
            }
        }

        System.out.println("Names in alphabetical
order:");
        for (int i = 0; i < n; i++) {
            System.out.println(names[i]);
        }
    }
}
```

Output:

Enter the number of names: *3*

Enter the names: *Ram Shyam Abhi*

Names in alphabetical order:

Abhi Ram Shyam

Practical No : 18

Write a java program that checks whether a given string is a palindrome or not. *Ex:* **MADAM** *is a palindrome.*

```
import java.lang.*;

public class Palindrome {
   public static void main(String[] args) {
        String str = "MADAM", reverseStr = "";

        int strLength = str.length();
        for (int i = (strLength - 1); i >= 0; --i) {
            reverseStr = reverseStr + str.charAt(i);
        }
        if
(str.toLowerCase().equals(reverseStr.toLowerCase())) {
            System.out.println(str + " is a Palindrome
String");
        } else {
            System.out.println(str + " is not a Palindrome
String");
        }
   }
}
```

Output:

MADAM is a Palindrome String

Practical No : 08

Write a java program to multiply two given matrices.

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

class MatrixMultiplication {
   int row,cols;
   int arr1[][];
   int arr2[][];
   int arr3[][];

   void getData(){
       Scanner sc = new Scanner(System.in);

       System.out.println("Enter Size Of Rows: ");
       row = sc.nextInt();
       System.out.println("Enter Size Of Column: ");
       cols = sc.nextInt();
       arr1 = new int [row][cols];
       arr2 = new int [row][cols];
       arr3 = new int [row][cols];

       System.out.println("Enter "+row+" Rows & "+cols+" Column
For First Matrix: ");
       for (int i = 0; i < row; i++ ){
           for(int j = 0; j < cols; j++){
               arr1[i][j] = sc.nextInt();
           }
       }

       System.out.println("Enter "+row+" Rows & "+cols+" Column
For Second Matrix: ");
       for (int i = 0; i < row; i++ ){
           for(int j = 0; j < cols; j++){
               arr2[i][j] = sc.nextInt();;
           }
       }
   }

   void calculation() {
```

```
        for (int i = 0; i < row; i++) {
            for (int j = 0; j < cols; j++) {
                arr3[i][j] = 0;
                for (int k = 0; k < cols; k++) {
                    arr3[i][j] += arr1[i][k] * arr2[k][j];
                }
            }
        }
    }

    void show(){
        for (int i = 0; i < row ; i++ ){
            for(int j = 0; j < cols; j++){
                System.out.print(+arr3[i][j]+" ");
            }
            System.out.println();
        }
    }

    public static void main(String[] args){
        MatrixMultiplication call = new MatrixMultiplication();
        call.getData();
        call.calculation();
        call.show();
    }
}
```

Output:

Enter Size Of Rows: 2

Enter Size Of Column: 2

Enter 2 Rows & 2 Column For First Matrix:
1 2 3 4
Enter 2 Rows & 2 Column For Second Matrix:
5 6 7 8

19 22
43 50

Practical No 10

Write a java program for Method overloading and constructor overloading.

```
import java.lang.*;

class EmployeeD{
   int id;
   String name;
   float salary;

   // Default Constructor
   public EmployeeD(){
       System.out.println("Default Constructor!");
   }

   // Parameterized Constructor
   public EmployeeD(int i, String n, float s){
       this.id = i;
       this.name = n;
       this.salary = s;
   }

   // Method Overloading
   public void printDetails() {
       System.out.println("No details provided");
   }
```

```
    public void printDetails(int id, String name, float
salary) {
        System.out.println("Employee ID: " + id + ",
Name: " + name + ", Salary: " + salary);
    }
}

public class ConstructorOverloading {
    public static void main(String[] args){
        EmployeeD call1 = new EmployeeD();
        EmployeeD call = new EmployeeD(1, "Shyam",
4999);

        call1.printDetails();
        call.printDetails(call.id, call.name,
call.salary);
    }
}
```

Output:

Default Constructor!
No details provided
Employee ID: 1
Name: Shyam
Salary: 4999.0

Practical No : 11

Write a java Program to represent Abstract class with example:

```java
import java.lang.*;
Import java.util.*;

abstract class Shape{
   abstract double area();
   void displayArea(){
       System.out.println("Area: "+area());
   }
}
class Circle extends Shape{
   double radius;
   Circle(double radius){
       this.radius = radius;
   }
   double area(){
       return Math.PI * radius * radius;
   }
}

class Rectangle extends Shape{

   double length, width;
   Rectangle(double length, double width){
       this.length = length;
       this.width = width;
   }

   double area(){
       return length*width;

   }
}

public class Abstract{
```

```
    public static void main(String[] args){
        Shape circle = new Circle(5);
        Shape rectangle = new Rectangle(4,6);
        circle.displayArea();
        rectangle.displayArea();
    }
}
```

Output:

Area: 78.53981633974483
Area: 24.0

Practical No: 12

Write a program to implement multiple inheritance:

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

interface Flyable {
    void fly();
}

interface Swimmable {
    void swim();
}

interface Quackable {
    void quack();
}
```

```java
class Duck implements Flyable, Swimmable, Quackable {
    @Override
    public void fly() {
        System.out.println("Duck is flying!");
    }

    @Override
    public void swim() {
        System.out.println("Duck is swimming!");
    }

    @Override
    public void quack() {
        System.out.println("Duck is quacking!");
    }
}

public class MultipleInheritance {
    public static void main(String[] args) {
        Duck duck = new Duck();
        duck.fly();
        duck.swim();
        duck.quack();
    }
}
```

Output:

```
Duck is flying!
Duck is swimming!
Duck is quacking!
```

Practical No : 13

Write a program to demonstrate method overriding and super keywords.

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

class Animal{
    void makeSound(){
        System.out.println("Animals Makes a Sound!");
    }
}

class Dog extends Animal{
    @Override

    void makeSound(){
        super.makeSound();
        System.out.println("Dog Barks!");
    }
}

public class Overriding {
    public static void main(String[] args) {
        Animal obj = new Animal();
        Dog obj1 = new Dog();
        obj.makeSound();
        obj1.makeSound();
    }
}
```

Output:

Animals Makes a Sound!
Animals Makes a Sound!
Dog Barks!