Name : Pratyay Dhond

Department : IT

Roll No : 1907011

Assignment 1 (Unit-1)

Q. 1) Write a Java program to check whether the number is prime or not.

→

Code :

```
import java.util.Scanner;

class Q1{

    public static boolean isPrime(int no){
        for(int i=2; i<(no/2+1) ; i++){
            if(no%i==0)
                return false;
        }
        return true;
    }

    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        int no = sc.nextInt();

        boolean result = isPrime(no);

        System.out.println("The number " + no +" is "
            + (result ? "prime" : " not prime"));
    }
}
```

Output :

CMD

```
javac Q1.java
java Q1
Enter a number : 37
The number 37 is prime
```

CMD

```
java Q1
Enter a number : 6
The number 6 is not prime.
```

Q2. Write a java program to generate a ladder of numbers.

→

Code :

```
import java.util.Scanner;
class Q2 {
    public static void main(String[] args){
```

```
Scanner sc = new Scanner(System.in);
System.out.print("Enter number of rows :");
int n = sc.nextInt();

    for(int i=1 ; i<=n; i++){
        for(int j=1; j<=i; j++){
            System.out.print(j+ " ");
        }
        System.out.println();
    }
}
```

Output:

CMD

```
java Q2
Enter number of rows : 5
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

Q3) Write a java programs which will read a line of integers and displays each integer and the sum of all integers using ~~Strik~~ String Tokenizer.

→

Code:

```
import java.util.Scanner;
import java.util.StringTokenizer;

class Q3 {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);
        System.out.print("Input number to add in single
                line");
        String str = sc.nextLine();

        StringTokenizer st = new StringTokenizer(str);
        int sum = 0;

        System.out.println("Input : ");

        while(st.hasMoreTokens()) {
            int temp = Integer.parseInt(st.nextToken());
            System.out.format("%6d\n", temp);
            sum += temp;
        }
```

```
        System.out.println("Sum of numbers : " + sum);
    }
}
```

Output:

CMD

```
javac Q.3.java
java Q3

Input numbers to add in a single line : 16 32 64 128 256 512 1024 2048
Input :
16
32
64
128
256
512
1024
2048
Sum of numbers : 4080
```

Q.4) Write a program to calculate the following:

1. Find the length of array.
2. Demonstrate a ~~one~~ one-dimensional array
3. Demonstrate a Two dimensional array.

→

Code:

```
import java.util.Scanner;

class Q4{

    public static void main(String[]args){

        int[] arr = {15,30,45,60,75,90,105,120};
        int length = arr.length;
        System.out.println("Length of array" + length);
        System.out.println("Demonstrate one-dimensional
          array");
        for(int i=0; i<length; i++){
          System.out.print (arr[i] + " ");
        }

        System.out.println(" Demonstrating two dimensional
          array : ");
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter number of rows :");
        int rows = sc.nextInt();
```

```
System.out.print("Enter number of columns : ");
int column = sc.nextInt();
int[][] matrix = new int[rows][column];

    for(int i= 0; i<rows; i++){
    : for(int j=0; j<column; j++){
        System.out.print("Enter dm element at position
            [" + (i+1) + "][" + (j+1) + "] : ");
        matrix[i][j] = sc.nextInt();
      }
    }

      System.out.println(" Printing Two-dimensional array ");
        for(int i=0; i<rows; i++){
          for(int j=0; j<columns; j++){
            System.out.print(" matrix[i][j] + " ");
          }
        System.out.println();
      }
    }
}
```

Output : [CMD]

```
javac Q4.java
java Q4

length of array : 8
Demonstrate-one dimensional array:
15 30 45 60 75 90 105 120

Demonstrating two-dimensional array
Enter ~~two~~ dim
Enter no. of rows :
3
Enter number of columns :
3
Enter elements at position [1][1] : 1
Enter elements at position [1][2] : 2
Enter elements at position [1][3] : 3
Enter elements at position [2][1] : 4
Enter elements at position [2][2] : 5
Enter elements at position [2][3] : 6
Enter elements at position [3][1] : 7
Enter elements at position [3][2] : 8
Enter elements at position [3][3] : 9
```

Printing two dimensional array:

1 2 3
4 5 6
7 8 9

Q5) Write a java program to display strings is sorted order.

Code:

```
import java.util.Scanner;
import java.util.Arrays;

class Q5{

public static void main(String[] args){
 Scanner sc = new Scanner(System.in);
 System.out.println("Enter five words : ");
 String[] arr = new String[5];

for(int i=0; i<5; i++){
 System.out.print("Enter word numno. " +(i+1)
  + " : ");
 arr[i] = sc.next();
}
```

```
System.out.println("Unsorted Array :");
for(int i = 0; i< arr.length ; i++) {
    System.out.printf format("%s\n", arr[i]);
  }

Arrays.sort(Arr arr);

System.out.println("Sorted Array :");
 for(int i =0; i< arr.length ; i++) {
    System.out.format("%s\n", arr[i]);
  }
 }
}
```

Output :

CMD

```
Enter five words :
Enter word no. 1 : India
Enter word no. 2 : Australia
Enter word no. 3 : England
Enter word no. 4 : America
Enter word no. 5 : Brazil

Unsorted Array :
India
Australia
England
America
```

Brazil

Sorted Array :

America
Australia
Brazil
England
India

Q.6) Explain in detail Scanner class, BufferedReader and Console Class with all its methods and usage.

→

Scanner Class :

- The Scanner class is used to get user input, and it is found in the java.util package.
- To use the Scanner class, created an object of the class and use any of the available methods found in the Scanner class documentation.
- Various different functions of Scanner class are :
  - next()
  - nextline()
  - nextInt()

- nextLong()
- nextFloat()
- nextDouble()

- next() - It finds and returns the next complete token from the Scanner.
- nextLine() - It moves the Scanner position to next Line and returns input as a string.
- nextInt() - It scans the next token ~~as a~~ of input as integer.
- nextLong() - It scans the next token of input as Long.
- nextFloat() - It scans the next token ~~input~~ of input as Float.
- nextDouble() - It scans the next token of input as double.

2> BufferedReader class:

- The java.io package contains BufferedReader class that takes input as an argument that is an object of InputStreamReader.
- For input the function - readLine() is used.
- This readLine function might generate an I/O Exception. So, we write it in try catch block.

```
InputStreamReader isr = new InputStreamReader
                        (System.in);
```

```
BufferedReader br = new BufferedReader(isr);
```

3) Use System.Console class :

- The console class is used to get input from console.
- It provides methods to read texts and password. If you read password using Console class, it will not be displayed to the user.
- Syntax,

```
String name = System.Console.ReadLine();
String password = System.Console.readPassword();
```

- The console is used to get input from the command line prompt.

Assignment No. 02 (Unit 2)

Q1) Create a class named Matrix which will have each of the following :

1- get the number of rows
2- get the number of columns
3- Set the elements of the matrix at given position (i,j)
4- Adding two matrices. If the matrices are not addable, "Matrices cannot be added", will be displayed.
5- Multiplying the two matrices.

→

```
import java.util.Scanner;

class Matrix {
    int rows;
    int columns;
    int [][] matrix;
    int[][] product;
    int[][] sum;

    Final Scanner sc = new Scanner(System.in);

    void getRows() {
        rows = sc.nextInt();
    }

    vo Matrix(){
        getRows();
        getColumns();
```

```
matrix = new int [rows] [columns];
Sum = new int [rows] [columns];
product = new int [rows] [columns];

Scanner sc = new Scanner (System.in);

for(int i=0; i< rows; i++){
    for (int j=0; j< columns; j++){
        System.out.println(" Enter element at index [" 
            + i + "][" +j + "] : ");
        matrix [i][j] = sc.nextInt();
    }
}
}

void getColumns(){
    System.out.println(" Enter number of columns :");
    columns = sc.nextInt();
}

void setAt(int i, int j, int value){
    matrix [i][j] = value;
}

void add (Matrix a){
    if(this.rows == a.rows && this.columns == a.columns){
        for(int i=0; i< this.rows; i++){
```

```
for(int j=0; j<columns; j++){
    sum[i][j] = this.matrix[i][j] + a.matrix[i][j];
  }
 }
 System.out.println("Addition result :");
 display(sum);
}else{
 System.out.println("Matrix cannot be added");
}
}

void multiply(Matrix a){

 for(int i=0; i<rows; i++){
  for(int j=0; j<columns; j++){
   try{{
    product[i][j] = matrix[i][j]
   for(int k=0; k<columns; k++){
    try{
     product[i][j] = matrix[i][j] *
                     a.matrix[k][j];
    }catch(Exception e){
    }finally{
     continue;
    }
   }
  }
 }
```

```
        }
    display(product);
}

void display(int[][] a){
    For(int i=0; i< matrix.length ;i++){
        For(int j=0; j< matrix[i].length; j++){
            System.out.println();
            System.out.print(a[i][j] +"  ");
        }
        System.out.println();
    }
}
}

class Q1{

    public static void main(String[] args){
        System.out.println("Enter Matrix 1 : ");
        Matrix m1 = new Matrix();
        System.out.println("Enter Matrix 2 : ");
        Matrix m2 = new Matrix();

        m1.add(m2);
        m1.multiply(m2);
        System.out.println(" Matrix after setting element at
            given position:");
```

```
        m1.setAt(0,0,11);
        m1.display(m1.matrix);
    }
}
```

Output : CMD

```
javac Q1.java
java Q1

Enter Matrix 1 :
Enter number of rows : 2
Enter number of columns : 2
Enter element at index [0][0] : 1
Enter element at index [0][1] : 2
Enter element at index [1][0] : 3
Enter element at index [1][2] : 4
Enter Matrix 2 :
Enter number of rows : 2
Enter number of columns : 2
Enter element at index [0][0] : 5
Enter element at index [0][1] : 6
Enter element at index [1][0] : 7
Enter element at index [1][1] : 8
Addition Result :
 6  8
10 12
```

Multiplication Result :

14 16

28 32

Matrix after setting element at given position :

11 2

3 4

Q2) Write a program to ~~write~~ print the area and perimeter of a triangle having sides of 3, 4, and 5 units by creating a class named triangle without any parameters in its constructor.

→

Code :

```
class Triangle {
    Triangle () {
        area (3,4,5);
        peri (3,4,5);
    }
    void area (int a, int b, int c ) {
        float s = (a+b+c)/2.0;
        Float a = (Float) Math.sqrt (s*(s-a)*(s-b)*(s-c));
        System.out.println (" Area of triangle ": " + A +
                "Units ");
    }
```

```
void peri(int a, int b, int c){
    System.out.println("Perimeter of triangle" + (a+b+c)
        + " Units");
    }
}

class Q2 {

    public static void main(String[] args){
        Triangle t = new Triangle();
    }
}
```

Output :

```
javac Q2.java
java Q2
Area of triangle : 6.0 units
Area of
Perimeter of triangle : 12 units.
```

Q.3) Create a class 'Complex' with seperate methods for each operation whose real and imaginary ~~roots~~ ~~are~~ parts are entered by user. WAP which prints the sum, difference and product of two complex numbers.

→

Code:

```
import java.util.Scanner;

class Complex {
    int a,b;

    Complex(){
        Scanner sc = new Scanner(System.in){
        System.out.println(" Enter the real part of equation:"
                    );
          a = sc.nextInt();
        System.out.println("Enter the imaginary part of
                  equation : ");
          b= sc.nextInt();
    }

        void add (Complex c1){
          System.out.println ("\n Addition: \n");
        System.out.println(" "+ a "+" + b + "j");
        System.out.println ("+ "+ c1.a " + " + c1.b + "j");
        System.out.println ("------------------------------");
        System.out.println (" "+(a+c1.a) +" +" (b+c1.b)+"j");
        System.out.println ("------------------------------");
    }

    void subtract (Complex c1){
        System.out.println("\n Subtraction : \n");
        System.out.println(" "+a "+" +b+"j ");
```

```
        System.out.println("- " + c1.a + " + " + c1.b + " j");
        System.out.println("----------");
        System.out.println("  " + (a - c1.a) + "  " + (b - c1.b) + " j");
        System.out.println("----------");
    }

    void Multiply (Complex c1) {
        System.out.println("\n Multiplication : \n");
        System.out.println("  " + a + " + " + b + " j ");
        System.out.println("x " + c1.a + " + " + c1.b + " j ");
        System.out.println("----------");
        System.out.println(" : " + (a x c1.a) + " + " + (b x c1.b) + " j");
        System.out.println("----------");
    }
}

class Q3 {
    public static void main (String[] args) {
        Complex c1 = new Complex();
        Complex c2 = new Complex();

        c1.add(c2);
        c1.subtract(c2);
        c1.multiply(c2);
    }
}
```

Output : CMD

```
javac Q3.java
java Q3

Enter the real part of equation:
1
Enter the imaginary part of equation:
2
Enter the real part of equation:
2
Enter the imaginary part of equation.

Addition:

  1 + 2j
+ 2 + 2j
  3 + 4j

Subtraction:

  1 + 2j
- 2 + 2j
 -1 + 0j
```

Multiplication :-

$$\begin{array}{r} 1 + 2j \\ \times\ 2 + 2j \\ \hline 2 + 4j \end{array}$$

Q4) Write a program to create a room class and display data, the attributes of this class is roomno, roomtype, roomarea, and ACMachine. In this class the methods are setdata and displaydata.

→ Code:

```
Class Room{
    int roomNo;
    String roomType;
    Float roomArea;
    boolean AC machine;

void setdata(int roomNo, String roomType, Float roomArea
, boolean ACMachine){
    this.roomNo = roomNo;
    this.roomType = roomType;
    this.roomArea = roomArea;
    this.ACmachine = ACMachine;
}
```

```
void displayData() {

    System.out.println("room no      : " + roomNo);
    System.out.println("Room type    : " + roomType);
    System.out.println("Room area    : " + roomArea);
    System.out.println("AC facility  : " + (ACMachine ?
                        "Available" : "Unavailable"));
    }
}

class Q4 {
    public static void main(String[] args) {
        Room r1 = new Room();
        r1.setData(101, "Deluxe Suite", 250.56F, true);
        r1.displayData();
    }
}
```

Output :

CMD :

```
javac Q4.java
java Q4
Room no      : 101
Room type    : Deluxe Suite
Room Area    : 250.56
AC facility  : Available
```

Q5) Write a program to demonstrate the use of static, this and final keyword.

→

code :

```
final class Student {
    final String mathsTeacher = "Laxmi ma'am";
    static String name;
    static int rollNo;

    void setData(String name, int rollNo) {
        this.name   = name;
        this.rollNo = rollNo;
    }
}

class Q5 {
    public static void display (Student Q) {
        System.out.println("Name      : " + Q.name);
        System.out.println("Roll No   : " + Q.rollNo);
        System.out.println("Maths     : " + Q.mathsTeacher);
    }

    public static void main(String[] args) {
        Student student = new Student();
        student.setData("Pratyay Dhond", 1907011);
        display(student);
    }
}
```

Output :

CMD

```
javac Q5.java
java Q5

Name : Pratyay Dhond
Roll No : 1907011
Maths : : Laxmi Ma'am
```

Assignment No. 03 (Unit 3)

Q1) Write a Java program to calculate the: area of rectangle, circle and triangle using the concept of abstract class.

→

Code:

```
package com.Company;

abstract class Area {
    final float PI = 3.14F;
    float half final float half = 0.5F;
    abstract float area(float a, float b);
}

class Rectangle extends Area {
    float area(float length, float breadth) {
        return length * breadth;
    }
}

class Circle extends Area {
    float area (float radius, float notRequired) {
        return PI * radius * radius;
    }
}

Class Triangle extends Area {
    float area (float height, float base) {
        return half
```

```
        return half * base * height;
    }
}

class Q1 {
    public static void main(String[] args) {
        Rectangle r1 = new Rectangle();
        System.out.println("Area of Rectangle : " +
                r1.area(10,20));
        Circle c1 = new Circle(10,0);
        System.out.println("Area of Circle : " +
                c1.area(10,0));
        Triangle t1 = new Triangle();
        System.out.println("Area of Triangle :" +
                t1.area(10,20));
    }
}
```

Output : CMD

```
javac   Q1.java
java    Q1
Area of Rectangle : 200.0
Area of Circle    : 314.0
Area of Triangle  : 100.0
```

Q2> Define an interface. Write a program which illustrates the design and implementation of an interface.

→

Interface : " Interfaces are basically a class which can have methods and variable with a major difference being that ~~its~~ interfaces define only abstract methods and final fields (data members)."

Code :

```
interface Employee {
    final static int diwaliBonus = 20000;
    int
    ~~float~~ compute(float salary);
}

class Programmer implements Employee {
    int
    float compute(float salary) {
        return salary + diwaliBonus;
    }
}

~~pubh class~~
class Q2 {
    public static void main(String[] args){
        Programmer p = new Programmer();
        System.out.println("Salary : " + p.compute(50000));
    }
```

Output :

CMD

```
javaC Q2.java
java Q2

Salary : 70000
```

Q.3) a) What is a package? Explain member access privilages.

→

Package - " Packages are Java language's way of grouping a variety of classes and/or interfaces together. Java packages act as "containers" for classes. Java packages are further classified into two types :

1) Java API package
2) User defined packages. "

Member access privilages :

1) Public
2) protected
3) default (friendly)
4) private protected
5) private

1) Public :

- Any variable or method is visible to the entire class in which it is defined.

Public :

- A variable or method declared public has the widest possible visibility and is accessible everywhere.
- Implemented by just adding public keyword before, e.g. public int number.

Protected :

- Visibility level is between public and ~~protected~~ friendly.
- The protected modifier makes the fields visible to all classes and sub classes in the same package, but also to subclasses in other packages.
- Non-subclasses in other packages cannot access the protected members.

Default :

- Also known as "friendly" level of access.
- Makes fields visible in all classes of the same package.
- The fields are not visible in other packages.

Private protected :

- Declaration :

  private protected int age ;

- Visibility is between "protected" and "private".
- The fields are visible to all subclasses regardless of the package.

- These fields are not accessible by other classes.

Private:

- Fields are accessible only by the class itself.
- These fields cannot be inherited by subclasses and hence, are not accessible in subclasses.

Access Location	Public	Protected	Friendly (default)	Private Protected	Private
Same class	Yes	Yes	Yes	Yes	Yes
Subclass in same package	Yes	Yes	Yes	Yes	No
Subclass in other packages	Yes	Yes	No	Yes	No
Other classes in same package	Yes	Yes	Yes	No	No
Non subclass in other packages	Yes	No	No	No	No

3.a.1. Visibility of fields.

b) Write a sample program to illustrate packages.

→

Code:

```
package mathematics;
public class Area {
    abstract float area();
}
```

public

Code: directory: 03/mathematics/circle.java

```
package mathematics;
public class circle {
    float diameter;
    float radius;
    float area;

    circle(float radius) {
        this.radius = radius;
        diameter = 2 * radius;

    void CalcArea() {
        System.out.println("Area : " + (3.14 *
            radius * radius);
```

```
}
}
```

Code 2: directory : Q3/Q3.java

```
import mathematics.circle;

class Q3{

    public static void main(String[] args){
        circle c1 = new circle (10);
        c1.CalcArea();
    }
```

Output :

CMD

```
javac Q3.java;
java Q3;

Area : 314.0
```

Q4) Explain about static import with an example.

→

- The static import feature eliminates the need of qualifying a static member with the class name.
- We can use the static import statement to import static members from classes and use them without qualifying the class name.

· Syntax :

import static package-name . subpackage-name . class-name . static-member-name ;

· Example :

import static java.lang.Math.* ;

OR import static java.lang.Math.PI ;

Code :

```
import static java.lang.Math.*;
import java.util.Scanner;
class Q4 {

    public static void main(String[] args){
        System.out.print("Enter radius : ");
        Scanner sc = new Scanner(System.in);
        int radius = sc.nextInt();
        Float area = PI * radius * radius;
        System.out.print("Area : " + area);
    }
}
```

Output : CMD

```
javac Q4.java
java Q4
Enter radius : 10
Area : 314.0
```

Q5) Write a corrected code/program for the details/ information given below.

Solution:

Code:

```
package p1;                          // different file
public class Teacher;
{ . . . . . }
public class Student
{ . . . . . }

package p2;                          // different file
public class for Courses.
{ . . . . . }
public class Student
{ . . . . . }

import p1.*                          // different file
import p2.*
p1.Student student1;
    OR
p2.Student student2;
```