

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;
```

} // end of for loop

```
    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"));

}

3. sum digit of a number

(2-3) & functions

Output: : shows output of a file

CMD

javac Q1.java

java Q1

: abcd

Enter a number : 37

The number 37 is prime

(on int) with random static fields

3++i : (i+s) [CMD] i = 1, s = 3

(o = i + ron) 31

java Q1

Enter a number : 6

The number 6 is not prime.

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



Code: : main() is static method

import java.util.Scanner;

"pi" <--> "random set" string -> integer

((String don) : class Q2 { })

public static void main(String[] args) {

To print a box Scanner sc = new Scanner(System.in);

No. of rows set 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();

}

Copy [Print] from the slide in a single

3

:(i,message) number n = 5

Output: java Q2.java

CMD : java

java Q2.java

Enter number of rows : 5

1 2 3 4 5

1 2 3 4 5

1 2 3 4 5

1 2 3 4 5

? (Question mark)

Ques: If the following segment : qmst-fai

: quest("m/f?")-format: fun, anskey?

: qmst->m/f

?

Q3) i. Write a java programs which will read a line of integers, and displays each integer and the sum of all integers using StringTokenizer.



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("%d\n", temp);
```

```
sum += temp;
```

}

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

private 3
private void add(int... i)
private void add(int... i) {
 int sum = 0;
 for (int n : i) sum += n;
 System.out.println("Sum of numbers : " + sum);
}

Output:

[CMD] > C:\Users\laptop\Desktop\Program

javac Q3.java

java Q3

Input numbers to add in a single

line : 16 32 64 128 256 512 1024 2048

Input: = 3080 []

Output: 3080 = 3080

(16 + 32 + 64 + 128 + 256 + 512 + 1024 + 2048)

long sum = 0;

for (long n : arr) sum += n;

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

16 + 32 + 64 + 128 + 256 + 512 + 1024 + 2048

= 3080

2048

Sum of numbers : 3080

[" + sum + "

int sum = 0;

for (int n : arr) sum += n;

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

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

1. Find the length of array.
2. Demonstrate a 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 element at position  
[" + (i+1) + "][" + (j+1) + "] : ");
```

```
matrix[i][j] = sc.nextInt();
```

3

3

```
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] + " ");
```

3 notice to friends who

Système RUE - Brigitte (2)

System.out.println();

3. *Colocar a mola de aterro entre*

3. Estas nádicas de Stromelia están

3. 開放、透明化、參與化、競爭化

3 - 100% 100% 100% 100% 100% 100% 100% 100% 100% 100%

Final version to submit to

• Classification des animaux selon

• : (e3 e3) adding the structure that

Output : [CMD] javac P4.java & java P4

javac P4.java \$12xen,28 = success fri

java P4 [array] fri sun = xisten [31] fri

length of array :: 8 or 21 : 0 = 31 fri

Demonstrate one dimensional array:

printing from 15 to 120

i (" : [" + (i+1) + "][" + (i+1) + "]")

Demonstrating two-dimensional array

Enter two dim

Enter no. of rows :

3

Enter number of columns :

3

3

Enter elements at position [2][1] : 1

Enter elements at position [1][2] : 2

Enter elements at position [2][3] : 3

Enter elements at position [2][2] : 4

Enter elements at position [2][1] : 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

(" : word list will) forming , then , method

Printing two dimensional array:

((1, 2, 3) forming , then , method)

4 5 6

7 8 9

((row - wise) form)

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

((++ i + first , second , i = 0 = i do))

Code: ((1) Java , " n/e/c") forming , then , method

```
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 no. " + (i+1)
```

```
        + " : ");
```

```
    arr[i] = sc.next();
```

}

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

```
System.out.println("Sorted Array :");  
for( int i = 0; i < arr.length; i++ ) {  
    System.out.format("%s\n", arr[i]);  
}  
3  
3  
3
```

Output:

CMD

Enter five words: India, China, America, Australia, England

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 user 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 of input as integer.
- nextLong() - It scans the next token of input as Long.
- nextFloat() - It scans the next token 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 );
```

(S find) so, in InputStreamReader

BufferedReader br = new BufferedReader (isr);

3) Use System.Console class:

(iii) ~~multiple~~ : The console class is used to get input from user and also to display output.~~multiple~~ 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.

System.out.println("Enter Name");

System.out.println("Enter Password");

String name = System.console().readLine();

String password = System.console().readLine();

(navigated) rename class = 02 rename Java

File > Save As > Java

File > Save As > Java

{}

File > Save As > Java

File > Save As > Java

File > Save As > Java

Assignment No. 02 (Unit 2)

PAGE NO. 14
DATE: / /

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 getRowsC() {  
    rows = sc.nextInt();  
}
```

```
void Matrix() {  
    getRowsC();  
    getColumnsC();
```

```

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.print ("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];
    }
}

else if (rows != a.rows || columns != a.columns) {
    System.out.println("Addition result : ");
    display(sum);
}
else if (rows != a.rows) {
    System.out.println("Matrix cannot be added");
}
else {
    System.out.println("Multiplication result : ");
    for(int i=0; i < rows; i++) {
        for(int j=0; j < columns; j++) {
            try {
                product[i][j] = matrix[i][j];
            }
            catch (Exception e) {
                continue;
            }
            for(int k=0; k < columns; k++) {
                try {
                    product[i][j] = matrix[i][j] *
                        a.matrix[k][j];
                }
                catch (Exception e) {
                    continue;
                }
            }
        }
    }
}

```

3

display (product); (return sm) unqsm

3

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

System.out.println();

3

3 : s : variable for entering value

3 s : [0][0] value to elements value

3 : [0][1] value to elements value

class Q1 { to enter the elements value

3 : value : s.nextInt() value

public static void main (String [] args) {

3 System.out.println ("Enter Matrix 1 : ");

2 : Matrix m1 = new Matrix();

2 : System.out.println ("Enter Matrix 2 : ");

2 : Matrix m2 = new Matrix();

3 : matrix addition

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);  
3
```

Output: [CMD] tom:~/Desktop]\$

```
javac P1.java
```

```
java P1
```

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][1] : 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 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");
```

Writing code to 3 reme print the result

3

5 11

4 8

```
class P2 {
```

```
public static void main(String[] args) {
```

```
    Triangle t = new Triangle();
```

3

Output:

[END]

javac P2.java

java P2

Area of triangle : 6.0 units

Area of

Perimeter of triangle : 12 units.

- Q.3) Create a class 'Complex' with separate methods for each operation whose real and imaginary 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();
```

```
b = sc.nextInt();
```

System.out.println("Enter the imaginary part of");

equation : ");

```
b = sc.nextInt();
```

}

void add(Complex c1) {

System.out.println("\nAddition: \n");

System.out.println(" " + a + " + " + " " + b + " j");

System.out.println(" " + " " + " " + " " + (c1.a * a) + " + " + (c1.b * b) + " j");

System.out.println("-----");

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

System.out.println("-----");

}

((5))

((5))

void subtract(Complex c1) {

System.out.println("\nA - Subtraction : \n");

System.out.println(" " + a + " + " + b + " j");

```

System.out.println("- " + "c1.a" + " + " + c1.b + "j");
System.out.println("-----");
System.out.println(" " + "(a - c2.a) + " + "( " + "c2.b" + " ) + " + "j");
System.out.println("-----");
3

```

void Multiply (Complex c1) {

```
System.out.println("In 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("-----");
```

3

class Q3 {

```
public static void main (String [ ] args) {
```

```
Complex c1 = new Complex();
```

```
Complex c2 = new Complex();
```

```
c1.add(c2);
```

```
c1.subtract(c2);
```

```
c2.multiply(c2);
```

3

Output:

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:

$$1 + 2j$$

$$+ 2 + 2j$$

$$\underline{3 + 4j}$$

Subtraction:

$$1 + 2j$$

$$- 2 + 2j$$

$$\underline{-1 + 0j}$$

Multiplication :

$$1 + 2j$$

$$\times 2 + 2j$$

$$\underline{1 + 2j}$$

$$+ 2 + 4j$$

$$\underline{3 + 6j}$$

- 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"));
    }
}

```

3 (Output for screen print) ~~Output~~

```

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

```

3 (Output) ~~Output~~ ~~Output~~

(Output) : roomno=101

(Output) : roomtype=Deluxe Suite

(Output) : roomarea=250.56f

javac Q4.java

java Q4

Roomno:101 room type:Deluxe Suite

Room type: Deluxe Suite

Room Area: 250.56f

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;

{
 System.out.println("Name : " + name);
 System.out.println("Roll No : " + rollNo);
 System.out.println("Maths : " + mathsTeacher);
}

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 Dhand", 1907011);
```

```
    display(student);
```

}

3

(C:\In0) > o.o4 javampeen

Output: with statements od mapping went to static (LB
leads to code to CMD) with given algorithm bnd shows

javac Q5.java	←
java Q5	: sh2)

↳ (uncommented part)

Name : Pratyay Dhond

Roll No : 19070111 from 2do

: Maths : I Laxmi Ma'am?

: 32.0 = Hard work leads fast success

(d) (a) (b) (c) (d) (e) (f) (g) (h) (i) (j) (k) (l) (m) (n) (o) (p) (q) (r) (s) (t) (u) (v) (w) (x) (y) (z)

↳ 3 mark questions element woks

3 (hard work, alms) work hard work

: alms * work harder

↳ 3 mark questions element woks

3 (hard work, alms) work hard work

: alms * work * is native

↳ 3 mark questions element woks

3 (hard work, alms) work hard work

: hard work

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;
```

```
import java.util.*; // import
```

```
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 $\frac{1}{2} \times \text{length} \times \text{breadth} \times \text{height}$; (Q1)

3. ~~square~~ \rightarrow ~~rectangle~~ \rightarrow ~~triangle~~ \rightarrow ~~circle~~

3

~~new Rectangle(10, 20);~~ \rightarrow ~~length = 10 and breadth = 20~~ : ~~rectangle~~

~~new Circle(10, 0);~~ \rightarrow ~~radius = 10~~ : ~~circle~~

~~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));~~

~~3. (Simple tool) ~~stargard~~ ~~start~~~~

~~3. ~~multiple + width + height~~~~

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 variables with a major difference being that its interfaces define only abstract methods and final fields (data members)."

+ : ~~signature is empty") nothing, due, methods~~

Code: ~~(((class, or, interface, etc))~~

(((interface Employee {

int static final int diwaliBonus = 20000;
int float compute(float salary);

((3) point class = to implement

+ : Employee ? class AProgrammer implements Employee {

((class, or, interface, etc))

int float compute(float salary) {

return salary + diwaliBonus;

3

3

: output

public class

class Q2 { : ((empty, to, run))

public static void main(String[] args) {

Programmer p = new Programmer();

System.out.println("Salary : " + p.compute(50000));

3

Output :

CMO

javac Q2.java

java Q2

Salary : 70000

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

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 privileges:

1) public

2) protected

3) default (friendly)

4) private

5) protected

2) Public :

"Sharing" here "means" 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.
- 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	Public	Protected	Friendly (default)	Private	Private
Location				Protected	
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:

```
class Area {
    float area() {
        float pi = 3.14;
        float r = 5;
        float area = pi * r * r;
        return area;
    }
}
```

Code:

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));

3

3

Code 2:

```
import mathematics.Circle;
class Q3 {
    public static void main(String[] args) {
        Circle c1 = new Circle(10);
        c1.CalcArea();
    }
}
```

Output:

```
C:\Users\Aman bhai\studeying>javac Q3.java
C:\Users\Aman bhai\studeying>java Q3; area = 314.0
C:\Users\Aman bhai\studeying>
```

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.*;  
public 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);  
}
```

3

Output: (CMD)

javac Q4.java	to compile program into .class file
---------------	-------------------------------------

java Q4	to run the program
---------	--------------------

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 Courses.

{.....}

public class Student

{.....}

import p1.* // different file

import p2.*

p1.Student student1;

OR

p2.Student student2;