**Batch: E2 Roll No.: 16010123325**

**Experiment / assignment / tutorial No. 9**

**Grade: AA / AB / BB / BC / CC / CD /DD**

**Signature of the Staff In-charge with date**

|  |
| --- |
| **TITLE :Java Packages** |

**AIM:** Create a package ‘myPackage’ which contains a class myMath. The class contains following static methods.

i)  power (x, y) – to compute xy

ii) fact (x) – to compute x!

Write a program to find the following series.

cos (x) = 1 – (x2/2!) + (x4/4!) – (x6/6!) + … upto n terms (n given by user).

(Do not make use of inbuilt functions. Use the functions of user defined class MyMath by importing mypackage.)

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Expected OUTCOME of Experiment:**

**CO4:** Explore the interface, exceptions, multithreading, packages.

**Books/ Journals/ Websites referred:**

1. Ralph Bravaco , Shai Simoson , “Java Programming From the Group Up” Tata McGraw-Hill.

2.Grady Booch, Object Oriented Analysis and Design .

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Pre Lab/ Prior Concepts:**

**Java Packages:**

A package in Java is a group of similar types of classes, interfaces, and sub-packages. They can be categorized into two categories, the built-in package ( java, lang, util, awt, javax, swing, net, io, sql et), and user-defined package.

They are used for the following tasks –

* To prevent the naming conflicts which can occur between the classes.
* Make the searching and locating of classes or enumerations or annotations much easier.
* Provide access control to the classes.
* Used for data encapsulation.

**Advantages of Java Package:**

* A Java package is mainly used for the categorization of classes and interfaces so that we can maintain them easily.
* They always provide access protection
* Used to bundle classes and interfaces.
* With the help of packages, we can reuse the existing code
* By using the package, we can easily locate the classes related to it.
* Also, remove the naming collision.

**Built-in Packages in Java**

Built-in is a part of Java API and it offers a variety of packages are –

lang – Automatically imported and it contains language support classes.

io – Contains classes for input and output operations.

util – Contains utility classes for implementing data structures.

applet – This package contains classes that create applets.

awt – Contain classes that implement compounds for GUI.

net – This package contains classes that support networking operations.

**User-defined Packages in Java**

1. package First;
2. public class MyClass
3. {
4. public void **getNames**(String name)
5. {
6. System.out.**println**(name);
7. }
8. }
9. package First;
10. import First.MyClass;
11. public class MyClass1 {
12. public static void **main**(String args[])
13. {
14. // Initializing the String variable with a value
15. String name = "Welcome";
16. // Creating an instance of class MyClass in the package.
17. MyClass obj = new **MyClass**();
18. obj.**getNames**(name);
19. }
20. }

.

**Class Diagram:**

|  |  |  |
| --- | --- | --- |
| Class Name | Parameters | Return Type |
| mymath | x: int, y: int | int |
|  | x: int | int |
| Mainclass | x: double, n: int | double |
|  | argos:String[] | void |

**Algorithm:**

**Package Declaration:**

* Create a package Mypackage with two classes: Mainclass and mymath.

**Import Statements:**

* Import Mypackage.mymath and java.util.Scanner for input.

**cosSeries Method:**

* Takes input x (value for cosine) and n (number of terms).
* Start with result = 1.0 (first term of cos series).
* Alternate signs using sign = -1.
* Loop through even powers (2, 4, 6, ...) up to 2\*n:
  + Use mymath.power(x, i) and mymath.fact(i) to calculate terms.
  + Add/subtract each term from result based on the sign.
* Return the final result.

**Main Method:**

* Ask the user to input x and n.
* Call cosSeries(x, n) to calculate the cosine series.
* Print the result.

**mymath Class:**

* power(x, y) calculates x^y using a loop.
* fact(x) calculates factorial of x using a loop.

**Implementation details:**

**Cos\_series.java**

import *myPackage.MyMath*;

import *java.util.\**;

*public* *class* cos\_series

{

*public* *static* *void* main(*String*[] *args*)

    {

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

        System.out.print("Enter the value of x (in radians): ");

*double* x = sc.nextDouble();

        System.out.print("Enter the number of terms n: ");

*int* n = sc.nextInt();

*double* cosX = 1;

*int* sign = -1;

        for (*int* i = 1; i < n; i++)

        {

*int* exp = 2 \* i;

*double* term = sign \* MyMath.power(x, exp) / MyMath.fact(exp);

            cosX += term;

            sign \*= -1;

        }

        System.out.println("Cosine of x using the series: " + cosX);

        sc.close();

    }

}

**MyMath.java**

package *myPackage*;

*public* *class* MyMath

{

*public* *static* *double* power(*double* *x*, *int* *y*)

    {

*double* result = 1;

        for (*int* i = 0; i < y; i++)

        {

            result \*= x;

        }

        return result;

    }

*public* *static* *int* fact(*int* *x*)

    {

*int* factorial = 1;

        for (*int* i = 1; i <= x; i++)

        {

            factorial \*= i;

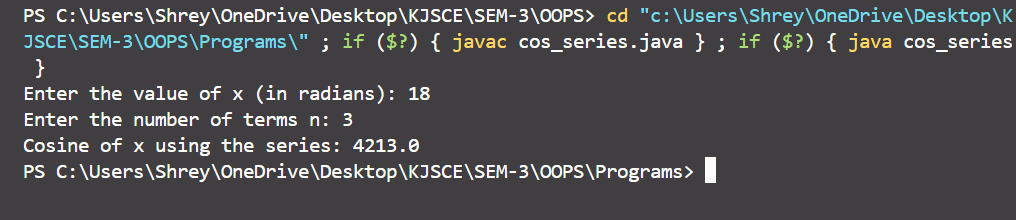
        }

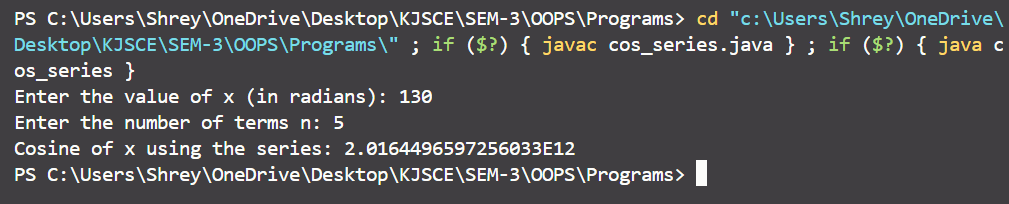
        return factorial;

    }

}

**Output:**





**Conclusion:**

In this experiment, we created a custom package myPackage with a class mymath to compute powers and factorials, using it to calculate the cosine series without built-in functions. This enhanced our understanding of packages and static methods in Java.

**Date: 11/10/24 Signature of faculty in-charge**

**Post Lab Descriptive Questions**

Q.1   What are Java Packages? What's the significance of packages?

Java packages are a way to organize and group related classes, interfaces, and sub-packages together. They help in managing large codebases by preventing naming conflicts and improving code organization.

### Significance of Packages:

* **Modularity:** Makes code easier to manage and maintain.
* **Reusability:** Classes can be reused in different projects.
* **Namespace Control:** Avoids class name conflicts by grouping them in namespaces.
* **Access Control:** Controls visibility of classes and methods (e.g., public, protected, private).

Q.2 Does Importing a package imports its sub-packages as well in Java?

No, importing a package in Java **does not** automatically import its sub-packages. Each package and sub-package must be imported separately. For example, importing java.util does not import java.util.regex. You need to import sub-packages explicitly if you want to use their classes.

Q.3 Write a program to create a package ‘myPack’ which contains a class Trigonometry. The

class contains following static methods.

i) sine() –accepts degree (0,30,60,90)

ii) cos() - accepts degree (0,30,60,90)

iii)tan()- accepts degree (0,30,60,90)

iv)cot()-- accepts degree (0,30,60,90)

v)cosec()-- accepts degree (0,30,60,90)

vi)sec()-- accepts degree (0,30,60,90)

(Do not make use of inbuilt functions. Use the functions of user defined class Trigonometry by

importing mypack.)

**1. Trigonometry.java**

package *myPackage*;

*public* *class* Trigonometry {

*public* *static* *double* sine(*int* *degree*) {

        switch (degree) {

            case 0: return 0;

            case 30: return 0.5;

            case 60: return 0.866;

            case 90: return 1;

            default: return Double.NaN;

        }

    }

*public* *static* *double* cos(*int* *degree*) {

        switch (degree) {

            case 0: return 1;

            case 30: return 0.866; *// sqrt(3)/2*

            case 60: return 0.5;

            case 90: return 0;

            default: return Double.NaN;

        }

    }

*public* *static* *double* tan(*int* *degree*) {

        switch (degree) {

            case 0: return 0;

            case 30: return 0.577;

            case 60: return 1.732;

            case 90: return Double.POSITIVE\_INFINITY;

            default: return Double.NaN;

        }

    }

*public* *static* *double* cot(*int* *degree*) {

        switch (degree) {

            case 0: return Double.POSITIVE\_INFINITY;

            case 30: return 1.732;

            case 60: return 0.577;

            case 90: return 0;

            default: return Double.NaN;

        }

    }

*public* *static* *double* cosec(*int* *degree*) {

        switch (degree) {

            case 0: return Double.POSITIVE\_INFINITY;

            case 30: return 2;

            case 60: return 1.154;

            case 90: return 1;

            default: return Double.NaN;

        }

    }

*public* *static* *double* sec(*int* *degree*) {

        switch (degree) {

            case 0: return 1;

            case 30: return 1.154;

            case 60: return 2;

            case 90: return Double.POSITIVE\_INFINITY;

            default: return Double.NaN;

        }

    }

}

**2. MainClass.java**

import *myPackage.Trigonometry*;

import *java.util.Scanner*;

*public* *class* Main

{

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

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

        System.out.print("Enter the angle (0, 30, 60, 90): ");

*int* degree = sc.nextInt();

        System.out.println("sin(" + degree + ") = " + Trigonometry.sine(degree));

        System.out.println("cos(" + degree + ") = " + Trigonometry.cos(degree));

        System.out.println("tan(" + degree + ") = " + Trigonometry.tan(degree));

        System.out.println("cot(" + degree + ") = " + Trigonometry.cot(degree));

        System.out.println("cosec(" + degree + ") = " + Trigonometry.cosec(degree));

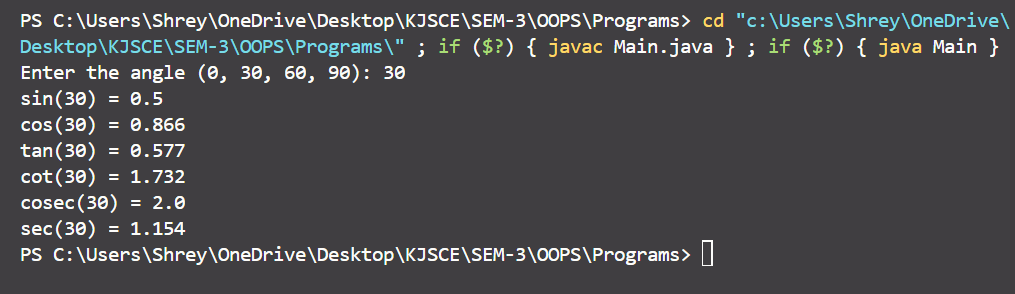
        System.out.println("sec(" + degree + ") = " + Trigonometry.sec(degree));

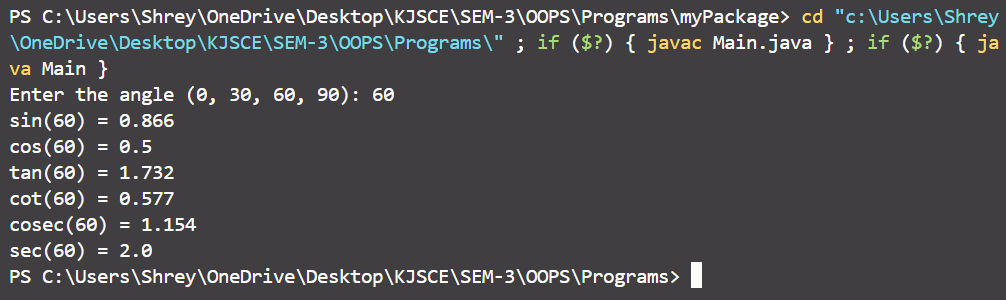
        sc.close();

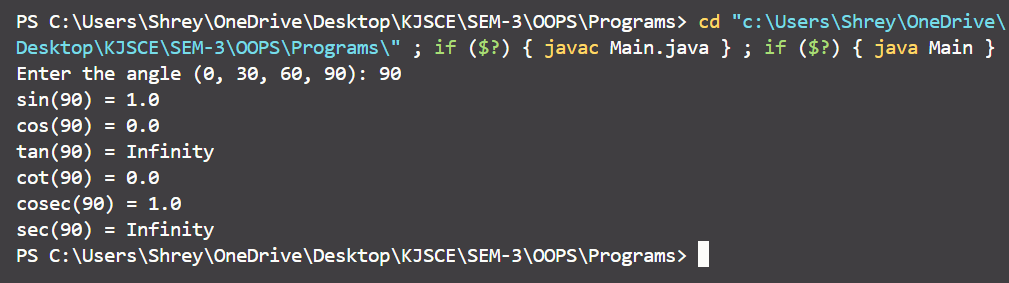
    }

}

**Output:**

****

****

****