Batch: D3 Roll No.: 16010123294

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

```
package First;

public class MyClass
{
public void getNames(String name)
{
System.out.println(name);
}

}
```

```
package First;
import First.MyClass;
public class MyClass1 {
public static void main(String args[])
{
// Initializing the String variable with a value
String name = "Welcome";
// Creating an instance of class MyClass in the package.
MyClass obj = new MyClass();
obj.getNames(name);
}
}
```

Algorithm:

1. Method: power(x, y)
Input:
x: The base number (a double).
y: The exponent (an integer).
Steps:
If y equals 0, return 1 (as any number raised to the power 0 is 1).
Otherwise, recursively call power(x, y - 1) and multiply the result by x.
Output:
The result of xyx^yxy (x raised to the power y).
2. Method: fact(x)
Input:
x: The number whose factorial is to be computed (an integer).
Steps:
If x equals 0 or 1, return 1 (as the factorial of 0 or 1 is 1).
Otherwise, recursively call fact(x - 1) and multiply the result by x.
Output:
The factorial of x (i.e., x!x!x!).

Cosine Series Algorithm:

Input:
Accept the angle x in degrees.
Accept the number of terms nnn to compute the approximation.
Convert degrees to radians:
radians=x×π180radians = x \times \frac{\pi}{180}radians=x×180π.

Initialize the series result:
Set result = 1 to account for the first term in the cosine series.
Set sign = -1 to alternate the signs of subsequent terms.
Iterate to compute the series terms:
Loop from i = 2 to 2×(n−1)2 \times (n - 1)2×(n−1), with an increment of 2.
For each iteration, add the term sign×xii!sign \times \frac{x^i}{i!}sign×i!xi to result.
Update the sign by multiplying it by -1 to alternate the signs.
Output the result:
The final result is the approximate value of cos (x)\cos(x)cos(x) using the series expansion.
End.
This algorithm uses the Taylor series expansion of the cosine function to approximate the value of cos (x)\cos(x)cos(x) based on the specified number of terms.

Implementation details:

MYPACKAGE:

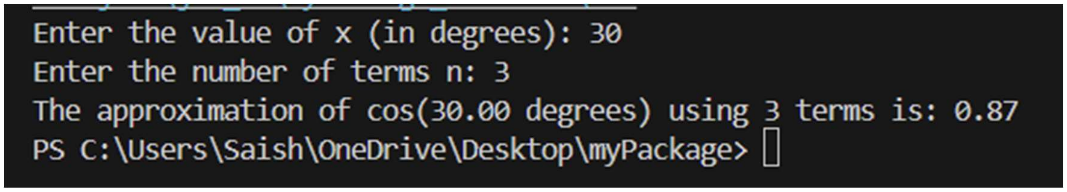
```
package mypackage;
```

```
public class myMath {



    public static double power(double x, int y) {
        if (y == 0) {
            return 1;
        }
        return x * power(x, y - 1);
    }

    public static int fact(int x) {
        if (x == 0 || x == 1) {
            return 1;
        }
        return x * fact(x - 1);
    }
}
```

Cosine Series

```
import mypackage.myMath;

import java.util.Scanner;


public class CosineSeries {


    public static double cosSeries(double x, int n) {

        double result = 1;

        int sign = -1;


        for (int i = 2; i <= 2 * (n - 1); i += 2) {

            result += sign * (myMath.power(x, i) / myMath.fact(i));

            sign *= -1;

        }


        return result;

    }


    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);


        System.out.print("Enter the value of x (in degrees): ");
        double degrees = scanner.nextDouble();
        System.out.print("Enter the number of terms n: ");
        int n = scanner.nextInt();

        double radians = degrees * Math.PI / 180;

        double cosValue = cosSeries(radians, n);

        System.out.printf("The approximation of cos(%.2f degrees) using %d terms is: %.2f%n", degrees, n, cosValue);

        scanner.close();
    }
}
```

Output:



```
Enter the value of x (in degrees): 30
Enter the number of terms n: 3
The approximation of cos(30.00 degrees) using 3 terms is: 0.87
PS C:\Users\Saish\OneDrive\Desktop\myPackage> 
```

Conclusion: Learned to make my own packages and import them for different usecases.

Date: __________ **Signature of faculty in-charge**

<u>Post Lab Descriptive Questions</u>

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

Java packages are a way to group related classes and interfaces together. They help organize code, prevent name conflicts, and control access.
Organization: Keeps code organized and manageable.
Namespace management: Avoids class name conflicts by grouping them in different packages.
Access control: Provides access levels (public, private, protected) between classes.
Reusability: Allows easy reusability of code across projects.

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. You need to import sub-packages separately. For example, importing java.util.* will not import java.util.stream.*; you would need to import java.util.stream.* explicitly.

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.)

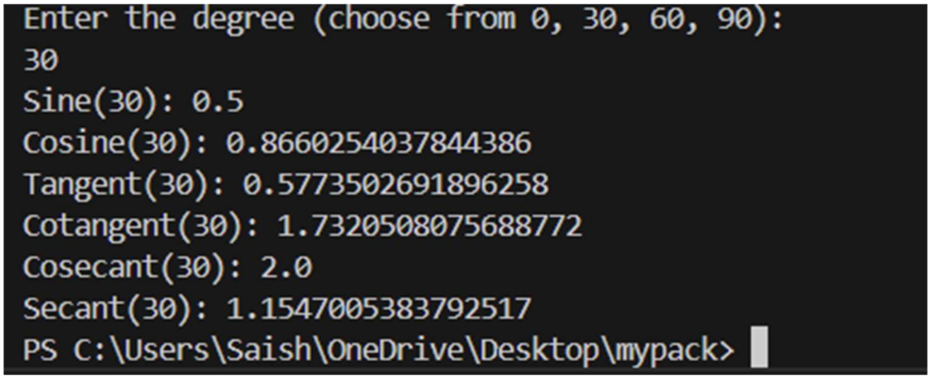
mypack

```
package mypack;

public class trigo{

   public static double sine(int degree) {
      switch (degree) {
         case 0: return 0.0;
         case 30: return 0.5;
         case 60: return Math.sqrt(3) / 2;
         case 90: return 1.0;
         default: throw new IllegalArgumentException("Invalid degree! Choose from 0,
30, 60, 90.");
      }
   }


   public static double cos(int degree) {
      switch (degree) {
         case 0: return 1.0;
         case 30: return Math.sqrt(3) / 2;
         case 60: return 0.5;
         case 90: return 0.0;
         default: throw new IllegalArgumentException("Invalid degree! Choose from 0,
30, 60, 90.");
      }
   }

   public static double tan(int degree) {
      switch (degree) {
         case 0: return 0.0;
         case 30: return 1.0 / Math.sqrt(3);
         case 60: return Math.sqrt(3);
         case 90: return Double.POSITIVE_INFINITY;
         default: throw new IllegalArgumentException("Invalid degree! Choose from 0,
30, 60, 90.");
      }
   }

   public static double cot(int degree) {
      switch (degree) {
         case 0: return Double.POSITIVE_INFINITY;
         case 30: return Math.sqrt(3);
         case 60: return 1.0 / Math.sqrt(3);
         case 90: return 0.0;
         default: throw new IllegalArgumentException("Invalid degree! Choose from 0,
30, 60, 90.");
      }
   }
   public static double cosec(int degree) {
      switch (degree) {
         case 0: throw new IllegalArgumentException("Cosecant undefined for 0
degrees.");
         case 30: return 2.0;
         case 60: return 2.0 / Math.sqrt(3);
         case 90: return 1.0;
         default: throw new IllegalArgumentException("Invalid degree! Choose from 0,
30, 60, 90.");
      }
   }

   public static double sec(int degree) {
      switch (degree) {
         case 0: return 1.0;
         case 30: return 2.0 / Math.sqrt(3);
         case 60: return 2.0;
         case 90: throw new IllegalArgumentException("Secant undefined for 90
degrees.");
```

```
            default: throw new IllegalArgumentException("Invalid degree! Choose from 0,
30, 60, 90.");
        }
    }
}
```

main.java

```
// File: Main.java
import mypack.trigo;
import java.util.Scanner;

public class main {
    public static void main (String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the degree (choose from 0, 30, 60, 90): ");
        int degree = sc.nextInt();

        try {
            System.out.println("Sine(" + degree + "): " + trigo.sine(degree));
            System.out.println("Cosine(" + degree + "): " + trigo.cos(degree));
            System.out.println("Tangent(" + degree + "): " + trigo.tan(degree));
            System.out.println("Cotangent(" + degree + "): " + trigo.cot(degree));
            System.out.println("Cosecant(" + degree + "): " + trigo.cosec(degree));
            System.out.println("Secant(" + degree + "): " + trigo.sec(degree));
        } catch (IllegalArgumentException e) {
            System.out.println(e.getMessage());
        }

        sc.close();
    }
}
```



```
Enter the degree (choose from 0, 30, 60, 90):
30
Sine(30): 0.5
Cosine(30): 0.8660254037844386
Tangent(30): 0.5773502691896258
Cotangent(30): 1.7320508075688772
Cosecant(30): 2.0
Secant(30): 1.1547005383792517
PS C:\Users\Saish\OneDrive\Desktop\mypack>
```