EXPERIMENT NO. 06: JAVA PACKAGES

Objective

To create a Java program that demonstrates the use of Java packages for modular programming and class organization.

Theory

Java packages provide a way to organize related classes and interfaces into namespaces, improving maintainability and reusability. There are two types of packages: built-in and user-defined. User-defined packages allow developers to group their own classes logically and avoid naming conflicts.

Advantages of Java Packages

- Avoid class name conflicts
- Controlled access using public, private, protected
- Better class management
- Easier to locate and use classes, interfaces, enums, etc.
- Supports hierarchical structure

Types of Packages

- 1. Built-in packages: Provided by Java API (e.g., java.util, java.io).
- 2. User-defined packages: Created by developers to group related classes.

Package Structure Example

Package: com.example.utilities

Contains: Calculator.java, DisplayHelper.java, etc.

Program Code – User-defined Package with Multiple Methods

```
// File: com/example/utility/MathHelper.java

package com.example.utility;

public class MathHelper {
    public int add(int a, int b) { return a + b; }
    public int subtract(int a, int b) { return a - b; }
    public int multiply(int a, int b) { return a * b; }
    public double divide(int a, int b) {
        if (b == 0) throw new ArithmeticException("Cannot divide by zero.");
        return (double) a / b;
}
```

```
public void printHeader() {
    System.out.println("== Java Package Utility ==");
}
```

```
import com.example.utility.MathHelper;

public class Main {
    public static void main(String[] args) {
        MathHelper helper = new MathHelper();
        helper.printHeader();
        System.out.println("Add: " + helper.add(10, 5));
        System.out.println("Subtract: " + helper.subtract(10, 5));
        System.out.println("Multiply: " + helper.multiply(10, 5));
        System.out.println("Divide: " + helper.divide(10, 5));
    }
}
```

Sample Output

```
== Java Package Utility ==
Add: 15
Subtract: 5
Multiply: 50
Divide: 2.0
```

Explanation

A package com.example.utility is created containing the class MathHelper. In the Main class, this class is accessed using its fully qualified package name.

Coding Task: Matrix Operations using Package

```
// File: Mathematics/MatrixOperations.java
package Mathematics;
public class MatrixOperations {
  private int[][] matrix;
  public MatrixOperations(int[][] matrix) {
     this.matrix = matrix;
  public static MatrixOperations add(MatrixOperations m1, MatrixOperations m2) {
     int[][] result = new int[2][2];
     for (int i = 0; i < 2; i++)
       for (int j = 0; j < 2; j++)
          result[i][j] = m1.matrix[i][j] + m2.matrix[i][j];
     return new MatrixOperations(result);
  public static MatrixOperations subtract(MatrixOperations m1, MatrixOperations m2) {
     int[][] result = new int[2][2];
     for (int i = 0; i < 2; i++)
       for (int j = 0; j < 2; j++)
          result[i][j] = m1.matrix[i][j] - m2.matrix[i][j];
     return new MatrixOperations(result);
  public void display() {
     for (int[] row : matrix) {
       for (int val : row)
          System.out.print(val + " ");
       System.out.println();
```

```
// File: MatrixMain.java
import Mathematics.MatrixOperations;

public class MatrixMain {
   public static void main(String[] args) {
     int[][] a = {{1, 2}, {3, 4}};
     int[][] b = {{5, 6}, {7, 8}};

     MatrixOperations mat1 = new MatrixOperations(a);
     MatrixOperations mat2 = new MatrixOperations(b);

     System.out.println("Matrix Addition:");
     MatrixOperations sum = MatrixOperations.add(mat1, mat2);
```

```
sum.display();

System.out.println("Matrix Subtraction:");
MatrixOperations diff = MatrixOperations.subtract(mat1, mat2);
diff.display();
}
```

Sample Output

Matrix Addition: 6 8 10 12 Matrix Subtraction:

-4 -4

-4 -4

Conclusion

Java packages allow for better organization and modularity in large-scale applications. This experiment demonstrates creating and using user-defined packages effectively.