| Experiment No. 5 |
| --- |
| Implement a program on Packages. |
| Date of Performance: |
| Date of Submission: |

## Aim: To use packages in java.

**Objective:** To use packages in java to use readymade classes available in them using square root method in math class.

**Theory:**

A java package is a group of similar types of classes, interfaces and sub-packages. Packages are used in Java in order to prevent naming conflicts, to control access, to make searching/locating and usage of classes, interfaces, enumerations and annotations easier, etc.

There are two types of packages-

1. Built-in package: The already defined package like java.io.*, java.lang.* etc are known as built-in packages.
2. User defined package: The package we create for is called user-defined package.

Programmers can define their own packages to bundle group of classes/interfaces, etc. While creating a package, the user should choose a name for the package and include a package statement along with that name at the top of every source file that contains the classes, interfaces, enumerations, and annotation types that you want to include in the package. If a package statement is not used then the class, interfaces, enumerations, and annotation types will be placed in the current default package.

**Code:**

```
import java.io.*;
public class Test

{
     public static void main(String args[])
     {
          double number = 64.0;
          double squareRoot = Math.sqrt(number);
          System.out.println("The square root of " + number + "
is: " + squareRoot);
     }
}
```

**OUTPUT:-**

**The square root of 64.0 is: 8.0**

```
C:\Users\admin\Desktop\JL>java Test.java
The square root of 64.0 is: 8.0
```

**Conclusion:**

Comment on the autoencoder architecture and the Image compression results.
→
**Autoencoder Architecture**

Autoencoders are a type of neural network used for unsupervised learning tasks, particularly for dimensionality reduction and feature learning. The general architecture of an autoencoder includes:

1. **Encoder**: This part of the network compresses the input into a lower-dimensional representation. It consists of one or more layers that map the input data to a smaller latent space.
2. **Latent Space**: The compressed representation of the data that captures its essential features.
3. **Decoder**: This part reconstructs the input data from the latent space representation. It mirrors the structure of the encoder but aims to approximate the original input as closely as possible.

Autoencoders can be used for various applications, including:

- **Dimensionality Reduction**: Reducing the number of features while retaining essential information.
- **Denoising**: Learning to remove noise from the input data.
- **Anomaly Detection**: Identifying unusual data points by measuring reconstruction error.

**Image Compression Results**

Autoencoders can be particularly useful for image compression:

- **Compression**: By encoding an image into a lower-dimensional latent space and then decoding it, autoencoders can compress image data while preserving as much of the original information as possible.
- **Quality**: The quality of the compressed image depends on the architecture and training of the autoencoder. Typically, more complex autoencoders with deeper networks and more sophisticated training techniques can achieve higher quality compression.
- **Evaluation**: The effectiveness of image compression can be evaluated using metrics like Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM). Higher PSNR and SSIM values generally indicate better reconstruction quality.