Vidyavardhini's College of Engineering and Technology
Department of Artificial Intelligence & Data Science

Experiment No.2
Accepting Input Through Keyboard
Date of Performance:
Date of Submission:

Aim: To apply basic programing for accepting input through keyboard.

Objective: To use the facility of java to read data from the keyboard for any program

Theory:

Java brings various Streams with its I/O package that helps the user perform all the Java input-output operations. These streams support all types of objects, data types, characters, files, etc. to fully execute the I/O operations. Input in Java can be with certain methods mentioned below in the article.

Methods to Take Input in Java

There are two ways by which we can take Java input from the user or from a file

1. BufferedReader Class
2. Scanner Class

Using BufferedReader Class for String Input In Java

It is a simple class that is used to read a sequence of characters. It has a simple function that reads a character another read which reads, an array of characters, and a readLine() function which reads a line.

InputStreamReader() is a function that converts the input stream of bytes into a stream of characters so that it can be read as BufferedReader expects a stream of characters. BufferedReader can throw checked Exceptions.

Using Scanner Class for Taking Input in Java

It is an advanced version of BufferedReader which was added in later versions of Java. The scanner can read formatted input. It has different functions for different types of data types.

The scanner is much easier to read as we don't have to write throws as there is no exception thrown by it.

It was added in later versions of Java

It contains predefined functions to read an Integer, Character, and other data types as well.

Syntax of Scanner class

Scanner scn = new Scanner(System.in);

Code:

- **Buffer Reader**

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

public class BufferedReaderExample
{
   public static void main(String[] args)
 {
      BufferedReader reader = new BufferedReader(new
InputStreamReader(System.in));

      try
        {
        System.out.print("Enter a string using BufferedReader: ");
        String input = reader.readLine();
        System.out.println("You entered: " + input);
        }
        catch (IOException e)
        {
        e.printStackTrace();
        }
   }
}
```

Scanner

```
import java.util.Scanner;


public class ScannerExample
 {
   public static void main(String[] args)
{
      Scanner scanner = new Scanner(System.in);


      System.out.print("Enter a string using Scanner: ");
      String input = scanner.nextLine();
      System.out.println("You entered: " + input);


      scanner.close();
   }
}
```

Conclusion:

BufferedReader is best for high-performance applications where efficiency is crucial and when reading large amounts of text.
Scanner is ideal for simpler applications where ease of use and versatility are prioritized, especially when dealing with various data types.