ACPCE
Where knowledge is second nature

Jawahar Education Society's
A. C. Patil College of Engineering, Kharghar
Navi Mumbai 410210

Student Name: Chetan Ingale

Course Name: C.S.E. (IoT CS BC)

Year: S.E.

Roll No.: 17

PRN No.: 221111030

Course code: CSL304

Semester: 3

Experiment Evaluation Sheet

Experiment No.: 2

Experiment Name:
Program on accepting input through keyboard.

Sr No.	Evaluation Criteria	Marks (Out of 9)	Performance Date	Correction Date and Signature of Instructor
1	Experiment Performance			
2	Journal Performance			
3	Punctuality			
Total				

Aim : Program on accepting input through keyboard.

Software required : Java, Javac.

Theory :

User Input (Scanner) :
The Scanner class is used to get user input, and it is found in the java.util package.

To use the Scanner class, create an object of the class and use any of the available methods found in the Scanner class documentation. In our example, we will use the nextLine() method, which is used to read Strings:

Input Types :

Method	Description
nextBoolean()	Reads a boolean value from the user
nextByte()	Reads a byte value from the user
nextDouble()	Reads a double value from the user
nextFloat()	Reads a float value from the user
nextInt()	Reads a int value from the user
nextLine()	Reads a String value from the user
nextLong()	Reads a long value from the user
nextShort()	Reads a short value from the user

User Input (BufferedReader) :
Reads text from a character-input stream, buffering characters so as to provide for the efficient reading of characters, arrays, and lines. The buffer size may be specified, or the default size may be used. The default is large enough for most purposes. In general, each read request made by a Reader causes a corresponding read request to be made of the underlying character or byte stream. It is therefore advisable to wrap a BufferedReader around any Reader whose read() operations may be costly, such as FileReaders and InputStreamReaders. Programs that use DataInputStreams for textual input can be localized by replacing each DataInputStream with an appropriate BufferedReader.

Input Types :

Method	Description
read()	Reads a single character.
readLine()	Reads a line of text. A line is considered to be terminated by any one of a line feed ('\n'), a carriage return ('\r'), or a carriage return followed immediately by a line feed.

Code 2.a :

```
import java.io.*;

public class userInput01{
   public static void main(String[] args)
      throws IOException
   {
      BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
      System.out.print("Enter your name: ");
      String name = reader.readLine();
      System.out.println("Your name is = " + name);
   }
}
```

Output 2.a :

```
student@csiot-ThinkCentre-M70s:~/Chetan17$ javac userInput01.java
student@csiot-ThinkCentre-M70s:~/Chetan17$ java userInput01
Enter your name: Chetan Ingale
Your name is = Chetan Ingale
student@csiot-ThinkCentre-M70s:~/Chetan17$
```

Code 2.b :

```
import java.util.Scanner;

public class userInput02{
   public static void main(String[] args) {
      String name;
      int age;
      char gender;
      double salary;

      Scanner console = new Scanner(System.in);

      System.out.print("What is your name : ");
      name = console.nextLine();

      System.out.print("What is your age : ");
      age = console.nextInt();

      System.out.print("What is your gender : ");
      gender = console.next().charAt(0);

      System.out.print("What is your salary : ");
      salary = console.nextDouble();

      System.out.print("\nName : " + name);
      System.out.print("\nAge : " + age);
      System.out.print("\nGender : " + gender);
      System.out.println("\nSalary : " + salary);

   }
}
```

Output 2.b :

```
student@csiot-ThinkCentre-M70s:~/Chetan17$ java userInput02
What is your name : Chetan Inagle
What is your age : 19
What is your gender : Male
What is your salary : 0

Name : Chetan Inagle
Age : 19
Gender : M
Salary : 0.0
```

Conclusion :

With this experiment we got familiar with taking user input in java.