**Input in Java**

**Java code**

**import** java.util.Scanner;

**public** **class** ScannerInput {

 **public** **static** **void** main(String[] args) {

 // Input using Scanner class

 Scanner in = **new** Scanner(System.***in***);

 String s = in.nextLine();

 System.***out***.println("You entered string "+s);

 **int** a = in.nextInt();

 System.***out***.println("You entered integer "+a);

 **float** b = in.nextFloat();

 System.***out***.println("You entered float "+b);

 }

}

**Exercise**

* Write code to enter a firstname, surname and age(integer) and output to screen on the same line.
* Write code to enter a Street address, town and county and output to screen on the seperate lines.

**3 types of input in Java**

There are three different ways to read the input from Java Console, they are –

* Using Java Bufferedreader Class
* Scanner Class in Java
* Console Class in Java

**Scanner Class in Java**

This is presumably the most favored technique to take input. The primary reason for the Scanner class is to parse primitive composes and strings utilizing general expressions, in any case, it can be utilized to peruse contribution from the client in the order line.

**Pros**

* Helpful strategies for parsing natives (nextInt(), nextFloat(), … ) from the tokenized input.
* General articulations can be utilized to discover tokens.

**Cons**

* The reading methods are not synchronized.
* Java File Class – java.io.File Class in Java

**Example of Scanner Class in Java**

import java.util.Scanner;

class GetInputFromUser

{

 public static void main(String args[])

 {

 Scanner in = new Scanner(System.in);

 String s = in.nextLine();

 System.out.println("You entered string "+s);

 int a = in.nextInt();

 System.out.println("You entered integer "+a);

 float b = in.nextFloat();

 System.out.println("You entered float "+b);

 }

}

**Java Bufferedreader Class**

This is the Java traditional technique, introduced in JDK1.0. This strategy is utilized by wrapping the System.in (standard information stream) in an InputStreamReader which is wrapped in a Java BufferedReader, we can read result include from the user in the order line.

**Pros**

* The information is cradled for productive perusing.

**Cons**

* The wrapping code is difficult to recall.

**Example of Java Bufferedreader Class–**

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

public class Test

{

 public static void main(String[] args) throws IOException

 {

 BufferedReader reader =

 new BufferedReader(new InputStreamReader(System.in));

 String name = reader.readLine();

 System.out.println(name);

 }

}

**Console Class in Java**

It has been turning into a favored route for perusing client’s contribution from the command line. In addition, it can be utilized for password key like contribution without resounding the characters entered by the client, the configuration string syntax structure can likewise be utilized (like System.out.printf()).

**Pros**

* Reading secret word without reverberating the entered characters.
* Reading strategies that are synchronized.
* Format string sentence structure can be utilized.

**Cons**

* Does not work in non-intelligent condition, (for example, in an IDE).
* Do You Know Difference Between Abstract Class and Interface in Java

**Example of Console Class in Java**

public class Sample

{

 public static void main(String[] args)

 {

 // Using Console to input data from user

 String name = System.console().readLine();

 System.out.println(name);

 }

}

4. Java Console Example

String name = null;

int number;

java.io.BufferedReader in = new BufferedReader(new InputStreamReader(System.in));

name = in.readLine();number = Integer.parseInt(in.readLine());

System.out.println("Name " + name + "\t number " + number);

java.util.Scanner sc = new Scanner(System.in).useDelimiter("\\s");

name = sc.next();

number = sc.nextInt();System.out.println("Name " + name + "\t number " + number);

java.io.Console cnsl = System.console();

if (cnsl != null) {

 name = cnsl.readLine("Name: ");

 System.out.println("Name entered: " + name);

}