# Chapter 3

## Input & Output Data

# Objectives

- At the end of this chapter, student should be able to:
  - understand the concept of  input and output representation in java.
  - obtain input from the console using the Scanner class.
  - obtain input using the JOptionPane input dialog boxes
  - examine ways to output results using output statements.

# Program Without Input

```
public class WithoutInput {
   public static void main(String[] args){
       System.out.println("How old are you?");
       System.out.println("Hi!!, you are 20 years
   old");
   }
}
```

```
C:\> javac WithoutInput.java
C:\> java WithoutInput
How old are you?
Hi!!, you are 20 years Old
```

# Program With Static Input

```
public class UsingStaticInput {
   public static void main(String[] args){
     int age;
     System.out.print("How old are you?");
     age = 20;
     System.out.println("Hi!!, you are "+age+ " years
   old");
   }
}
```

```
How old are you?
Hi!!, you are 20 years Old
```

**How to make an interactive program?**

# Input

- Input can be obtained from:
  - console
  - input dialog
- To read input from the console, Java uses `System.in` to refer to the standard input device.
- By default, the input device is the keyboard.
- When the computer gets the data from the keyboard, the user is said to be acting interactively.

# Reading Input from the Console

- `Scanner` class can be used to create an object to read input from System.in.
- `Scanner` is a predefined Java class.
- The Scanner class is in the `java.util` package
- To read data:
  - Create an input stream object of the class `Scanner` and associate it with the standard input device as the following statements:

    ```
    Scanner console = new Scanner(System.in);
    ```
  - Use the methods such as `next()`, `nextLine()`, `nextInt()`, `nextShort()`, `nextByte()`, `nextLong()`, `nextFloat()` and `nextDouble()`

# Structure of a program

//Java Package

//class

//main method

//open program, {

//body

//close program, }

```
import java.lang.*;

public class Welcome{
  public static void
  main(String[] args){

  System.out.println("Welc
  ome");
}
}
```

# Implicit Import and Explicit Import

```
java.util.* ; // Implicit import

java.util.Scanner; // Explicit
Import
```

No performance difference

# Methods for Scanner Object

| Methods | Description |
|---|---|
| nextByte() | reads an integer of the byte type. |
| nextShort() | reads an integer of the short type. |
| nextInt() | reads an integer of the int type. |
| nextLong() | reads an integer of the long type. |
| nextFloat() | reads a number of the float type. |
| nextDouble() | reads a number of the double type. |
| next() | reads a string that ends before a whitespace character. |
| nextLine() | reads a line of text (i.e., a string ending with the Enter key pressed). |

# Using `Scanner` Class for Input

age 17

```
import java.util.*;
public class UsingScannerClass {
   public static void main(String[] args){
      Scanner input = new Scanner(System.in);
      int age;
      System.out.print("How old are you?");
      age = input.nextInt();
      System.out.println("Hi!!, you are "+age+ " years old");
   }
}
```

```
How old are you?        17
Hi!!, you are 17 years old
```

# Exercise

- Write the java program using the given pseudocode.

1. Start
2. Get a radius
3. Calculate area where area = π x r x r
4. Display area
5. End

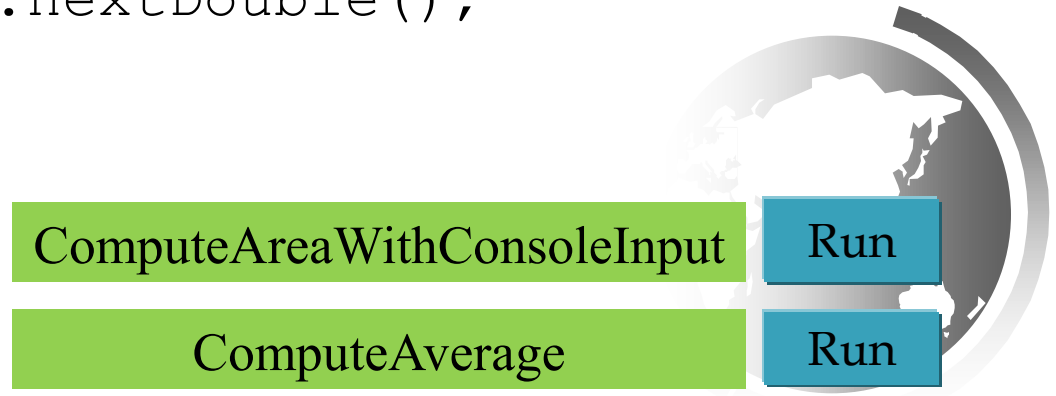
# Reading Input from the Console

1. Create a Scanner object

```
Scanner input = new Scanner(System.in);
```

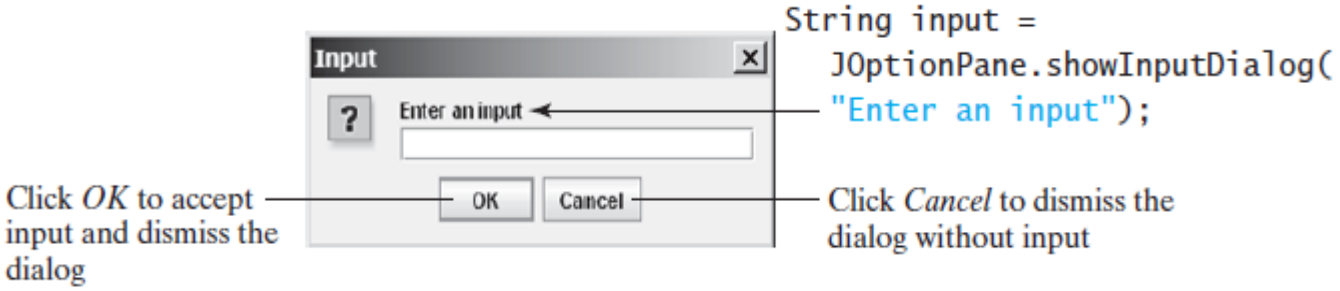
2. Use the method nextDouble() to obtain to a double value. For example,

```
System.out.print("Enter a double value: ");
Scanner input = new Scanner(System.in);
double d = input.nextDouble();
```



# Getting Input from Input Dialogs

- invoking the `JOptionPane.showInputDialog` method

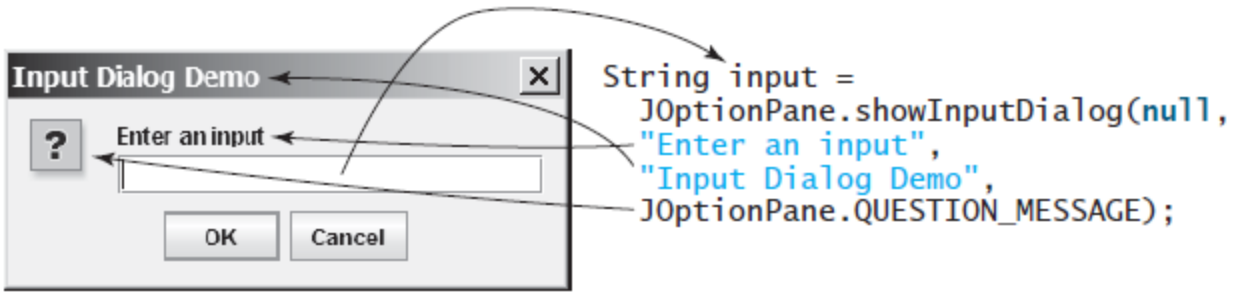



# Getting Input from Input Dialogs

- Two ways to invoke `showInputDialog` method:
  - `JOptionPane.showInputDialog(x);`

    where `x` is a string for the prompting message.
  - `String string = JOptionPane.showInputDialog(null, x, y, JOptionPane.QUESTION_MESSAGE);`

    where `x` is a string for the prompting message and y is a string for the title of the input dialog box, as shown in the example below.



# Output

- In Java, output on the standard output device is accomplished by using the standard output object System.out.
- The object System.out has access to two methods, to output a string on the standard output device:
  - `print`
  - `println`
  - `printf`: print and format the output
- Exercise: open Java API and understand the methods

# Output

- Syntax:

```
System.out.print(expression);
System.out.println(expression);
System.out.println();
System.out.printf(format, arguments);
System.out.printf(locale, format, arguments);
```

# Output

- The expression is evaluated, and its value is printed at the current insertion point on the output device.
- The method `print()` leaves the insertion point after the last character of the value of expression,
- The method `println()` positions the insertion point at the beginning of the next line.
- The statement `System.out.println();` only positions the insertion point at the beginning of the next line.
- The method `printf()` produces output in a specific format and leaves the the insertion point after the last character of the value of expression

# Output

- In an output statement, if expression consists of only one string or a single constant value, then expression evaluates to itself.
- If expression consists of only one variable, then expression evaluates to the value of the variable. When an output statement outputs

```
char ch='A';
System.out.println(ch);
System.out.println('A');
System.out.printf("Hello %s!%n", "World");
```

```
A
A
Hello World!
```

- The operator + works with strings and numeric values

# Output

```
public class Hello {
   public static void main(String[] args){
      System.out.println("Hello World");
      System.out.println("Welcome to Java ");
      System.out.printf("%s", "Hello World");
   }
}
```

```
C:\> javac Hello.java
C:\> java Hello
Hello World
Welcome to Java
Hello World
```

# Output

```
public class Hello {
   public static void main(String[] args){
        System.out.println(29 / 4);
        System.out.println("Hello there.");
        System.out.println(12);
        System.out.println("4 + 7");
        System.out.println(4 + 7);
        System.out.println('A');
        System.out.println("4 + 7 = " + (4 + 7));
        System.out.println(2 + 3 * 5);
        System.out.println("Hello \nthere.");
   }
}
```

```
7
Hello there.
12
4 + 7
11
A
4 + 7 = 11
17
Hello
there.
```

# Exercises

1. What is the result of `25 / 4`? How would you rewrite the expression if you wished the result to be a floating-point number?

2. Show the result of the following code:

```
System.out.println(2 * (5 / 2 + 5 / 2));
System.out.println(2 * 5 / 2 + 2 * 5 / 2);
System.out.println(2 * (5 / 2));
System.out.println(2 * 5 / 2);
```

3. Are the following statements correct? If so, show the output.

```
System.out.println("25 / 4 is " + 25 / 4);
System.out.println("25 / 4.0 is " + 25 / 4.0);
System.out.println("3 * 2 / 4 is " + 3 * 2 / 4);
System.out.println("3.0 * 2 / 4 is " + 3.0 * 2 / 4);
```

# Escape Character

**Table 2-4** Commonly Used Escape Sequences

| | Escape Sequence | Description |
|---|---|---|
| `\n` | Newline | Insertion point moves to the beginning of the next line |
| `\t` | Tab | Insertion point moves to the next tab stop |
| `\b` | Backspace | Insertion point moves one space to the left |
| `\r` | Return | Insertion point moves to the beginning of the current line (*not* the next line) |
| `\\` | Backslash | Backslash is printed |
| `\'` | Single quotation | Single quotation mark is printed |
| `\"` | Double quotation | Double quotation mark is printed |

# Escape Character

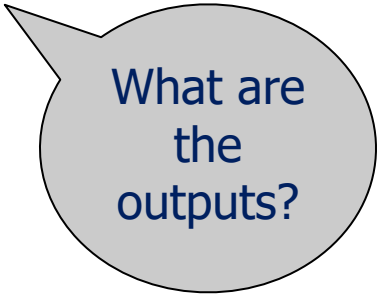
```
public class Hello {
   public static void main(String[] args){
      System.out.print("Hello World!! n Welcome to Java");

      System.out.print("Hello World!!");
      System.out.print("\n Welcome to Java");

      System.out.println("Hello World!!");
      System.out.print("\n Welcome to Java");
   }
}
```

What are the outputs?

```
Hello World!!
Welcome to Java
```

# Escape Character

```
public class Escape_Character {
   public static void main(String[] args){
    System.out.println("The newline escape sequence is \\n");
     System.out.println("The string \"Java\"");
     System.out.println("The tab character is \'\\t\'");
   }
}
```




# The `showMessageDialog` Method

```
JOptionPane.showMessageDialog(null,
  "Welcome to Java!",
  "Example 1.2 Output",
  JOptionPane.INFORMATION_MESSAGE));
```

# Two Ways to Invoke the Method

- There are several ways to use the `showMessageDialog` method. For the time being, all you need to know are two ways to invoke it.
  - use a statement as shown in the example:

    ```
    JOptionPane.showMessageDialog(null, x,
    y, JOptionPane.INFORMATION_MESSAGE));
    ```

    where x is a string for the text to be displayed, and y is a string for the title of the message dialog box.
  - The other is to use a statement like this:

    ```
    JOptionPane.showMessageDialog(null, x);
    ```

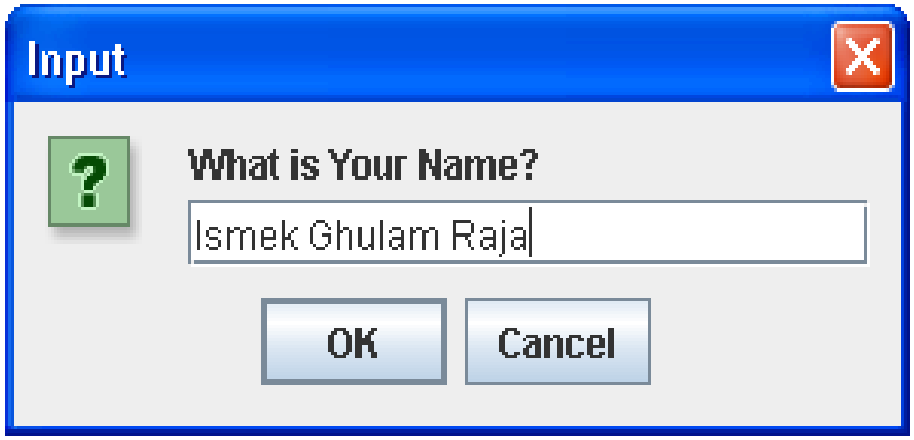
    where x is a string for the text to be displayed.

# Example

```
import javax.swing.JOptionPane;
public class Welcome {
   public static void main(String[] args) {
     String name;
     name = JOptionPane.showInputDialog("What is Your Name?");
     JOptionPane.showMessageDialog(null,"Hi!!! "+name+
       "\nWelcome\nTo\nJava\nWorld");
   }
}
```

# Example

```
import javax.swing.JOptionPane;
public class Welcome {
   public static void main(String[] args) {
        String name;
        name = JOptionPane.showInputDialog("What
           is Your Name?");
        JOptionPane.showMessageDialog(null,
        "Hi!!! "+name+"\nWelcome\nTo\nJava\nWorld");

   }
}
```



# Example

```
import javax.swing.JOptionPane;
public class Welcome {
   public static void main(String[] args) {
        String name;
        name = JOptionPane.showInputDialog("What is Your
   Name?");
        JOptionPane.showMessageDialog(null,"Hi!!!
   "+name+ "\nWelcome\nTo\nJava\nWorld");
   }
}
```


# Common Pitfall 1: Redundant Input Objects

```
Scanner input = new Scanner(System.in);
System.out.print("Enter an integer: ");
int v1 = input.nextInt();

Scanner input1 = new Scanner(System.in);
System.out.print("Enter a double value: ");
double v2 = input1.nextDouble();
```

# Exercises

1. Write the java program using the given pseudocode.
   - Please use dialog box (input dialog+ message dialog) approach;

1. Start
2. Get year_of_birth
3. Calculate age where age = 2021 - year_of_birth
4. Display age
5. End

2. Write a program that reads a number in feet, converts it to meters, and displays the result. One foot is 0.305 meter. Here is a sample run:

```
Enter a value for feet: 16.5 Enter
16.5 feet is 5.0325 meters
```