# Elementary Programming

## Input & Output (+ Case Study 1)


EECS1021:
Object Oriented Programming:
from Sensors to Actuators
Winter 2020

Original slides: DR. CHEN-WEI "JACKIE" WANG
Updates: DR. JAMES ANDREW SMITH

# Topics in “Elementary Program’g” series

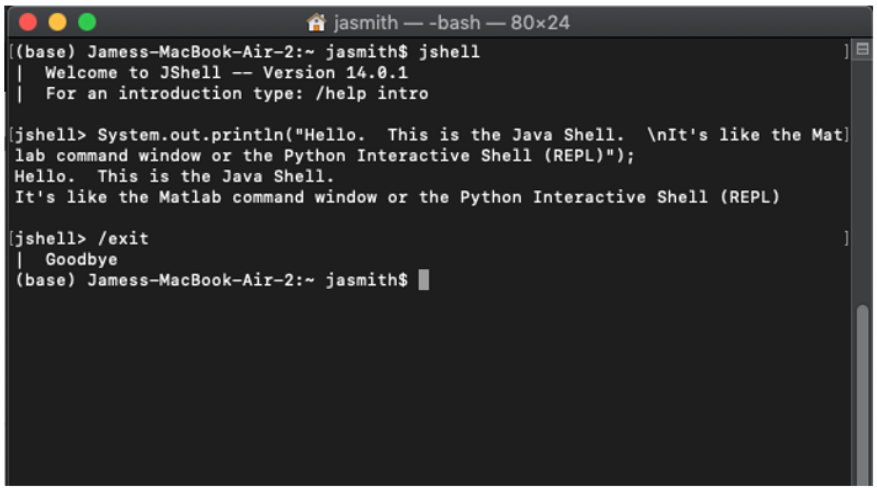
LASSONDE SCHOOL OF ENGINEERING

- Intro to Java (General)
- Operations and Data
- *Input & Output (+ Case Study 1)*
- More I/O (+ Case Study 2)
- Numbers Types & Conversions
- Software Development High Level Process

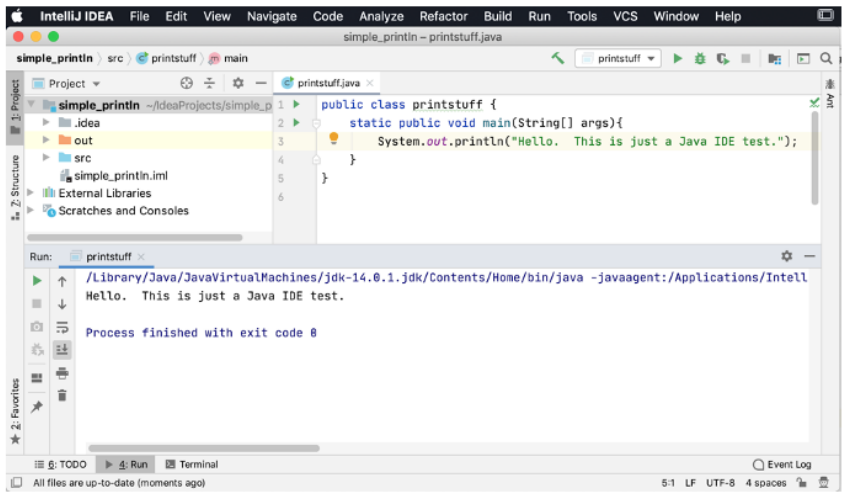
# Simple outputs with `println`

We often use `System.out.println(someString)` to write things to the user.

- e.g. *`System.out.println`*`("Hello. This is the Java Shell.");`




Printing within the Java Shell, jshell.




Printing within an IDE like IntelliJ

# print vs. println

Executing `System.out.println(someString)` is the same as executing `System.out.print(someString + "\n")`.

- e.g.,

```
System.out.print("Hello");
System.out.print("World");
```

```
HelloWorld
```

- e.g.,

```
System.out.println("Hello");
System.out.println("World");
```

```
Hello
World
```

# Common Print Statements

You can execute *print statements* to the console (terminal or within the IDE).

e.g., Print statements involving literals or named constants only:

```
final double PI = 3.14; /* a named double constant */
System.out.println("Pi is " + PI); /* str. lit. and num. const. */ %
```

e.g., Print statements involving variables:

```
String msg = "Counter value is "; /* a string variable */
int counter = 1; /* an integer variable */
System.out.println(msg + counter);
counter = 2; /* re-assignment changes variable's stored value */
System.out.println(msg + counter);
```

# Case Study 1: Compute the Area of a Circle

**Problem**: declare two variables `radius` and `area`, initialize `radius` as 20, compute the value of `area` accordingly, and print out the value of `area`.

```
public class ComputeArea {
  public static void main(String[] args) {
    double radius; /* Declare radius */
    double area; /* Declare area */
    /* Assign a radius */
    radius = 20; /* assign value to radius */
    /* Compute area */
    area = radius * radius * 3.14159;
    /* Display results */
    System.out.print("The area of circle with radius ");
    System.out.println(radius + " is " + area);
  }
}
```

It would be more flexible if we can let the user specify the inputs via keyboard!

# Input and Output

*Reading* input from the console enables *user interaction*.

```
import java.util.Scanner;
public class ComputeAreaWithConsoleInput {
  public static void main(String[] args) {
    /* Create a Scanner object */
    Scanner input = new Scanner(System.in);
    /* Prompt the user to enter a radius */
    System.out.print("Enter a number for radius: ");
    double radius = input.nextDouble();
    /* Compute area */
    final double PI = 3.14169; /* a named constant for π */
    double area = PI * radius * radius; /* area = πr^2 */
    /* Display result */
    System.out.println(
      "Area for circle of radius " + radius + " is " + area);
  }
}
```

# Useful Methods for Scanner

- *nextInt()* which reads an integer value from the keyboard
- *nextDouble()* which reads a double value from the keyboard
- *nextLine()* which reads a string value from the keyboard

# Variables: Common Mistakes (1)

**Mistake**: The same variable is declared more than once.

```
int counter = 1;
int counter = 2;
```

*Fix 1*: Assign the new value to the same variable.

```
int counter = 1;
counter = 2;
```

*Fix 2*: Declare a new variable (with a different name).

```
int counter = 1;
int counter2 = 2;
```

Which fix to adopt depends on what you need!

# Variables: Common Mistakes (2)

**Mistake**: A variable is used before it is declared.

```
System.out.println("Counter value is " + counter);
int counter = 1;
counter = 2;
System.out.println("Counter value is " + counter);
```

*Fix*: Move a variable's declaration before its very first usage.

```
int counter = 1;
System.out.println("Counter value is " + counter);
counter = 2;
System.out.println("Counter value is " + counter);
```

Remember, Java programs are always executed, line by line, *from top to bottom*.