CS1101 Programming and Problem Solving

Dr. Gina Bai

Spring 2023

Logistics

- **PA02 – W1, W2, A, B** on **zyBook > Chap 11**
  - Due: Thursday, Feb 2, at 11:59pm
- **ZY-3** on **zyBook > Assignments**
  - Due: Saturday, Feb 4, at 11:59pm

Recap

1. In the method heading
   `public static double sum (int a, char b)`
   - `a` and `b` are called **parameters**
   - the type of the value returned by the method is **double**

2. When the type of a method is defined as `void`, this indicates
   A. The method returns zero
   B. The method returns nothing

Recap

Q: What's the exact output of the following code?

```
public class RecapMethod {
    public static void main(String[] args){
        int z = 6;
        int x = p(z);
        System.out.print(x);
    }

    public static int p(int temp){
        System.out.print("B");
        return temp + 3;
    }
}
```

B9

Use a Method – When and How?

- Place statements into a method when:
  - The statements are related structurally, and/or
  - The statements are repeated

- You should typically not create methods for:
  - An individual print statement
  - Unrelated or weakly related statements

Passing Parameters

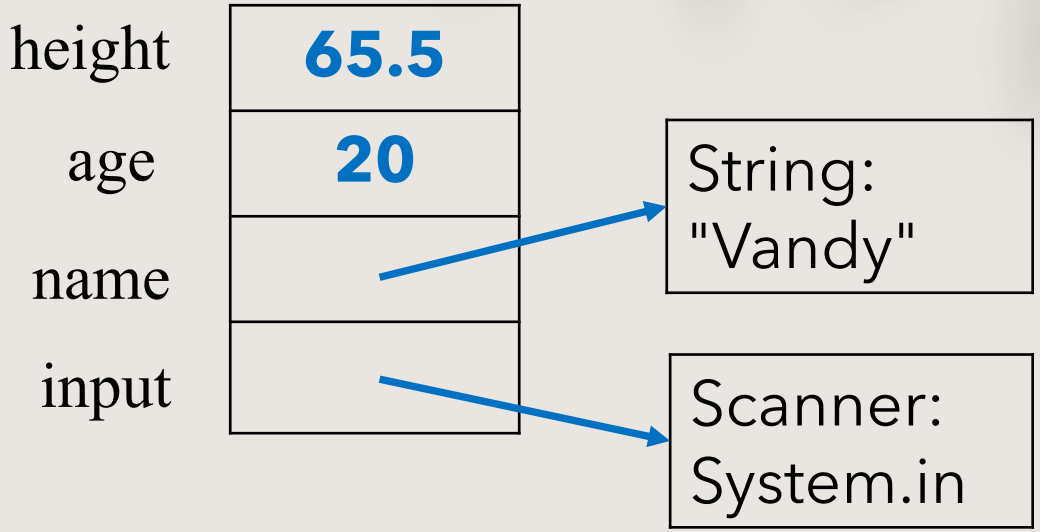
zyBook Chap 3.2, 3.3

Data – <type> <name>

- For **primitive** data types, **values** are stored.
- For **objects**, **references** to objects are stored.

Example:

```
double height = 65.5;
int age = 20;
String name = "Vandy";
Scanner input = new Scanner(System.in);
```



Passing Parameters

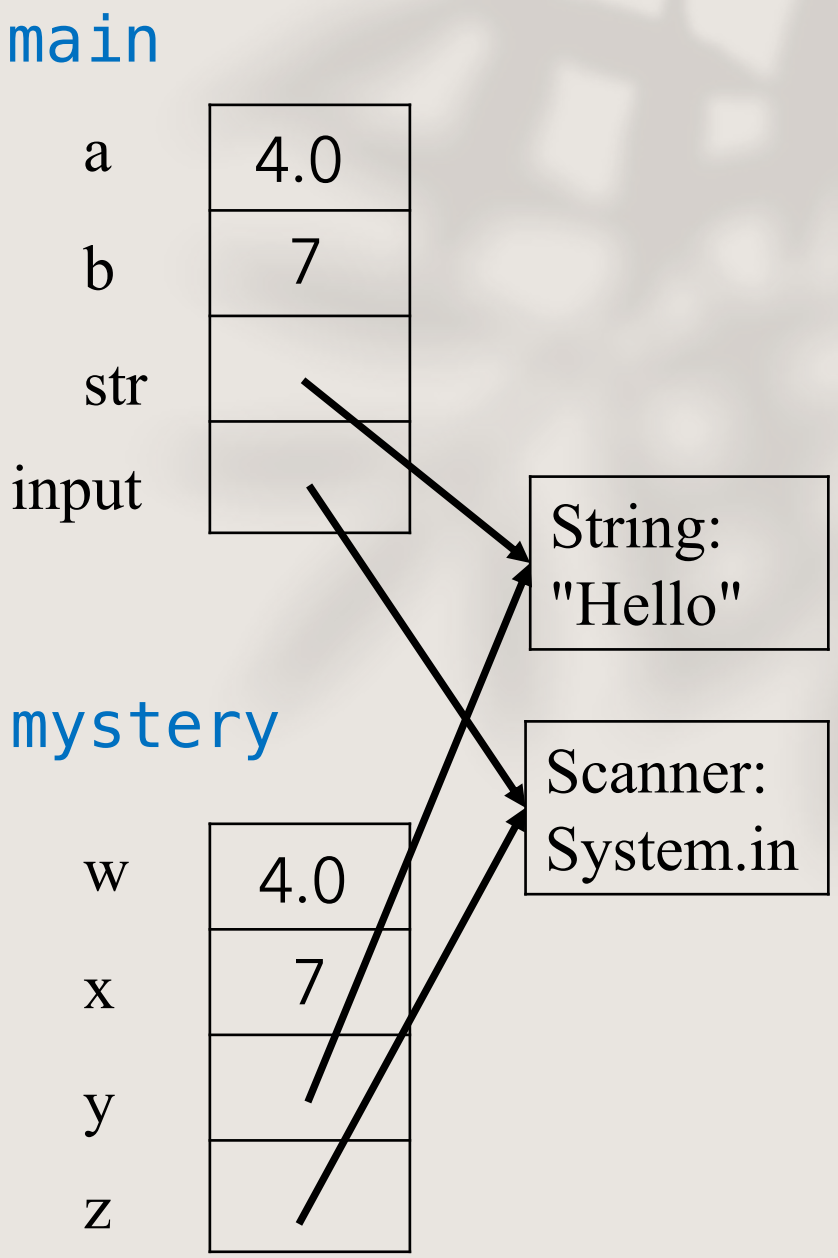
- When a **primitive** data type is passed as a parameter
  - the **value** is copied
  - "pass by value"
- When an **object** is passed as a parameter
  - the **reference** is copied
  - "pass by reference"

```java
import java.util. Scanner;

public class Mystery {

    public static void main (String[] args) {
        double a = 4.0;
        int b = 7;
        String str = "Hello";
        Scanner input = new Scanner(System.in);
        mystery(a, b, str, input);
    }


    public static void mystery (double w, int x, String y, Scanner z) {
        //......
    }
}
```



Scope

- The scope of a variable or a constant is the portion of the code that it is visible.

- A variable or constant is visible **from** the time it is **declared** **until** the **closing curly brace** (}).
  - A variable or constant declared in a method is only visible for that execution of the method.
  - The scope of **parameters** is the method.

Scope

```
import java.util. Scanner;

public class Mystery {

    public static void main (String[] args) {
        double a = 4.0;   // The scope of a starts here
        int b = 7;        // The scope of b starts here
        String str = "Hello";    // The scope of str starts here
        Scanner input = new Scanner(System.in);  // The scope of input starts here
        mystery(a, b, str, input);
    } // The scope of a, b, str, and input ends here

    public static void mystery (double w, int x, String y, Scanner z) {
        //... Scope of w, x, y, z ...
    }
}
```

Q: What's the exact output of the following code?

```
public class ParameterMystery {
    public static void main(String[] args) {
        int a = 4;
        int b = 7;
        int c = -2;

        mystery(a, b, c);   // mystery(4, 7, -2);
        mystery(c, 3, a);
        mystery(a + b, b + c, c + a);
    }
```

c = 4, a = 7, b = -2

```
    public static void mystery(int c, int a, int b) {
        b -= 2;         // b -= 2
        c = a + 5;      // c = 7 + 5
        a -= b;         // a -= -4
        System.out.println(b + " + " + c + " = " + a);
    }                           // -4 + 12 = 11
}
```

Q: What's the exact output of the following code?

```java
public class ParameterMystery {
    public static void main(String[] args) {
        int a = 4;
        int b = 7;
        int c = -2;

        mystery(a, b, c);   // mystery(4, 7, -2);
        mystery(c, 3, a);   // mystery(-2, 3, 4);
        mystery(a + b, b + c, c + a);  // mystery(11, 5, 2);
    }

    public static void mystery(int c, int a, int b) {
        b -= 2;
        c = a + 5;
        a -= b;
        System.out.println(b + " + " + c + " = " + a);
    }
}
```

```
$ javac ParameterMystery.java
$ java ParameterMystery
-4 + 12 = 11
2 + 8 = 1
0 + 10 = 5
```

TopHat

```java
public class PassingParam {

    public static void main(String [] args) {
        int x = 3;
        int y = 2;
        doTheThing(y, x);   // doTheThing(2, 3);
    }

    public static void doTheThing(int x, int y) {
        // Hence, x is 2, y is 3
        x = y - 1;  // x = 3 - 1;
        y = x + 1;  // y = 2 + 1;
        System.out.println(x + " " + y);
    }
}
```

```
$ javac PassingParam.java
$ java PassingParam
2 3
```

Scanner as a Parameter

Scanner as Parameter

- You should only have **one Scanner** for console input.
- If you need to use the console Scanner in multiple methods, it should be passed as a parameter.
  - The parameter is passed by reference.

Code Example

```java
import java.util.Scanner;

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

        Scanner input = new Scanner(System.in);

        System.out.print("Enter course: ");
        String course = input.next();

        System.out.print("Enter credit hours: ");
        int creditInt = input.nextInt();

        System.out.println("[" + course + "] "
                           + creditInt + " credit hours");
    }
}
```

NOTE:
When passed as a parameter, though it's the SAME Scanner, it could be named differently in each method.

```java
import java.util.Scanner;

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

        Scanner input = new Scanner(System.in);

        String course = getCourse(input);

        int creditInt = getHours(input);

        System.out.println("[" + course + "] "
                           + creditInt + " credit hours");
    }

    /**
     * Reads in and returns the course name entered by the user
     * @param console input scanner
     * @return course name
     */
    public static String getCourse(Scanner console) {
        System.out.print("Enter Course: ");
        return console.next();
    }

    /**
     * Reads in and returns the number of credit hours entered by the user
     * @param scnr input scanner
     * @return credit hours
     */
    public static int getHours(Scanner scnr) {
        System.out.print("Enter credit hours: ");
        return scnr.nextInt();
    }
}
```

Coding Practice

Starter Code →

preLec9
PDF document

Eggs
JAVA File

A farm sells organic brown eggs to local customers. They charge $3.25 for a dozen eggs, or 45 cents for individual eggs that are not part of a dozen.

Write a class called **Eggs** that

1. prompts the user for the number of eggs in the order, and then
2. Displays the amount owed with a full explanation in the following format (NOTE: keep 2 decimal places for total):

```
$ javac Eggs.java
$ java Eggs
How many eggs would you like to buy? 27
You ordered 27 eggs. That is 2 dozen at $3.25 per dozen
and 3 loose eggs at 45 cents each for a total of $7.85.
```

Sample Solution

```java
import java.util.Scanner;

public class Eggs {

    public static final double DOZEN_PRICE = 3.25;
    public static final double LOOSE_PRICE = 0.45;

    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        int numEggs = getNumEggs(input);
        calcPrice(numEggs);
    }

    public static int getNumEggs(Scanner input) {
        System.out.print("How many eggs would you like to buy? ");
        return input.nextInt();
    }

    public static void calcPrice(int numEggs) {
        int numDozenEggs = numEggs / 12;
        int numLooseEggs = numEggs % 12;
        double totalPrice = numDozenEggs * DOZEN_PRICE + numLooseEggs * LOOSE_PRICE;

        System.out.printf("You ordered %d eggs. That is %d dozen at $%.2f per dozen\n",
                          numEggs, numDozenEggs, DOZEN_PRICE);
        System.out.printf("and %d loose eggs at %.0f cents each for a total of $%.2f.\n",
                          numLooseEggs, LOOSE_PRICE * 100, totalPrice);
    }
}
```