Lecture 3: Method Parameters

Building Java Programs: A Back to Basics Approach
by Stuart Reges and Marty Stepp

Strings

String is a type used in Java to store literal expressions. It is NOT a primitive type.

```
public static void main(String[] args)
{
        String x="This is a string.";
        System.out.println("x"); // x
        System.out.println(x);//This is a string.
        System.out.println("This is another string.");
}
```

Output:

```
x
This is a string.
This is another string.
```

Method Parameters

Methods in Java can have parameters or input arguments.

Syntax:

```
public static void methodName(type var1)
{ <statements>} // declare
```

The parameters in the method header are **formal parameters.**

```
methodName(expression); // call
```

The parameters in the method header are **actual parameters.**

Method Parameters

Example:

```
public static void printDouble(int x)
{
        System.out.println("Your number " + x
                                + "doubled is" + 2*x +".");
        x=x+2;

}

To call:
printDouble(5); //Your number 5 doubled is 10.
printDouble(3.4); //Error! Incompatible types!
```

Method Parameters

Methods cannot change the values of primitive types.

```
public static void main(String[] args)
{
        int x=3;
        printDouble(4);
        printDouble(x);
        System.out.println("x is now " + x);
        }
```

```
Output:
Your number 4 doubled is 8.
Your number 3 doubled is 6.
x is now 3
```

Note: The value of x did not change.

Method Parameters

Methods in Java can have return types.

```
public static type methodName(type var1,…, type var2)
{ <statements>} // declare
```

Example:

```
public static int doubleThis(int n)
{
        int m=2*n;
        return m;
        // or simply return 2*n;
}

int a=doubleThis(9); //a is now 18
```

Return

```
public static void main(String[] args){
      int x=6;
      int y;
      doubleThis(x); //returned value NOT SAVED!
      y=doubleThis(x); // y is now 12.
      System.out.println("x doubled is "
                                    + doubleThis(x));
      System.out.println("x is now " + x);

}
```

Output:

```
x doubled is 12
x is now 6
```

Value semantics

- **value semantics**: When primitive variables (`int`, `double`) are passed as parameters, their values are copied.
  - Modifying the parameter will not affect the variable passed in.

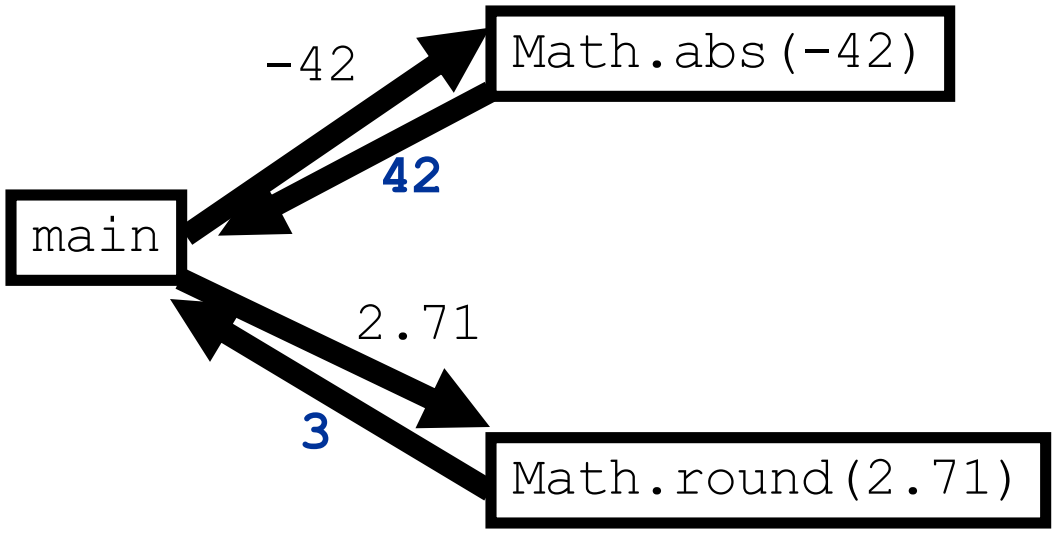
```
public static void strange(int x) {
    x = x + 1;
    System.out.println("1. x = " + x);
}

public static void main(String[] args) {
    int x = 23;
    strange(x);
    System.out.println("2. x = " + x);
    ...
}
```

Output:

```
1. x = 24
2. x = 23
```

Return

- **return**: To send out a value as the result of a method.
  - The opposite of a parameter:
    - Parameters send information *in* from the caller to the method.
    - Return values send information *out* from a method to its caller.
      - A call to the method can be used as part of an expression.



Common error: Not storing

- Many students incorrectly think that a `return` statement sends a variable's name back to the calling method.

```
public static void main(String[] args) {
    slope(1, 0, 6, 3);
    System.out.println("The slope is " + result);  // ERROR:
}                                                  // result not defined

public static double slope(int x1, int x2, int y1, int y2) {
    double dy = y2 - y1;
    double dx = x2 - x1;
    double result = dy / dx;
    return result;
}
```

Fixing the common error

- Instead, returning sends the variable's *value* back.
  - The returned value must be stored into a variable or used in an expression to be useful to the caller.

```
public static void main(String[] args) {
    double s = slope(1, 0, 6, 3);
    System.out.println("The slope is " + s);
  //System.out.println("The slope is " + slope(1,0,6,3));
}

public static double slope(int x1, int x2, int y1, int y2) {
    double dy = y2 - y1;
    double dx = x2 - x1;
    double result = dy / dx;
    return result;
}
```

Return examples

- Example:

```
// Returns the slope of the line between the given points.
public static double slope(int x1, int y1, int x2, int y2) {
    double dy = y2 - y1;
    double dx = x2 - x1;
    return dy / dx;
}
```

  - `slope(1, 3, 5, 11)` returns `2.0`

Return examples

```
// Converts degrees Fahrenheit to Celsius.
public static double fToC(double degreesF) {
    double degreesC = 5.0 / 9.0 * (degreesF - 32);
    return degreesC;
}

// Computes triangle hypotenuse length given its side lengths.
public static double hypotenuse(int a, int b) {
    double c = Math.sqrt(a * a + b * b);
    return c;
}
```

- You can shorten the examples by returning an expression:

```
public static double fToC(double degreesF) {
    return 5.0 / 9.0 * (degreesF - 32);
}
```

Find the errors

```
Public class Parameters{
      public static void main(String[] args){
            double bubble=867.53, x=10.01, y=5.3;
            printer(double x, double y);
            printer(x);
            printer("barack", "obama");
            System.out.println("z = " + z);
      }
      public static void printer(x, y){
        int z = 5;
        System.out.println("x = " + double x+ " and y= "+y);
        System.out.println("The value from main is
                                                "+bubble);
}}
```

Find the errors

```
Public class Parameters{
      public static void main(String[] args){
            double bubble=867.53, x=10.01, y=5.3;
            printer(double x, double y);
            printer(x); // need two arguments.
            printer("barack", "obama");// not right types
            System.out.println("z = " + z);// z is local to
                                              //printer
      }
      public static void printer(x, y){
        int z = 5;
        System.out.println("x = " + double x+ " and y= "+y);
        System.out.println("The value from main is
                                              "+bubble);
}}
```

Corrected Version

```
public class Parameters{
        public static void main(String[] args){
                double bubble=867.53, x=10.01, y=5.3;
                printer(double x, double y);
                printer(x,y); // need two arguments.
                printer("barack", "obama");// not right types
                int z=8;
                System.out.println("z = " + z);// z is local to printer
        }
        public static void printer(double x, double y){
                int z = 5;
                System.out.println("x = " + double x+ " and y= "+y);
                System.out.println("The value from main is " + bubble);
}}
```

Lab

Create a folder called Parameters in CS50 IDE.

Write a class with the following five static methods.

```
// given two integers x and y, returns their average.
public static double average(int x, int y)
{…}


// given two points (x1, y1) and (x2,y2), returns
// the slope of the line through them. You may assume
// x1 is not equal to x2.
public static double slope(int x1,int y1,int x2,int y2)
{…}
```

Lab

```
// given two integers x and y, returns the difference x-y
public static int difference(int x, int y)
{…}

// given an integer x returns its square x*x.
public static int square(int x)
{…}

// given two points on the plane, returns the distance between them.
// You MUST CALL the methods difference and square above.
// In addition, you CANNOT use subtraction nor multiplication in this method.
public static double distance(int x1, int y1, int x2, int y2)
{…}
```

Lab

Write your main() method so that your program has an output similar to:

```
The average of 8 and 9 is 8.5
The slope of the line between (8,9) and (2,4) is 0.8333333333333334
The distance between (8,9) and (2,4) is 7.810249675906654
```

Notice the format of the points on the coordinate plane.

Lab 2

Modify the previous program by adding a method printMath so that the call

```
printMath(8,9,2,4);
```

Gives the output:

```
The average of 8 and 9 is 8.5
The slope of the line between (8,9) and (2,4) is 0.8333333333333334
The distance between (8,9) and (2,4) is 7.810249675906654
```

Use the following method header:

```
public static void printMath(int x, int y, int
a, int b) {…}
```