CS 106A, Lecture 7
Parameters

reading:

Art & Science of Java, 5.1 - 5.5

Redundant recipes

- Recipe for baking **20** cookies:
  - Mix the following ingredients in a bowl:
    - **4** cups flour
    - **1** cup butter
    - **1** cup sugar
    - **2** eggs
    - **40** oz. chocolate chips ...
  - Place on sheet and Bake for about 10 minutes.

- Recipe for baking **40** cookies:
  - Mix the following ingredients in a bowl:
    - **8** cups flour
    - **2** cups butter
    - **2** cups sugar
    - **4** eggs
    - **80** oz. chocolate chips ...
  - Place on sheet and Bake for about 10 minutes.

Parameterized recipe

- Recipe for baking **20** cookies:
  - Mix the following ingredients in a bowl:
    - **4** cups flour
    - **1** cup sugar
    - **2** eggs
    - **...**

- Recipe for baking **N** cookies:
  - Mix the following ingredients in a bowl:
    - **N/5** cups flour
    - **N/20** cups butter
    - **N/20** cups sugar
    - **N/10** eggs
    - **2N** oz. chocolate chips ...
  - Place on sheet and Bake for about 10 minutes.

- **parameter**: A value that distinguishes similar tasks.

Drawing boxes

boxOfStars

- Consider the task of printing the following boxes:

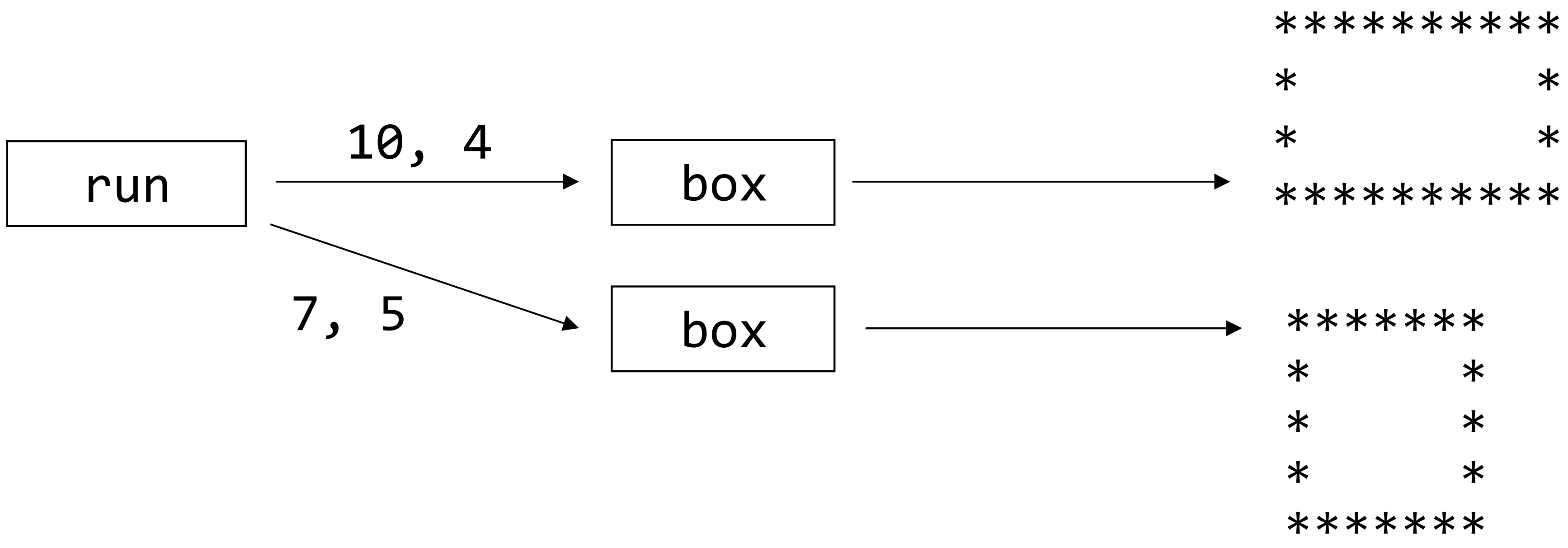
```
**********
*        *
*        *
**********

*******
*     *
*     *
*     *
*     *
*******
```

  - The code to draw each box will be very similar.
    - Would variables help? Would constants help?

Parameters

- **parameter**: A value passed to a method by its caller.
  - Write a method **box** to draw a box of any size.
    - When *declaring* the method, we will state that it requires the caller to tell it the width and height of the box.
    - When *calling* the method, we will specify the width and height to use.



```
***********
*         *
*         *
***********
```

```
*******
*     *
*     *
*     *
*******
```

Declaring a parameter

Stating that a method requires a parameter in order to run

```
public void name(type name) {
    statements;
}
```

- Example:

```
public void password(int code) {
    println("The password is: " + code);
}
```

  - When `password` is called, the caller must specify the integer code to print.

Passing a parameter

Calling a method and specifying values for its parameters

```
methodName(expression);
```

- Example:

```
public void run() {
    password(42);
    password(12345);
}
```

Output:

```
The password is 42
The password is 12345
```

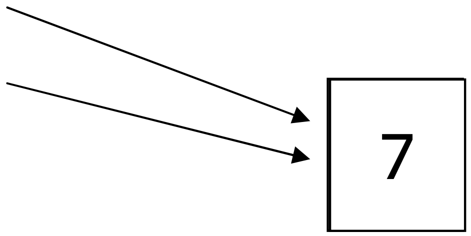
- Illegal to call without passing an `int` for that parameter.

```
password();       // Error
password(3.7);    // Error
```

How params are passed

- When the method is called:
  - The value is stored into the parameter variable.
  - The method's code executes using that value.

```
public void run() {
    chant(3);
    chant(7);
}
```



```
public void chant(int times) {
    for (int i = 0; i < times; i++) {
        println("Java is great!");
    }
}
```

Multiple parameters

- A method can accept multiple parameters separated by commas: `,`
  - When calling it, you must pass values for each parameter.

- Declaration:

```
public void name(type name, ..., type name) {
    statements;
}
```

- Call:

```
name(value, value, ..., value);
```

- *Exercise:* Write the box-drawing program using parameters.

Boxes solution

```
public class Boxes extends ConsoleProgram {
    public void run() {
        box(10, 3);
        box(5, 4);
        box(20, 7);
    }

    public void line(int count) {                // Prints the given
        for (int i = 1; i <= count; i++) {       // number of stars
            print("*");                          // plus a line break.
        }
        println();
    }

    public void box(int width, int height) {      // Prints a box of *
        line(width);                              // of the given size.
        for (int line = 1; line <= height - 2; line++) {
            print("*");
            for (int space = 1; space <= width - 2; space++) {
                print(" ");
            }
            println("*");
        }
        line(width);
}   }
```

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 void strange(int x) {
    x = x + 1;
    println("2: x = " + x);
}

public void run() {
    int x = 23;
    println("1: x = " + x);
    strange(x);
    println("3: x = " + x);
}
```

Output:

```
1: x = 23
2: x = 24
3: x = 23
```

"Parameter mystery"

ParameterMystery1

- **Q:** What is the output of the following program?

```
public class ParameterMystery extends ConsoleProgram {
    public void run() {
        int x = 9;
        int y = 2;
        int z = 5;

        mystery(z, y, x);
        mystery(y, x, z);
    }

    public void mystery(int x, int z, int y) {
        println(z + ", " + (y - x));
    }
}
    //  A.      B.       C.       D.        E.
    //  2, 4    5, -7    9, 3     z, y-x    N/A
    //  9, 3    5, -7    2, 4     y, x-z
```

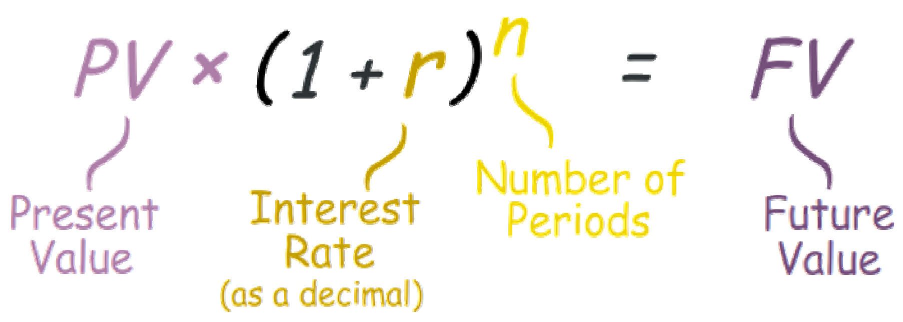
Investment exercise

- Given this formula for compound interest, write a program **Investment** that calculates money earned by two investors.
  - Also report the overall "quality" of the investment as from the table below.

```
Investor #1:
Initial amount? 100.00
Interest rate%? .03
Num. of months? 5
Final amount = $115.93
Profit = $15.93 (16%)
medium

Investor #2:
Initial amount? 5.25
Interest rate? .08
Num. of months? 24
Final amount = $33.29
Profit = $28.04 (534%)
strong

Have a nice day!
```



Profit	Category
0 - 10%	weak
10 - 50%	medium
over 50%	strong

Investment solution (1/3)

```
/*
 * Prompts the user for information about two investments
 * with compound interest and calculates the final amount
 * along with a quality rating for each investment.
 */
import acm.program.*;

public class Investment extends ConsoleProgram {
    public void run() {
        invest(1);
        invest(2);
        println("Have a nice day!");
    }

    ...
```

Investment solution (2/3)

```
// Reads investment info for one person.
public void invest(int number) {
    // prompt for investment information
    println("Investor #" + number + ":");
    double initial = readDouble("Initial amount? ");
    double percent = readDouble("Interest rate%? ");
    int months = readInt("Num. of months? ");

    // calculate final amount using compound interest
    double finalAmount = initial;
    for (int i = 0; i < months; i++) {
        finalAmount += percent * finalAmount;
    }

    // report results
    println("Final amount = $" + finalAmount);
    report(initial, finalAmount);
}
```

Investment solution (3/3)

```
    // Calculates profit earned and reports overall quality
    // of the investment as weak, medium, or strong.
    public void report(double initial, double finalAmount) {
        // compute profit
        double profit = finalAmount - initial;
        double percent = 100.0 * profit / initial;
        println("Profit = $" + profit + " ("
                + percent + "%)");

        // report quality of investment
        if (percent < 10) {
            println("weak");
        } else if (percent < 50) {
            println("medium");
        } else {
            println("strong");
        }
        println();   // blank line
    }
}
```

Formatting with printf

```
printf("format string", parameters);
```

- A format string can contain placeholders to insert parameters:

`%d`	integer
`%f`	real number
`%s`	string

  - these placeholders are used instead of + concatenation
  - write %% to print a % sign

- Example:

```
int w = 13;
int h = 7;
printf("size is %d by %d!\n", w, h);
// output:  size is 13 by 7!
```

  – `printf` does not drop to the next line unless you write \n

Specifying a width

`%`*`W`*`d`	integer, *W* characters wide, right-aligned
`%-`*`W`*`d`	integer, *W* characters wide, left-aligned
`%`*`W`*`f`	real number, *W* characters wide, right-aligned
…	

- Example:

```
for (int i = 1; i <= 3; i++) {
    for (int j = 1; j <= 10; j++) {
        printf("%4d", i * j);
    }
    println();   // to end the line
}
```

- Output:

```
   1   2   3   4   5   6   7   8   9  10
   2   4   6   8  10  12  14  16  18  20
   3   6   9  12  15  18  21  24  27  30
```

Specifying precision

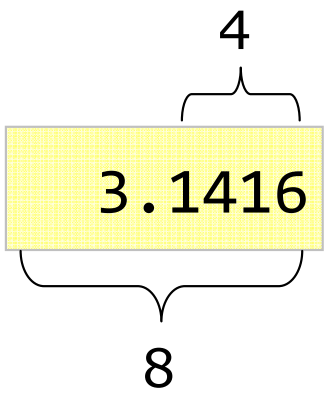
`%.`***`D`***`f`	real number, rounded to *D* digits after decimal
`%`***`W.D`***`f`	real number, *W* chars wide, *D* digits after decimal
`%-`***`W.D`***`f`	real number, *W* wide (left-align), *D* after decimal

- Example:

```
double pi = 3.14159;
printf("Pi is %.2f\n", pi);
printf("More precisely: %8.4f\n", pi);
```

- Output:

```
Pi is 3.14
More precisely:   3.1416
```



BMI exercise

Given this formula for body mass index (BMI):

$$BMI = \frac{weight}{height^2} \times 703$$

BMI	Category
below 18.5	class 1
18.5 - 24.9	class 2
25.0 - 29.9	class 3
30.0 and up	class 4

- Write the following program:

```
Person 1's information:
height (in inches)? 70.0
weight (in pounds)? 194.25
BMI = 27.868928571428572
class 3

Person 2's information:
height (in inches)? 62.5
weight (in pounds)? 130.5
BMI = 23.485824
class 2

Have a nice day!
```

BMI solution

```
/* This program computes two people's body mass index (BMI) and
 * compares them.  The code uses methods with parameters.
 */
import acm.program.*;

public class BMI extends ConsoleProgram {
    public void run() {
        person(1);
        person(2);
        println("Have a nice day!");
    }

    /* Reads info for one person and computes their BMI */
    public void person(int number) {
        println("Person " + number + "'s information:");
        double height = readDouble("height (in inches)? ");
        double weight = readDouble("weight (in pounds)? ");
        double bmi = weight * 703 / height / height;
        report(bmi);
    }

    ...
```

BMI solution, cont'd.

```
    /* Outputs information about a person's BMI and weight status */
    public void report(double bmi) {
        println("BMI = " + bmi);
        if (bmi < 18.5) {
            println("class 1");
        } else if (bmi < 25) {
            println("class 2");
        } else if (bmi < 30) {
            println("class 3");
        } else {
            println("class 4");
        }
    }
}
```