"We're flooding people with information. We need to feed it through a processor. A human must turn information into intelligence or knowledge. **We've tended to forget that no computer will ever ask a new question.**"

— Rear Admiral Grace Murray Hopper
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.
Consider the task of printing the following lines/boxes:

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

**********

***********************************

**********

*****

*   *

*   *

*****

**********

*        *

**********

*****

*   *

*   *

*****

```

Redundant figures
A redundant solution

public class Stars1 {
    public static void main(String[] args) {
        lineOf13();
        lineOf7();
        lineOf35();
        box10x3();
        box5x4();
    }
    public static void lineOf13() {
        for (int i = 1; i <= 13; i++) {
            System.out.print("*");
        }
        System.out.println();
    }
    public static void lineOf7() {
        for (int i = 1; i <= 7; i++) {
            System.out.print("*");
        }
        System.out.println();
    }
    public static void lineOf35() {
        for (int i = 1; i <= 35; i++) {
            System.out.print("*");
        }
        System.out.println();
    }
    ...
Parameterization

- **parameter**: A value passed to a method by its caller.

  - Instead of `lineOf7`, `lineOf13`, write `line` to draw any length.

  - When *declaring* the method, we will state that it requires a parameter for the number of stars.
  - When *calling* the method, we will specify how many stars to draw.
Declaring a parameter

Stating that a method requires a parameter in order to run

```java
public static void <name>(<type> <name>) {
    <statement>(s);
}
```

Example:

```java
public static void sayPassword(int code) {
    System.out.println("The password is: " + code);
}
```

– When `sayPassword` is called, the caller must specify the integer code to print.
Passing a parameter

Calling a method and specifying values for its parameters

\[
\text{name}(\text{expression}) ;
\]

Example:

```java
public static void main(String[] args) {
    sayPassword(42);
    sayPassword(12345);
}
```

Output:

```
The password is 42
The password is 12345
```
A parameter can guide the number of repetitions of a loop.

```java
public static void main(String[] args) {
    chant(3);
}

public static void chant(int times) {
    for (int i = 1; i <= times; i++) {
        System.out.println("Just a salad...");
    }
}
```

Output:

Just a salad...
Just a salad...
Just a salad...
Just a salad...
How parameters are passed

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

```java
public static void main(String[] args) {
    chant(3);
    chant(7);
}
```

```java
public static void chant(int times) {
    for (int i = 1; i <= times; i++) {
        System.out.println("Just a salad...");
    }
}
```
Common errors

- If a method accepts a parameter, it is illegal to call it without passing any value for that parameter.
  
  ```java
  chant(); // ERROR: parameter value required
  ```

- The value passed to a method must be of the correct type.
  
  ```java
  chant(3.7); // ERROR: must be of type int
  ```

- Exercise: Change the Stars program to use a parameterized method for drawing lines of stars.
// Prints several lines of stars.
// Uses a parameterized method to remove redundancy.
public class Stars2 {
    public static void main(String[] args) {
        line(13);
        line(7);
        line(35);
    }

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

- A method can accept multiple parameters. (separate with , )
  - When calling it, you must pass values for each parameter.

- Declaration:
  ```java
  public static void <name>(<type> <name>, ..., <type> <name>) {
    <statement>(s);
  }
  ```

- Call:
  ```java
  <name>(<exp>, <exp>, ..., <exp>);
  ```
public static void main(String[] args) {
    printNumber(4, 9);
    printNumber(17, 6);
    printNumber(8, 0);
    printNumber(0, 8);
}

public static void printNumber(int number, int count) {
    for (int i = 1; i <= count; i++) {
        System.out.print(number);
    }
    System.out.println();
}

Output:
444444444
171717171717
000000000

Modify the Stars program to draw boxes with parameters.
Stars solution

// Prints several lines and boxes made of stars.
// Third version with multiple parameterized methods.

public class Stars3 {
    public static void main(String[] args) {
        line(13);
        line(7);
        line(35);
        System.out.println();
        box(10, 3);
        box(5, 4);
        box(20, 7);
    }

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

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

```java
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
```
public class ParameterMystery {
    public static void main(String[] args) {
        int x = 9;
        int y = 2;
        int z = 5;

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

public static void mystery(int x, int z, int y) {
    System.out.print(z + " " + (y - x) + " ");
}

A. 5 -7 5 -7  
B. 9 -3 5 7  
C. 2 4 9 3  
D. 9 -3 5 12  
E. None of A through D
Clicker 2 - What is output by the following code?

int x = 2;
int y = 5;
mystery2(x, y);
System.out.print(x + " " + y + " ");

public static void mystery2(int x, int y) {
    System.out.print(x + " " + y + " ");
    x *= y + 3;
    y--;
    x++;
    System.out.print(x + " " + y + " ");
}

A. 2 5 17 4 2 5
B. 2 5 17 4 17 4
C. 17 4 2 5 17 4
D. 2 5 2 5 17 4
E. None of A through D
Recall: Strings

- **string**: A sequence of text characters.

  String `<name>` = "<text>";
  String `<name>` = <expression resulting in String>;

- Examples:

  String name = "Marla Singer";
  int x = 3;
  int y = 5;
  String point = "(" + x + ", " + y + ")";
Are Strings a primitive data type just like \texttt{int} and \texttt{double}? \\
A. No \\
B. Yes
Strings as parameters

```java
public class StringParameters {
    public static void main(String[] args) {
        sayHello("Marty");
        String teacher = "Bictolia";
        sayHello(teacher);
    }

    public static void sayHello(String name) {
        System.out.println("Welcome, " + name);
    }
}
```

Output:

Welcome, Marty
Welcome, Bictolia

Modify the Stars program to use string parameters. Use a method named `repeat` that prints a string many times.
Stars solution

// Prints several lines and boxes made of stars.
// Fourth version with String parameters.

public class Stars4 {
    public static void main(String[] args) {
        line(13);
        line(7);
        line(35);
        System.out.println();
        box(10, 3);
        box(5, 4);
        box(20, 7);
    }

    // Prints the given number of stars plus a line break.
    public static void line(int count) {
        repeat("*", count);
        System.out.println();
    }

    ...
}
Stars solution, cont'd.

... 

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

// Prints the given String the given number of times.
public static void repeat(String s, int times) {
    for (int i = 1; i <= times; i++) {
        System.out.print(s);
    }
}