Parameterized recipe

- 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 N cookies:
  - Mix the following ingredients in a bowl:
    - \( \frac{N}{5} \) cups flour
    - \( \frac{N}{20} \) cups butter
    - \( \frac{N}{20} \) cups sugar
    - \( \frac{N}{10} \) eggs
    - 2N oz. chocolate chips...
  - Place on sheet and Bake for about 10 minutes.

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

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.

Redundant figures

Consider the task of printing the following lines/boxes:

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

- Parameter: A value that distinguishes similar tasks.
A redundant solution

```java
public class Stars1 {
   public static void main(String[] args) {
      lineOf3();
      lineOf7();
      lineOf13();
      box4();
   }
   public static void lineOf13() {
      for (int i = 1; i <= 13; i++) {
         System.out.println("*");
      }
   }
   public static void lineOf7() {
      for (int i = 1; i <= 7; i++) {
         System.out.println("*");
      }
   }
   public static void lineOf3() {
      for (int i = 1; i <= 3; i++) {
         System.out.println("*");
      }
   }
   public static void box4() {
      System.out.println();
   }
   ...
```

- This code is redundant.
- Would variables help? Would constants help?
- What is a better solution?
- line - A method to draw a line of any number of stars.
- box - A method to draw a box of any size.

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.

```
public class Stars2 {
   public static void main(String[] args) {
      line(7);
      line(13);
   }
   public static void line(int length) {
      for (int i = 1; i <= length; i++) {
         System.out.println("*");
      }
   }
}
```

Declaring a parameter

**Stating that a method requires a parameter in order to run**
```
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**
```
$name (<expression>) ;
```

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

**Output:**
The password is 42
The password is 12345
Parameters and loops

- 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...

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);
}

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.

Stars solution

```java
// 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, you must pass values for each parameter.

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

- Call:
  <name>(<exp>, <exp>, ..., <exp>);

Multiple parameters example

```java
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:
44444444
1717171717
00000000

- Modify the Stars program to draw boxes with parameters.

Stars solution

```java
// 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("*");
        }
        System.out.println();
    }

    // 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);
    }
}
```

Stars solution, cont'd.

...
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.

```java
public static void main(String[] args) {
    int x = 23;
    mystery2(x);
    System.out.println("2. x = "+x);
}
```

```
A. 5 -7   B. 9 -3   C. 2 4   D. 9 -3   E. None of
      5 -7   5 7   9 3   5 12   A - D
```

What is output by the following code?

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

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

```
A. 5 2   B. 2 5   C. 2 5   D. 5 2   E. 2 5
  5 2   5 2   5 2   5 2   5 2
```

Recall: Strings

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

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

- Examples:

```java
String name = "Marla Singer";
int x = 3;
int y = 5;
String point = "(" + x + ", " + y + ")";
```
clicker

Are Strings a primitive data type just like int and double?
A. Yes
B. No

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

```java
// 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();
    }
    // 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.println(" ");
            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);
        }
    }
}
```

Stars solution, cont'd.

...