Functions!!!

we've already been using lots of functions:

- `System.out.println()`
- `scanner.nextLine()`
- `scanner.nextInt()`

System → class
out → class (inside System class)
println() → function (inside of 'out' class)
scanner → class instance
nextLine() → function (inside of scanner class instance)
nextInt() → function (inside of scanner class instance)
Functions!!!

- Why functions? Functions provide good way to design systems!
- And we can define our own functions...
Functions!!!

```java
/**
 * @param number
 * @return true if the <code>number</code> is even; otherwise false
 */
private static boolean isEven(int number) {
    return (number%2 == 0);
}
```
Coding Design!

- DRY principle - Don't Repeat Yourself!
  - Loops help with this too
  - But functions (procedures/methods) help even more!
- Modular design
  - Functions/procedures/modules help with this!
Functions!!!

• comprised of:
  • name – use names that describe what the function does
  • parameters (optional) - variables passed to the function
  • code - statements that run on given parameters
  • return value (optional) - result returned by the function
Variable Scope

• Scope is the context in which a variable is accessible (what code is the variable accessible in)

• Two main types of variables and their scope:
  • Global variables
  • Local variables
Global Variables

- Global variables - variables declared in the class, but outside of all the function.
  - accessible globally and modifiable from any function in that class
Global Variables

The Bad:

• variables take up space (memory) while in existence and global variables exist as long as that class exists
• variables exist outside of where they may be needed cluttering up the class
• makes functions dependent on outside data
Local Variables

• Local variables - variables that are defined inside a specific code context and are not accessible outside of that context. In this case inside of a function.

• In functions we sometimes call these 'function variables'
Local Variables

The Good:

- Deleted after function is done running (saves memory)
- Using local variables instead of global variables ensures good modular design
  - in general, a function shouldn't be dependent on code outside of itself, it should be self contained.
Parameters

Parameters are variables that are passed to a function.

- Create local copies of the value passed to it
- Java is 'pass-by-value'
  - Changing the contents of a parameter does not change the contents of a original variable passed to that function.
- But we must go deeper...
Variables

Variables give us access to data that is stored in memory.

Two types of variables:

- Primitives (char/int/float/boolean/etc.)
- Classes/Objects/Arrays (Strings/Integer/Float/etc.)
Variables

- Primitives store the actual value
- Classes/Objects/Arrays store a reference to the location in memory where the data is stored
Return Values

Pass variables in, option to pass one back
Libraries

- A library is a set of utilities that provide functionality that apply to specific topics (using functions).
  - system
  - networking
  - graphics
  - crypto
  - I/O
  - etc.