Functions (I)

What are functions good for?

- Help **avoid** code **repetition**
  - parameterization of functional blocks, encapsulation of code
- **Scalability**: build large and robust software systems
  - modular, top-down design, unit testing of code modules
- Portability and **re-use**

Terminology

- **Subroutine/Procedure**: A call/return block of code that **does not return a value**, but may take arguments and modify their value.
- **Function**: A call/return block of code that **returns a value**. Functions may take arguments and modify their value.
- **Method**: A function or subroutine that is **part of a class**.

- **Argument**: An **expression used when calling a function** or subroutine.
- **Parameter**: The **identifier declared in** a function or subroutine **definition** that will connect to the corresponding argument in a function call.

Parameter Passing

- **Pass by value**: the computer evaluates the argument expression and places a copy of its value into the memory address referenced by the parameter. (C, C++, Java)
- **Pass by reference**: the computer evaluates the argument expression and places a reference to the result into the memory address referenced by the parameters. (C++)

- Show Swap.java, ask the output for swap(int, int) [won’t swap a and b]
- ask the output for swap(Integer, Integer) [won’t swap a and b]
- ask the output for resetVal (Item item) [reset the item val]
- Why? [Java is pass by value]
```java
public class Swap {
    public static void swap (int a, int b) {
        int t = a;
        a = b;
        b = t;
    }

    // If change the reference to refer to another location,
    // any changes to the reference are not reflected
    // back in the main function
    public static void swap (Integer a, Integer b) {
        Integer t = a;
        a = b;
        b = a;
    }

    // If change the member of an object referred by the reference
    // the changes are reflected back, as even it's a copy of
    // the object's reference, it refers to the same object
    public static void resetVal (Item item) {
        item.val = 0;
    }

    public static void main (String[] args) {
        int a = 5;
        int b = 10;
        System.out.println("Before swapping a = " + a + " b = " + b);
        swap(a, b);
        System.out.println("After swapping a = " + a + " b = " + b);
        Integer i = 5;
        Integer j = 10;
        System.out.println("Before swapping i = " + i + " j = " + j);
        swap(i, j);
        System.out.println("After swapping i = " + i + " j = " + j);
        Item t = new Item(5);
        System.out.println("Before resetting t.val = " + t.val);
        resetVal(t);
        System.out.println("After resetting t.val = " + t.val);
    }
}

class Item {
    int val;

    public Item (int v) {
        val = v;
    }
}
```

```
mpb-190250:Functions yingli$ javac Swap.java
mpb-190250:Functions yingli$ java Swap
Before swapping a = 5 b = 10
After swapping a = 5 b = 10
Before swapping i = 5 j = 10
After swapping i = 5 j = 10
Before resetting t.val = 5
After resetting t.val = 0
mpb-190250:Functions yingli$
```