CS139 – Methods

Let's Look At Houses...

Advantages of Methods

- Avoid code repetition – Methods are re-usable. We can mix and match existing methods to solve new problems.
- Simplify problem-solving – Humans can only solve large problems by decomposing them into smaller problems, which may themselves need to be broken into smaller problems...
- Simplify testing – Individual methods may be tested in isolation.

Terminology

- There are several names for the same general idea:
  - procedure (most often used when no value is returned)
  - function (most often used when a value *is* returned)
  - subroutine
  - method
- In Java (and most OO languages) we use the term method.

Quiz 1: What Will Be Printed?

```java
public class MethodDemo {

   public static void main(String[] args) {
       int a;
       int b;
       int c;

       a = 2;
       b = 3;
       c = methodOne(b, a);
       System.out.println(a + " " + b + " " + c);
   }

   public static int methodOne(int a, int b) {
      int result;

      result = a * 2 + b;
      a = 6;
      return result;
   }

}
```

Two Issues

- Scope – The region of code where a variable can be seen/accessed.
  - Variables defined inside methods are called local variables – visible only inside that method.

- Pass by value – In Java, methods receive a copy of their arguments Changing the parameter variable only changes the copy.

Quiz 2: What Will Be Printed?

```java
public class MethodDemo {

    public static int methodOne(int a, int b) {
        int result;

        result = a * 2 + b;
        return result;
    }

    public static void main(String[] args) {
         System.out.println(methodTwo(4, 5));
    }

    public static String methodTwo(int a, int b) {
        String result;

        result = "answer: " + methodOne(b, a);
        return result;
    }
}
```

Quiz 3: What Will Be Printed?

```java
public class MethodDemo {

   public static int methodOne(int a, int b) {
      int result;

      result = a * 2 + b;
      return result;
   }

   public static void main(String[] args) {
       methodTwo(4, 5);
       methodOne(3, 4);
   }

   public static String methodTwo(int a, int b) {
      String result;

      result = "answer: " + methodOne(b, a);
      return result;
   }
}
```

Separate Compilation

- Most programs involve multiple .java files.
- Many of the arguments for methods apply here as well – another level of task decomposition.
- Example: The StdDraw class from lab.
  - We can re-use the class whenever we need to draw stuff.

Review: println

```
String name = "Bob";
double amount = 1000000.0 + 1.0/3.0;
System.out.println("Hi " + name + ", you owe me $" + amount + ".");
```

- Output is:

```
Hi Bob, you owe me $1000000.3333333334.
```

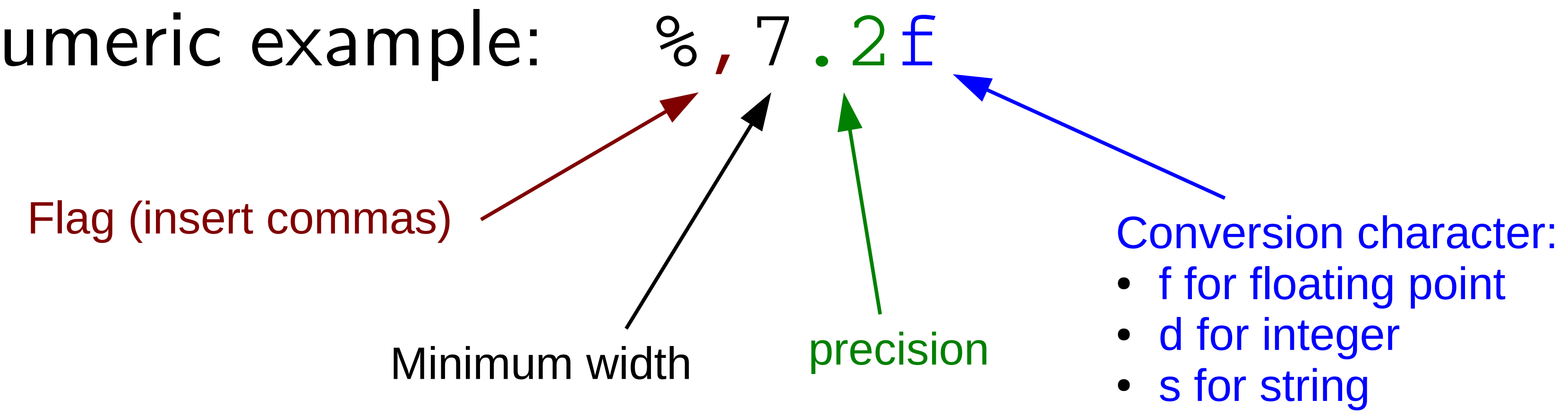
printf and Format Specifiers

```
String name = "Bob";
double amount = 1000000.0 + 1.0/3.0;
System.out.printf("Hi %s, you owe me $%,.2f.\n", name, amount);
```

- Output is:

```
Hi Bob, you owe me $1,000,000.33.
```

- `%s` and `%,.2f` are format specifiers
- Numeric example: `%,7.2f`


%,7.2f
Flag (insert commas)
Minimum width
precision
Conversion character:
• f for floating point
• d for integer
• s for string