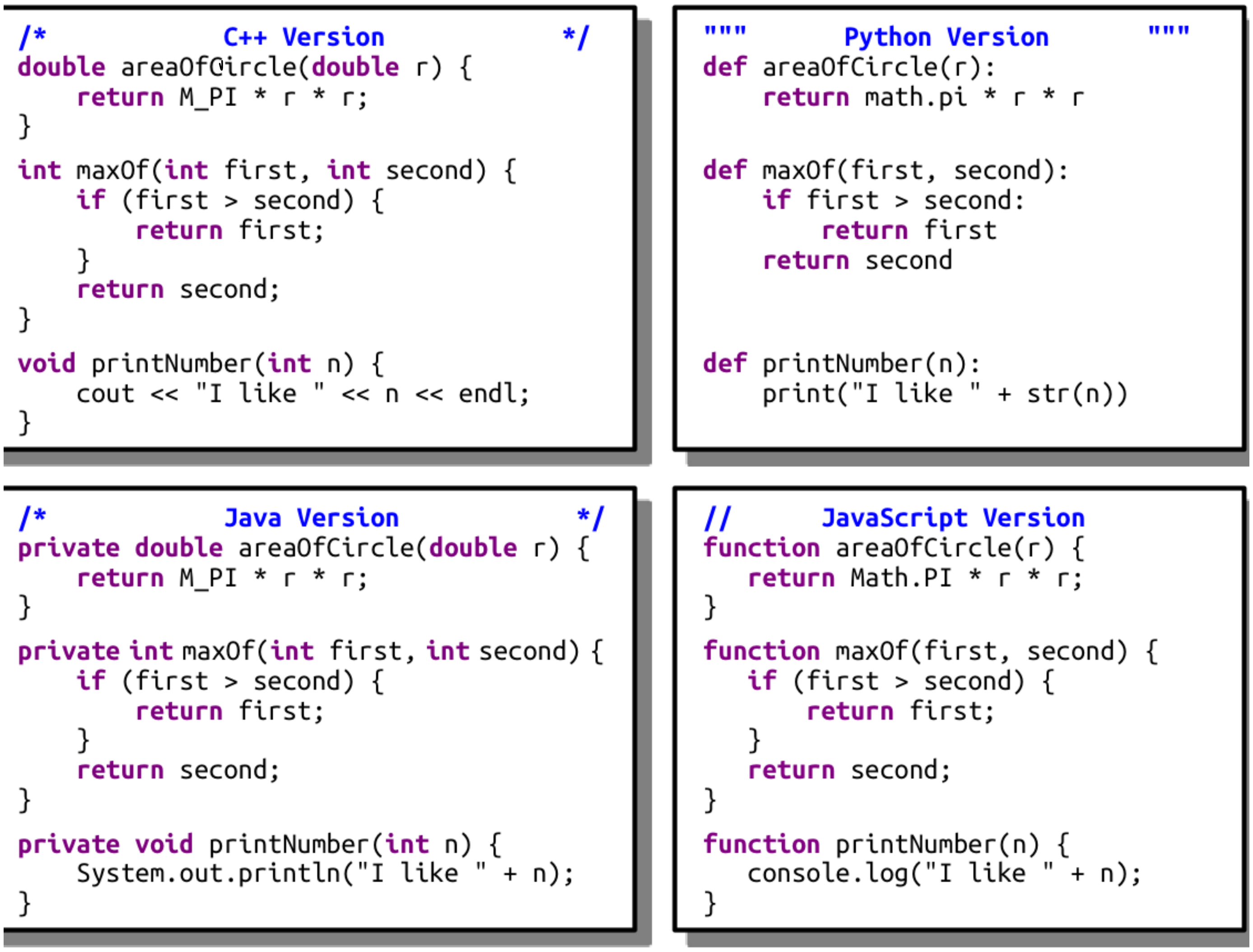
Functions in C++

- Functions in C++
  - How C++ organizes code
- Some Simple Functions
  - Getting comfortable with the language
- Intro to Recursion



```
/*              C++ Version             */
double areaOfCircle(double r) {
    return M_PI * r * r;
}
int maxOf(int first, int second) {
    if (first > second) {
        return first;
    }
    return second;
}
void printNumber(int n) {
    cout << "I like " << n << endl;
}
```

```
"""          Python Version          """
def areaOfCircle(r):
    return math.pi * r * r

def maxOf(first, second):
    if first > second:
        return first
    return second


def printNumber(n):
    print("I like " + str(n))
```

```
/*              Java Version             */
private double areaOfCircle(double r) {
    return M_PI * r * r;
}
private int maxOf(int first, int second) {
    if (first > second) {
        return first;
    }
    return second;
}
private void printNumber(int n) {
    System.out.println("I like " + n);
}
```

```
//       JavaScript Version
function areaOfCircle(r) {
   return Math.PI * r * r;
}
function maxOf(first, second) {
   if (first > second) {
       return first;
   }
   return second;
}
function printNumber(n) {
   console.log("I like " + n);
}
```

Functions in C++ work like functions in Python/JavaScript or like methods in Java. They (optionally) take in parameters, perform a calculation, then (optionally) return a value.

All variables in C++ need a type. Some common types include **int** (integer), **double** (real number), and **bool** (true/false).

You define a function by writing

```
return-type fn-name(args) {
    // … code goes here …
}
```

If a function does not return a value, its return type should be the cool-but-scary-sounding **void**.

A C++ program begins execution in a function called main with the following signature

```
int main() {
    /* … code to execute … */
    return 0;
}
```

By convention, main should return 0 unless the program encountes an error.

Forward Declarations (Function prototypes)

- A forward declaration is a statement that tells the C++ compiler about an upcoming function
- Forward declarations look like this:

  return-type function-name (parameters)

Dividing two integers in C++ always produces an integer by dropping any decimal value.

Thinking Recursively

- Solving a problem with recursion requires two steps
- First, determine how to solve the problem for simple cases
  - This is called the base case
- Second, determine how to break down larger cases into smaller instances
  - This is called the recursive step

Recap

- The C++ compiler reads from the top of the program to the bottom. You can't call a function that hasn't either been prototyped or defined before the call site

- Each time you call a function, C++ gives you a fresh copy of all the local variables in that function. Those variables are independent of any other variables with the same name found everywhere
- You can split a number into "everything but the last digit" and "the last digit" by dividing and modding by 10
- A recursive function is one that calls itself. It has a base case to handle easy cases and a recursive step to turn bigger to turn bigger versions of the problem into smaller ones.
- Functions can be written both iteratively and recursively.