Functions!

What's a function?

A function is a place where you can put code that can be called to run at any time. Its really good for things that are always the same. Use functions to break up you code into smaller pieces that are easier to manage instead of just having everything in main.

Data in/out of a function

You can give data to functions through parameters. Parameters can be any type. When you call a function, you have to make sure you give those parameters in the **proper order**. Inside of a function, you can use parameters just like normal variables. They already have values in them.

To get data out of a function, you get to return one value. If you declare your function as "void", then you do not have to return. To return a value, all you need to do is type `return` and then a value/variable.

You have already been using functions!

Every time that you call something by name and have parenthesis, you are calling a function. Here are some functions that you have already been using:

- `printf`
- `scanf`
- `sin, cos, tan`

Making a function

To make a function, there are a few things that you need to take care of. At the top of the file (under any #include, #define), you need to make a function **prototype**. A prototype is just a way of telling the compiler what a function does before it actually defines it. It's like when you declare your variables up at the top, but don't use them for a while. Note that you must end prototypes with a **semicolon.**

Now after the prototype, anywhere in the file, you can have you **function definition**. In the definition, you have to put the **header** and the **body**. The header is just like the prototype except that you must put the parameter names and there is no semicolon. Then you have the body of the function surrounded by braces (just like the body of an if or loop).

Prototype

```
return_type function_name(param_type);
```

Examples:

```
void myFun1(int a, int b, double c);
void myFun1(int, int, double);
```

Note that in the above examples you don't actually have to have parameter names (they actually don't matter at all).

Function header

```
void myFun1(int a, int b, double c)
{
     printf("Functions are cool!");
     return a;
}
```

Function Checklist

Here are the things you have to make sure that you have when writing functions.

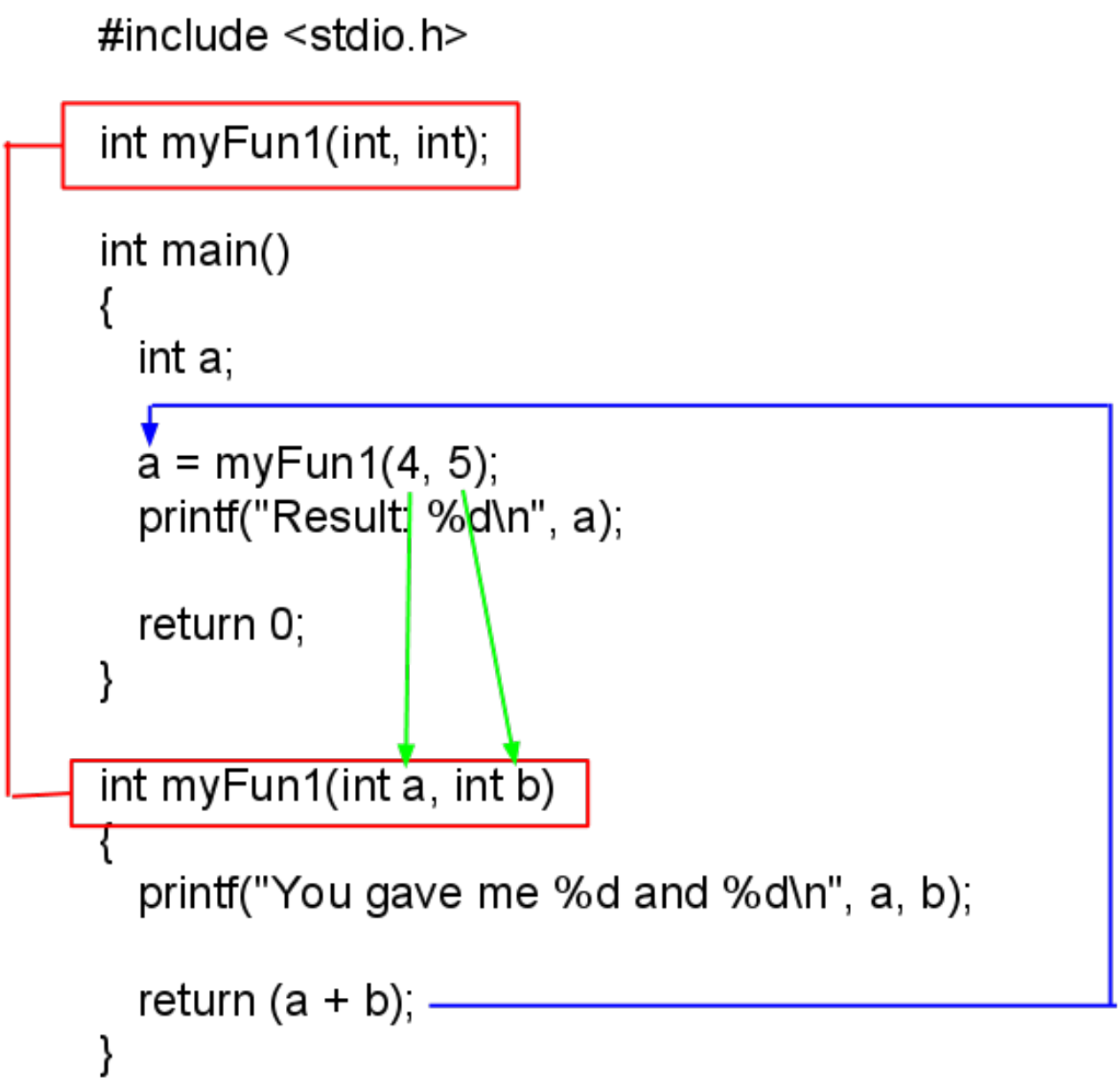
1. Prototype
    1. return type
    2. function name
    3. parameter types
    4. semicolon
2. Header (like prototype, but you have variable names and no semicoln)
3. open brace
4. body
5. close brace

Calling (using) a function

You have already used functions, but lets review. To use a function, you have to say its name and then put its parameters into parenthesis. The values that you put in the parenthesis are matched one-for-one to the parameters in the function header. After the function finishes, it will return its value (if not void) to you. Since function calls return a value, you can store its value in a variable like this:

```
int c = myFun1(3, 4, 2.0);
```

Example



This program outputs:

```
You gave me 4 and 5
Result: 9
```

Program

Write me a program that uses a function. This function must take in two ints and then return the **larger** of the two. You do not have to worry about equal numbers.

Sample Runs (input in **bold**)

```
Please enter two ints: 1 2
Larger of 1 and 2 is 2

Please enter two ints: 7 6
Larger of 7 and 6 is 7

Please enter two ints: 123123123 1231231
Larger of 123123123 and 1231231 is 123123123
```