Stubbing the Implementation

Always start by writing a "skeleton" or stub for your function. Make sure your code starts out syntactically correct, and then stays that way.

1. **Copy the prototype or declaration** into the implementation file. You don't need to bring the documentation with the prototype, but you may.
2. **Remove the semicolon** at the end of the prototype and **add some braces** to supply a **body** for the function.
3. Unless your function is a procedure (`void` function), you must create a return variable to hold the result. Look at the function return type to decide what type to make this variable. Initialize it to the "empty" value.
4. Add a `return` statement at the very end of your function.

> ***Warning.*** *Make sure your stubs always include a* `return` *statement of the correct type, or your function may crash at runtime.*

Here's a **stubbed-out** version of one function:

```
#include "digits.h"

int firstdigit(int n)
{
    int result{};
    return result;
}
```

Once you've stubbed out the two other functions, you'd expect your program to compile and link, but it does not. You get the same linker errors that appeared earlier.

Why? Since you now have two, separately compiled portions of object code, you have to tell the compiler **how to link them together**. You do that with a **makefile**.