The website for these SI sessions is `https://github.com/benblazak/2014-fall-si-cpsc120`.

Many of these examples are from `https://github.com/benblazak/2014-spring-si-cpsc120`, which is full of stuff I wrote for a lab last semester. If you're looking for extra practice, this is one of many places you might start.

Some Code

1. ```
   /**
    * Ben Blazak
    * 2014-09-12
    *
    * Simple "hello world" program.  What does it do?
    */

   #include <iostream>
   using namespace std;

   int main() {
       cout << "hello world\n";
       return 0;
   }
   ```

2. This version has no `using namespace std;`, so `cout` must be fully qualified as `std::cout`.

   ```
   #include <iostream>

   int main() {
       std::cout << "hello world\n";
       return 0;
   }
   ```

   In the classroom, `using namespace std;` is almost always used. In the real world, I've read that it's considered bad practice. You should be aware that there are different ways to do things. If the instructor cares, of course, you kind of have to write it how they wish. Past that though, just think about who you're writing for – whether it's your friend (because what friend doesn't want to see your C++ projects? duh), or your future self in a few months when you're studying for your midterm. Think about what they would find prettier and easier to read, and write it that way.
   ```

3. Here, we only return a value, we don't `cout` anything. As a consequence we do not need the `#include <iostream>` at the beginning. How do you think this would effect the preprocessed and compiled files?

   What if we return a value other than 0?

   ```
   int main() {
       return 42;
   }
   ```

   In real code, returning anything other than `0` from `main()` usually means that an error occurred.

   Notice that `42` and `0` are both integers. The `int` in front of `main()` is what tells the compiler that we're planning to return an integer. Since `main()` is a special function – it's the function that the operating system calls when you ask it to run your program – it must have this signature (i.e. it must return an integer). Other functions can return different things.

4. Wait, we can have more than one `return`?

   ```
   #include <iostream>

   int main() {
       std::cout << "hello world 1\n";
       return 1;
       std::cout << "hello world 2\n";
       return 2;
       std::cout << "hello world 3\n";
       return 3;
   }
   ```

   Having more than one `return` in a function won't be useful until we talk about control flow (e.g. `if...else` statements), and some people consider it dangerous. In general, I think it's okay, as long as it makes your code more clear – which is to say, as long as you're **careful**; as with everything :) .

Variables, Basic Operators, and Assignment

- What's the difference between `int x;` and `int x = 0;`?
- What are the two functions of the `<<` operator? What about the `>>` operator?
- What is the value of `x` in memory after the statement `int x = 3 + 5 * 3 + 10 / 2;` in
  - Decimal?
  - Binary?
  - Hexadecimal?
- If you compile and run the following program

```
/**
 * Ben Blazak
 * 2014-09-12
 *
 * Create a variable, `x`, and print its address.
 */

#include <iostream>
using namespace std;

int main() {
    int x = 5;
    cout << &x << endl;

    return 0;
}
```

and

```
0x7fff52b26ad8
```

is printed to the screen (true story), what does this mean?

Program Design

- How many different numbers can you represent using only the fingers of one hand?
- How would you write a program (just a general design, not actual code, for now) to figure this out (without doing any mathematical analysis first)?
- Is there a simple mathematical formula to find the answer?
- Which method do you think would be quicker for the computer to compute?