Namespaces

- in C++ we can group a set of definitions together and give this collection a name, hence creating a namespace
- the definitions could be of constants, variables, functions, structs, etc
- later we can add more definitions to that namespace if desired
- different namespaces could have different definitions for any given item
- when a program wishes to use a definition coming from a specific namespace it identifies the namespace
- we'll look at ways to create a namespace, add to a namespace, and a variety of ways to identify which namespace definitions we wish to use

Creating a namespace

- we create a namespace and give it a name as follows:

```
namespace YourChosenName {
  // the various definitions would go here
};
```

- for example, creating a cs160 namespace with a definition for a LineLength constant and an intArr struct

```
namespace cs160 {
    const int LineLen = 96;
    struct intArr {
      int* arr;
      int sizeAlloc, sizeInUse;
    };
};
```

Adding to a namespace

- later in the code we can add more definitions to the namespace with the exact same syntax, e.g.

```
namespace cs160 {
    struct floatArr {
      float* arr;
      int sizeAlloc, sizeInUse;
    };
};
```

Using namespace definitions

- to use a definition from a namespace we specify the namespace name :: definition name
- for example, if we wanted to declare a floatArr variable based on the cs160 namespace:

```
cs160::floatArr x;  // gets the cs160 defn of floatArr
```

- anyone reading the code can thus see exactly where the definition of floatArr is coming from

Shorthand: “using xxxx::yyyy;”

- while the previous syntax shows the source of the definition very clearly, sometimes programmers don't want to rewrite the “`cs160::`” every time they use that particular definition
- as a shorthand, we can tell the compiler (and other programmers) the namespace to use for a specific definition with the syntax

```
using cs160::floatArr;
```

- in the rest of the code we can then simply write `floatArr` instead of `cs160::floatArr`

Using an entire namespace

- an even more substantial shortcut is to say we wish to use all the definitions from a namespace, with syntax

```
using namespace cs160;
```

- we've been doing this with the `std` namespace
- the downsides to this are that:
  - the namespace might have definitions we don't actually want and that clash with the names of definitions we've created ourselves in the current program
  - the reader of the code can't immediately see where a definition comes from (is it from a namespace or from somewhere in the current code)