Supplement IV.B: namespaces

For Introduction to C++ Programming
By Y. Daniel Liang

In a large project, you will use programs from different sources. The scope of an identifier (e.g., variable, constant, etc) may overlap with the scope of another identifier, which causes naming conflict. You can solve this problem using namespaces. Each namespace defines a scope for identifiers. Listing 1 gives a simple program that demonstrates the use of namespaces.

Listing 1 NamespaceDemo.cpp

```
#include <iostream>
using namespace std;

namespace namespace1
{
  double x = 1;

  void p()
  {
    cout << "from namespace1" << endl;
  }
}

namespace namespace2
{
  double x = 2;

  void p()
  {
    cout << "from namespace2" << endl;
  }

}

using namespace namespace1;

int main()
{
  cout << namespace1::x << endl;
  cout << namespace2::x << endl;

  p();
  namespace2::p();

  return 0;
```

```
        }
```

Sample output

```
     1
     2
     from namespace1
     from namespace2
```

Namespaces are defined using the keyword **namespace**. Two namespaces named **namespace1** and **namespace2** are defined in lines 4 and 14. Lines 28 and 29 access variable **x** in **namespace1** and **namespace2** using the binary scope resolution operator **::** in **namespace1::x** and **namespace2::x**.The syntax **namespaceName::identifier** is known as the full qualified name for the **identifier**.

You may also use a **using** directive or **using** declaration. The syntax for the **using** directive is

```
        using namespace namespaceName;
```

This statement tells the compiler to recognize all members in the specified namespace. For example,

```
        using namespace namespace1; // line 24
```

Throughout the book, you have used

```
        using namespace std;
```

This statement tells the compiler to recognize all members in the **std** namespace. The contents of **<iostream>**, **<cmath>**, **<ctime>**, and many other functions are defined in the **std** namespace.

The syntax for the **using** declaration is

```
        using namespaceName::member;
```

For example, in Listing 2.4 in Chapter 2, you used

```
        using std::cout;

        using std::cin;

        using std::endl;
```

to explicitly specify the members **cout**, **cin**, and **endl** in the std namespace.