91. Namespace

Objectives

- Create a namespace
- Place identifiers in a namespace

As you know values in variables/objects are lost when you close a program. But there are times when you want to save their values. For instance if a player reaches a new high score, you might want to save his/her high score (and his/her name – or he/she would not be happy...) and then print that person's name and high score every time that game runs. In this set of notes, I'll show you how to save data into text files.

Writing to a File

Run this:

```
#include <iostream>

namespace yliow
{
    int x;
    void f()
    {
        std::cout << "hello world" << std::endl; }
    class C
    {};
}
int main()
{
    yliow::x = 42;
    yliow::f();
    yliow::C obj;
    return 0;
}
```

A **namespace** is just a container of names (i.e., variables, functions, structs, classes.)

Why?

Allows you to put names into a namespace to avoid naming conflicts.

For instance suppose John and Mary work on the same project. They can choose different namespaces for their own things:

- `john::print_helloworld()`
- `mary::print_helloworld()`

Openness of namespace

Namespaces are **open**: you can open it after it's closed.

```
namespace yliow
{
    class C
    {};
}

namespace yliow // open again!!!
{
    class D
    {};
}
int main()
{
    yliow::C obj1;
    yliow::D obj2;
    return 0;
}
```

The most common usage of namespace is probably to hold classes and function prototypes:

```
// GameLib.h
namespace GameLib {
    class vec2d { ... };
    class graphics { ... };
    class sound { ... };
    class physics { ... };
}
```

If you're writing games and scientific simulations. You can split the above.

```
// Physics.h
namespace GameLib
{
    class vec2d {…};
    class physics {…};
}
```

```
// GameLib.h
#include "Physics.h"
namespace GameLib
{
    class graphics {…};
    class sound {…};
}
```

In this way, when you're writing a text-based sci simulations, you don't need to compile with the graphics and sound class.

Using

You can avoid typing namespace when you use it:

```
namespace yliow
{
    class X {};
    class Y {};
}

using namespace yliow;
int main()
{
    X obj; // do not need yliow::X
    return 0;
}
```