**Lab Sections**

1. Objectives
2. Introduction
3. Definitions
4. Experiments

|  |
| --- |
|  |

Namespace

**Namespace**

1. **Objectives**

**After you complete this experiment you will be able to:**

1. **Explain the meaning of: “#include <iostream>” and “using namespace std;”**
2. **Explain the concept of a namespace.**
3. **Introduction**

Namespaces allows us to group a set of global classes, objects and/or functions under a name. Provided are three examples that describe namespaces and what they are capable of.

1. **Definitions**

We will define terms that you will be using in the semester. They are as follows:

1. **#include** is a keyword that allows us to use a library in C++. A library contains definitions of keywords used in programs. If you follow #include with the name of the library, you can use any keyword inside that library. Think of these like a collection of **function definitions** in C language but can hold many definitions.
2. **namespace** refers to a programmer-defined region in your program. It may vary in size from a few variables to many files. Think of these like a collection of **function prototypes** in C language but can be used on a larger scale than just a single file.

More information on these and other keywords can be found in your course textbook and on the web.

1. **Experiments**

**Step 1: Without Namespace**

**In this experiment you will create a program using the ‘std’ namespace. Enter, save, compile and execute the following program in MSVS. Call the new project “namespace1” and the program “namespace1.cpp”.**

#include <iostream>

int main ()

{

std::cout << "Hello world!" << std::endl;

return 0;

}

**This example shows a classic program that displays “Hello World” while using a symbol (::) called the scope resolution operator. The name of the namespace we are using is called ‘std’. This means anything located inside the ‘std’ namespace can be accessed using ‘std’ followed by the scope resolution operator and the keyword we wish to use. In this case, our keyword ‘cout’ works exactly as ‘printf’ in the C language. While the keyword ‘endl’ works exactly like ‘\n’ in the C language.**

**Step 2: With namespace**

**Enter, save, compile and execute the following program in MSVS. Call the**

**project“namespace2” and the program “namespace2.cpp”.**

#include <iostream>

using namespace std;

int main () {

cout << "Hello world!" << endl;

return 0;

}

**The previous example shows a program that displays “Hello World” while using no scope resolution operator. This is possible because we added the keyword ‘using namespace std;’ at the top of the file. If you specify ‘using namespace std;’ then you don't have to put std:: throughout your code. The program will know to look in the ‘std’ namespace to find the object.**

**So the question remains, why then do we need #include <iostream> if ‘cout’ is included in the namespace std? Or in other words: what is the difference between ‘#include <iostream>’ and ‘using namespace std’?**

**The answer is simple. ‘cout’ and ‘endl’ are defined in the file called ‘iostream’, and this ‘iostream’ file is located inside of the ‘std’ namespace. But it just maybe that another ‘cout’ exists outside of the scope of ‘std’.**

**Therefore ‘using namespace std’ defines the** *scope* **of the terms we write.**

**And ‘#include <iostream>’ refers to the file that holds the** *actual definition* **of the terms we write.**

**We now have one more example to show you.**

**Step 3: User-defined namespace**

**Enter, save, compile and execute the following program in MSVS. Call the**

**project“namespace3” and the program “namespace3.cpp”.**

#include <iostream>

using namespace std;

namespace myspace

{

double value = 3.14159;

}

int main()

{

cout << myspace::value << endl; //outputs 3.14159

return 0;

}

**In this final example, we define our own user namespace called ‘myspace’. In ‘myspace’ we have defined a ‘value’ that is of type double and is equal to 3.14159. This example illustrates that you can not only define your own namespaces, but you may access that information as well.**

**As it appears, the ‘std’ namespace is much like our ‘myspace’ namespace, but is much, much larger with many files, keywords, and definitions. In fact, everything inside the ‘std’ namespace is so large that it has been given a special name among C++ programmers. It is known as the “C++ Standard Library”.**

**You have now concluded the Namespace Lab.**