Namespace

- C++ has one name rule which can cause name clashes
- Name space pollution: too many names
- Namespace mechanism – *group a collection of names that logically belong together (enclose names in a scope)*

Create a Namespace

```
namespace mySpace
{ // all constants, functions, class/struct definitions
  void getData(int&);
};
namespace yourSpace
{ // all constants, functions, class/struct definitions
  void getData(int&);
};
```

Access Members within Namespace

- Qualify each reference everywhere in the program:

  ```
  mySpace::getData(int& dataValue);
  yourSpace:: getData(int& dataValue);
  ```

- "using" declaration (for qualified id only, `getData`):

  ```
  using mySpace::getData(int&);
  ```

  Any reference to the qualified id following this declaration will refer to the one declared above namespace

- "using" directive (all ids within namespace):

  ```
  using namespace mySpace;
  ```

  Any reference to all the ids within the namespace can be done directly after the directive statement.

Namespace std

All the identifiers in standard C++ header files are part of the `std` namespace.

For example, `cin` and `cout` maybe written as `std::cin` and `std::cout`

Rules for Use of Namespace std

- Qualify names in prototypes and/or function definitions heading:

```
std::cout
```

- If only one name from namespace std is used within a block and it is used more than once in a function block, use a using declaration:

```
using std::cout;
```

- If more than one name are used from namespace std are used within a block, use a using directive:

```
using namespace std;.
```

- “using” directive should not be used outside a block.

```
using namespace std;.
```