Lab 1

From One Translation Unit to Another

In this workshop, you implement aspects of linkage, storage duration, namespaces, guards, and operating system interfaces.

**LEARNING OUTCOME**

Upon successful completion of this workshop, you will have demonstrated the ability

* to link to a variable in another translation unit
* to declare a local variable that lasts the lifetime of the program
* to guard a class definition from repetition
* to define a class within its own namespace
* to pass arguments to program from the command line

**SPECIFICATIONS**

This workshop consists of three modules:

* **w1**
* **process**
* **CString**

CString Module

Write the header and implementation files for a class named **CString**.  Wrap your header file in a conditional macro that guards against repeated use.  Embed your class definition and its implementation in a namespace named **w1**.  Include in your class definition:

* a constant definition of the number of characters to be stored by an object of your class
* a constructor that receives the address of a C-style null-terminated string and stores the first MAX characters of the string.  Check for receipt of the null address.  Store an empty string in that case. //store up to the max allowance of character and if the string is null, store empty string
* a member query named **display()**that receives a reference to an **ostream** object and displays the string as stored in your **CString** object
* a helper non-friend operator that inserts the stored string into the left **ostream** operand.  This operator prefaces the string with the number of the insertion and increment that number

Do not use the **string** class of the standard library in this workshop.  Use the **cstring** functions.

Include in your implementation file:

* a definition of a global variable initialized to the number of characters stored by an object of your class.

process Module

Write the header and implementation files for a function named **process**.  Your function receives the address of a C-style null-terminated string and a **std::ostream** reference. The **process** module uses an object of your **CString** class to store a possibly truncated version of the string.  Leave your prototype and function definition in the global namespace.  In your definition:

* construct a **CString** object from the string received
* insert the **CString** object into the **std::ostream** object and terminate with a newline

main Module

Update the **main()** module to include an external variable to access the max number of characters stored by your **CString** object. Print the max number of characters to the **std::ostream** object.

Sample Output

The output from your program looks like:

Command Line : C:\Users\...\Debug\Lab1.exe oop345 dps941

Maximum number of characters stored : 3

0: oop

1: dps