In completing this workshop, I have learned:

1. How to set the command-line arguments in Visual Studio:

Right-click the project, then go to Properties. Choose the Debugging in the left menu, and the command-line arguments setting is on the right.

1. Even section for the command line arguments is blank, there is always at least one argument. Thus, when sending the command-line argument set by us to the console, the iteration shall start from 1, i.e. the second argument, instead of 0.
2. The lifetime of static variable is through the whole program, and the variable is visible in all scopes in the translation unit in which it is declared. Thus it is good to be used for counting how many times a function is called during the execution of a program.
3. The size of C-style null-terminated string is predefined, unless dynamic allocation is used to allocate memory for the string in the run-time, so it is of any length and take an uncertain number of characters.
4. The difference between internal and external linkage using examples from your code

Internal linkage has a keyword “static”. For example, in my code:

//String.cpp

static int count = INITIAL;

count is visible in all scopes only in String.cpp, but invisible in other translation units.

External linkage has a keyword “extern”. For example, in my code:

//w1.cpp

int INITIAL = 3;

//String.cpp

extern int INITIAL;

Since INITIAL is initialized in w1.cpp, the keyword “extern” is omitted. INITIAL is external linkage, so it is visible in all the translation units in the same program.

1. The execution effect of the code in the **process()** function that creates the **String** object which is inserted into standard output (which member function(s) does this code call)

In the process() function, the following line of code creates the String object:

String a = String(str);

It calls the copy constructor, which means it creates a temporary object to String and assigns it to a.

The following line of code inserts a into standard output:

std::cout << a << std::endl;

It calls the member function display() and the member operator <<, which not only inserts a to the standard out, but also increments the static variable “count”.

1. The changes that you made in upgrading your **String** class.

The data member has been changed to a pointer to null-terminated C string, so String class can manage a C string of any length.

The definition of 0-argument constructor has been changed to define the pointer as “nullptr”.

The copy constructor and copy assignment have also been upgraded to manage the dynamic allocation.

The destructor has been specifically defined to deallocate the class resource.