Week 3 Lab—Data Types and Assignment Statements

# Scenario

In this week’s lab, you will look at data types and memory leaks and look at a variety of IDEs.

# Rubric

Point distribution for this activity:

|  |  |  |
| --- | --- | --- |
| **Lab Activity** | | |
| **Document** | **Points possible** | **Points received** |
| Part A | 20 |  |
| Part B | 30 |  |
| **Total Points** | **50** |  |

# PART A:

One of the dangers with C++ pointers is memory leaks. Run the following code in C++.

#define \_CRTDBG\_MAP\_ALLOC

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdlib.h>

#include <crtdbg.h>

#include <string>

void memLeak()

{

int \*p = new int;

char \* string1 = new char[20];

char \* string2 = new char[25];

strcpy(string1, "Sheldon");

string2 = string1;

delete p;

}

int main(int argc, char\* argv[])

{

memLeak();

\_CrtDumpMemoryLeaks();

return 0;

}

When you run this code you should see the following showing the memory leaks: Detected memory leaks!

Dumping objects ->

{72} normal block at 0x02BA6250, 25 bytes long.

Data: < > CD CD CD CD CD CD CD CD CD CD CD CD

{71} normal block at 0x02BA97F8, 20 bytes long.

Data: <Sheldon > 53 68 65 6C 64 6F 6E 00 CD CD CD CD CD CD

How would you fix it so that the value in string1 and string2 are both “Sheldon” but with no memory leaks?

The original code was making string2 equal string1 which is an address because they are pointers, so the original string2 address was lost which created the memory leak. To fix this, I added a loop that takes the value at the address of string1 equal the value at the address of string2 for each character in the string.

C++ code:

#define \_CRTDBG\_MAP\_ALLOC

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdlib.h>

#include <crtdbg.h>

#include <string>

void memLeak()

{

int \*p = new int;

char \* string1 = new char[20] {0};

char \* string2 = new char[25] {0};

strcpy(string1, "Sheldon");

for (int i = 0; \*(string1+i) != NULL; i++)

{ \*(string2+i) = \*(string1+i); }

delete p;

delete[] string1;

delete[] string2;

}

int main(int argc, char\* argv[])

{

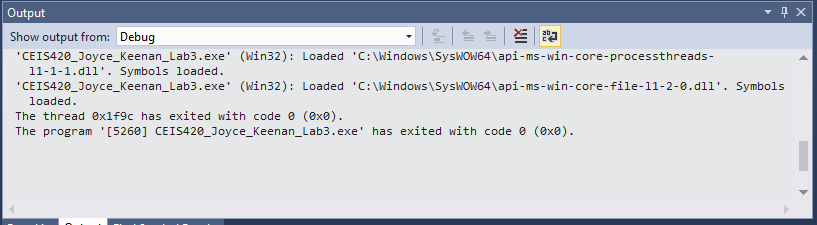
memLeak();

\_CrtDumpMemoryLeaks();

return 0;

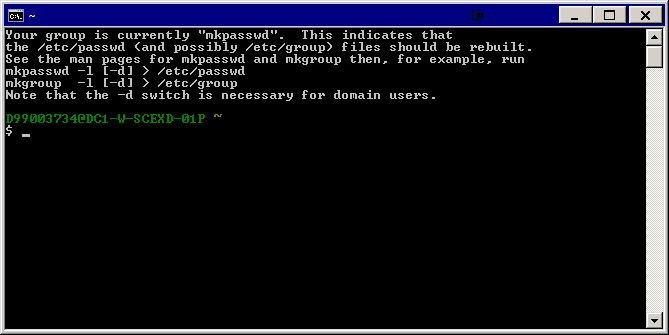
}

Screenshot of output:



# Part B:

In our labs, we have always used an IDE. Now let’s compare code by creating the same code using a command line. Either download Cygwin on your machine, use your Raspberry Pi, any other Linux machine, or open Cygwin from <http://lab.devry.edu>



Type vi hello.cpp

In vi, you need to type i to get in insert mode and escape to go to command mode. Type i then the following code.

#include <iostream>

using namespace std;

int main(void)

{

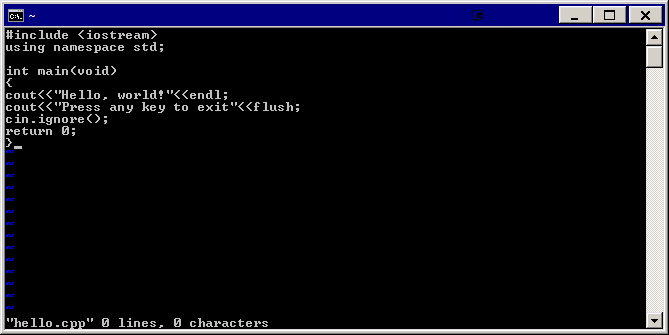
cout << "Hello, World!" << endl;

cout << "Press any key to exit" << flush;

cin.ignore( );

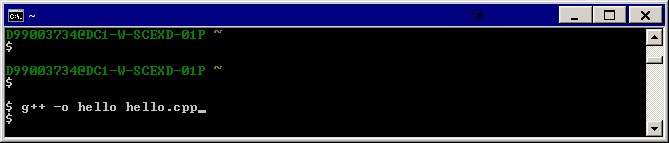
return 0;

}



Once you’ve typed in the code, press the escape key, then type in :wq! This will save and quit.

Now type in g++ -o hello hello.cpp



Then ./hello



This will run your code.

Next run Hello World in C#, Java (download Eclipse or use it in citrix: lab.devry.edu or try Netbeans), and Python (IDLE). Python is available on citrix: lab.devry.edu or on the Raspberry Pi or by downloading Python <https://www.python.org/downloads/> . Feel free to try another language (ruby/kotlin/C/perl).

Paste screenshots of all code here.

1. C++ command line in linux
2. C#
3. Java
4. Python
5. Optional—another language

Compare: g++ command line, Visual Studio, Netbeans/Eclipse, and Idle. Which do you like the best? Why?