Compiling and Debugging Lab Csc 305 Prof. Jonathan Parziale

**Intro:**

In Linux and Unix system in order to create programs/applications we need to use the command line equivalent of a compiler and debugger.

On Linux systems there is a compiler that is open source and provided as a package that an administrator can install. These are Known as GNU Compilers, the packages that are included is for c and c++ as well as some older programing languages such as Fortran and perl. We will be focusing on gcc(c compiler) and g++(c++ compiler).

The debugger in linux is known gdb, or GNU debugger, it is a command line tool that allows you step, and trace through a program, to discover run time bugs and logical errors.

**Lab:**

Objective: To learn how to compile and debug code on Linux platforms using the command line tools gdb and g++, There will be 3 syntax errors and 4 logical errors in the code.

**Deliverables:**

Take a screen shot of the program running before each correction, and after each correction, in order to show progressing through the errors, and upload all the screenshots to blackboard

**Todo:**

Goal: Correct all syntax and logical errors, using the tools specified to find the errors

Stage 1: correct all syntax errors (4 in total)(use the compiling instructions below)

Stage 2: correct all logical errors. (3 in total)(use the debugging instructions below)

* Perform all Tasks on the Linux server,

Login to server:

*If working on a linux or mac computer:*

Command:ssh username@163.238.35.166

Password:firstnamelast4ofID

*If working on a windows computer:*

Open up “putty”

Enter the ip in host box:163.238.35.166

Enter your username:LastName

Enter you password:firstnamelast4ofID

**Stage 1:Task Compiling:**

1. Download the two files from blackboard, circle.cpp, and circle.h
2. Copy the files to the server using “winscp” application on windows or scp cli on linux and mac.

3) Create a new directory to store the files in

4) Compile the code

a) Run the command g++ circle.cpp -o executable.exe, this will compile the code both for circle.cpp and the referenced file circle.h

note: -o parameter specifies a new name for the executable, if you do not specify the “-o” parameter the default name of the executable is “a.out”

5) Using the error messages that the compiler gives you correct the syntax errors

**6) repeat this process until all syntax errors are resolved**.

Compiling:

When compiling code, first the compiler checks the source files for syntax errors and link errors, if it finds any errors, it displays them on the screen and aborts compiling the code. Other Compilers such as Visual Studio, gcc/g++, and Eclipse function in a similar manor; one of the notable differences is in how they word the errors. Some error messages may look similar and some may not. The process to find the syntax errors is straightforward, every time you compile the code, the compiler will tell you the syntax errors. Each syntax error will follow the same format. Logical errors on the other hand, a compiler cannot help you with, in these cases we use what is known as a debugger.

Debugging:

A debugger is an application that acts as a wrapper for the application execution, which gives the user/developer the ability to control execution of the program. A debugger gets a map of the application memory of the program, and the program counter, so the user can track the values of each variable after each line executes. The debugger works in conjunction with a compiler, when the source code is compiled, the compiler can generate symbols that the debugger uses in order to understand how to read the code. Without these symbols a debugger can not understand how to read the binary file.

**Stage 2: Task Debugging:**

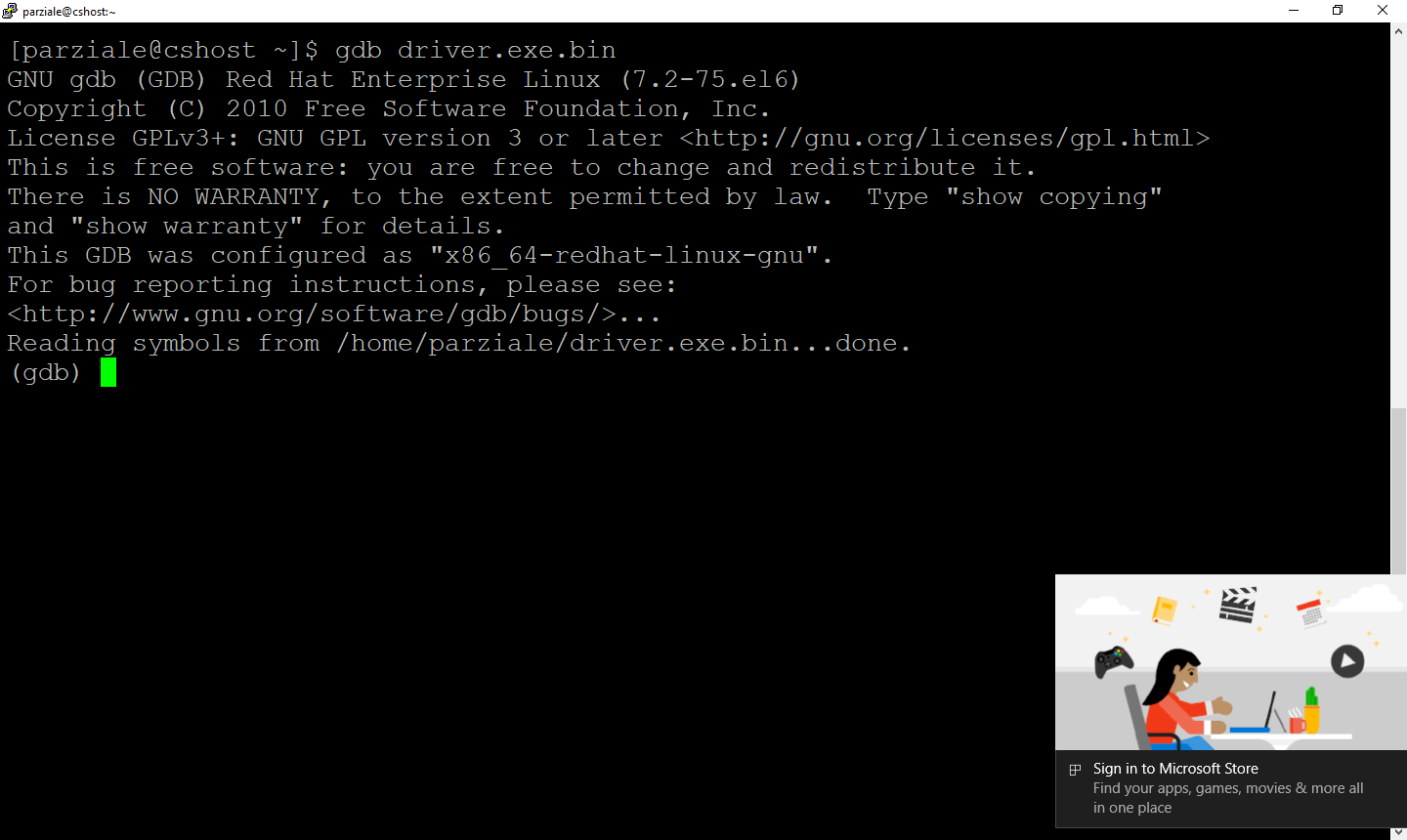
Debugging applications in Linux using the debugger known “gdb”

Reference:http://web.eecs.umich.edu/~sugih/pointers/gdbQS.html

1) Compile the code using “g++ circle.cpp –o executable\_name –g”, this will Compile the code and create symbols for the debugger to use.

2) Run gdb on your executable

“gdb executable\_name”



3) When in gdb(when you have a prompt that looks like (gdb)) type the command “start”

This will execute the code for the executable that you are working on.

4) Type “continue” to execute the next line of code in your executable.

If you hit an error the debugger quits the execution of the code,

NOTE: if you would like to see each line of code as it runs use “next” instead of “continue”

5) Read the error messages and return to the bash shell by typing “quit” in gdb

note:gdb will tell you exactly which line in the source code produced the error

6) Make the corrections to the code and re-compile the code.

**7) Repeat this process until all logical errors are corrected.**