CS 277 Lab 4
SSH and IA-32 Assembly
Due 11:59pm Sunday March 16, 2014

Purpose

The purpose of this lab is to gain some experience problem solving and writing IA-32 assembly language code. Remember that when writing IA-32 assembly code, compilation will only be successful if you are on a system that supports the architecture. There is a good chance you won't be able to complete the assignment using your personal system. This makes for a good time to become familiar with the **ssh** command-line utility if you have not already.

Assignment – Part 1

Your work in Part 2 must be compiled and run on lab machines that support the IA-32 architecture we covered in class. These machines are named:

cs6
cs10
cs17
cs26

You do not have to physically be at these computer systems in order to compile and run your code on them. Instead, you should log into them via *ssh*. **ssh** stands for *secure shell*. *ssh* is a program that allows you to open a connection to a remote computer system and execute commands in a terminal session on that system. For this lab, you will need to ssh into one of the four systems above for compilation and execution. You can develop on any system. Complete the following steps before moving on to Part 2:

1) Open two terminal windows and place them side by side.
2) In one window, issue the ssh command with your username and the name of the system to which you wish to connect. To ssh into one of the Lab 4 approved machines:

ssh username@cs6

this will log your user account into a terminal session on cs6.

3) Once logged in, type **pwd** and press enter. You will notice that you are in your home directory. By default, ssh will open your remote session with your home directory as your working directory location.
4) Type **ls** and press enter and you will see all of your directories and files are present on the remote system. Your home directory and files all "live" on a network file server. This gives you the same access to your data no matter which computer you physically log into or which systems you log into via ssh.
5) Now **cd** into your lab4 directory in both terminal windows so that both your local and your remote terminal sessions are in your lab 4 directory.
6) In your local window, type **make** and press enter. Does the compilation succeed?
7) Now in your remote window type **make** and press enter. Does the compilation succeed?
8) Now try running the compiled executable, named *guesser*, on both the local and remote systems. What happens?
9) When you are finished with your ssh session, simply type exit and the session will be closed. Your terminal is again an active session on your local system.

As you work on Part 2, you can develop your code on any system. However, you must compile and run the code on one of the four Lab 4 systems listed above. Be sure you are comfortable logging into one of these systems via *ssh* before continuing to Part 2.

Assignment – Part 2

In this part of the assignment you will write IA-32 assembly code to complete the evaluation function of a random number guessing game. You will use the *make* utility to compile your solution. The makefile will produce an executable called *guesser* upon successful compilation. Guesser prompts the player for a number between 1 and 100 and lets the player know if the number is larger or smaller than the secret number. Play proceeds until the player guesses the correct number. Your code should be placed in the file **asm.s.** Your job is

to implement a procedure named *is_correct()*. The C prototype looks like this:

```
int is_correct(int, int, int *);
```

The three function parameters are (in order): the secret number, the user's guess and an output parameter in the form of an integer pointer. The return value of the function should be the number **1** if the user's guess is a valid guess (>=1 and <= 100) and should be **0** otherwise. If the user's guess is the same as the secret number, then **0** is passed back *via the pointer variable* as an output parameter. If the user's guess is less than the secret number then a **negative number** should be passed back via the pointer variable. If the guess is greater than the secret number then a **positive number** should be passed back via the pointer variable.

Submission

To submit, email your **asm.s** file to mharmon@lclark.edu before the due date and time.

Hints and Tips

- You do not need to worry that the pointer variable will be uninitialized, NULL or in some other way pointing to an invalid memory address. The main program will always ensure the pointer is valid.

Evaluation

+10 compiles

I will run your code against a set of test data that includes numbers both inside and outside of the valid number range. Correctness of your code will be based on how it performs over the set of inputs:

+15 runs without error
+15 correctly returns 1 if guess is in the valid guess range
+15 correctly returns 0 otherwise
+15 correctly returns 0 via the output parameter when the guess and secret numbers match
+15 correctly returns a positive number via the output parameter when guess is > than secret number
+15 correctly returns a negative number via the output parameter when guess is < than secret number