CMPT350: Lab Assignment 1

A hangman game in MIPS *5%*

Due: Tuesday, October 4, 17:00

This lab assignment will have you make a game of Hangman in MIPS.

Laboratory Procedure:

On the course Moodle, look for the Laboratory 1 assignment in the Laboratories section. Inside is a premade skeleton MIPS files, `laboratory1.s` Save this somewhere handy on your machine. You may use both the command line version, `spim`, and the GUI version, QtSpim to run your program. As well, you will need a text editor to edit your `laboratory1.s`.

Assignment Requirements:

- **Basics**
  - Run your program successfully using either QtSpim or the `spim` command line tool.
  - Sufficient commenting is present in the program, including the preamble comment block at the beginning of the file. **This preamble block must contain a line which contains the contents "# Author: <YourName>"**
  - The program terminates properly.
  - Output looks "proper" (e.g. appropriate newlines, easy to read)
- **Hangman**
  - The program prompts the user for an input word between 5 and 10 letters.
  - The program keeps a "progress" string showing which letters have and haven't been guessed, which is updated and displayed after each character guess
  - If an incorrect guess is made, the program tells the user that the last guess was incorrect.
  - The program has a fixed number of "incorrect guesses", which if the user exceeds, triggers the loss message "YOU LOSE!" displayed on its own line. and the program exits.
  - If the user guesses all the letters, then the win message "YOU WIN!" is displayed on its own line and the program exits.
  - In addition to the "YOU WIN!" or "YOU LOSE!" messages, the program must also exit with the appropriate exit code, exit code 1 for a win, and exit code 2 for a loss.

***** NOTE:** Your code **must be interpreted successfully and execute** within the **laboratory environment, using the command line version of SPIM**. Failure to do so will result in a **mark of ZERO** for the program. *******

Creating a trace file

As part of all demo and laboratory submissions, you must submit what's called a trace file. This is a text file you'll create that in essence replicates what your terminal looks like when you demonstrate your programs. To assist with this, a tool called `script2` has been provided for you. To create a trace file, simply run `script2` in your terminal, and then proceed to use the terminal as normal to navigate to and run your programs. When you have demonstrated everything, press CTRL+D to signal that you are done, and `script2` will exit, creating a file called typescript in the directory you ran `script2` from. This is your trace file. **Please rename your tracefile to include your name, e.g. baker_typescript**

Submission:

Your program should be named using your name and end in a .s extension. For example, BakerLaboratory1.s

When you feel your program meets all of the above requirements, create a zip archive with your code submission and trace file included. An easy way to do this is with the `zip` command. For example, if your solution file and trace file are in a folder called `baker`, then the command `zip -r baker.zip baker/` will create a zip file containing your submission files called baker.zip (Replace this with your name when creating your zip file). Upload this zip file to Moodle.

Example Output:

Example screenshots are posted under the Laboratory 1 section on Moodle.

Hints:

- The hardest aspect of this lab will be managing the string tracking which letters have been revealed in the solution word. Recall Lab 2 for how to read and write bytes to our strings
- Likewise, like in C, it is a useful convention to end a string with a null byte to indicate that this is the string boundary. This will have to be manually managed by you when working with your strings, but will help you when determining exit conditions for your string loops
- Check your syscall tables to learn how to exit your program with a particular exit code
- To check that your program exited with the correct exit code, you can use `echo $?` in the terminal immediately after running your program