Homework #2 – Assembly Programming

Due Monday, February 15 at 5:00pm

You must do all work individually, and you must submit your assignment electronically via Sakai.

All submitted code will be tested for suspicious similarities to other code, and the test will uncover cheating, even if it is "hidden" (by reordering code, by renaming variables, etc.).

MIPS Instruction Set

1) [5 points] What MIPS instruction is this? 0000 0011 0011 0010 1000 0000 0010 0010

2) [5] What is the binary representation of this instruction? lw $r1, 3($r2)

Assembly Language Programming in MIPS

For the MIPS programming questions, use the QtSpim simulator that you used in Recitation #4. Please note that the TAs will be grading your assembly programs using SPIM. If you use some other MIPS simulator (e.g., MARS), there is NO guarantee that what works on that simulator will also work on SPIM. We will grade strictly based on what works when your program runs on SPIM.

3) [20] Write a MIPS program called palindromes.s that prints out the first N positive numbers that are at least two digits and palindromes (i.e., are the same if read left-to-right or right-to-left), where N is an integer that is input to the program.

   You will upload palindromes.s into Sakai (via Assignments).

4) [40] Write a MIPS program called recurse.s that computes f(N), where N is an integer greater than zero that is input to the program. f(N) = 3*(N+1)+f(N-1)-1. The base case is f(0)=1. Your code must be recursive, and it must follow proper MIPS calling conventions. **The key aspect of this program is to teach you how to obey calling conventions → code that is not recursive or that does not follow MIPS calling conventions will be severely penalized!**

   You will upload recurse.s into Sakai (via Assignments).

5) [50] Write a MIPS program called HoopsStats.s that is similar to the C program you wrote in Homework #1. However, instead of reading in a file, your assembly program will read in lines of input as strings. That is, each line will be read in as its own input (using spim's syscall support for reading in inputs of different formats). The file is a series of player stats, where each player entry is 3 input lines long. The first input is a string for the player's last name, the second input is his jersey number (an int), and the third input is his points per game (rounded to an int). The last line of the file is the string "DONE". For example:

```
Hurley
11
15
Hill
33
17
DONE
```

IMPORTANT: There is no constraint on the number of players, so you may not just allocate space for, say, 10 basketball player records; you must accommodate an arbitrary number of players. You must allocate space on the heap for this data. **Code that does not accommodate an arbitrary number of players will be penalized!** Furthermore, you may not read the entire file to first find out how many players there are and *then* do a single dynamic allocation of heap space; instead, you must dynamically allocate memory on-the-fly as you read the file. To perform dynamic allocation in MIPS assembly, I recommend looking at:
http://chortle.ccsu.edu/assemblytutorial/Chapter-33/ass33_4.html

Your program should output a number of lines equal to the number of players, and each line is the player's name and the difference between his jersey number and his points per game (e.g., 11-15=-4 for Hurley, 33-17=16 for Hill). We'll call that stat his JPG. The lines should be sorted in ascending order of JPG, and you must write your own sorting function (you can't just use the qsort library function). For example, the output for the sample statsfile above should be:

```
Hurley -4
Hill 16
```

You will upload HoopStats.s into Sakai (via Assignments).