CDA3101 - Spring 2015
Assignment 3
Writing a Simulator for a Subset of the MIPS Instruction Set

Objectives: Learn how simple machine instructions are decoded and executed.

Your assignment is to write a simulator for a subset of the MIPS instruction set. Your simulator should read a machine code file that was output from the previous assignment. The machine code file is specified as the first argument on the command line. For instance, if your solution's executable is named *sim.exe* and the machine code file to be simulated is named *sum.obj*, then you should use the following command to simulate the execution.

```
sim.exe sum.obj
```

After reading the instructions and the data, the simulator should start the simulation from the first instruction in the machine code file and should continue the simulation until it encounters a *syscall* that exits.

You are required to simulate all the types of instructions specified in the previous assignment. The following table shows the system calls that you are required to process. The *system call code* is passed in the *$v0* register.

System Call Code	Argument	Explanation
1	*$a0* = integer	Print an integer value followed by a newline character to standard output.
5		Read an integer value from standard input and assign the value to *$v0*.
10		Exit the simulation.

The details of the simulation should be printed to a *log.txt* file in the current directory. This includes the disassembled instructions and the initial values of the data memory. At each point an instruction is executed, you should print the disassembled instruction, the current values of the registers, and the current values of the words in the data segment of memory.

You can assume that both the maximum number of instructions and the maximum number of data words is 32768. You should assume that *$zero* always has the value of zero and any write to this register will not change its value.

Below are the exceptions you must be able to recognize. For each exception you should print an appropriate message to *stderr* and exit the simulation.

Type	Explanation
illegal instruction	Illegal combination of *opcode* and *funct* field values.
illegal instruction address	PC is referencing an address outside of the text segment.
illegal data address	Load or store referencing an address outside of the data segment.
divide by zero	Integer divide by zero.

You have access to my executable by using the following command:

`~whalley/cda3101exec/sim.exe` *name*`.obj`

Be sure to only attempt to run this executable file on *linprog*, which is also where your solution will be tested. Both your output to *stdout* and the *log.txt* file are required to match my output exactly.

You can use any programming language that can be compiled or interpreted on *linprog*. You should have comments at the top of the file indicating your name, this course, the assignment, and the command used to compile or interpret your program on *linprog*. For example:

```
/*****************************************************
*                                                    *
*       Name: Joe Shmoe                              *
*      Class: CDA 3101                               *
* Assignment: Asg 3 (a simulator of a subset of the  *
*                    MIPS instruction set)           *
*    Compile: "gcc -g sim.c"                         *
*                                                    *
*****************************************************/
```

You should submit this assignment as a single file (e.g. *sim.c*) attachment in an e-mail message to your TA for your recitation section (*kiswani@cs.fsu.edu* or *yannes@cs.fsu.edu*). Do not attach a zip file as some e-mail servers will filter a message with a zip file as spam. The assignment is due at the beginning of class on February 5.