CS 2200 Homework 1

Spring 2019

Instructions:

- Please print a copy of the assignment and hand write your answers. No electronic submissions are allowed. **Please print as one double-sided page. Do NOT staple multiple sheets together. There will be a 10 point penalty if you do this improperly.**
- This is an individual assignment. You may discuss concepts but not the answers.
- Due Date: **23rd January 2019 – 6:15 PM** in recitation. Bring your BuzzCard. Show up on time.

Name:_____ **GT Username:**_____ **Section:**_____

1) Give two reasons why it is preferable to use registers over making memory accesses.

2) What are two differences between a Complex Instruction Set (CISC) ISA and a Reduced Instruction Set (RISC) ISA? Which type of ISA is LC-2200?

3) For the struct defined below, show how a smart compiler might pack the data to minimize wasted space and follow alignment restrictions. Pack in such a way that you can **guarantee aligned accesses** to all the elements of the struct. Assume the compiler **cannot** intelligently reorder fields of the struct in memory. Assume a char is 1 byte, int is 4 bytes, and a short is 2 bytes. Moreover, assume the architecture is **big endian** and supports load word, load byte, and load half word instructions.

```
struct x {
    char a;      // value 0x96
    short b;     // value 0xCAF3
    int c[2];  // values {0xDEADB33F, 0xFACEFE3D}
    char d;      // value 0x51
};
```

In the following memory picture each row represents a memory word comprising of 4 bytes, and each cell represents a byte. You do not necessarily need to use all rows. Write each byte in with the hexadecimal values from the comments above.

+3	+2	+1	+0	Starting address
				0x1000
				0x1004
				0x1008
				0x100C

4) Fill in the missing lines below. The LC-2200 assembly program should **increment** the value in the memory location **pointed to by x** (assume x is already in $s0) by each **even** number from 2 to 10. The C code is provided below. Some operands and instructions are given.

```
int x = 0;
for (int i = 2; i < 12; i += 2) {
    x += i;
}
```

```
     addi $t0, $zero, ____# loop counter
     addi $t1, ____, ____ # loop limit

loop:   ____ $t0, ____, bye
        ____ $t2, 0x0(____)
        add ____, ____, ____
        sw  ____, ____($s0)
        ____ $t0, $t0, ____
        beq ____, ____, ____

bye: ...
```

5) List **one benefit** of supporting function calls by utilizing a shadow register set and **one benefit** of supporting function calls with a calling convention (as described by the LC-2200 ISA).