CPSC 213, Summer 2007— Midterm

Date: May 30, 2007; Instructor: Mike Feeley

This is a closed book exam. No notes. Electronic calculators are permitted. Answer in the space provided; use the backs of pages if needed. There are **6** questions on **5** pages, totaling **50** marks. You have **50 minutes** to complete the exam.

NAME: _____

STUDENT NUMBER: _____

Q1	/ 12
Q2	/ 6
Q3	/ 6
Q4	/ 10
Q5	/ 12
Q6	/ 4
Total	/ 50

1. (12 marks) Short Answers.

1a. What is wrong with this explanation of how a computer accesses a variable (lets call it A): "The computer asks the memory to give it the value of the variable named A."?

1b. Consider the C global declaration: `int a[10]`. Does the compiler know the address of a[4]? If so, how does it calculate it?

1c. What is *pc-relative* addressing and why is it used?

1d. Why must the Java compiler use a double indirect (or indirect) jump (i.e., Gold `croo`, `droo`, or `eri-`) to implement certain method invocations while the C compiler can use a direct jump to implement all procedure calls (i.e., `b--- aaaaaaaa`)?

2. **(6 marks)** This program prints out the initial value of two variables `global` and `local`.

```
int global;
int main () {
    int local;
    printf ("global=%d, local=%d, k=%d\n", global, local);
}
```

When executed, it outputs `global=0, local=-1877226936` (and in fact the value of `local` appears to be random). Explain why these two variables have different initial values. Be sure to indicate where in memory the variables are stored and how their storage location impacts their initial value. State whether you think there is a good reason they have different initial values and if so clearly explain what that reason is.

3. **(6 marks)** Consider a language, call it D, that is exactly like C, except that it allows programers to declare static variables whose size is determined at runtime. In the example below, `a` is such a variable; its size is determined at runtime by the value of `i` passed in to `foo`.

```
void foo (int i) {
    int a[i];
    int j;
}
```

Your job is to assess the impact of this new feature on the machine code generated by the compiler. There are two important things that change about this machine code. What are they? Explain.

4. **(10 marks)** Consider the following Java class.

```
public class Foo {
    static int i;
    int j;
    static void foo (int k) {
        int l;

        i=0;
        j=0;
        k=0;
        l=0;
    }
}
```

Give the Gold Machine instruction(s) for assigning each of the variables `i` through `l` the value 0.

Assume 0 is stored in `r0` and that `r5` stores the stack pointer and `r7` stores the address of the object. Also assume that any static variables allocated by the compiler start at address 0x1000. Clearly state any other assumptions you make.

Your answer should be numeric Gold machine instructions. Include a comment next to each instruction explaining what it does (the disassembler-style comment is fine).

4a. The code for `i=0`:

4b. The code for `j=0`:

4c. The code for `k=0`:

4d. The code for `l=0`:

5. **(12 marks)** Here is a Gold ISA implementation of a procedure that takes two arguments and returns a value. Your gold in this question is to figure out what this procedure does. Don't panic. You can do this if you first place helpful comments next to each line of the program and then use those comments to translate the program into C-like pseudo code. The procedure has a loop in it and it computes a simple, useful function on its inputs.

```
00000100:   6605            deca r5
00000102:   6605            deca r5
00000104:   0000 00000000   ld  $0x0, r0
0000010a:   3005            st  r0, 0x0(r5)
0000010c:   3015            st  r0, 0x4(r5)
0000010e:   1050            ld  0x0(r5), r0
00000110:   1351            ld  0xc(r5), r1
00000112:   6701            not r1
00000114:   6301            inc r1
00000116:   6101            add r0, r1
00000118:   9109            beq  r1, 0x12a
0000011a:   1251            ld  0x8(r5), r1
0000011c:   2101            ld  (r1,r0,4), r1
0000011e:   1152            ld  0x4(r5), r2
00000120:   6112            add r1, r2
00000122:   3215            st  r2, 0x4(r5)
00000124:   6300            inc r0
00000126:   3005            st  r0, 0x0(r5)
00000128:   80f3            br  0x10e
0000012a:   1150            ld  0x4(r5), r0
0000012c:   6405            inca r5
0000012e:   6405            inca r5
00000130:   c600            jmp 0x0(r6)
```

5a. Place useful comments above where you think necessary to understand the program (and for part marks). Don't just say things like "adds four to register 5". Either say nothing or say something a little more useful like "discards top word from stack".

5b. Translate the program into C-like pseudo code and explain in English (briefly) what function the program computes.

6. (4 marks) BLUE TEAM ONLY
Your instruction set does not include indexed load and store instructions such as Gold `2isd` and `4sid`. Give one advantage and one disadvantage of this design choice. Give an example of a C or Java language feature (or part of one) that requires more Blue machine instructions than Gold, in light of this choice; and explain why.

6. (4 marks) GREEN TEAM ONLY
Your instruction set has five instructions that read or write memory, but does not include base plus displacement instructions such as Gold `1osd` and `3sod`. What was the tradeoff you made? Give one advantage and one disadvantage of this design choice. Give an example of a C or Java language features that requires more Green machine instructions than Gold, in light of this choice.

6. (4 marks) MAUVE TEAM ONLY
Your instruction set does not include base-plus-displacement load and store instructions such as Gold `1osd` and `3sod`. Give one advantage and one disadvantage of this design choice. Give an example of a C or Java language feature that requires more Mauve machine instructions than Gold, in light of this choice.

6. (4 marks) TERRACOTTA TEAM ONLY
In your instruction set both jumps and branches can be conditional, but your branch instructions use a 4-bit pc offset. In the Gold machine, only branches can be conditional, but these instructions have an 8-bit pc offset. What design tradeoff did you make (think about why you have only 4-bit offsets)? Give one advantage and one disadvantage of your choice. Give an example of something Terracotta can due in fewer instructions than Gold and vice versa.

First, CIRCLE YOUR COLOUR, then answer below: