Midterm Exam #1 for ECE 250, Fall 2012

name: _____

Duke students are bound by an academic integrity standard:

1. I will not lie, cheat, or steal in my academic endeavors, nor will I accept the actions of those who do.

2. I will conduct myself responsibly and honorably in all my activities as a Duke student.

Please sign your name below to acknowledge that you follow this standard:

---

1) [10 points]

(a) Add the following base-10 numbers using 6-bit 2s complement math: +12, -7. **Show your work!**

(b) You have a 32-bit machine. In C, a “long” is a 64-bit integer. How many bytes are required to hold a long* as in the following code? Explan your answer.

```
long a = 12345678;
long* b = &a;
```

2) [10] The IEEE 754 floating point standard specifies that floating point numbers have one sign bit, an 8-bit exponent (with a bias of 127), and a 23-bit significand. Represent the number -5.125 in this format.

3) [10] In the following code snippet, what are the values of the following four expressions after the program completes? **Explain your answer to the last one.**

*(p+7) =

p[3] =

p[0] =

a - p =

```
int* p = (int*) malloc(40*sizeof(int));
for (int i=0; i<40; i++){
    p[i] = i*i;
}
*p = 7;
p[7] = 3;
int* a = &p[7];
```

4) [25] Write MIPS assembly code for the following C/C++ code. **Use appropriate MIPS conventions for procedure calls**, including the passing of arguments and return values, as well as the saving/restoring of registers. Assume that there are 2 argument registers ($a1-$a2), 2 return value registers ($v1-$v2), 2 callee-saved registers ($s1-$s2), and 2 caller-saved registers ($t1-$t2). The return address register ($ra) is callee saved. Don't worry about satisfying conventions with respect to the procedure that called main(), but make sure that main() satisfies conventions with respect to foo(). Feel free to continue your answer on the next page.

```
int main (){
    int x = foo(3,4); // you must put x in $t1
}
int foo(int x, int y){
    int z = x-y; // you must put z in $s1
    int q = 1; // you must put q in $t2
    if (z<0){
        q = negate(z);
    }
    q = q + 1;
    return z+q;
}
int negate(int m){
    int p = -1*m; // you must put p in $s1
    return p;
}
```

5) [15] (a) Explain how an assembler could create a pseudo-instruction for "`NAND rd, rs, rt`" if the NAND instruction doesn't exist in the ISA but NOR and NOT do exist in the ISA. Show the code that the assembler would generate to replace a NAND instruction.

(b) A computer is a finite state machine. What constitutes the architectural (i.e., software-visible) state of a computer?

(c) Why is the immediate value in most branch instructions (e.g., bgt r1, r2, imm) negative?