Short Warmup Test 1

This “warm-up” test is by no means meant to be comprehensive (see the study guide for a comprehensive list), but it will give you some sample problems on which to work. I recommend that you work on them with a friend from class. Remember no calculators will be permitted when you take the test.

1. Perform the following conversions:
   (a) \(27_{10} = \) ____________
   (b) \(23_{4} = \) ____________
   (c) \(1011011_{2} = \) ____________

2. Assume the following hex value represents a IEEE 754 floating point value. What is the value in decimal?
   \(C220\ 0000\)

3. Assume the following hex values represent two IEEE 754 floating point values? Calculate the sum of the two values. Show your work.
   \(4210\ 0000\)
   \(4150\ 0000\)

4. Give the hex for the largest positive value that can be stored in a 32-bit signed integer.

5. Give the hex for the largest value that can be stored in a 32-bit unsigned.

6. Give the hex for largest negative value that can be stored in a 32-bit signed integer.

7. Compute each of the following binary calculations on 8 bit signed numbers and indicate whether it results in a positive overflow, negative overflow or no overflow.
   (a) \(10000000 + 11000001\)
   (b) \(01110010 + 00011001\)
   (c) \(01110011 + 11100001\)
   (d) \(01101011 + 00100100\)

8. Assuming that %eax contains the value 3 and %ecx contains the value 2, indicate the value that would be stored in %edx after each of the following statements.
   (a) leal 4(%eax, %ecx), %edx
   (b) leal 4(%eax, %ecx, 3), %edx
   (c) leal (%eax, %ecx), %edx
9. The function compute below has this prototype: \texttt{int compute(int x, int y)};

```assembly
compute:
pushl %ebp
movl %esp, %ebp
subl $4, %esp
movl $0, -4(%ebp) #\text{mem[-4 + %ebp]} contains local variable i
.L3:
movl 8(%ebp), %eax #\text{mem[8 + %ebp]} contains x
cmpl 12(%ebp), %eax #\text{mem[12 + %ebp]} contains y
j1 .L5
jmp .L4
.L5:
leal -4(%ebp), %eax
incl (%eax)
incl 8(%ebp)
jmp .L3
.L4:
movl -4(%ebp), %eax
leave
ret
```

(a) Assuming the value of \texttt{x} is 3 and the value of \texttt{y} is 5, what value is returned by the compute function?

(b) Give equivalent C code using gotos.

10. What is the difference between a little endian and a big endian machine?

11. Recall the following function show\_bytes that is used to print the byte representation of program objects.

```c
typedef unsigned char * byte_pointer;
void show_bytes(byte_pointer start, int len)
{
    int i;
    for (i = 0; i < len; i++)
        printf("%.2x ", start[i]);
    printf("\n");
}
```

Suppose I write a program that calls show\_byte like this:

```c
int j = 0x12345678;
show_bytes((byte_pointer) &j, sizeof(int));
```

and the output is:

```
78 56 34 12
```

What does this say about the endianness of the machine?

12. Write a C function called bitClear that takes as input two ints called \texttt{value} and \texttt{bitNo} and returns \texttt{value} with the bit at position \texttt{bitNo} in \texttt{value} cleared (set to zero). For example,

```c
printf("%x", bitClear(0xffffffff, 3));
```

would cause the value \texttt{0xffffffff7} to be displayed. You may assume that \texttt{bitNo} contains a number between 0 and 31 (inclusive).