CSCI-UA.0201-001

Computer Systems Organization

Midterm Exam Fall 2015 (time: 60 minutes)

Last name: First name:

Notes:

- **If you perceive any ambiguity in any of the questions, state your assumptions clearly.**
- **Questions vary in difficulty; it is strongly recommended that you do not spend too much time on any one question.**

1. [1 point] Why do we need to know such information as an integer is 4 bytes in length?

To know the upper and lower bound of the numbers a variable can carry in our programs. Going below the lower bound or above the upper bound will cause wrong number in the variable, leading to a bug that may not be easily found if we didn't know this info,

2. [2 points] Beside dynamic allocation, state two other reasons as to why do we need pointers.

- Passing a complex data structure (e.g. arrays, structures, …) to a function as an argument.
- Allow a function to modify its arguments.

3. [4 points] The following C code is buggy. List all the bugs you can find. No need to fix them.

```
struct  _node{
        int x;
        int y; }

int populate_list( int M){

            struct _node * employees;
            int i;
           employees = (struct _node *) malloc(M * sizeof(struct _node *));

           for( i = 0;  i < M;  i++){
                  employees[i].x = i;
                  employees[i].y = i*2;
               }
  }
```

4. [2 points] Can the zero flag (ZF) and the sign flag (SF) be both 1 at the same time? If yes, give an example of an operation that does this (no need for assembly code, just describe the operation). If not, explain why not.

No, they cannot.
Because the ZF is set to 1 when all the bits of the result is 0. SF is set to 1 when the most significant bit of the result is 1. So, the most significant bit cannot be 0 and 1 at the same time.

5. [2 points] State two reasons for why do we need an assembler and not making the compiler generate the binary presentation right away.

- To be able to compare assembly code with HLL code. This enables professional programmers to enhance the quality (i.e. performance, etc) of their code. So, we need assembly code to be generated.
- The compiler is already a complicated piece of software. We don't want to add another task to it.
- Compiler and assembler are doing two different tasks, hence each one can be optimized differently.

6. [4 points] Suppose we have the following C code (assuming a, b, and b are unsigned integers):

```
if( a == b && b >  c)
    c += a + b;
```

Write the corresponding assembly code, assuming:
a will go in %eax, b in %ebx, and c in %ecx)

```
     cmpl %ebx, %eax
     jne out
     cmpl %ecx, %ebx
     jb out
     je out
L1:  addl %eax, %ebx
     addl %ebx, %ecx
out:
```

7. Suppose x is an integer (i.e. 4 bytes). We want to test whether the 3rd least significant bit of x is 1 or not (i.e. the 3rd bit from the right), so we wrote the expression:

if((x & mask) != 0)

a. [1 point] What is the value of mask, both in binary and hexadecimal?

0x00000004
0000 0000 0000 0000 0000 0000 0000 0100

b. [2 points] Which of the following expressions generate correct mask? Circle ALL correct answers. There may be more than one correct answer, or there may be none!

- 1 << 3
- [✓ 1 << 2]
- [✓ two’s complement of 0xFFFFFFFC]
- two’s complement of (-2)

c. [2 points] Please give the expression that sets the 3rd bit from left of x to 1 and leave all the other bits unchanged.

x |= 0x20000000