CSCI-UA.0201-003

Computer Systems Organization

Final Exam - Fall 2016 (time: 75 minutes)

Last name: First name: NetIT:

(Total 30 points)

1. (5 points) Circle the correct answer among the choices given. If you circle more than one answer, you will lose the grade of the corresponding question.

(A) Whenever there is a cache miss, a replacement policy must be used.

a. The above statement is true.

b. The above statement is false.

c. It depends on the design of the cache.

(B) During execution, the CPU encountered a "divide by zero" event, what will happen then?

a. execution of a system call

b. execution of an exception

c. execution of an interrupt

d. none of the above

(C) Which of the following does not decrease even if we increase cache associativity and the total cache size?

a. conflict misses

b. compulsory misses

c. capacity misses

d. none of the above

(D) Getting a TLB hit means level 1 cache will not be accessed

a. True

b. False

c. Depends on the OS

e. Cannot tell

(E) Assume a signed int x, what does the following expression present:

(1 + (x<<3) + ~x + x)

a. 7x

b. 8x

c. 8x + 1

d. 7x + 1

e. none of the above

2. [12 points] Given the following x86_64 assembly code and its corresponding C code, fill-in the blanks in the corresponding C code. Also on the far right, fill in the correspondence between each register and its corresponding variable in C. (Hint: you can neglect edx because it does not correspond to any variable in the C code and is used only by the compiler for the translation)

```
foo: xorl %eax, %eax
     movl $32, %ebx
     movl $0, %ecx

L0: cmpl %ebx, %ecx
    jge L1
    cmpl  %edi, %eax
    jle L2
    addl  %esi, %eax
    jmp L3

L2: movl %edi, %edx
     subl  %esi, %edx
     addl  %edx, %eax

L3: addl  $2, %ecx
     jmp L0
L1: Ret
```

```
int foo(int x, int y){
   int j = __32_____;
   int sum = ____0____;
   int i;

    for(i = _0_; i < j_; __i+= 2_){
        if( sum > x)
         ___sum += y_____;
         else
         ___sum += x-y___ ;
          }
    return sum;
}
```

edi	x
esi	y
ebx	j
eax	sum
ecx	i

3. [5 points] A cache has an access time of 1 cycle. The computer with that cache experienced an average memory access time of 10 cycles, and the cache hit rate is 90%. What is the access time of the main memory? Did we benefit from having a cache in this system? Why? [A correct final answer without the steps leading to that answer will earn you a ZERO]

avg mem access = m + (1-p)M from the problem: avg = 10 m = 1 p = 0.9

→ M = 90 cycles

yes, we benefit from the cache because with cache avg mem access = 10 cycles and without cache it is 90 cycles

4. For the following piece of code:

```
void doit() {
        if (fork() != 0) {
                fork();
                printf("hello\n");
                }
        return;
}

int main() {
        printf("hello\n");
        doit();
        printf("hello\n");
        exit(0);
}
```

a. [1 point] How many times will “Hello” be printed?

6

b. [1 point] Before any of the processes exit*;* how many stacks exist?

3

c. [1 point] If the system that is executing the above code is a single-core system and is using one-level page table. How many TLBs exist in the system *before* any of the processes execute exit(0)?

1

5. [5 points] Suppose we have the following C function:

```
void memory(){
        int * p;
        int * q;
        int * r;
        int * k;

        p = (int *) malloc(100*sizeof(int));    ← 1
        q = (int *) malloc(50*sizeof(int));     ← 2
        free(p);
        r = (int *) malloc(60*sizeof(int));     ← 3
        k = (int *) malloc(50*sizeof(int));     ← 4
        free(r);
        p = (int *) malloc(100*sizeof(int));    ← 5
          ….. rest of the code …
}
```

For each one of the numbered malloc above, indicate in the following table whether that call needs to do a syscall or not. You do not need to explain. We assume that the allocator will allocate exactly what is specified in malloc and not more.

malloc #	**syscall Yes/No**
1	Y
2	Y
3	N
4	Y
5	N