CPSC 3220 – Quiz #3 February 7, 2018 Name: ___**Answer Key**_

1. (2 points) Can UNIX/LINUX fork return an error? Why or Why not?

Yes, the system runs out of resources (memory, Thread Control Blocks(TCB), etc.)

2. (2 points) Given the code to the right;

   Explain what must happen for the UNIX wait (line 8) to return immediately and successfully.

```
1 main() {
2   int child_pid = fork();
3   if (child_pid == 0) {
4     printf (“I am process #%d\n”, getpid());
5     return 0;
6   } else {
7     printf (“I am process #%d\n”, getpid());
8     wait(child_pid);
9     printf (“I am the parent of process #%d\n”, child_pid);
10    return 0;
11  }
12 }
```

The child process of the fork has run to completion before the wait system call is made.

3. (3 points) Given the code to the right,

   How many different copies of the variable x are there?

   What are their values where their process finishes?

```
1 main() {
2   int child = fork();
3   int x = 5;
4   if (child == 0) {
5     x += 5;
6   } else {
7     child = fork();
8     x += 10
9     if (child ) {
10       x += 5;
11  }
12 }
```

3 copies, **{20, 15, 10}**

Continued on Back

4. (2 points) UNIX/LINUX uses one system command “open(args)” for I/O.

Why does Unix NOT use “open/create/exists” for I/O?

The UNIX/LINUX system call open(args) is an all-purpose atomic instruction.

The arguments to the system call control the behavior (opening,creating,appending,reseting,…) of the I/O.

A single system call simplifies the interface to the user apps.

5. (2 points) How is a microkernel “better” than a Monolithic kernel?

Smaller core kernel

More modular libraries

Easier O/S updates

Simpler code base

6. (1 point) Which UNIX/LINUX system call creates a new process?

exec() or fork()

fork() is the system call that creates a new process by “cloning” the currently running process.

exec() is used to run a different process (but does not create a new process)