Com S 352
Assignment 1
Due: September 8, 2017

1.14 (10 points) Under what circumstances would a user be better off using a timesharing system rather than a PC or a single-user workstation?

1.20 (10 points) Direct memory access is used for high-speed I/O devices in order to avoid increasing the CPU's execution load.

a. How does the CPU interface with the device to coordinate the transfer?
b. How does the CPU know when the memory operations are complete?
c. The CPU is allowed to execute other programs while the DMA controller is transferring data. Does this process interfere with the execution of the user programs? If so, describe what forms of interference are caused.

1.25 (10 points) Describe a mechanism for enforcing memory protection in order to prevent a program from modifying the memory associated with other programs.

2.18 (10 points) What are the two models of inter-process communication? What are the strengths and weaknesses of the two approaches?

2.23 (10 points) How are iOS and Android similar? How are they different?

6. (**25 points**) Review the memory management for a C program (read the recitation notes if you haven't got chance to attend the recitation), and then study the following C code and answer questions:

```
char a;
int main()
{
        foo();
        ...
}
int foo()
{
        char *b;
        char *c;
```

```
        char d;
        b=(char *)malloc(10);
        c=(char *)&a;
        d=10;
        return d;
}
```

Where (data segment, stack, or heap) are
(1) variable a,
(2) variable b,
(3) the space pointed by b,
(4) variable c,
(5) the space pointed by c,
and (6) variable d
stored, respectively? After function foo finishes its execution, which of the above variable/space are/is reclaimed by the OS?

7. (**25 points**)When the following C program is run in a Linux system, the execution of which line(s) must trigger invocation(s) of system call and/or exception? When the program will terminate?

```
int main()
    {
        double w=1234.789;
        double x=0;
        double y;
        double *z;

(1)     z=(double *)&w;
(2)     scanf("%f", &y);
(3)     *z=sqrt(w);
(4)     y=y*2.0;
(5)     printf("y=%f, *z=%f\n", y, *z);
(6)     y=y/x;
(7)     w=y*z;
(8)     return 0;
    }
```