Chapter 1, 2, 4 and 6

1. Define these terms in one sentence!
   a. OS
   b. Process
   c. Interrupt
   d. System call
   e. Context swapping
   f. Semaphore
   g. Spooling
   h. Ready queue
   i. Busy waiting
   j. Critical section
   k. Multiprogramming
   l. Time-sharing
   m. CPU bound
   n. Deadlock
   o. Polling
   p. PCB
   q. Starvation
   r. Mutual exclusion
   s. Long-term scheduling
   t. Concurrency!

2. Why is DMA access to main memory (often) given a higher priority than processor access to main memory?

3. Consider a concurrent system with two processes, P and Q, shown in the following code. A, B, C, D, and E are arbitrary atomic statements. Assume that these two processes are executing concurrently.

   ```
   Process P
   { A; D; B; E; C; }
   }
   Process Q
   }
   ```

   Show all the possible interleavings for the execution of P and Q. That is, give each possible execution "trace", in terms of the atomic statements, that could occur.

4. Consider the following program:

   ```
   const int n = 50;
   int tally;
   void total(){
     int count;
     for (count=1; count<=n; count++)
       tally = tally + 1;
   }
   void main(){
     tally=0;
     Cobegin{
       total();
       total();
     }
     Cout<< tally <<'\n';
   }
   ```
5. Consider the following software solution for the mutual exclusion problem that was presented in the January 1966 issue of *Communications of the ACM*:

```cpp
boolean blocked [2];
int turn;

void P(int id){
    while (true){
        blocked[id] = true;
        while (turn != id)
            while (blocked[1-id])
                turn = id;

        /* critical section */
        MutexBegin()
            blocked[id] = false;
        MutexEnd()

        /* Reminder section */
    }
}

void main(){
    blocked[0] = false;
    blocked[1] = false;
    turn = 0;
    cobegin{
        P(0);
        P(1);
    }
}
```

Is this a correct solution?! Describe your answer completely.

6. Show that Semaphores and Monitors have the same ability and effect.
   a. By implementing a Semaphore with Monitors
   b. By implementing a Monitors with Semaphore