

# COS214 Tutorial 10

*Roberto Togneri, 2000*

1. The UNIX *ln* function is used to create a link to a file. Show how it can be used as an effective mechanism for mutual exclusive access to a file if there are multiple processes attempting to access a file. Are there any problems with this solution?
2. The readers and writers problem can be formulated in several ways with regard to which category of processes can be started and when. Carefully describe three different variations of the problem, each one favoring (or not favoring) some category of processes. For each variation, specify what happens when a reader or a writer becomes ready to access the database, and what happens when a process is finished using the database. Carefully analyse the monitor solution presented in lectures, which variation does it represent?
3. *The Cigarette-Smokers Problem.* Consider a system with three smoker processes and one agent process. Each smoker continuously rolls a cigarette and then smokes it. But to roll and smoke a cigarette, the smoker needs three ingredients: tobacco, paper, and matches. One of the smoker processes has paper, another has tobacco, and the third has matches. The agent has an infinite supply of all three materials. The agent places two of the ingredients on the table. The smoker who has the remaining ingredient then makes and smokes a cigarette, signaling the agent on completion. The agent then puts out another two of the three ingredients, and the cycle repeats. Complete the following program fragment to synchronize the agent and the smokers:

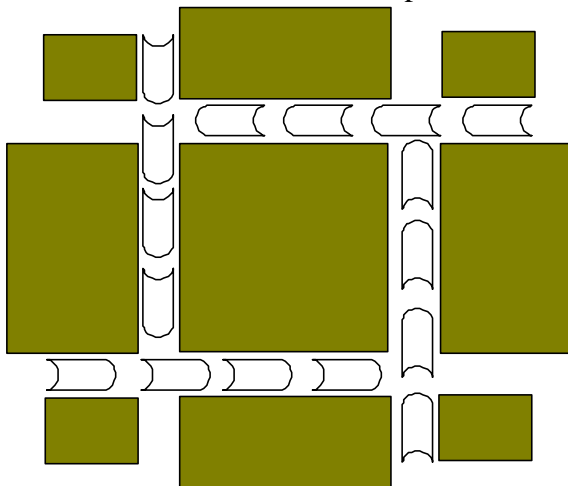
```
semaphore a[3] = 0;          /* a[0] for tobacco, a[1] for paper, a[2] for matches */
semaphore agent = 1;
```

```
Agent(void) {
    int i,j;
    repeat
        i = 3 * drand48( );    /* returns a random integer 0, 1 or 2 for i */
        j = 3 * drand48( );    /* returns a random integer 0, 1 or 2 for j */
        while (i != j) {      /* i and j must be different */
            the rest of it
        }
    until false;
}
```

```
Smoker(int r) {              /* r indicates which ingredients this smoker has */
    repeat
        the rest of it
    until false;
}
```

4. (a) Consider a system consisting of a number of processes attempting to access three identical line printers. Write a monitor that allocates line printers to these processes.
- (b) Parallel execution of list operations (e.g.  $\max()$ ,  $\min()$ ,  $\text{sort}()$ ) is trivially accomplished using a master-slave paradigm. Consider the case of one master and two slave processes. The master takes the list and divides it into two halves and passes one half to  $\text{slave}[0]$  and the other half to  $\text{slave}[1]$ . The slave processes then concurrently perform the list operation on their local list (e.g. find the local maximum for that half of the list) and return their result to the master (e.g. the local maximum). The master then merges the result from each slave to derive the global result (e.g. the maximum of each local maximum returned is the global maximum of the whole list). Describe the synchronisation between the master and the two slaves using message passing.

5. Consider the traffic deadlock depicted below:



- (a) Show that the four necessary conditions for deadlock indeed hold in this example.
  - (b) State a simple rule that will avoid deadlocks in this system.
6. In an electronic funds transfer system, there are hundreds of identical processes that work as follows. Each process reads an input line specifying an amount of money, the account to be credited, and the account to be debited. Then it locks both accounts and transfers the money, releasing the locks when done. With many processes running in parallel, there is a very real danger that having locked account  $x$  it will be unable to lock  $y$  because  $y$  has been locked by a process waiting for  $x$ . Devise a scheme that avoids deadlocks. Do not release an account record until you have completed the transactions (e.g. don't lock one account and then release it immediately if the other account is found to be locked). Why is this last condition important?