

Problem No. 1

In the fall of 1996, Hurricane Fran hit Durham, which is why you *still* see so many trees down in the woods. For the sake of local history, here is a Hurricane Fran updating of the traditional cigarette man's problem.

Three men are in a neighborhood with hundreds of fallen trees. To cut up a fallen tree, each man needs all three of the following items: a chainsaw with a sharp chain, gasoline mix, and chain lubricating oil. There is a mailorder loggers' supplier with new chains, gas, and oil in ample quantity making deliveries to the area. Each neighbor has his own chainsaw but only one of them has a sharpener to keep his chain sharp. Another neighbor has a large tank of the gasoline mixture. The third neighbor has lots of lubricating oil. The action begins when a delivery of two of these items arrives in the neighborhood which would allow one of the neighbors to cut up a tree until the delivered items become exhausted (e.g. for the third neighbor this would mean that his chain gets dull and his gas is gone, but he still has enough oil). When the lucky neighbor who got the benefit of the last shipment is done, he places another order to the mailorder supply, which sends two more items (at random), thus enabling another of the neighbors to obtain all three items and cut up a tree.

Representing the men and the supplier as threads, write the pseudocode that solves the synchronization constraints in this problem using semaphores.

First attempt (BAD!!!):

```
// All semaphores except lock are initialized to 0.  
// lock is initialized to 1, and is a mutex variable.
```

```
do forever {  
    P( lock );  
    randNum = rand( 1, 3 );  
    if ( randNum == 1 ) {  
        // Put chain on table  
        // Put mix on table  
        V(chain); // Wake up man with chain  
        V(mix); // Wake up man with mix  
    } else if ( randNum == 2 ) {  
        // Put mix on table  
        // Put oil on table  
        V(mix); // Wake up man with mix  
        V(oil); // Wake up man with oil  
    } else {  
        // Put chain on table  
        // Put oil on table  
        V(chain); // Wake up man with chain  
        V(oil); // Wake up man with oil  
    } // Wake up man with mix  
    V( lock );  
    P( agent ); // Agent sleeps  
} // end forever loop
```

the code for one of the men. The others are analogous.

```
1  do forever {  
2      P(chain);  
3      P(mix);  
4      P( lock );  
5      // Pick up chain  
6      // Pick up mix  
7      V( agent );  
8      V( lock );  
9      // cut up a tree  
10 }
```

Any problems with this code?

```
1.    V(chain); // Wake up man with chain
2.    V(mix);    // Wake up man with mix
...
5.    P( agent );
```

can't proceed

```
P(chain);
P(mix);
```

can't proceed

```
3. P(chain);
   P(oil);
```

can't proceed

```
4. P(mix);
   P(oil);
```

can't proceed

DEADLOCK!

correct solution

// Adapted from: <http://www.cs.umd.edu/~hollings/cs412/s96/synch/smokers.html>

// All semaphores except lock are initialized to 0.
// lock is initialized to 1, and is a mutex variable.

```
1  do forever {
2    P( lock );
3    randNum = rand( 1, 3 );
4    if ( randNum == 1 ) {
5      // Put chain on table
6      // Put mix on table
7      V(oil); // Wake up man with oil
8    } else if ( randNum == 2 ) {
9      // Put mix on table
10     // Put oil on table
11     V(chain); // Wake up man with chain
12   } else {
13     // Put chain on table
14     // Put oil on table
15     V(mix); } // Wake up man with mix
16   V( lock );
17   P( agent ); // Agent sleeps
18   } // end forever loop
```

I will give code to one of the men. The others are analogous.

```
1  do forever {
2    P(oil); // Sleep right away
3    P( lock );
4    // Pick up chain
5    // Pick up mix
6    V( agent );
7    V( lock );
8    // cut up a tree
9  }
```