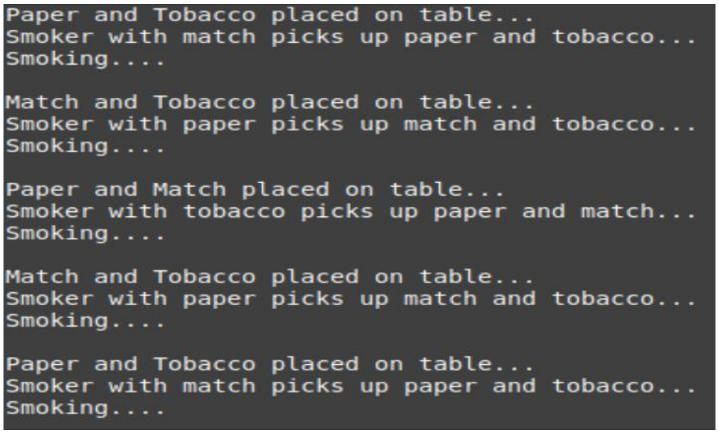
Matthews Cardenas

OS Lab



```
Paper and Tobacco placed on table...
Smoker with match picks up paper and tobacco...
Smoking....

Match and Tobacco placed on table...
Smoker with paper picks up match and tobacco...
Smoking....

Paper and Match placed on table...
Smoker with tobacco picks up paper and match...
Smoking....

Match and Tobacco placed on table...
Smoker with paper picks up match and tobacco...
Smoking....

Paper and Tobacco placed on table...
Smoker with match picks up paper and tobacco...
Smoking....
```

Both semaphores and mutexes where used to synchronize the agent and smoker processes.

Semaphores were used to accomplish this task by first locking all the semaphores except for lock which was initialized to 1. Then a parent agent process and 3 child smoker processes were created. For the agent, a for-loop was used to run the process 10 times. Going into the for-loop, wait was used on the lock semaphore, then a random number picked (1-3). Whichever the number picked corresponds to, a signal is used on one of the smoker semaphores, and the loop ends using signal on the lock and agent semaphore. The child process consists of an infinite while loop waiting to use wait on lock and one of the smoker semaphores which are both signaled by the agent process. After, the print statements are executed, and a signal is used on the agent and lock semaphore.

To accomplish this task with mutexes, the same implementation was used but with all the semaphores being replaced with mutexes and all processes being put into functions which threads can call. The mutexes were created and initialized, and all except for the lock mutex were locked. Then threads were created to run each function/process. The logic of locking and unlocking of the mutexes is the same as the waiting and signaling of the semaphore. Lastly, the program waits for the thread running the agent function/process to end before terminating.

In this case both semaphores and mutexes had a similar use since the semaphore needed was binary. Although they were used identically here, their uses differ. Mutexes are used to lock a resource while semaphores are used to signal when a process is done with the resource others are requesting.