Fall 2015
CS3413
Lab3

Semaphores

Note: for this lab compile programs with "-lpthread" option

The Smoker's Problem
There are four processes in this problem: three smoker processes and an agent process. Each of the smoker processes will make a cigarette and smoke it. To make a cigarette requires tobacco, paper, and matches. Each smoker process has one of the three items. i.e., one process has tobacco, another has paper, and a third has matches. The agent has an infinite supply of all three. The agent randomly places two of the three items on the table, and the smoker that has the third item makes the cigarette. After having a smoke a process sleeps for a random amount of time. Write a pthread program using semaphores to synchronize the processes.

Once you complete a task, please submit your solution via D2L.

POSIX semaphore functions and types are defined in semaphore.h.
To define a semaphore object, use

```
sem_t sem_name;
```

To initialize a semaphore, use sem_init():

```
int sem_init(sem_t *sem, int pshared, unsigned int value);
```

- sem points to a semaphore object to initialize
- pshared is a flag indicating whether or not the semaphore should be shared with fork()ed processes. LinuxThreads does not currently support shared semaphores
- value is an initial value to set the semaphore to

Example of use: `sem_init(&sem_name, 0, 10);`

To wait on a semaphore, use sem_wait:

```
int sem_wait(sem_t *sem);
```

- If the value of the semaphore is negative, the calling process blocks; one of the blocked processes wakes up when another process calls sem_post.

Example of use: `sem_wait(&sem_name);`

To increment the value of a semaphore, use sem_post:

```
int sem_post(sem_t *sem);
```

- It increments the value of the semaphore and wakes up a blocked process waiting on the semaphore, if any.

Example of use: `sem_post(&sem_name);`