The essential system calls for Project #2 (CS314-001/002, Spring 2019)

(a) Fork system call

- Structure
- Variables (which variables are inherited and which variables will not be inherited between the parent and the child process)
- Risk in calling “fork” in a loop

(b) Shared memory

1. Define the contents of a shared memory (as C “struct”)
2. Obtain the size of the shared memory
3. Create the shared memory (“shmget”)
4. Attach the shared memory (“shmat”) – “shmat” returns a pointer
5. Initialize the contents in the shared memory (using the pointer)
6. Detach the shared memory (“shmdt”)
7. Delete the shared memory (“shmctl”)

(c) Semaphores

1. Prepare “semaphore operation array” (“struct sembuf operations[1];” – for manipulating a semaphore)
2. Prepare “semaphore control data structure” (“union semun” – for initializing a semaphore)
3. Set the initial value of a semaphore (argument.val = 1;)
4. Create a semaphore (“semget”)
5. Initialize the new semaphore (“semctl”)
6. Use the semaphore by using “semaphore operation array” (“semop”)
   - for “wait” operation
   - for “signal” operation
7. Delete a semaphore (“semctl”)