1. /* counting semaphore, opposite to binary semaphore or mutex lock, means more than one process can have connection at the same time. */
   // here is an example of counting semaphore.

```
struct Semaphore {
        int value = N; // N is the number of connections allowed at the same time
        vector<process> v; // using a vector to store all the awaiting processes

}

void HandleNextIncomingConnection(Semaphore s) {
        s.value--;
        if (s.value < 0) {
                v.push(pid); // put current process to wait list
                block();
        }
        else{return;}
}

void CloseExistingConnection (Semaphore s) {
        s.value++;
        if (s.value <= 0) {
                // remove process p from queue
                p = v.pop();
                wakeup(p);
        }
        else{return; }
}
```

2. The maximum number of processes that can be in the critical section at the same time will be 12. Since one of the process does an increment of one on the semaphore instead of decrement of one, before the end of the process, the value of the semaphore will be higher than the correct one by two, which allows two more processes to enter the critical section. Therefore, it is possible that 10 + 2 = 12 processes enter the critical section at the same time.

3. // this code is written for female employees by default. male code is similar.
   // assume the queue for bathroom is FIFO
   struct Semaphore {
   int value = 3; // 3 is the max number of employees allowed at the same time
   vector<process> E; // using a vector to store all the awaiting employee
   }

   void checkEmpty(Semaphore s) {
   if(s.value == 3) {
   Empty = true;
   }else {
   Empty = false;
   }
   }

   void Enter(Semaphore s) {
   s.value--;
   if (s.value < 0) {
   E.push(e); // put current employee to wait line
   block();
   }
   else{return;}
   }

   void Leave(Semaphore s) {
   s.value++;
   checkEmpty(s); // check if the last employee in the bathroom is leaving
   if (s.value <= 0) {
   // remove process p from queue
   Process p = E.pop();
   wakeup(p);
   }
   else{return; }
   }
   Semaphore B; // Bathroom
   bool Empty = true;
   int women_multiplex = 10;
   int men_multiplex = 10;
   for(;;) { // forever
   // this code is for female employees. male code is similar.
   // thus assume all processes here are female
   <here the employee is at his/her desk>
   // here comes the code for the employee to access the bathroom.
   If(women_multiplex==0) {// stop for same sex

```
        while(!Empty){
                sleep(1000); // sleep till the bathroom is empty
        }
        women_multiplex = 10; // restore status
        sleep(10000); // sleep to wait other sex to enter bathroom
        while(!Empty){
                sleep(1000); // sleep till the other sex releases the bathroom
        }
}
        if(men_multiplex == 10 || Empty) {/* female only enter bathroom if no male in
the bathroom*/
                women_multiplex--;
                Enter(s);
                <here the employee is in the bathroom>
                Leave(s);
        }else {
                E.push(e); // put current employee to wait line
                block();
        }
}
```