Xingjian Shen

NUID: 25024803

HW2

1.

```
void lock(int * key) {
        do XCHG(key, &bolt)
        while(*key == 1);
        return;
}

void unlock(int * key) {
        XCHG(key, &bolt);
}
```

2.

No, this approach will not suffer from race conditions.

Explain: As long as there are mails in the mailbox, the process receiving mails is not blocked and it is the case the other way around, in which when the sending process does not have an error sending to full mailbox, sending can be done. The two processes keep checking if the mailbox is empty or full until they receive an error, so the race condition would not happen because the increment or decrement on the amount of mails in the mailbox is updated every time the two processes check.

3.

```
sem passenger_num = 0;
sem car_num = 0;

void CarThread(int i) {
        while(1) {
                sem_wait(passenger_num);
                sem_signal(car_num);
                ride();
        }
}


void PassengerThread(int i) {
        sem_signal(passenger_num);
        sem_wait(car_num);
}
```

4.

```
sem mutex= 1; sem full = 0; int count = 0;

void bear(void) {
        while(1) {
                semWait(&full);
                semWait(&mutex);
                eat();
                count = 0;
                semSignal(&mutex);
        }
}


void honeybee(void) {
        while(1) {
                semWait(&mutex);
                if(count < H) {
                        fill();
                        count++;
                }
                semSignal(&mutex);
        }
}
```