**Задача 1**

Покажем, что если в doorway-секции метода lock поменять местами (want[t].store(true)) и victim (victim.store(t)), то полученный алгоритм не будет гарантировать взаимное исключение.

void lock(int t) {

victim.store(t);

want[t].store(true);

while (want[1 - t].load() && victim.load() == t) {

// wait

}

}

Пусть порядок выполнения команд для потоков A и B такой (поток A - 0, B - 1):

**A:** victim.store(0) -> **B:**victim.store(1) -> **B**:want[1].store(true) -> **B:** while (want[0].load() && victim.load() == 1) -> **A:** want[0].store(true) -> **A:** while (want[1].load() && victim.load() == 0)

Тогда, когда поток B проверяет условие для входа в цикл, он видит, что want[0] == false, значит, он пройдет в критическую секцию. Далее, когда поток A проверяет условие (происходит до того, как вызван unlock() потока B), want[1] = true и victim == 1, значит, он тоже пойдет в критическую секцию, и в ней будет два потока одновременно, что противоречит взаимному исключению.