1. Distributed laxlock program without busy waiting

```
program LaxDist2(N) {
  Map v
  v[0] ← startSystem(Lax2(0))
  v[1] ← startSystem(Lax2(1))
} // LaxDist2
```

Obtain program `Lax2` such that

- `LaxDist2(N)` implements the laxlock service `DistLax(N)`, and
- `Lax2` has no busy waiting and no local thread.

2. Proof of implements

Prove that `LaxDist2(N)` implements `DistLax(N)`.

That is, introduce atomicity breakpoints in `Z`, state the appropriate assertions, and prove that `Z` satisfies them.

```
program Z(int N) {
  ic {N ≥ 1}
  inputs(); outputs()
  Map v
  v[0] ← startSystem(Lax2(0))
  v[1] ← startSystem(Lax2(1))
  si ← startSystem(LaxLockInverse(N,v))
}
```

Note:

In problem 1, try to develop the program along with key assertions. (What does this mean? Ask in class.)

Then in problem 2, you won't have to invent predicates to satisfy proof rules.