1)

Consider the following invariants:

[inv1] m(p1)+m(p2)+m(p3)+m(p4)+m(p5) = ph1+ph2+ph3+ph4+ph5

[inv2] |m(p7)|+|m(p2)| = 4

[inv3] LF(m(p4))+RF(m(p4)) + m(p6) = f1+ f2+ f3+ f4+ f4 + f5

Now consider two cases:

(a) m(p4)+m(p5) ≠ 0. Then either left\_fork or return\_right\_fork\_and\_exit\_dining\_room can be fired.

(b) m(p4)+m(p5) = 0. Then, using [inv3], we get LF(m(p3)) + m(p6) = f1+ f2+ f3+ f4+ f4 +f5. Also, using [inv1], we get m(p1)+m(p2)+m(p3) = ph1+ph2+ph3+ph4+ph5.

We then use the definition of LF(x) and RF(x) to lay out the foundation of the proof:

LF(x) != RF(x) for all x, where x = ph1, ph2, ph3, ph4, or ph5.

Therefore, when m(p3) != 0, then take\_right\_fork is fired. Then, if m(p2) != 0, then take\_left\_fork is fired.

If m(p1) != ph1+ph2+ph3+ph4+ph5, this must mean that either m(p3) != 0 or m(p2) != 0.

If m(p1) = ph1+ph2+ph3+ph4+ph5, this must mean that m(p2) = 0, which means that by [inv2], m(p7) = 4. This means that enter\_dining\_room can be fired.

2)

a) Solved in q2a.lts.

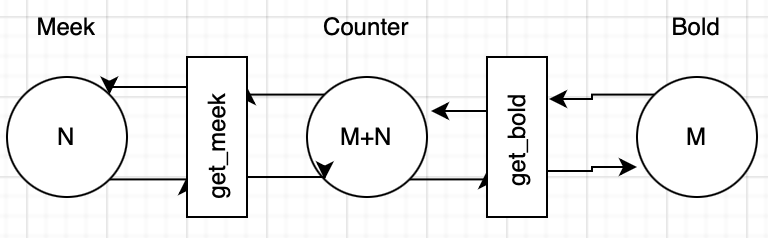
b) Solved in q2b.lts. There is a violation of the property.

c) Solved in q2c.java.

3)

a) Solved in q3a.lts

b)



c) Solved in q3c.lts.

Note that meek.getcheese will always be starved, as bold.getcheese will always be executed.

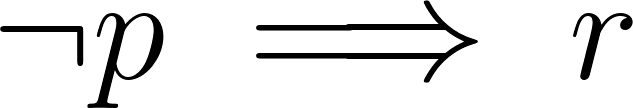
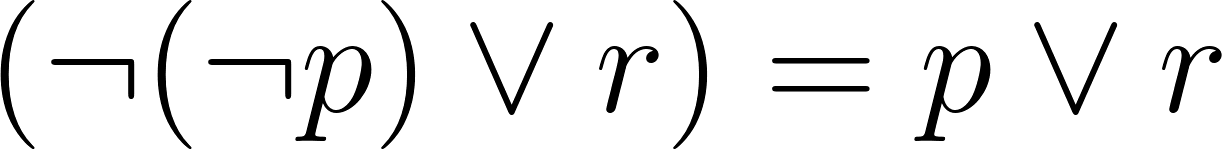
4) Solved in q4.lts

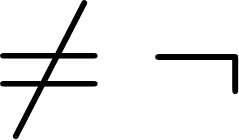
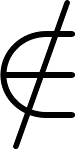
5) Solved in q5.java

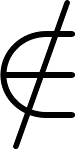
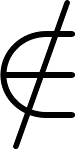
6) Solved in q6.lts

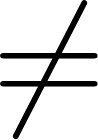
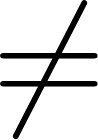
7)

a)

i) ( [](https://www.codecogs.com/eqnedit.php?latex=%5Cneg%20p%20%5Cimplies%20r#0)) is equivalent to [](https://www.codecogs.com/eqnedit.php?latex=(%5Cneg%20(%5Cneg%20p)%20%5Clor%20r)%20%3D%20p%20%5Clor%20r#0). We have L(s0) = {r} so M, s0 |= φ. We have L(s2) = {p,q}, so M, s0 |= φ

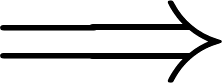
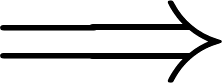
ii) It is established that r [](https://www.codecogs.com/eqnedit.php?latex=%5Cin#0) L(s0) and r [](https://www.codecogs.com/eqnedit.php?latex=%5Cin#0) L(s1). There is an infinite path, which is s0 -> s1 -> s1… . This means that M, s0 |= EG r. It follows that M, s0 | [](https://www.codecogs.com/eqnedit.php?latex=%5Cneq%20%5Cneg#0) EG r. As r [](https://www.codecogs.com/eqnedit.php?latex=%5Cnotin#0) L(s2) = {p,q}, this means that M, s2 |= [](https://www.codecogs.com/eqnedit.php?latex=%5Cneg#0) EG r, as the future includes the present.

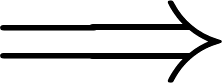
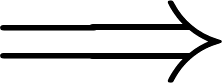
iii) Since t [](https://www.codecogs.com/eqnedit.php?latex=%5Cnotin#0) L(s0) and t [](https://www.codecogs.com/eqnedit.php?latex=%5Cnotin#0) L(s2), it follows that both M, s0 |

[](https://www.codecogs.com/eqnedit.php?latex=%5Cneq#0) E(t U q) and M, s2 |[](https://www.codecogs.com/eqnedit.php?latex=%5Cneq#0) E(t U q)

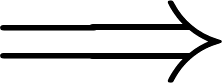
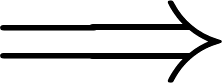
vi) As q [](https://www.codecogs.com/eqnedit.php?latex=%5Cin#0) L(s2) and we have infinite paths from s0 -> s2…, we have M, s0 |= Fq. It follows that as q [](https://www.codecogs.com/eqnedit.php?latex=%5Cin#0) L(s2), then s0 |= Fq, as the future includes the present.

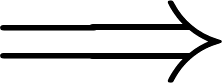
b)

LTL - G ( Fp [](https://www.codecogs.com/eqnedit.php?latex=%5Cland#0) (p [](https://www.codecogs.com/eqnedit.php?latex=%5Cimplies#0) Fs) [](https://www.codecogs.com/eqnedit.php?latex=%5Cland#0) (p [](https://www.codecogs.com/eqnedit.php?latex=%5Cimplies#0) Ft))

CTL - AG( AFp [](https://www.codecogs.com/eqnedit.php?latex=%5Cland#0) AG(p [](https://www.codecogs.com/eqnedit.php?latex=%5Cimplies#0) AFs) [](https://www.codecogs.com/eqnedit.php?latex=%5Cland#0) AG( p [](https://www.codecogs.com/eqnedit.php?latex=%5Cimplies#0) AFt))

c)

LTL - G ( Fq [](https://www.codecogs.com/eqnedit.php?latex=%5Cland#0) Fr [](https://www.codecogs.com/eqnedit.php?latex=%5Cland#0) (q [](https://www.codecogs.com/eqnedit.php?latex=%5Cimplies#0) ([](https://www.codecogs.com/eqnedit.php?latex=%5Cneg#0) p U r) [](https://www.codecogs.com/eqnedit.php?latex=%5Cland#0) (q [](https://www.codecogs.com/eqnedit.php?latex=%5Cimplies#0) (FT U r)))

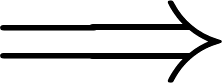
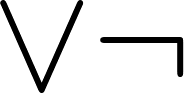
CTL - AG ( AFq [](https://www.codecogs.com/eqnedit.php?latex=%5Cland#0) AFr) [](https://www.codecogs.com/eqnedit.php?latex=%5Cland#0) AG( q [](https://www.codecogs.com/eqnedit.php?latex=%5Cimplies#0) A( [](https://www.codecogs.com/eqnedit.php?latex=%5Cneg#0) p U r))

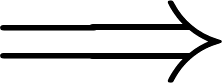
d)

LTL - s |= G (F Φ)

CTL - s |= AG(AF Φ)

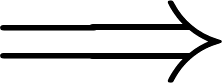
e)

LTL - G( p [](https://www.codecogs.com/eqnedit.php?latex=%5Cimplies#0) XG ( [](https://www.codecogs.com/eqnedit.php?latex=%5Cneg#0) q [](https://www.codecogs.com/eqnedit.php?latex=%5Clor%20%5Cneg#0) r U t))

CTL - AG ( p[](https://www.codecogs.com/eqnedit.php?latex=%5Cimplies#0) AX AG([](https://www.codecogs.com/eqnedit.php?latex=%5Cneg#0) q [](https://www.codecogs.com/eqnedit.php?latex=%5Clor#0) A [](https://www.codecogs.com/eqnedit.php?latex=%5Cneg#0) r U t))

f)

LTL - G( Fq [](https://www.codecogs.com/eqnedit.php?latex=%5Cland#0) Fr [](https://www.codecogs.com/eqnedit.php?latex=%5Cland#0) ( [](https://www.codecogs.com/eqnedit.php?latex=%5Cneg#0) q [](https://www.codecogs.com/eqnedit.php?latex=%5Clor#0) ( [](https://www.codecogs.com/eqnedit.php?latex=%5Cneg#0) p U r))

CTL - AG (AFq [](https://www.codecogs.com/eqnedit.php?latex=%5Cland#0) AFr) [](https://www.codecogs.com/eqnedit.php?latex=%5Cland#0) AG(q[](https://www.codecogs.com/eqnedit.php?latex=%5Cimplies#0) A ([](https://www.codecogs.com/eqnedit.php?latex=%5Cneg#0) p U r))

8)

lpri - local processing of reader i, i=1,2,

lpwi - local processing of writer i, i=1,2,

tri - reader i, i=1,2, requests reading,

twi - writer i, i=1,2, requests writing,

ri - reader i, i=1,2, is reading,

wi - writer i, i=1,2, is writing,

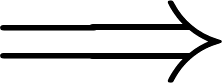
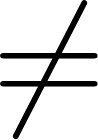
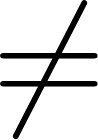
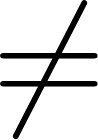
Note that the solution restricted to writers only should be the same as Mutual Exclusion considered in class! Hence to avoid similar problems we have to introduce additional boolean variables (or atomic predicates): turn=w1, turn=w2 and turn=r, to indicate that worlds where writer 1 will write (turn=w1), writer 2 will write (turn=w2), or readers (one or both) will read (turn=r).

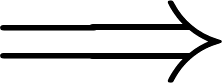
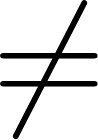
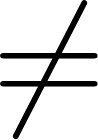
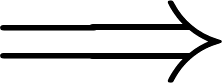
Now, states can be identified by atomic predicates of the form: (read1, read2, write1, write2, turn) where read1 [](https://www.codecogs.com/eqnedit.php?latex=%5Cin#0) {Ipr1, tr1, r1}, read2 [](https://www.codecogs.com/eqnedit.php?latex=%5Cin#0) {Ipr2, tr2, r2}, write1 [](https://www.codecogs.com/eqnedit.php?latex=%5Cin#0) {Ipw1, tw1, w1}, write2 [](https://www.codecogs.com/eqnedit.php?latex=%5Cin#0) {Ipw2, tw2, w2}, and turn [](https://www.codecogs.com/eqnedit.php?latex=%5Cin#0) {turn=w1, turn=w2, turn=r}.

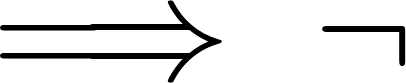
The life of a reader is a simple cycle: (Ipr1, \*, \*, \*, \*) -> (tr1, \*, \*, \*, \*) -> (r1, \*, \*, \*, \*) -> back to beginning

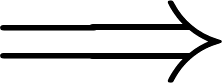
The writer is similar : (\*,\*,Ipw1,\*,\*) -> (\*,\*,tw1,\*,\*) -> (\*,\*,w1,\*,\*) -> back to beginning.

Not all combinations of atomic predicates are allowed, for example:

stw1 = w1 [](https://www.codecogs.com/eqnedit.php?latex=%5Cimplies#0) str1 [](https://www.codecogs.com/eqnedit.php?latex=%5Cneq#0) r1 [](https://www.codecogs.com/eqnedit.php?latex=%5Cland#0) str2 [](https://www.codecogs.com/eqnedit.php?latex=%5Cneq#0) r2 [](https://www.codecogs.com/eqnedit.php?latex=%5Cland#0) stw2 [](https://www.codecogs.com/eqnedit.php?latex=%5Cneq#0) w2,

str1 = r1 [](https://www.codecogs.com/eqnedit.php?latex=%5Cimplies#0) stw1 [](https://www.codecogs.com/eqnedit.php?latex=%5Cneq#0) w1 [](https://www.codecogs.com/eqnedit.php?latex=%5Cland#0) stw2 [](https://www.codecogs.com/eqnedit.php?latex=%5Cneq#0) w2 stw1

Safety LTL - G ( w1 [](https://www.codecogs.com/eqnedit.php?latex=%5Cimplies%20%5Cneg#0) (w2 [](https://www.codecogs.com/eqnedit.php?latex=%5Clor#0) r1 [](https://www.codecogs.com/eqnedit.php?latex=%5Clor#0) r2))

Liveliness LTL - G(tr1 [](https://www.codecogs.com/eqnedit.php?latex=%5Cimplies#0) Fr1)