1. Study Warford, Vega, and Staley, *A Calculational Deductive System for Linear Temporal Logic*.

   `https://dl.acm.org/doi/10.1145/3387109`

2. Fill in the blank entries in the table below. From the table, do you believe the *until* operator $\mathcal{U}$ is associative? Explain why or why not.

σ	s_0	s_1	s_2	s_3	s_4	s_5	s_6	s_7	…
p	F	F	T	T	T	T	T	F	…
q	F	T	F	T	T	F	F	F	…
r	F	F	F	T	F	F	T	F	…
$p \,\mathcal{U}\, q$	F								…
$q \,\mathcal{U}\, r$	F								…
$p \,\mathcal{U}\, (q \,\mathcal{U}\, r)$	F								…
$(p \,\mathcal{U}\, q) \,\mathcal{U}\, r$	F								…

For the following proofs, note that there is no case analysis proof technique in temporal logic.

3. Prove (6) Distributivity of $\bigcirc$ over $\equiv$: $\bigcirc (p \equiv q) \equiv \bigcirc p \equiv \bigcirc q$.

4. Prove (8) Falsehood of $\bigcirc$: $\bigcirc \mathit{false} \equiv \mathit{false}$.

5. Prove (20) Right zero of $\mathcal{U}$: $p \,\mathcal{U}\, \mathit{true} \equiv \mathit{true}$.

6. Prove (45) Expansion of $\Diamond$: $\Diamond p \equiv p \vee \bigcirc \Diamond p$.

7. Prove (72) Absorption of $\Box$: $\Box \Box p \equiv \Box p$.

8. Prove (77) Strengthening of $\Box$: $\Box p \Rightarrow \Diamond p$.

   You may hand in the written exercises on paper or electronically. If you hand them in electronically they must be in a PDF document named

   `a21written.pdf`

9. Implement Dekker's algorithm in C++.
   The source file to modify is `Dekker.cpp` in the `cosc450CppDistr` software distribution. Execute process p 15 times in a loop with control variable `i`. In the noncritical section, output

   `Process p, i=`*i*

   where *i* is the value of `i`. Execute process q 10 times in a loop with control variable `j`. In the noncritical section, output

   `Process q, j=`*j*

   where *j* is the value of `j`. To show that Dekker's algorithm really works, put a random 10ms delay between *every* statement in the preprotocol section and the postprotocol section as well as between the assignments to `temp` and `n` in an attempt to produce incorrect results. The final value should be 25. For this problem, hand in

`Dekker.cpp`

10. Implement Dekker's algorithm in Java.
The source file to modify is `Dekker.java` in the `cosc450JavaDistr` software distribution. Execute process p 15 times in a loop with control variable `i`. In the noncritical section, output

`Process p, i=`*i*

where *i* is the value of `i`. Execute process q 10 times in a loop with control variable `j`. In the noncritical section, output

`Process q, j=`*j*

where *j* is the value of `j`. To show that Dekker's algorithm really works, put a random 10ms delay between *every* statement in the preprotocol section and the postprotocol section as well as between the assignments to `temp` and `n` in an attempt to produce incorrect results. The final value should be 25. For this problem, hand in

`Dekker.java`