Übung 3

- Algorithmus von Lamport

Der von Lamport entwickelte Algorithmus erfordert es, dass für jede REQUEST-Nachricht eine ACK-Nachricht als Bestätigung gesendet wird. Ist dies wirklich nötig oder gibt es Situationen bei denen die Bestätigung einer REQUEST-Nachricht entfallen kann?

Übung 3

- Algorithmus von Lamport

Die Bestätigung kann entfallen wenn der Knoten bereits eine **jüngere REQUEST-Nachricht (größerer Zeitstempel) versendet** hat. Der anfragende Knoten bekommt oder hat in diesem Fall schon eine Bestätigung bekommen, dass keine Nachricht mit einem kleineren Zeitstempel von dem befragten Knoten unterwegs ist.

Die Bestätigung kann entfallen wenn der Knoten bereits eine ältere REQUEST-Nachricht versendet hat oder sich im kritischen Abschnitt befindet. Der anfragende Knoten erhält dann die benötigte Bestätigung durch die RELEASE-Nachricht.

Übung 3

- Algorithmus von Ricart und Agrawala

Es gibt Anwendungen bei denen zwischen Lese- und Schreibzugriffen unterschieden werden kann. Meist soll es dann möglich sein, dass eine **beliebige Anzahl an Teilnehmern Daten lesen**, darf aber zu einem bestimmten Zeitpunkt **nur ein Teilnehmer schreiben kann**. Erweitern sie den Algorithmus von Ricart und Agrawala für den gegenseitigen Ausschluss zwischen einem Schreiber und einer beliebigen Anzahl an Lesern.

Übung 3

- Algorithmus von Ricart und Agrawala

Es muss zwischen Lese- und Schreibanfragen unteschieden werden. Es gibt nun die Nachrichten R-REQUEST, W-REQUEST, REPLY.

Ein Knoten kann zu einem Zeitpunkt nur eine Art von Anfrage stellen.

Übung 3

- Beschreibung des Algorithmus
  - Eintrittswunsch des Knotens K_i:

    Entweder R-REQUEST(t, i) an alle oder
    W-REQUEST(t,i) an alle
  - Bearbeitung von W-REQUEST:
    - Wenn K_j eine W-REQUEST-Nachricht von K_i erhält, sendet er eine REPLY-Nachricht an K_i, falls er
      - weder selbst seinen kritischen Abschnitt betreten will noch ihn gerade betreten hat, oder
      - er selbst eine Anforderung gestellt hat, aber der Zeitstempel von K_i's Anforderung kleiner ist als der seiner eigenen Anforderung.
    - In allen anderen Fällen wird das Senden der REPLY-Nachricht aufgeschoben (Eintrag in Warteschlange!).

Übung 3

- Beschreibung des Algorithmus
  - Bearbeitung von R-REQUEST:
    - Wenn K_j eine R-REQUEST-Nachricht von K_i erhält, sendet er eine REPLY-Nachricht an K_i, falls er
      - weder selbst seinen kritischen Abschnitt als **Schreiber** betreten will noch ihn gerade als **Schreiber** betreten hat, oder
      - er selbst eine Schreib-Anforderung gestellt hat, aber der Zeitstempel von K_i's Anforderung kleiner ist als der seiner eigenen Anforderung.
    - In allen anderen Fällen wird das Senden der REPLY-Nachricht aufgeschoben.

Übung 3

- Beschreibung des Algorithmus
  - Betreten kritischer Abschnitte
    - K_i kann kritischen Lese- oder Schreib-Abschnitt betreten, wenn er auf seine Anforderung von allen anderen eine REPLY-Nachricht erhalten hat.
  - Verlassen des kritischen Abschnitts durch K_i
    - K_i versendet zu allen aufgeschobenen REQUESTs eine REPLY-Nachricht.

Übung 3

- Algorithmus von Maekawa

Erweitern Sie den Algorithmus von Maekawa, so dass bis zu k Knoten gleichzeitig in den kritischen Abschnitt eintreten können.

Übung 3

- Lsg:
  Man setzt k-Instanzen des Algorithmus auf, die parallel ablaufen und jeweils einen kritischen Abschnitt verwalten. Will ein Knoten in den kritischen Abschnitt, so sendet er einen Request an jede Instanz. Erhält er mehr als einen kritischen Abschnitt gibt er diese sofort wieder frei.