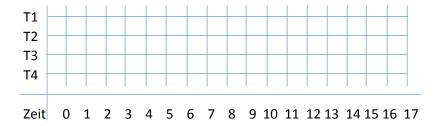
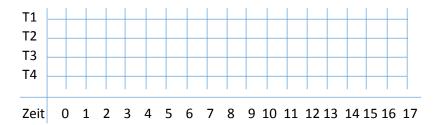
- 1. Scheduling
 - a) Was sind die Ziele, die ein Scheduler beachten muss. Geben Sie 6 Ziele aus der Vorlesung an. (3 Punkte)
 - b) Geben Sie ein Beispiel für zwei in Konflikt stehende Ziele und erläutern Sie, worin der Konflikt besteht. (2 Punkte)
 - c) Wann findet beim Scheduling eine Verdrängung statt? Geben Sie drei unterschiedliche Ereignisse an, aufgrund derer der Scheduler einen Prozess verdrängen kann. (3 Punkte)
 - d) Geben Sie die drei wesentlichen Zustände an, in denen sich die Prozesse im Rahmen des Scheduling befinden können. Wieviele Prozesse können sich auf einem Einprozessorsystem (Single-Core, kein Hyperthreading o.ä.) mit unbegrenztem Speicher in den jeweiligen Zuständen befinden? (3 Punkte)
 - e) Verschiedene Scheduling-Verfahren unterscheiden sich maßgeblich bzgl. des Zugriffs auf die Prozesse in **einem** der obigen Zustände. In welchem? Geben Sie eine kurze Begründung. (2 Punkte)
 - f) Nach welchen drei Kriterien lassen sich Scheduling-Verfahren klassifizieren? (1,5 Punkte)
 - g) Was ist der Vorteil von sehr einfachen Scheduling-Verfahren (z.B. ohne Verdrängung, ohne Prioritäten, etc.)? (1 Punkt)
 - h) Was versteht man unter "Prioritätsinvertierung"? Geben Sie ein Beispiel an. (3 Punkte)
- 2. In einem Betriebssystem sind folgende aperiodische Tasks zu planen (die Zeiten, die Scheduler und Dispatcher selbst benötigen, sollen vernachlässigbar sein):

Anforderung	Rechenzeit	Deadline
(Zeitpunkt)	(Zeiteinheiten)	(Zeitpunkt)
1	3	7
5	4	12
0	6	10
1	2	16
	(Zeitpunkt)	1 3 5 4

a) Zeichnen Sie die Scheduling-Abfolge nach dem "Shortest Job First"-Verfahren (SJF) **mit** Verdrängung in das Zeitdiagramm ein. (3 Punkte)



 b) Dieselben Tasks sollen in einem Echtzeit-Betriebssystem mit dem "Earliest Deadline First"-Verfahren (EDF) geplant werden. Zeichnen Sie auch hier im Zeitdiagramm die Scheduling-Abfolge. (4 Punkte)



- c) Werden alle Deadlines eingehalten? Falls ja, bitte anhand der Endzeiten begründen. Falls nein, bitte die Task(s) aufzählen, deren Deadline(s) nicht eingehalten wurde(n). (1 Punkt)
- d) Für welche Art von Prozessen bzw. Tasks kann das "Rate Monotonic Scheduling" (RMS) angewendet werden? (1 Punkt)
- e) Zwei periodische Tasks T₁ und T₂ haben die Rechenzeit t₁ = 6 Zeiteinheiten (ZE) und t₂ = 3 ZE sowie die Periodendauern P₁ = 12 ZE und P₂ = 10 ZE.
 Kann das "Rate Monotonic Scheduling" (RMS) für diese beiden Prozesse alle Deadlines einhalten? Leiten Sie Ihre Antwort rechnerisch her. (3 Punkte)

3. Synchronisation

a) Zwei nebenläufige Prozesse P_0 und P_1 verwenden die binären Semaphore S_0 und S_1 , die beide mit dem Wert 0 initialisiert sind. Jeder Prozess P_i (i = 0, 1) führt folgenden Code aus (verallgemeinerter Pseudocode):

```
P<sub>i</sub>:
    /* Beginn Code-Teil 1 */
    ...
    /* Ende Code-Teil 1 */
    signal(S<sub>i</sub>);
    wait(S<sub>1-i</sub>);
    /* Beginn Code-Teil 2 */
    ...
    /* Ende Code-Teil 2 */
```

Sie können davon ausgehen, dass die beiden Code-Teile 1 und 2 **keine** weiteren Semaphor-Operationen enthalten.

Kreuzen Sie die richtigen Aussagen an (4 Punkte):

- O P₀ und P₁ können sich gleichzeitig im Code-Teil 1 befinden.
- O P₀ und P₁ können sich **nicht** gleichzeitig im Code-Teil 2 befinden.

- O P₀ beginnt mit Code-Teil 2 erst, nachdem P₁ Code-Teil 1 abgeschlossen hat.
- O P₁ beginnt mit Code-Teil 2 erst, nachdem P₀ Code-Teil 1 abgeschlossen hat.
- O Zwischen P_0 und P_1 kann beim Warten auf S_0 bzw. S_1 ein **Deadlock** auftreten.
- O Die beiden Semaphore schützen **zusammen** einen kritischen Abschnitt.
- O Jedes der beiden Semaphore schützt einen kritischen Abschnitt.
- O Durch die Semaphore wird kein kritischer Abschnitt geschützt.
- b) Welche Methoden außer Semaphoren gibt es zur Sicherung kritischer Abschnitte? Nennen Sie drei weitere Methoden, die Sie aus der Vorlesung kennen. (1,5 Punkte)
- c) Zwei nebenläufige Threads T1 und T2 eines Prozesses greifen wie folgt auf eine gemeinsame Variable v zu:

```
void T1()
{
    v = 2;
    v = v * 4;
}

void T2()
{
    v = 5;
}
```

Sofern **keine** Synchronisation zwischen den Threads stattfindet, was sind die möglichen Werte für v nach Ende der Ausführung beider Threads? Geben Sie für jeden Wert die Reihenfolge(n) der Schritte an, durch die der Wert zustande kommt. (4 Punkte)

d) Wie nennt man die Art von Thread-Konflikt, der für den obigen Nicht-Determinismus verantwortlich ist? (1 Punkt)