

# Vorlesung Betriebssysteme 1

## Aufgabenblatt 3 zu Unit OS4

(Besprechung im Übungstermin gemäß Ankündigung auf der Vorlesungs-Website)

Die Aufgaben auf diesem und den folgenden Übungsblättern sollen Ihnen helfen, Ihr Selbststudium zu strukturieren und Ihnen ein Gefühl dafür geben, welchen Wissensstand Sie bisher in der Veranstaltung erlangt haben sollten. Um die gemeinsame Zeit mit den Tutoren optimal nutzen zu können, sollten Sie versuchen, bis zum jeweiligen Tutorengespräch möglichst viele der Aufgaben zu bearbeiten.

Die Bearbeitung sollte in einer Art und Weise erfolgen, dass a) Sie und Ihre Gruppe gut mit den Ergebnissen arbeiten können und b) sich im Gespräch mit den Tutoren über den Stoff der Aufgaben austauschen können. Es bestehen keine weiteren formalen Anforderungen an die Aufgabenbearbeitung unsererseits.

1. Die Aufgaben des Scheduling in Betriebssystemen verteilen sich auf drei Komponenten. Den *Short-Term Scheduler*, den *Long-Term Scheduler* und den *Dispatcher*. Was ist die Rolle dieser drei Komponenten?
2. Was ist *Echtzeit*? Unterscheiden Sie dabei *harte* und *weiche Echtzeit-Anforderungen*.
3. Was ist ein *Thread-Kontext*? Beschreiben Sie den Ablauf eines *Kontextwechsels*.
4. Unterscheiden Sie *kooperatives* und *preemptives* Scheduling.
5. Was ist ein *Quantum*? Welchen Einfluss hat die Länge des Quantums auf das Verhalten des Betriebssystems?
6. Während ihrer Laufzeit im System nehmen Threads mehrere *Zustände* im Scheduling ein. Zeichnen Sie ein Diagramm und visualisieren Sie diese Zustände und die Übergänge zwischen ihnen. Wodurch werden die Zustandsübergänge jeweils verursacht?

7. Betrachten Sie die folgende Menge an *Tasks* (*Task-Set*):

Thread ID	Startzeit $t_s$ (in ms)	Ausführungszeit $e$ (in ms)	Priorität
1	0	8	9
2	2	6	11
3	2	6	10
4	4	10	11

- (a) Zeichnen Sie für das gegebene Task-Set ein Gantt-Diagramm für einen *FiFo-Scheduler*.
- (b) Zeichnen Sie für das gegebene Task-Set ein Gantt-Diagramm für einen *prioritätenlosen Round-Robin-Scheduler* mit einer Quantenlänge von 3 Zeiteinheiten.
- (c) Zeichnen Sie für das gegebene Task-Set ein Gantt-Diagramm für einen *prioritätenbasierten Round-Robin-Scheduler* mit einer Quantenlänge von 3 Zeiteinheiten.

Nehmen Sie dabei jeweils an, dass für einen Kontextwechsel keine Zeit verloren geht.

8. Berechnen Sie für die oben gezeichneten Gantt-Diagramme jeweils für jeden Task die *Wartezeit* (waiting-time) und die *Gesamtabarbeitungszeit* (turnaround-time), sowie den *Durchsatz* des Schedulers. Vergleichen Sie die Ergebnisse.

9. In Prioritätenbasierten Scheduling können Threads niedriger Priorität *verhungern*. Beschreiben Sie, wie moderne Betriebssysteme mit diesem Problem umgehen und gehen Sie auf ein Beispiel genauer ein. Betrachten Sie dazu auch den Programmrahmen zur Starvation Avoidance:

<https://github.com/osmmpi/osmthread>

Der gegebene Programmrahmen enthält eine Implementierung für prioritätenbasiertes Scheduling von User-Mode Threads, sowie ein Beispielprogramm welches die Implementierung verwendet, und ein Szenario demonstriert, in dem Starvation auftritt. Erweitern Sie den Programmrahmen um einen aus den Materialien bekannten Ansatz zur Starvation Avoidance.

10. Was ist ein *Multilevel-Queue Scheduler*? Beschreiben Sie die verschiedenen *Levels* des Multilevel-Queue Schedules *Completely Fair Scheduler* (CFS) in Linux und wozu sie verwendet werden.

