| **Prüfung Betriebssysteme** | **Prof. Dr. Tobias Lauer** |
|---|---|
| **Wintersemester 2019/20** | **05.02.2020** |

Name: _____ Matrikelnummer: _____

Studiengang/Semester: _____ Unterschrift: _____

| **Aufgabe** | **1** | **2** | **3** | **4** | **Summe** |
|---|---|---|---|---|---|
| **Mögliche Punkte** | **18** | **14** | **12** | **15** | **59** |
| **Erreichte Punkte** | | | | | |

1. Die Klausur darf nur in prüfungsfähigem Zustand geschrieben werden.

2. Schreiben Sie auf diesen Aufgabenbogen Ihren **Namen, Matrikelnummer, Studiengang, Semester** und **unterschreiben Sie**. Geben Sie den gesamten Aufgabensatz ab.

3. Prüfen Sie zu Beginn der Prüfung den Aufgabensatz auf **Vollständigkeit** (9 Seiten).

4. Die Aufgabenblätter bieten Platz für die Lösungen der Aufgaben. Sollten Sie zusätzlichen Platz benötigen, können Sie die Rückseiten der Blätter verwenden oder Sie bekommen zusätzliches Prüfungspapier. Tragen Sie dort auch Ihren Namen und die Matrikelnummer ein, und beschriften Sie die Lösungen mit der Aufgabennummer.

5. Bei Rechenaufgaben muss der **Lösungsweg** ersichtlich sein. Geben Sie die dazugehörigen **Einheiten** an.
   Bei Angaben von Speichergrößen gelten wie in der Vorlesung immer die **binären Vielfachen**: 1 KB (KiB) = $2^{10}$ Byte; 1 MB (MiB) = $2^{10}$ KB (KiB) = $2^{20}$ Byte; usw.

6. Schreiben Sie **deutlich lesbar** mit **Kugelschreiber** oder **Füller**. Antworten, die in **rot** oder mit **Bleistift** geschrieben sind oder unleserlich sind, **werden nicht gewertet**.

7. Als Hilfsmittel sind neben Schreibwerkzeug (Stifte, Lineal) **ein handbeschriebenes DIN A4 Blatt** (Vorder- und Rückseite) sowie ein **nicht programmierbarer**, **nicht funk- oder netzwerkfähiger Taschenrechner** zugelassen. Weitere Hilfsmittel sind nicht erlaubt.

8. Jeder Täuschungsversuch führt zum Nichtbestehen der Prüfung.

9. Die Klausur dauert 60 Minuten.

**Viel Erfolg!**

**Aufgabe 1** (18 Punkte)

(a) Welche Art von Synchronisationskonflikt entsteht, wenn man einen **kritischen Abschnitt nicht wirksam** sichert? Benennen und erläutern Sie den Konflikt. (2 Punkte)

(b) Zur softwareseitigen Sicherung eines kritischen Abschnitt soll eine globale Variable `lock` verwendet werden, die zu Beginn auf 0 gesetzt ist und die durch den Wert 1 anzeigen soll, dass ein Thread bzw. Prozess gerade auf im kritischen Abschnitt ist. Dazu ruft jeder Thread direkt **vor** seinem Eintritt in den kritischen Abschnitt die Methode `lockSection()` auf und direkt **nach** dem Austritt die Methode `unlockSection()`, die folgendermaßen definiert sind:

```
void lockSection() {
    while (lock == 1) {
        // tue nichts, sondern warte, bis lock nicht mehr auf 1
    }
    lock = 1;     // Setze nun selbst lock auf 1
}


void unlockSection() {
    lock = 0;     // Setze lock auf 0
}
```

Ist dieses Vorgehen geeignet, um den kritischen Abschnitt wirksam zu schützen? Begründen Sie Ihre Antwort genau. (4 Punkte)

(c) Welche Möglichkeiten haben Sie in der Vorlesung kennengelernt, um kritische Abschnitte **wirksam** zu sichern? Nennen Sie zwei solcher Möglichkeiten. (1 Punkte)

(d) Zwei Prozesse P1 und P2 verwenden gemeinsame Betriebsmittel R1, R2 und R3, die jeweils durch die Semaphore S1, S2, S3 gesichert werden (d.h. dass zwischen dem `wait`- und dem `signal`-Befehl für dasselbe Semaphor das jeweilige Betriebsmittel vom ausführenden Prozess gehalten wird).
Gegeben ist der folgende Programmcode (Pseudocode) für P1 bzw. P2:

```
P1:
wait(S2)
…
wait(S1)
…
signal(S1)
wait(S3)
…
signal(S3)
signal(S2)
```

```
P2:
wait(S3)
wait(S1)
…
signal(S3)
…
wait(S2)
…
signal(S2)
signal(S1)
```

P1 und P2 werden nun nebenläufig ausgeführt. P1 sei gerade in der Zeile `wait(S3)`, während sich P2 in der Zeile `wait(S1)` befinde. Stellt diese Situation ein **Deadlock** dar? Begründen Sie Ihre Antwort mit Hilfe des in der Vorlesung eingeführten **Ressourcengraphen** (Kreise für Prozesse, Rechtecke für Betriebsmittel). (3 Punkte)

(e) Kann bei der nebenläufigen Ausführung von P1 und P2 in einer anderen Situation als der in Teil (b) ein Deadlock auftreten? Begründen Sie Ihre Antwort in Worten oder leiten Sie sie mit einem Diagramm her.
Falls ein Deadlock auftreten kann, markieren Sie im obigen Pseudocode für P1 und P2 jeweils mit einem Pfeil den Befehl, bei dem sich P1 bzw. P2 beim Deadlock befindet. (3 Punkte)

(f) Beschreiben Sie (am einfachsten mit Pseudocode), wie man mit Hilfe von 2 Semaphoren und einer Hilfsvariable eine **Barriere für *n = 5* nebenläufig ausgeführte Threads** erzeugen kann, d.h. **jeder** Thread muss jeweils alle seine Befehle vor der Barriere ausgeführt haben, bevor er weitermachen (d.h. die Befehle nach der Barriere ausführen) darf.
Geben Sie an, wie die beteiligten Semaphore und die Hilfsvariable initialisiert werden müssen und mit welcher Abfolge von Befehlen die Barriere im Thread-Code implementiert wird.
(5 Punkte)

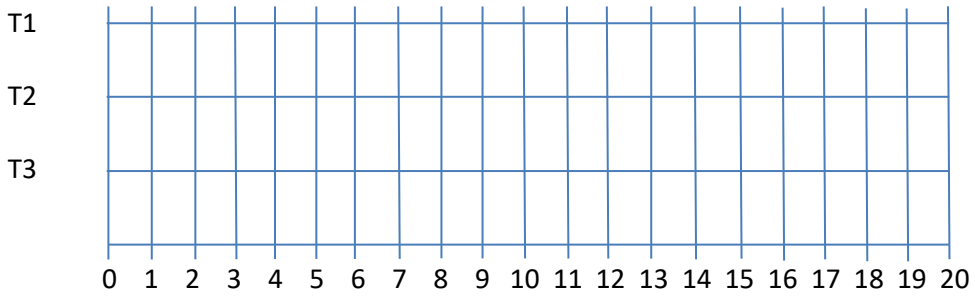
**Aufgabe 2** (14 Punkte)

Die folgenden periodischen Tasks T1, T2 und T3 sollen in einem Echtzeit-Betriebssystem (auf einem Rechner mit nur einem Prozessor, kein Hyperthreading o.ä.) mit dem Verfahren **„Rate Monotonic Scheduling“ (RMS)** geplant werden:

| | **Erster Anforderungs-zeitpunkt A** | **Ausführungs-dauer** | **Perioden-dauer** |
|---|---|---|---|
| **Task T1**: | 0 | 3 | 8 |
| **Task T2**: | 1 | 2 | 5 |
| **Task T3**: | 4 | 1 | 6 |

(a) Ermitteln Sie, ob ein erfolgreiches Scheduling der obigen Tasks mit dem RMS-Verfahren grundsätzlich möglich ist, d.h. nennen Sie die entsprechende **notwendige Bedingung** und prüfen Sie, ob diese erfüllt ist. (2 Punkte)

(b) Zeichnen Sie in das folgende Zeitdiagramm die Scheduling-Abfolge für das RMS-Verfahren während der ersten 20 Zeiteinheiten ein. Markieren Sie die Zeitpunkte, an denen eine Verdrängung stattfindet, mit einem „V“ und Zeitpunkte, an denen eine Deadline überschritten wird, mit einem „X“. (4 Punkte)
(**Hinweis:** Die Zeit, die Scheduler und Dispatcher selbst benötigen, werden vernachlässigt.)




(c) Kann RMS für die obigen Tasks die Einhaltung aller Deadlines **garantieren**? Begründen Sie Ihre Antwort rechnerisch. (2 Punkte)

(d) Was versteht man beim Scheduling unter dem Begriff „Prioritätsinvertierung“? Geben Sie Ihre Erläuterung anhand eines Beispiels. (4 Punkte)

(e) Wie könnte man das Problem der Prioritätsinvertierung lösen? (2 Punkte)

**Aufgabe 3** (12 Punkte)

Gegeben sei ein Rechnersystem mit 32-Bit-Adressierung, 1 GiB physikalischem Hauptspeicher und einer Rahmen- bzw. Kachelgröße von 2 KiB.

a) Wie hoch ist die Anzahl der Rahmen im Hauptspeicher (Zweierpotenz genügt)? (1 Punkt)

b) Ein Programm greift auf die virtuelle Speicheradresse **`0x8B0D09F8`** zu. Geben Sie die **kleinste** sowie die **größte virtuelle Adresse** an, die **auf derselben Seite** liegt wie die angegebene Adresse. Geben Sie die gesuchten Adressen als vollständige 32-Bit-Hexadezimalzahlen an. (2 Punkte)
**Hinweis**: Teilen Sie die Adresse auf in Seitennummer und Offset. Am einfachsten geht das, wenn Sie die Adresse ins Binärsystem übertragen.

c) Ein Programm legt ein Array der Länge 9172 Byte an. Wieviele physikalische Rahmen werden dazu **mindestens**, wieviele **höchstens** benötigt? Belegen Sie Ihre Antworten rechnerisch. (2 Punkte)

d) Wenn das Array aus (c) an der virtuellen Adresse **`0x31BA2FFE`** beginnt, wieviele physikalische Rahmen werden dann benötigt? Begründen Sie Ihre Antwort. (2 Punkte)

e) In einer Hauptspeicherverwaltung mit 3 Rahmen R1, R2, R3 werden nacheinander die Seiten

5, 1, 2, 1, 3, 4, 5, 1, 5, 4, 2, 1, 5

angefordert. Tragen Sie in der Tabelle Abfolge von Seitenersetzungen ein, die bei der **FIFO-Strategie** (**First In First Out**) entsteht. Markieren Sie Seitenfehler mit „F“. Die ersten drei Seiten 5, 1, 2 sind bereits eingetragen. (3 Punkte)

| | | 1 | 3 | 4 | 5 | 1 | 5 | 4 | 2 | 1 | 5 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| R1 | 5 | | | | | | | | | | |
| R2 | 1 | | | | | | | | | | |
| R3 | 2 | | | | | | | | | | |
| Seitenfehler | F | | | | | | | | | | |

f) Um zu beurteilen, wie gut die in (e) genannte Strategie ist, soll sie mit einer **optimalen** Abfolge von Seitenersetzungen vergleichen werden. Tragen Sie diese in die unten stehende Tabelle ein, und markieren Sie auch hier Seitenfehler mit „**F**“. (2 Punkte)

| | | 1 | 3 | 4 | 5 | 1 | 5 | 4 | 2 | 1 | 5 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| R1 | 5 | | | | | | | | | | |
| R2 | 1 | | | | | | | | | | |
| R3 | 2 | | | | | | | | | | |
| Seitenfehler | F | | | | | | | | | | |

**Aufgabe 4** (15 Punkte)

Die Festplatte eines Rechnersystems erlaubt den Zugriff auf einzelne Blöcke der Größe **512 Byte**. Ein Dateisystem A nutzt zur Verwaltung der Festplatte **verkettete Listen** (ohne Tabelle). In jedem Block wird dafür eine 16-Bit-Zahl (also 2 Byte) für den Verweis auf den nächsten Block benötigt. Ein anderes Dateisystem B verwendet **verkettete Listen mit Tabelle** (File Allocation Table).

(a) Wieviel Platz auf der Festplatte wird für eine Datei der Größe **156 Byte**, wieviel für eine Datei der Größe **3065 Byte** in Dateisystem A verbraucht? Wieviel sind es in Dateisystem B? Bitte begründen Sie Ihre Antworten rechnerisch. (2 Punkte)

(b) Auf manche Arten von Dateien ist ein **wahlfreier Zugriff (Random Access)** nötig. Erläutern Sie, was damit gemeint ist. Welches der Dateisysteme A und B eignet sich dafür besser und warum? (2 Punkte)

(c) Beim Auslagern von Hauptspeicherseiten auf die Festplatte verwendet ein Betriebssystem eine Seitengröße von 2 KiB = 2048 Byte (wie in Aufgabe 4). Die ausgelagerten Seiten werden dabei in eine Datei auf der Festplatte geschrieben.
Welches der obigen Dateisysteme A und B eignet sich für solch eine Auslagerungsdatei besser? Geben Sie auch hier eine Begründung. (2 Punkte)

(d) Was ist in Dateisystem B die Obergrenze für die Größe einer einzelne Partition (d.h. einer „logischen Festplatte“), wenn die Einträge der Tabelle aus **16-Bit-Zahlen bestehen**, also auf $2^{16}$ verschiedene Blöcke verwiesen werden kann?
Wieviel Platz im Hauptspeicher wird für die Tabelle selbst verbraucht? (3 Punkte)

(e) Ein anderes Dateisystem nutzt die **zusammenhängende Belegung** von Blöcken eines Datenträgers. Nennen Sie **einen Vorteil** und **einen Nachteil** diese Art der Speicherung von Dateien. Bei welcher **Verteilung von Schreib- und Lesezugriffen** lässt sich diese Art der Belegung sehr gut einsetzen? (3 Punkte)

(f) Ein weiteres Dateisystem verwendet **Indexknoten** (I-Nodes). Ein I-Node soll dabei insgesamt **zwölf** Listeneinträge enthalten: **acht** direkte Blockadressen, **zwei** Adressen von Blöcken für eine **einfach indirekte** Adressierung (die selbst wiederum 256 Blockadressen speichern), und **zwei** Adressen für eine **doppelt indirekte** Adressierung.
Wie groß kann in diesem Dateisystem eine beliebige Datei höchstens sein? Geben Sie die Größe als Dezimalzahl oder in Form von Zweierpotenzen an. (3 Punkte)