

(Prüfungs-)Aufgaben zum Thema Scheduling

Mit einem **D** gekennzeichnete Aufgaben sind nur für die Prüfung im Diplomstudiengang relevant.

- 1)** Geben Sie die beiden wichtigsten Kriterien bei der Wahl der Größe des Quantums beim Round-Robin-Scheduling an.

- 2)** In welchen Situationen und von welchen (Betriebssystem-)Routinen wird an einem System mit präemptivem Multitasking und Prioritätensteuerung der Scheduler aufgerufen?

- 3)** Ein System mit präemptivem Multitasking werde hauptsächlich zur Programmentwicklung benutzt. Die Größe des Quantums an diesem System sei einstellbar. Ist die Anpassung des Quantums eine wichtige Aufgabe für den System Manager? Begründen Sie Ihre Antwort.

- 5)** Wie kann ein Betriebssystem dafür sorgen, daß kein Thread aufgrund niedriger Priorität „verhungert“? (Sie können z.B. beschreiben, wie Windows NT das macht.)

- 6)** Erläutern Sie den Begriff des Quantums an einem System mit präemptivem Multitasking. Welche Kriterien spielen bei der Festlegung der Größe des Quantums eine Rolle?

- 7)** Ein Betriebssystem arbeite mit prioritätengesteuertem Scheduling sowie Round-Robin-Scheduling für Threads gleicher Priorität. Beschreiben Sie, in welchen grundlegend verschiedenen Situationen der Scheduler aufgerufen wird. Geben Sie für jede dieser Situationen ein Beispiel, und geben Sie dabei jeweils auch an, von welcher Betriebssystemkomponente der Scheduler aufgerufen wird.

- 8)** Erläutern Sie die dynamische Anpassung von Prioritäten durch ein Betriebssystem.

- 9)** Was versteht man unter "soft affinity"? Wie kann soft affinity beim Scheduling auf einem Mehrprozessorsystem berücksichtigt werden und warum kann dies sinnvoll sein?

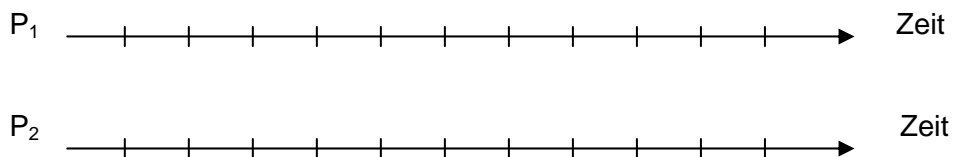
- 12)** Wie wird auf den meisten Betriebssystemen der Scheduler aufgerufen? Warum ist dies so implementiert?

13) Auf einem System mit zwei Prozessoren P_1 und P_2 stehen 4 Threads zur Bearbeitung an, die in der Reihenfolge A - B - C - D in der Warteschlange bereiter Threads stehen. Die Threads haben folgende Eigenschaften:

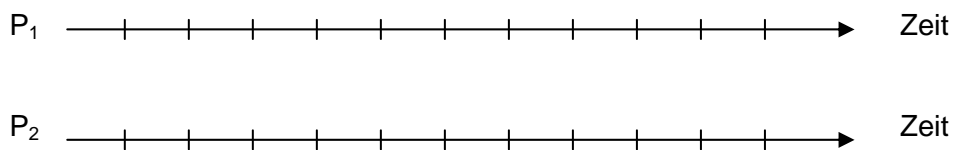
- A: Laufzeit 6 Zeiteinheiten, blockiert alle 3 Zeiteinheiten
- B: Laufzeit 6 Zeiteinheiten, blockiert alle 2 Zeiteinheiten
- C: Laufzeit 6 Zeiteinheiten, blockiert alle 3 Zeiteinheiten
- D: Laufzeit 2 Zeiteinheiten, blockiert nach jeder Zeiteinheit

Das Blockieren dauert jeweils weniger als eine Zeiteinheit. Wenn er wieder bereit wird, stellt sich ein Thread hinten wieder an. Es findet keine Verdrängung laufender Threads statt.

a) Wie werden die Threads ausgeführt?



b) Wie werden die Threads ausgeführt, wenn der Scheduler von allen bereiten Threads immer den ersten in der Warteschlange bereiter Threads auswählt, der "soft affinity" zu dem freien Prozessor hat (bzw. den vordersten Thread, wenn keiner der Threads "soft affinity" hat)?



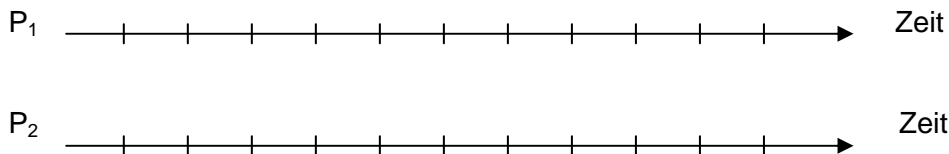
c) Zusatzaufgabe (+4 Punkte)

Welche Auswirkungen hat also hier die Berücksichtigung der "soft affinity" auf die Verweilzeiten der Threads? Durch welche (unrealistische) Vorgabe in der Aufgabenstellung wird hier der eigentliche Vorteil der Berücksichtigung der soft affinity verdeckt?

14) Auf einem System mit zwei Prozessoren P_1 und P_2 stehen 4 Threads zur Bearbeitung an, die in der Reihenfolge A - B - C - D in der Warteschlange bereiter Threads stehen. Jeder Thread nutzt die CPU für jeweils eine "Laufeinheit" (es findet keine Verdrängung laufender Threads statt), blockiert sich dann für eine halbe Zeiteinheit und stellt sich danach hinten wieder in der Warteschlange an. Die Eigenschaften der Threads sind:

- A: benötigt 2 "Laufeinheiten" à 3 Zeiteinheiten
- B: benötigt 3 "Laufeinheiten" à 2 Zeiteinheiten
- C: benötigt 2 "Laufeinheiten" à 3 Zeiteinheiten
- D: benötigt 2 "Laufeinheiten" à 1 Zeiteinheit

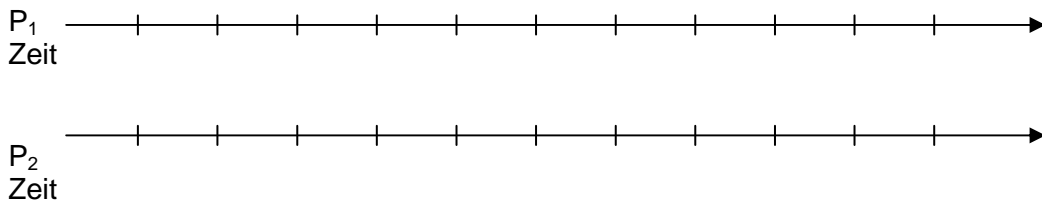
a) Wie werden die Threads ausgeführt?



Berechnen Sie:

	A	B	C	D	Durchschnitt
Laufzeit					
Verweilzeit					

b) Wie werden die Threads ausgeführt, wenn der Scheduler von allen bereiten Threads immer den ersten in der Warteschlange bereiter Threads auswählt, der "soft affinity" zu dem freien Prozessor hat (bzw. den vordersten Thread, wenn keiner der Threads "soft affinity" hat)? *Die Länge einer Laufeinheit verkürze sich dabei um 25%, wenn der Thread auf derjenigen CPU läuft, auf der er auch zuletzt lief.*



Berechnen Sie:

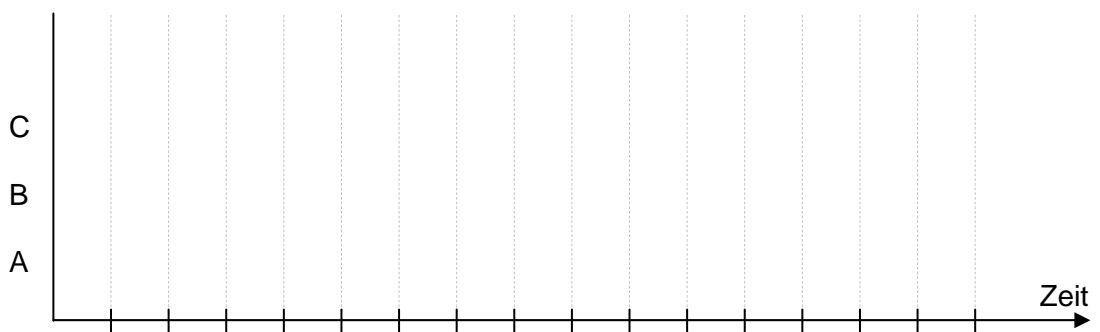
	A	B	C	D	Durchschnitt
Laufzeit					
Verweilzeit					

15) Auf einem System stehen 3 Threads A, B und C zur Bearbeitung an, die folgende Eigenschaften haben:

- A: Priorität 5, Laufzeit 6 Zeiteinheiten, blockiert alle 3 Zeiteinheiten für zwei Zeiteinheiten
- B: Priorität 4, Laufzeit 4 Zeiteinheiten, blockiert alle 2 Zeiteinheiten für eine Zeiteinheit
- C: Priorität 5, Laufzeit 2 Zeiteinheiten, blockiert nach jeder Zeiteinheit für zwei Zeiteinheiten

- Die beiden Threads mit Priorität 5 stehen ursprünglich in der Reihenfolge A - C in der Warteschlange.
- Ein höherer Wert bedeutet auch eine höhere Priorität.
- Es findet *kein* Round-Robin-Scheduling (also keine Verdrängung nach Ablauf einer bestimmten Zeit) statt.

Wie werden die Threads ausgeführt?



16) Ein System arbeitet mit prioritätenbasiertem Scheduling und Round-Robin-Scheduling für Threads gleicher Priorität. Hohe numerische Werte stehen für hohe Prioritäten. Die Länge des Quantums hängt von der Priorität ab (wie z.B. bei System V R4). Es findet *keine* dynamische Anpassung der Prioritäten statt.

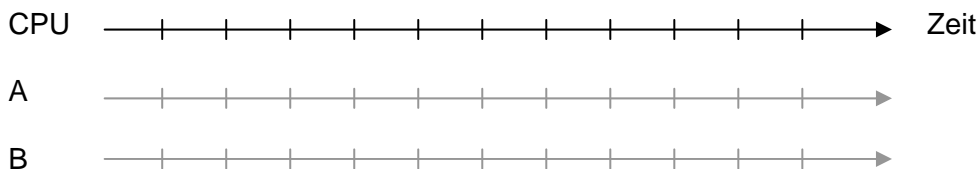
Jeder Thread nutzt die CPU für die angegebene "Nutzungszeit", blockiert sich dann für die angegebene "Blockierzeit" und stellt sich danach hinten wieder in der Warteschlange an.

Es stehen 3 Threads zur Bearbeitung an mit folgenden Eigenschaften (ZE=Zeiteinheit):

- A: Priorität 4, Quantum: 1 ZE, Nutzungszeit: 1 ZE, Blockierzeit: 2 ZE
- B: Priorität 4, Quantum: 1 ZE, Nutzungszeit: 2 ZE, Blockierzeit: 2 ZE
- C: Priorität 2, Quantum: 2 ZE, Nutzungszeit: unendlich, Blockierzeit: 0 (rein CPU-lastig)

Zu Beginn sind alle Threads bereit und Thread A steht vorne in der Prio4-Warteschlange (vor B).

Wie werden die Threads in den ersten 11 ZE ausgeführt? Markieren Sie (als Hilfe für Sie, ohne Bewertung) auf den Achsen für A und B, wann der jeweilige Thread blockiert ist.



17D)

- a) Was versteht man unter dem Problem der Prioritätsumkehr (priority inversion)?
- b) Beschreiben Sie *entweder* für Windows NT/2000/XP *oder* für ein Unix-Derivat (in der Vorlesung wurde hier Solaris behandelt), welche Mechanismen das Betriebssystem für die Lösung des Problems der Prioritätsumkehr enthält.

Es folgt die Beschreibung für das Betriebssystem _____ .

18)

- a) Was versteht man unter "soft affinity"? Warum sollte der Scheduler soft affinity berücksichtigen?
- b) Wie kann der Scheduler soft affinity berücksichtigen, wenn er einen bereiten Thread für eine frei gewordene CPU aussucht? Worauf muß dabei evtl. noch geachtet werden?
- c) Wie sollte der Scheduler soft affinity berücksichtigen, wenn ein Thread bereit wird und mehrere freie CPUs zur Verfügung stehen?
- d) (Zusatzaufgabe mit Zusatzpunkten, bitte Rückseite verwenden)
Diskutieren Sie, ob und evtl. wie der Scheduler soft affinity berücksichtigen kann, wenn ein Thread bereit wird, und alle CPUs belegt sind.

19) Erläutern Sie Idee und Prinzipien der dynamischen Anpassung von Prioritäten durch ein Betriebssystem (keine Details der Implementierung).

20)

- a) Was ist das prinzipielle Kriterium, nach dem ein Betriebssystem die Prioritäten von Threads dynamisch verändert? (Ohne Spezialfälle wie z.B. die Lösung des Problems der Prioritätsumkehr.)
- b) Beschreiben Sie, wie dieses Prinzip in einem Betriebssystem umgesetzt werden kann. (Sie können z.B. die Prioritätenanpassung in Windows oder Unix beschreiben.)

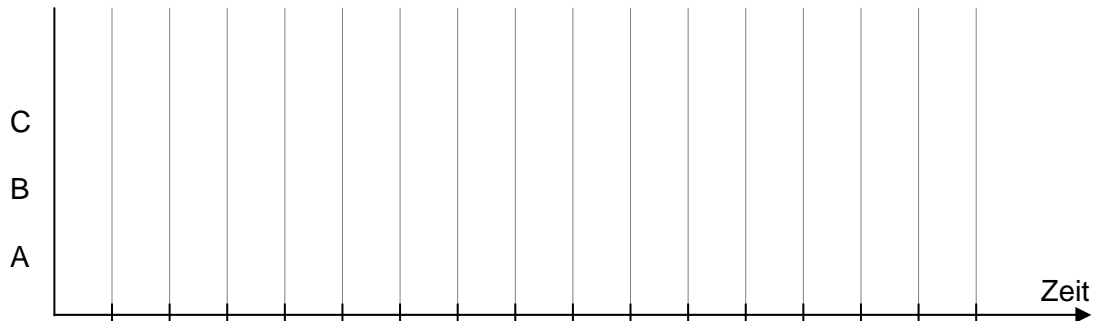
22)

Auf einem System stehen 3 Threads A, B und C zur Bearbeitung an, die folgende Eigenschaften haben:

- A: Priorität 6, Laufzeit 6 Zeiteinheiten, blockiert alle 2 Zeiteinheiten für zwei Zeiteinheiten
B: Priorität 3, Laufzeit 4 Zeiteinheiten, blockiert alle 2 Zeiteinheiten für zwei Zeiteinheiten
C: Priorität 6, Laufzeit 2 Zeiteinheiten, blockiert nach jeder Zeiteinheit für sechs Zeiteinheiten

- Ein höherer Wert bedeutet auch eine höhere Priorität.
- Die beiden Threads mit Priorität 6 stehen ursprünglich in der Reihenfolge A - C in der Warteschlange.
- Es findet *kein* Round-Robin-Scheduling (also keine Verdrängung nach Ablauf einer bestimmten Zeit) statt.

Wie werden die Threads ausgeführt? Markieren Sie, welcher Thread zu welcher Zeit am Laufen ist. (Tipp: markieren Sie sich auch, wann ein Thread blockiert ist.)



23)

- a) Sowohl in Windows wie auch in Solaris gibt es zwei „Arten“ oder „Klassen“ von Prioritäten, von denen die eine als „Echtzeitprioritäten“ bezeichnet wird. Erläutern Sie für *eines* dieser Betriebssysteme die prinzipiellen Unterschiede dieser beiden Arten von Prioritäten.

Es folgt die Beschreibung für das Betriebssystem _____ .

- b) Beschreiben Sie *entweder* für Windows *oder* für Solaris die dynamische Anpassung der Prioritäten durch das Betriebssystem. (Es kommt nicht auf die letzten Details an, die insbesondere bei Solaris sehr umfangreich sind.)

Es folgt die Beschreibung für das Betriebssystem _____ .

24)

Ein System arbeitet mit prioritätenbasiertem Scheduling, es findet aber kein Round-Robin-Scheduling für bereite Threads gleicher Priorität statt. Hohe numerische Werte stehen für hohe Prioritäten. Es findet eine dynamische Anpassung der Prioritäten wie folgt statt:

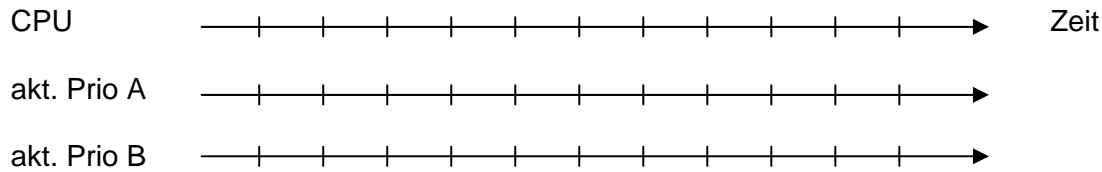
- Wenn ein blockierter Thread bereit wird, wird seine aktuelle Priorität um 2 erhöht, jedoch maximal auf den Wert 10.
- Wenn ein Thread eine Zeiteinheit CPU-Zeit erhalten hat, wird seine aktuelle Priorität um 1 erniedrigt, jedoch nicht unter seine Basispriorität. Hat er nach der Erniedrigung die gleiche Priorität wie andere bereite Threads, so darf er weiterlaufen (zwecks Einsparen eines Kontextwechsels).
- Werden zwei Threads gleicher Priorität gleichzeitig bereit, so ist es Zufall, welcher von beiden die CPU erhält. (In einem realen System werden zwei Threads natürlich nie wirklich gleichzeitig bereit).

Jeder Thread nutzt die CPU für die angegebene "Nutzungszeit" und blockiert sich dann für die angegebene "Blockierzeit".

Es gibt 2 Threads (und den Idle Thread) mit folgenden Eigenschaften (ZE=Zeiteinheit):

- A: Basispriorität 3, Nutzungszeit: 1 ZE, Blockierzeit: 1 ZE
 B: Basispriorität 4, Nutzungszeit: 2 ZE, Blockierzeit: 2 ZE
 Idle: Basispriorität 0, Nutzungszeit: unendlich, Blockierzeit: 0

Wie werden die Threads in den ersten 11 ZE ausgeführt? Markieren Sie auf der Achse CPU den laufenden Thread, auf den anderen beiden Achsen die aktuelle Priorität der Threads, und ob ein Thread gerade blockiert ist (z.B. durch 7b = Priorität 7, blockiert).



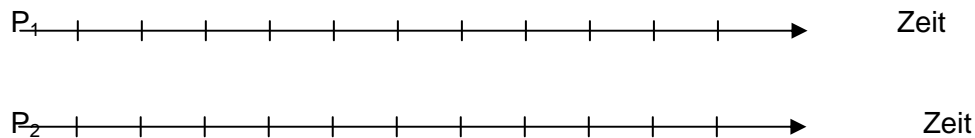
Beschreiben Sie verbal die Auswirkung der dynamischen Anpassung der Prioritäten auf die beiden Threads A und B.

25) Auf einem System mit zwei Prozessoren P_1 und P_2 stehen 3 Threads A, B und C zur Bearbeitung an, die folgende Eigenschaften haben:

- A: Priorität 4, blockiert alle 2 Zeiteinheiten für 2 Zeiteinheiten
- B: Priorität 2, blockiert nie
- C: Priorität 4, blockiert alle 3 Zeiteinheiten für 2 Zeiteinheiten

Priorität 4 ist höher als Priorität 2.

a) Wie werden die Threads in den ersten 10 Zeiteinheiten ausgeführt?



Treffen Sie nach 4 und nach 8 Zeiteinheiten die Scheduling-Entscheidung so, wie dies ein realer Scheduler auch tun würde, und geben Sie eine kurze Begründung dafür an.

b) Konzeptionell könnte man in der Situation, die in obigem Beispiel nach 4 oder 8 Zeiteinheiten vorliegt, auch eine andere Entscheidung treffen (dies wäre allerdings schwierig zu implementieren). Welchen evtl. Vorteil und welchen Nachteil hätte diese andere Entscheidung?

26)

a) Was versteht man unter "soft affinity"? Warum sollte der Scheduler soft affinity berücksichtigen?

b) Wie sollte der Scheduler auf einem Mehrprozessorsystem vorgehen, wenn ein Thread bereit wird, und mehrere freie CPUs zur Verfügung stehen?

c) Wie könnte (sollte) der Scheduler auf einem Mehrprozessorsystem vorgehen, wenn ein Thread bereit wird, alle CPUs belegt sind, und er entscheiden muss, ob einer der laufenden Threads verdrängt werden soll, und falls ja, welcher?
(Es gibt mehr als eine „richtige“ Vorgehensweise. Begründen Sie deshalb Ihre Antwort.)

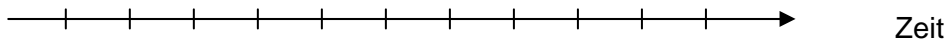
27D)

Auf einem System stehen 4 Threads zur Bearbeitung an, die in der Reihenfolge A - B - C - D eingetroffen sind. Die Threads haben folgende Laufzeiten:

A: 2 Zeiteinheiten, B: 4 Zeiteinheiten, C: 3 Zeiteinheiten, D: 1 Zeiteinheit

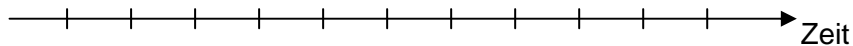
Der Scheduler arbeitet mit Round Robin Scheduling und einem Quantum von einer Zeiteinheit, d.h. nach jeder Zeiteinheit wird ein Kontextwechsel zum ersten Thread in der Warteschlange durchgeführt. Die Anordnung der wartenden Threads (nicht das eigentliche Scheduling!) erfolge auf verschiedene Arten:

- a) Zu Beginn werden die wartenden Threads nach dem FCFS-Prinzip geordnet. Ein verdrängter Thread stellt sich hinten in der Warteschlange an. Wie werden die Threads ausgeführt?



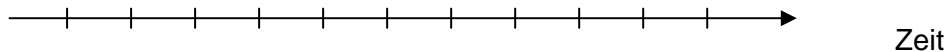
Geben Sie die Verweilzeiten der einzelnen Threads und die durchschnittliche Verweilzeit an.

- b) Zu Beginn werden die wartenden Threads nach dem SJN-Prinzip geordnet. Ein verdrängter Thread stellt sich hinten in der Warteschlange an. Wie werden die Threads ausgeführt?



Geben Sie die Verweilzeiten der einzelnen Threads und die durchschnittliche Verweilzeit an.

- c) Die Warteschlange wird zu jeder Zeit nach dem SJN-Prinzip geordnet. Wie werden die Threads ausgeführt?



Geben Sie die Verweilzeiten der einzelnen Threads und die durchschnittliche Verweilzeit an.

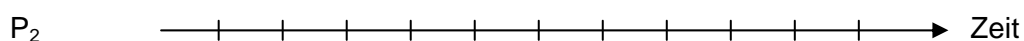
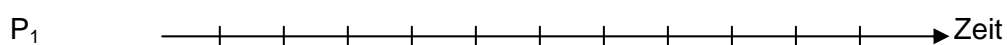
28)

Auf einem System mit zwei Prozessoren P_1 und P_2 stehen 3 Threads A, B und C zur Bearbeitung an, die folgende Eigenschaften haben:

- A: Priorität 3, blockiert nach jeweils 1 Zeiteinheit für 2 Zeiteinheiten
- B: Priorität 4, blockiert nach jeweils 2 Zeiteinheiten für 2 Zeiteinheiten
- C: Priorität 2, blockiert nie

Ein höherer numerischer Wert bedeutet auch eine höhere Priorität.

Wie werden die Threads in den ersten 10 Zeiteinheiten ausgeführt?



29)

- a) Erläutern Sie den Begriff des Quantums beim Round-Robin-Scheduling.
- b) Welche zwei Kriterien müssen bei der Festlegung der Größe des Quantums vor allem berücksichtigt werden?
- c) Was ist die Größenordnung, die an realen Systemen für das Quantum verwendet wird? Warum erfüllt dieser Wert die beiden Kriterien aus Teil b)?

30) Beantworten Sie die folgende Frage *entweder* für Linux *oder* für Windows:

Kann es passieren, dass ein Thread „verhungert“, d.h. im Zustand bereit ist, aber nie die CPU bekommt? Falls ja, beschreiben Sie wie es dazu kommen kann, falls nein, beschreiben Sie, wodurch dies verhindert wird.

Die Antwort für das Betriebssystem _____ lautet:

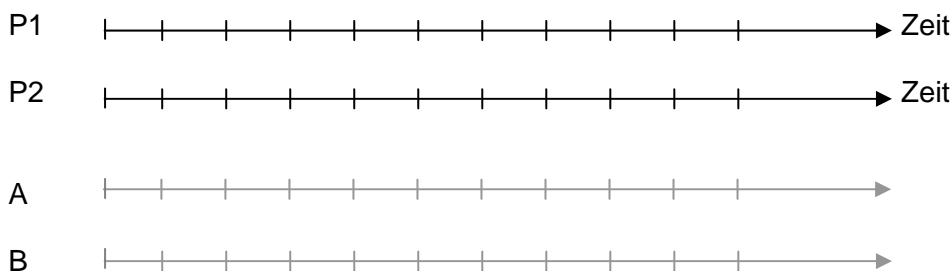
31)

- a) Ein System mit zwei Prozessoren P1 und P2 arbeitet mit prioritätenbasiertem Scheduling. Hohe numerische Werte stehen für hohe Prioritäten. Jeder Thread nutzt die CPU für die angegebene "Nutzungszeit", und blockiert sich dann für die angegebene "Blockierzeit".
Wenn zu einem Zeitpunkt ein Thread blockiert und ein anderer bereit wird, so seien *beide* Ereignisse bereits passiert, bevor der Scheduler läuft.

Es stehen 3 Threads zur Bearbeitung an mit folgenden Eigenschaften (ZE=Zeiteinheit):

- A: Priorität 4, Nutzungszeit: 1 ZE, Blockierzeit: 2 ZE
- B: Priorität 3, Nutzungszeit: 2 ZE, Blockierzeit: 2 ZE
- C: Priorität 2, Nutzungszeit: unendlich, Blockierzeit: 0 (rein CPU-lastig)

Wie werden die Threads in den ersten 10 ZE ausgeführt? Markieren Sie (als Hilfe für Sie, ohne Bewertung) auf den Achsen für A und B, wann der jeweilige Thread blockiert ist.



- b) Wie ändert sich der Ablauf der Threads, wenn Soft Affinity berücksichtigt wird (und zwar so, wie dies auch auf realen Systemen - z.B. Unix und Windows - gemacht wird)?
(Wie immer: mit Begründung)

32) Geben Sie entweder für Windows *oder* für Solaris *oder* für Linux an, welche verschiedenen „Klassen“ vom Scheduler unterschieden werden, und wie das Scheduling für Threads in den verschiedenen Klassen prinzipiell durchgeführt wird (Stichworte genügen, keine Details).

Es folgt die Beschreibung für das Betriebssystem _____ .

33)

a) Auf einem Ein-Prozessor-System, das wie Windows oder Unix (nicht Linux) arbeitet (die Unterschiede im Detail spielen für die Aufgabe keine Rolle) laufen zwei völlig CPU-intensive Threads mit gleicher Basispriorität, und sonst nichts. Was beobachten Sie für die folgenden Größen:

	Thread 1	Thread 2
CPU-Benutzung		
Kontextwechsel /sec		
Aktuelle Priorität		

Von welcher Größe am System hängt die Zahl der Kontextwechsel ab?

b) Ein I/O-intensiver Thread mache, wenn er alleine auf dem obigen System läuft, ca. 2000 Ein-/Ausgaben pro Sekunde. Was beobachten Sie für die folgenden Größen (ungefähr):

	I/O-intensiver Thread
CPU-Benutzung User Mode	
CPU-Benutzung Kernel Mode	
Kontextwechsel /sec	
Aktuelle Priorität	

Interrupts/sec am System:

c) Was beobachten Sie für die folgenden Größen (ungefähr), wenn der I/O-intensive Thread und einer der CPU-intensiven Threads gleichzeitig laufen, und beide Threads dieselbe Basispriorität haben:

	CPU-intensiver Thread	I/O-intensiver Thread
CPU-Benutzung		
Kontextwechsel /sec		
I/Os / sec		
Aktuelle Priorität		

Interrupts/sec am System: