Aufgabe 1 (5 Punkte) (Multiple Choice)

Beantworten Sie folgende Fragen durch Ankreuzen der richtigen Antwort. Für jede falsche Antwort wird ein Punkt abgezogen (es werden minimal 0 Punkte vergeben).

Welche der folgenden Aussagen gelten?

a) Im allgemeinen gilt: ein Deadlock tritt auf gdw. der Resource-Allocation Graph einen Zykel enthält.
ja ☐ **nein** ☐

b) Das Round-Robin Verfahren gehört zu den nicht-preemptiven Strategien.
ja ☐ **nein** ☐

c) Der Banker's Algorithmus wird zur Synchronisation von Prozessen eingesetzt.
ja ☐ **nein** ☐

d) Die Belady Anomalie kann beim FIFO Algorithmus auftreten.
ja ☐ **nein** ☐

e) Deadlocks werden in Linux und Windows unterschiedlich behandelt.
ja ☐ **nein** ☐

Aufgabe 2 (5+2+2+3=12 Punkte) (Prozess-Management)

a) Definieren Sie den Begriff Prozess. In welchen Zuständen kann sich typischerweise ein Prozess befinden? Stellen Sie die Zustände in einem Graphen dar, aus dem ersichtlich wird, von welchem Prozesszustand in welchen anderen Prozesszustand gewechselt werden kann.

b) Definieren Sie den Begriff Thread. Welche Vorteile bietet die Nutzung mehrerer Threads im Gegensatz zur Nutzung mehrer Prozesse?

c) Mit welchem Kommando kann man unter Unix Prozesse abbrechen? Mit welchem Kommando kann man sich die aktiven Prozesse darstellen lassen?

d) Durch welche zwei Verfahren kann Interprozesskommunikation erfolgen? Erläutern Sie kurz die beiden Verfahren.

Aufgabe 3 (4+3=7 Punkte) (Prozess-Scheduling)

Gegeben seien fünf Prozesse $P_1, ..., P_5$ mit den Laufzeiten $L_1 = 2, L_2 = 7, L_3 = 3, L_4 = 6, L_5 = 5$. Hierbei sei L_i die Laufzeit des Prozesses P_i für $i = 1, ..., 5$. Diese Prozesse sollen mittels der Verfahren FCFS, SJF, Round-Robin mit Zeitquantum vier (RR4) auf zwei CPUs A und B verteilt werden. Alle Prozesse sind bereits im System vorhanden und sind nach aufsteigendem Index angekommen (also P_1 zuerst). Die Umschaltzeiten zwischen den Prozessen sollen vernachlässigt werden.

a) Geben Sie für die drei Verfahren an, zu welchen Zeitpunkten welche Prozesse die CPUs belegen! Verwenden Sie hierzu folgende Tabelle:

		Zeitpunkte															
	CPU	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
FCFS	A																
	B																
SJF	A																
	B																
RR4	A																
	B																

b) Berechnen Sie für jedes der drei Verfahren die mittlere Wartezeit!

Aufgabe 4 (4+4=8 Punkte) (Wechselseitiger Ausschluss)

Ein Cluster von n Rechnern ist mit drei Druckern ausgestattet. Da es sich beim ersten Drucker um einen hochwertigen Farb-Laserdrucker handelt, wird dieser wesentlich häufiger genutzt, als der zweite, veraltete Schwarz/Weiss-Laserdrucker. Um nun die Auslastung für den zweiten Drucker zu erhöhen, führen Sie folgende Strategie ein: Jeder Druckjob wird zentral von einem Scheduler (mit Druckerpuffer) verwaltet und wie folgt auf die Drucker verteilt:

- Der 1. Drucker nimmt maximal 4 Jobs auf.
- Der 2. Drucker nimmt maximal 10 Jobs auf.
- Der 3. Drucker, ein veralteter Matrix-Drucker, weist keine Beschränkung auf.

Für jeden Job wird zunächst versucht, ihn auf dem 1. Drucker zu starten. Gelingt dies nicht, so wird versucht, ihn auf dem 2. Drucker zu starten. Bei erneutem Fehlversuch wird er in die Queue des 3. Druckers geschickt.

a) Der Algorithmus `scheduler` verteilt die eintreffenden Druckjobs vom Cluster auf die Drucker. Markieren Sie die Zeilen, die den Algorithmus in Pseudocode beschreiben.

☐	`init (S,1);`	☐	`init (job,1);`
☐	`scheduler(PrinterJob S) {`	☐	`scheduler(PrinterJob job) {`
☐	`wait(S);`	☐	`signal(S);`
☐	`if(q1.size() >= 4) then {`	☐	`if(q1.size() < 4) then {`
☐	`job=q1.dequeue(); }`	☐	`q1.enqueue(&job); }`
☐	`else { if(q2.size() < 10) then {`	☐	`if(q2.size() == 10) then {`
☐	`q1.enqueue(&job); }`	☐	`q2.enqueue(&job); }`
☐	`else { job=q3.dequeue(); }`	☐	`else { q3.enqueue(&job); } }`
☐	`signal(S); }`	☐	`wait(S); }`

b) Der Algorithmus `drucker1` verarbeitet die Jobs des ersten Druckers. Markieren Sie die Zeilen, die den Algorithmus in Pseudocode beschreiben.

☐	`drucker1 {`	☐	`drucker1(job.dequeue()) {`
☐	`PrinterJob S;`	☐	`PrinterJob *job;`
☐	`if(q1.size()==0) then {`	☐	`repeat {`
☐	`job=NULL;`	☐	`job=q1.enqueue();`
☐	`wait(S);`	☐	`q1.dequeue();`
☐	`wait(S);`	☐	`if(q1.size()>0) then {`
☐	`q1.enqueue(&job);`	☐	`job=q1.dequeue(); }`
☐	`if(q1.size()==0) then {`	☐	`signal(S);`
☐	`if(job!=NULL) then {`	☐	`job=q1.dequeue();`
☐	`signal(S); }`	☐	`PrintIt(job); }`
☐	`until false; }`	☐	`wait(S); }`

Aufgabe 5 (4+6=10 Punkte) (Deadlocks)

Gegeben seien vier Prozesse P0, P1, P2 und P3, die drei exklusive Betriebsmittel BM1, BM2 und BM3 benutzen wollen. Es existieren 9 Exemplare von BM1, 7 von BM2 und 5 von BM3: $\text{Available} = (9, 7, 5)$. Ferner sei der maximale Bedarf der Prozesse bekannt:

- $\text{Max}(0) = (7, 5, 3)$
- $\text{Max}(1) = (6, 2, 2)$
- $\text{Max}(2) = (9, 7, 1)$
- $\text{Max}(3) = (3, 2, 0)$

a) Prüfen Sie für die Zustände A und B, ob sich das System in einem sicheren Zustand befindet. Begründen Sie Ihre Antwort!

$\text{Allocation\_A}(0) = (0, 2, 2),$	$\text{Allocation\_B}(0) = (1, 2, 1),$	$\text{Allocation\_C}(0) = (0, 2, 0)$
$\text{Allocation\_A}(1) = (3, 1, 1),$	$\text{Allocation\_B}(1) = (3, 1, 2),$	$\text{Allocation\_C}(1) = (2, 2, 1)$
$\text{Allocation\_A}(2) = (3, 2, 1),$	$\text{Allocation\_B}(2) = (2, 2, 0),$	$\text{Allocation\_C}(2) = (2, 1, 1)$
$\text{Allocation\_A}(3) = (1, 1, 0),$	$\text{Allocation\_B}(3) = (1, 1, 0),$	$\text{Allocation\_C}(3) = (1, 1, 0)$

b) Gehen Sie von dem sicheren Zustand C aus. Welche der drei einzeln gegebenen Anforderungen führen in einen sicheren Zustand, d.h. welche Zuteilungen dürfen erlaubt werden und welche nicht? Begründen Sie Ihre Antwort!

a) $\text{Request}(3) = (0, 2, 0)$
b) $\text{Request}(2) = (2, 0, 0)$
c) $\text{Request}(1) = (2, 0, 0)$

Aufgabe 6 (2+6=8 Punkte) (Speicherverwaltung)

a) Was versteht man unter virtuellem Speicher?

b) Der Referenzstring ω beschreibe die Folge der Seitenzugriffe. Die Tabellen zeigen die Belegung der Seitenrahmen im Hauptspeicher. Es stehen einmal drei und einmal vier Rahmen zur Verfügung. Vervollständigen Sie die Tabellen unter Verwendung der theoretischen, optimalen Strategie (OPT). Markieren Sie die Seitenfehler mit einem $\star$.

$\omega =$	3	1	4	0	2	1	3	4	0	2	2	1	4	5	6	2	1	3	4	2	3
Seitenfehler				$\star$																	
Seitenrahmen 1	3	1	4	0																	
Seitenrahmen 2		3	1	4																	
Seitenrahmen 3			3	1																	

Seitenfehler																					
Seitenrahmen 1	3	1	4	0																	
Seitenrahmen 2		3	1	4																	
Seitenrahmen 3			3	1																	
Seitenrahmen 4				3																	

Aufgabe 7 (2+1+2+3=8 Punkte) (Speicherverwaltung)
Für die Speicherverwaltung nach dem Segmentierungsverfahren sei für ein Programm die folgende Segmenttabelle gegeben (Länge in Speicherworten):

Segment	Basis	Länge
0	3300	305
1	1900	198
2	1865	25
3	1710	145
4	2700	182
5	4300	55
6	4205	65

a) Was ist der Unterschied zwischen *externer* und *interner* Fragmentierung?

b) Wieviele Speicherworte stehen dem Programm im physikalischen Speicher zur Verfügung?

c) Welche ist die kleinste und welche die größte für das Programm verfügbare physikalische Adresse?

d) Berechnen Sie zu den folgenden physikalischen Adressen jeweils die logischen Adressen. Welche Anfragen lösen einen Segmentation Fault aus?

   a) 4270
   b) 1810
   c) 2888
   d) 1935
   e) 2777
   f) 4222

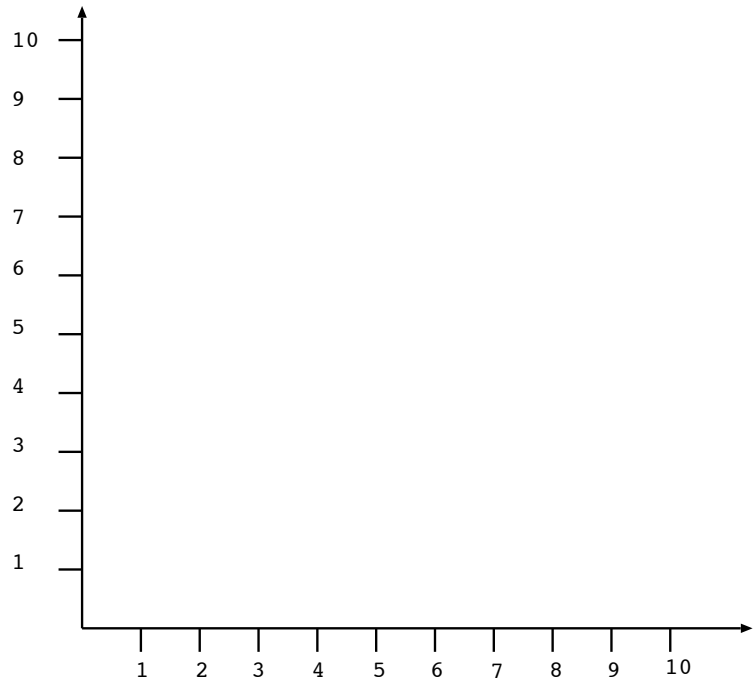
Aufgabe 8 (5+2=7 Punkte) (Lifetime-Funktion)

a) Geben Sie die Lifetime-Funktion $L(m)$ zu einem Programm an, das durch den Referenzstring ω mit

$$\omega = 2\,3\,2\,3\,1\,3\,1\,3\,0\,3\,0\,1\,2\,3\,2\,3\,2\,1\,2\,0\,2\,1\,2\,0$$

gekennzeichnet ist. Die Zeit, die zwischen zwei Seitenanfragen vergeht, soll jeweils eine Zeiteinheit betragen. Die Seitenersetzung erfolge mittels LRU-Strategie (Least Recently Used). Gehen Sie davon aus, dass das Working Set zu Beginn mit anderen als den angegebenen Seiten gefüllt ist.

Zeichnen Sie den Graphen der Lifetime-Funktion $L(m)$ für $m = 1 \ldots 10$ und geben Sie zusätzlich die Zahl der Seitenfehler für $m = 1 \ldots 10$ in einer Tabelle an.



b) Welche optimale Einstellung der Speichergrösse ergibt sich bei Anwendung des primären Knie-Kriteriums? Zeichnen Sie zusätzlich die zugehörige Gerade in Ihren Graphen ein.

Aufgabe 9 (3+4+2=9 Punkte) (Working Set)

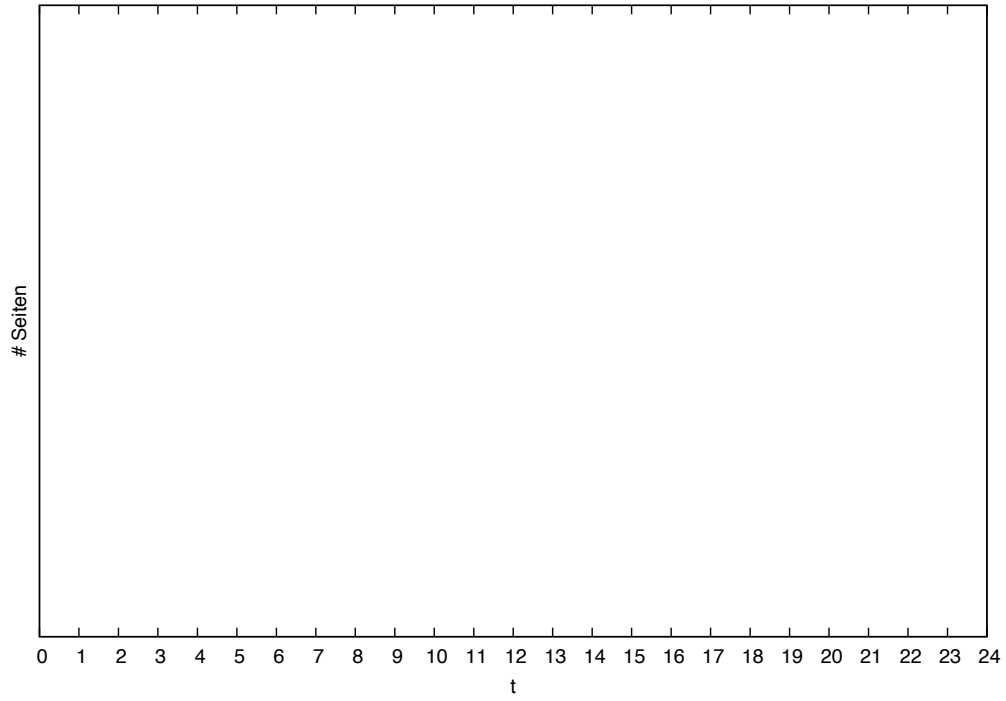
Es sei der folgende Referenzstring eines Prozesses gegeben:

$$\omega = 2\ 1\ 5\ 0\ 3\ 4\ 2\ 3\ 2\ 2\ 5\ 6\ 1\ 2\ 3\ 4\ 5\ 6\ 0\ 1\ 6\ 5\ 6\ 1$$

Der Wert h soll für diese Aufgabe $h := 5$ betragen.

a) Geben Sie die formale Definition des Working Sets $W(t, h)$ zum Zeitpunkt t mit Rückwärtsfenster h an.

b) Konstruieren Sie einen Graphen, auf dessen x-Achse die Zeit t (von 0 bis 24) und auf dessen y-Achse die Anzahl der momentan dem Prozess zugeteilten Seiten aufgetragen wird. Gehen Sie davon aus, dass das Working Set zu Beginn, also bei $t = 0$, mit keiner Seite gefüllt ist und die Zeit, die zwischen zwei Seitenanfragen vergeht, jeweils eine Zeiteinheit beträgt.



c) Geben Sie die Working Sets der lokalen Bereiche zu den Zeitpunkten 10, 15, 19 und 24 an.

Aufgabe 10 (4 Punkte) (UNIX Inodes)

Das Dateissystem von Unix basiert auf Inodes. Jede Datei enthält 4 Pointertypen: 12 direct pointers sowie je einen single indirect, double indirect und einen triple indirect pointer, die zusammen auf alle Blöcke der Datei verweisen.

Die direct pointer verweisen *direkt* auf Datenblöcke, während der single indirect pointer auf einen Block zeigt, dessen Inhalt direct pointers sind. Entsprechend zeigt der double indirect pointer auf einen Block mit single indirect pointern und der triple indirect pointer auf einen Block mit double indirect pointern.

Sei die Blockgrösse 2 KBytes (=2048 Bytes) und die Pointergrösse 2 Bytes. Wieviele Bytes lassen sich so durch jeden der 4 Pointertypen adressieren? Eventuell vorkommende Potenzen müssen nicht ausgerechnet werden.

Aufgabe 11 (4+5=9 Punkte) (Stackalgorithmus)
Zu den Seitenzugriffen eines Programms sei der folgende Referenzstring ω gegeben:

$$\omega = 6\ 5\ 4\ 3\ 2\ 5\ 6\ 3\ 1$$

a) Zeigen Sie anhand des gegebenen Referenzstrings, dass die Seitenersetzungsstrategie LIFO die Inklusionseigenschaft erfüllt.

b) Geben Sie die Prioritätsliste π_t und die resultierende Stack-Belegungsfolge s_t zum Referenzstring ω für die Zeitpunkte $t = 1, \ldots, 9$ an.