Systemprogrammierung 2

Fragensammlung

06. Januar 2015

DISCLAIMER: Dieser Fragenkatalog wurde Basis von Bearbeitungen von Mini- und Altklausuren erstellt und Übernimmt keinen Anspruch auf Vollständigkeit und Richtigkeit. Er stellt keine offizielle Veröffentlichung des Lehrstuhl 4 für Informatik der FAU Erlangen-Nürnberg dar.

1 Single-Choice

Frage 1: Bei RAID 0 wird durch das Verteilen der Daten auf mehreren Platten ein sogenanntes gestreiftes Plattensystem erzeugt. Welche Aussage dazu ist richtig?

- ☑ Der Lesezugriff auf ein gestreiftes Plattensystem ist schneller, da mehrere Platten gleichzeitig beauftragt werden können.
- ☐ Die Paritätsinformation wird gleichmäßig über alle beteiligten Platten verteilt.
- ☐ Es dürfen nicht mehr als 5 Festplatten beteiligt sein, da sonst die Paritätsinformation nicht mehr gebildet werden kann.
- ☐ Die Daten auf einem RAID-0-Festplattensystem sind besonders sicher, da beim Ausfall von nur einer Platte die verlorengegangenen Daten wieder hergestellt werden können.

Frage 2: Bei der Ausführung eines Programms in einem virtuellen Adressraum erfolgt die Adressabbildung wie folgt:

- ☐ logische Adresse → Programmadresse → virtuelle Adresse → physikalische Adresse
- ☑ Programmadresse → logische Adresse → virtuelle Adresse → physikalische Adresse
- ☐ Programmadresse → virtuelle Adresse → logische Adresse → physikalische Adresse
- ☐ virtuelle Adresse → logische Adresse → Programmadresse → physikalische Adresse

Frage 3: Bei der Behandlung von Ausnahmen (Traps oder Interrupts) unterscheidet man zwei Bearbeitungsmodelle. Welche Aussage hierzu ist richtig?

- ☐ Das Wiederaufnahmemodell dient zur Behandlung von Interrupts (Fortführung des Programms nach einer zufällig eingetretenen Unterbrechung). Bei einem Trap ist das Modell nicht sinnvoll anwendbar, da ein Trap ja deterministisch auftritt und damit eine Wiederaufnahme des Programms sofort wieder den Trap verursachen würde.
- ☐ Nach dem Beendigungsmodell werden Interrupts bearbeitet. Gibt man z. B. CTRL-C unter UNI über die Tastatur ein, wird ein Interrupt-Signal an den gerade laufenden Prozess gesendet und dieser dadurch beendet.
- ☑ Interrupts dürfen auf keinen Fall nach dem Beendigungsmodell behandelt werden, weil überhaupt kein Zusammenhang zwischen dem unterbrochenen Prozess und dem Grund des Interrupts besteht.
- ☐ Das Betriebssystem kann Interrupts, die in ursächlichem Zusammenhang mit dem gerade laufenden Prozess stehen, nach dem Beendigungsmodell behandeln, wenn eine sinnvolle Fortführung des Prozesses nicht mehr möglich ist.

Frage 4: Bei einer prioritätengesteuerten Prozess-Auswahlstrategie (Scheduling-Strategie) kann es zu Problemen kommen. Welches der folgenden Probleme kann auftreten?

- ☐ Die Anzahl der Prioritäten reicht nicht aus, wenn nur wenige Prozesse vorhanden sind.
- ☑ Prozesse können ausgehungert werden.
- ☐ Das Phänomen der Prioritätsumkehr hungert niedrig-priore Prozesse aus.
- ☐ Die Auswahlstrategie arbeitet ineffizient, wenn viele Prozesse im Zustand bereit sind.

Frage 5: Beim Blockieren in einem Monitor muss der Monitor freigegeben werden. Warum?

- ☐ weil der Faden sonst aktiv warten würde
- ☐ weil sonst die Monitordaten inkonsistent sind
- ☑ weil ein anderer Faden die Blockierungsbedingung nur aufheben kann, wenn er den Monitor betreten darf
- ☐ weil kritische Abschnitte immer nur kurz belegt sein dürfen

Frage 6: Beim Blockieren in einem Monitor muss der Monitor freigegeben werden. Warum?

- ☐ weil sonst die Monitordaten inkonsistent sind.
- ☑ weil ein anderer Thread die Blockierungsbedingung nur aufheben kann, wenn er den Monitor betreten darf.
- ☐ weil kritische Abschnitte immer nur kurz belegt sein dürfen.
- ☐ weil der Thread sonst aktiv warten würde.

Frage 7: Beim Einsatz von RAID 5 wird durch eine zusätzliche Festplatte Datensicherheit erzielt, so dass der Ausfall einer Festplatte den laufenden Betrieb nicht stören kann. Welche Aussage dazu ist richtig?

- ☑ Die Paritätsinformation wird gleichmäßig über alle beteiligten Platten verteilt.
- ☐ Der Lesezugriff auf ein Plattensystem mit RAID 5 ist langsamer als bei normalen Plattenzugriffen, da der Zugriff auf die Platten komplexer ist.
- ☐ Es sind mindestens 5 Festplatten nötig.
- ☐ Es dürfen nicht mehr als 5 Festplatten beteiligt sein, da sonst die Paritätsinformation nicht mehr gebildet werden kann.

Frage 8: Beim Einsatz von RAID 5 wird durch eine zusätzliche Festplatte Datensicherheit erzielt, so dass der Ausfall einer Festplatte den laufenden Betrieb nicht stören kann. Welche Aussage dazu ist richtig?

- ☐ Es sind mindestens 5 Festplatten nötig.
- ☑ Die Paritätsinformation wird gleichmäßig über alle beteiligten Platten verteilt.
- ☐ Auf der zusätzlichen Festplatte wird Paritätsinformation über den Inhalt der anderen Festplatten gespeichert. Dies belastet die zusätzliche Platte besonders stark.
- ☐ Der Lesezugriff auf ein Plattensystem mit RAID 5 ist langsamer als bei normalen Plattenzugriffen, da der Zugriff auf die Platten komplexer ist.

Frage 9: Beim Einsatz von RAID-Systemen kann durch zusätzliche Festplatten Fehlertoleranz erzielt werden. Welche Aussage dazu ist richtig?

- ☐ Bei RAID 4 Systemen wird Paritätsinformation gleichmäßig über alle beteiligten Platten verteilt.
- ☑ Bei allen RAID-Systemen ist ein höherer Lese-Durchsatz als bei einer einzelnen Platte möglich, da mehrere Platten gleichzeitig beauftragt werden können.
- ☐ Bei RAID 4 und 5 darf eine bestimmte Menge von Festplatten nicht überschritten werden, da es sonst nicht mehr möglich ist, die Paritätsinformation zu bilden.
- ☐ RAID 0 erzielt Fehlertoleranz durch das Verteilen der Daten auf mehrere Platten.

Frage 10: Beim Einsatz von RAID-Systemen wird durch zusätzliche Festplatten ein fehlertolerierendes Verhalten erzielt. Welche Aussage dazu ist richtig?

- ☑ Der Lesezugriff auf ein gestreiftes Plattensystem insbesondere auch auf ein RAID 5 System ist schneller, da mehrere Platten gleichzeitig beauftragt werden können.
- ☐ Bei RAID 4 und 5 darf eine bestimmte Menge von Festplatten nicht überschritten werden, da es sonst nicht mehr möglich ist, die Paritätsinformation zu bilden.
- ☐ Bei RAID 4 Systemen wird die Paritätsinformation gleichmäßig über alle beteiligten Platten verteilt.
- ☐ Bei RAID 5 Systemen sind mindestens 5 Festplatten nötig.

Frage 11: Beim Zugriff auf Datei-Inhalte wird typischerweise zwischen sequentiellem und wahlfreiem Zugriff unterschieden. Welche Aussage ist richtig?

- ☐ wahlfreier Zugriff erfolgt nach beliebigem Muster und ist ideal bei allen denkbaren Speichermedien geeignet
- ☑ bei wahlfreiem Zugriff ist ein wohlgeordnetes Zugriffsmuster nicht erkennbar und das Verfahren ist z. B. bei Festplatten geeignet
- ☐ bei sequentiellem Zugriff dürfen verschiedene Benutzer nur hintereinander auf die Datei zugreifen
- ☐ bei wahlfrei organisiertem Massenspeicher ist sequentieller Zugriff nicht möglich

Frage 12: Betrachten Sie folgende Aussagen zur Speicherung der Daten einer Datei auf Festplatte. Welche Aussage ist falsch?

- ☐ Bei kontinuierlicher Speicherung kann das Auffinden eines genügend großen zusammenhängenden Speicherbereiches schwierig sein.
- ☑ Kontinuierliche Speicherung ist gänzlich unmöglich, falls Dateien dynamisch erweiterbar sein sollen.
- ☐ Eine kontinuierliche Speicherung der Daten in aufeinanderfolgenden Blöcken erhöht die Lese- und Schreibleistung gegenüber verstreuter Speicherung.
- ☐ Bei kontinuierlicher Speicherung kann die Ortsinformation lediglich aus der Nummer des ersten Plattenblocks und der Dateilänge bestehen.

Frage 13: Ein Betriebssystem setzt logische Adressräume auf der Basis von Segmentierung ein. Welche Aussage ist falsch?

- ☐ Die Segmentierung schränkt den logischen Adressraum derart ein, sodass nur auf gültige Speicheradressen erfolgreich zugegriffen werden kann.
- ☐ Über gemeinsame Segmente kann innerhalb von zwei verschiedenen logischen Adressräumen auf dieselben Speicherzellen zugegriffen werden.
- ☐ Mit der Segmentierung kann einem Prozess mehr Speicher zugeordnet werden als physikalisch vorhanden ist, da ein Segment teilweise ausgelagert werden kann.
- ☑ Segmente können verschiedene Länge haben. Eine Längenbegrenzung wird üblicherweise bei der Speicherabbildung geprüft.

Frage 14: Ein Prozess wird in den Zustand bereit überführt. Welche Aussage passt nicht zu diesem Vorgang?

- ☐ Der Prozess wurde von einem Prozess mit einer höheren Priorität verdrängt.
- ☐ Der Prozess hat auf Daten von der Festplatte gewartet und die Daten stehen nun zur Weiterbearbeitung bereit.
- ☑ Der Prozess wartet auf eine Tastatureingabe.
- ☐ Der Prozess ist grundsätzlich lauffähig und wird im Rahmen der mittelfristigen Prozesseinplanung eingelagert.

Frage 15: Ein System setzt Segmentierung und Seitenadressierung ein. Zwei Prozesse sollen ein Segment gemeinsam benutzen. Wie geht das Betriebssystem vor, um das gemeinsame Segment einzurichten?

- ☑ Das Betriebssystem legt eine Seitentabelle für das gemeinsame Segment an und trägt diese bei beiden Prozessen in die jeweiligen Segmenttabellen ein.
- ☐ Das Betriebssystem trägt jeweils eine Seitentabelle in die prozesseigene Segmenttabelle ein und sorgt dafür, dass die Einträge jeweils auf die gleichen Seitenrahmen im Hauptspeicher verweisen.
- ☐ Gemeinsame Segmente können nur durch den Prozess jedoch nicht vom Betriebssystem eingerichtet werden.
- ☐ Die beiden Prozesse bekommen die gleiche Segmenttabelle zugewiesen.

Frage 16: Ein laufender Prozess wird durch den Scheduler verdrängt. Welcher Zustandsübergang findet statt?

- ☐ Der Prozess wechselt vom Zustand bereit in den Zustand blockiert.
- ☐ Der Prozess wechselt vom Zustand laufend in den Zustand blockiert.
- ☑ Der Prozess wechselt vom Zustand laufend in den Zustand bereit.
- ☐ Der Prozess wechselt vom Zustand laufend in den Zustand beendet.

Frage 17: Ein laufender Prozess wird in den Zustand blockiert überführt. Welche Aussage passt zu diesem Vorgang?

- ☐ Der Prozess terminiert.
- ☑ Der Prozess wartet auf Daten von der Festplatte.
- ☐ Es ist kein direkter Übergang von laufend nach blockiert möglich.
- ☐ Der Prozess liest von der Festplatte mit nichtblockierenden Eingabeoperationen.

Frage 18: Es gibt verschiedene Ursachen, wie Nebenläufigkeit in einem System entstehen kann (gewollt oder auch ungewollt). Was gehört nicht dazu?

- ☐ durch Interrupts
- ☑ durch die MMU
- ☐ durch preemptives Scheduling
- ☐ durch Threads auf einem Monoprozessorsystem

Frage 19: Es gibt verschiedene Ursachen, wie Nebenläufigkeit in einem System entstehen kann (gewollt oder auch ungewollt). Was gehört nicht dazu?

- ☐ durch Interrupts
- ☐ durch Signale an einen UNIX-Prozess
- ☑ durch Koroutinen
- ☐ durch Threads und preemptives Scheduling auf einem Mono(/Multi)prozessorsystem

Frage 20: Es gibt verschiedene Ursachen, wie Nebenläufigkeit in einem System entstehen kann (gewollt oder auch ungewollt). Was gehört nicht dazu?

- ☐ durch Interrupts
- ☐ durch preemptives Scheduling
- ☑ durch Traps
- ☐ durch Threads auf einem Multiprozessorsystem

Frage 21: Für lokale Variablen, Aufrufparameter, etc. einer Funktion wird bei vielen Prozessoren ein sog. Aktivierungsblock (activation record oder stack frame) auf dem Stack angelegt. Welche Aussage ist richtig?

- ☐ Nach dem Rücksprung aus einer Funktion sind Zeiger auf die Speicherzellen ihres Aktivierungsblocks nicht mehr gültig. Ein Zugriff über solch einen Zeiger führt dann zu einem Segmentation fault.
- ☐ Bei rekursiven Funktionsaufrufen kann der Aktivierungsblock in jedem Fall wiederverwendet werden, weil die gleiche Funktion aufgerufen wird.
- ☑ Über Zeiger kann man alle Daten des Aktivierungsblocks der aufrufenden Funktion verändern.
- ☐ Der Compiler legt zur Übersetzungszeit fest, an welcher Position im Aktivierungsblock der main-Funktion die globalen Variablen angelegt werden.

Frage 22: Für welchen Zweck wird der Systemaufruf listen() benutzt?

- ☐ Damit das Betriebssystem überhaupt Systemaufrufe annimmt, muss es erst mit listen() in einen Modus des Zuhörens gebracht werden.
- ☑ Mit listen() wird ein Socket für die Verbindungsannahme vorbereitet. Ein Parameter gibt an, wie viele Verbindungsanfragen vor deren Annahme gepuffert werden können.
- ☐ Der Aufruf von listen() wartet solange an einem Socket, bis eine einkommende Verbindungsanfrage vorliegt.
- ☐ Mit listen() wird ein Socket für die Verbindungsannahme vorbereitet. Ein Parameter gibt an, wie viele laufende Verbindungen maximal möglich sind.

Frage 23: Gegeben sei folgendes Szenario: zwei Fäden werden auf einem Monoprozessorsystem mit der Strategie First Come First Served verwaltet. In jedem Faden wird die Anweisung i++; auf die gemeinsame, globale Variable i ausgeführt. Welche der folgenden Aussagen ist richtig?

- ☑ In einem Monoprozessorsystem ohne Verdrängung ist keinerlei Synchronisation erforderlich.
- ☐ Während der Inkrementoperation müssen Interrupts vorübergehend unterbunden werden.
- ☐ Die Inkrementoperation muss mit einer CAS-Anweisung nicht-blockierend synchronisiert werden.
- ☐ Die Operation i++ ist auf einem Monoprozessorsystem immer atomar.

Frage 24: Gegeben sei folgendes Szenario: zwei Threads werden auf einem Monoprozessorsystem mit der Strategie First Come First Served verwaltet. In jedem Faden wird die Anweisung i++; auf die gemeinsame, globale Variable i ausgeführt. Welche der folgenden Aussagen ist richtig:

- ☐ Die Inkrementoperation muss mit einer CAS-Anweisung nicht-blockierend synchronisiert werden.
- ☐ Die Operation i++ ist auf einem Monoprozessorsystem immer atomar.
- ☐ Während der Inkrementoperation müssen Interrupts vorübergehend unterbunden werden.
- ☑ In einem Monoprozessorsystem ohne Verdrängung ist keinerlei Synchronisation erforderlich.

Frage 25: Im regulären Ablauf eines Anwendungsprogramms und einer Signalbehandlungsfunktion (bzw. im Ablauf eines Betriebssystems und einer Unterbrechungsbehandlungsfunktion) wird auf gemeinsame Daten modifizierend zugegriffen. Welche Aussage ist falsch?

- ☐ Falscher Einsatz von Schlossvariablen (lock/unlock-Operationen) kann zu Verklemmungen führen
- ☑ Der Aufruf von P-Operationen im Anwendungsprogramm und dazu korrespondierenden V-Operationen in der Signalbehandlung führt nicht zu Verklemmungen
- ☐ Alle Verfahren der mehrseitigen Synchronisation können eingesetzt werden
- ☐ Alle Verfahren nicht-blockierender Synchronisation sind einsetzbar

Frage 26: In einem Segmentdeskriptor werden verschiedene Informationen über ein Segment eines logischen Adressraums gehalten. Was gehört sicher nicht dazu?

- ☑ die Benutzer-IDs, für die diese Zugriffsrechte gelten
- ☐ die physikalische Adresse des Segmentanfangs im Hauptspeicher
- ☐ Zugriffsrechte (z. B. lesen, schreiben, ausführen)
- ☐ die Länge des Segments

Frage 27: In einem Seitendeskriptor werden verschiedene Informationen über eine Seite eines virtuellen Adressraums gehalten. Was gehört sicher nicht dazu?

- ☑ Die Benutzerkennung in einem Mehrbenutzersystem
- ☐ Die Position der Seite im Hauptspeicher
- ☐ Zugriffsrechte (z. B. lesen, schreiben, ausführen)
- ☐ Ein Zähler, der ein Maß für das Alter der Seite enthält

Frage 28: In einem Seitendeskriptor werden verschiedene Informationen über eine Seite eines virtuellen Adressraums gehalten. Was gehört sicher nicht dazu?

- ☐ Die Adresse der Seite im physikalischen Hauptspeicher
- ☑ Die Position der Seite im logischen Adressraum
- ☐ Zugriffsrechte (z. B. lesen, schreiben, ausführen)
- ☐ Ein Zähler, der ein Maß für das Alter der Seite enthält

Frage 29: In einem Seitendeskriptor werden verschiedene Informationen über eine Seite eines virtuellen Adressraums gehalten. Was gehört sicher nicht dazu?

- ☐ Zugriffsrechte (z. B. lesen, schreiben, ausführen)
- ☑ die Größe der Seite
- ☐ die Blockadresse der Seite auf dem Hintergrundspeicher
- ☐ ein Zähler, der ein Maß für das Alter der Seite enthält

Frage 30: In einem Seitendeskriptor werden verschiedene Informationen über eine Seite eines virtuellen Adressraums gehalten. Was gehört sicher nicht dazu?

- ☐ Zugriffsrechte (z. B. lesen, schreiben, ausführen)
- ☐ die Adresse der Seite auf dem Hintergrundspeicher
- ☑ die Zahl der page-faults
- ☐ die Information, ob die Seite seit dem letzten Einlagern modifiziert wurde

Frage 31: In einem Seitendeskriptor werden verschiedene Informationen über eine Seite eines virtuellen Adressraums gehalten. Was gehört sicher nicht dazu?

- ☐ die Adresse der Seite im Hauptspeicher
- ☑ die Position der Seite im virtuellen Adressraum
- ☐ Zugriffsrechte (z. B. lesen, schreiben, ausführen)
- ☐ ein Zähler, der ein Maß für das Alter der Seite enthält

Frage 32: In einem UNIX-UFS-Dateisystem gibt es symbolische Namen/Verweise (Symbolic Links). Welche Aussage ist richtig?

- ☐ Der Systemaufruf stat() liefert im Gegensatz zum Systemaufruf lstat() die Dateiattribute des symbolischen Verweises und nicht die Attribute vom Ziel des Verweises.
- ☐ Ein symbolischer Verweis kann ausschließlich auf reguläre Dateien verweisen.
- ☑ Beim Zugriff auf einen Symbolic Link kann ein 'No such file or directory.'-Fehler auftreten (errno==ENOENT), obwohl der Symbolic Link existiert.
- ☐ In jedem Inode ist ein Referenzzähler gespeichert, welcher die Anzahl der Symbolic Links angibt, die auf ihn verweisen.

Frage 33: In einem UNIX-UFS-Dateisystem gibt es symbolische Verweise (Symbolic Links) und feste Verweise (Hard Links) auf Dateien. Welche Aussage ist falsch?

- ☑ Hard Links können nur vom Systemadministrator angelegt werden.
- ☐ Ein Hard Link kann nur auf Dateien, jedoch nicht auf Verzeichnisse verweisen.
- ☐ Ein Symbolic Link kann auf Verzeichnisse verweisen.
- ☐ Symbolic Links können existieren, obwohl die verwiesene Datei oder das verwiesene Verzeichnis bereits gelöscht wurde.

Frage 34: In einem UNIX-UFS-Dateisystem gibt es symbolische Verweise (Symbolic Links) und feste Verweise (Hard Links) auf Dateien. Welche Aussage ist richtig?

- ☐ Ein Symbolic Link kann nur auf Dateien nicht jedoch auf Verzeichnisse verweisen.
- ☐ Ein Hard Link kann nur auf Verzeichnisse nicht jedoch auf Dateien verweisen.
- ☐ Hard Links können nur vom Systemadministrator angelegt werden.
- ☑ Symbolic Links können existieren, obwohl die Ziel-Datei oder das Ziel-Verzeichnis bereits gelöscht wurde.

Frage 35: In einem UNIX-UFS-Dateisystem gibt es symbolische Verweise (Symbolic Links) und feste Verweise (Hard Links) auf Dateien. Welche Aussage ist richtig?

- ☐ Ein Symbolic Link kann nicht auf Dateien anderer Dateisysteme verweisen.
- ☐ Ein Hard Link kann nur auf Verzeichnisse nicht jedoch auf Dateien verweisen.
- ☐ Wird der letzte Symbolic Link auf eine Datei gelöscht, so wird auch die Datei selbst gelöscht.
- ☑ Für jede reguläre Datei existiert mindestens ein Hard-Link im selben Dateisystem.

Frage 36: In einem kritischen Abschnitt soll ein Element aus einer doppelt verketteten Liste entnommen und in eine andere, einfach verkettete Liste eingehängt werden. Welches Synchronisationsverfahren ist hierbei für den Einsatz in einem Multiprozessorsystem am besten geeignet?

- ☐ Nicht-blockierende Synchronisationsverfahren, weil sie am besten mit den parallelen Abläufen in einem Multiprozessorsystem zurechtkommen.
- ☑ Eine Lock-Variable zur Absicherung des kritischen Abschnitts, verbunden mit aktivem Warten falls der kritische Abschnitt gerade belegt ist.
- ☐ Einseitige Synchronisation (wie Unterbrechungssperren), weil ja nur kurzzeitig (wenige Maschinenbefehle lang) der Zugriff auf die Listen durch andere Prozesse verhindert werden muss.
- ☐ Ein Monitor, verbunden mit einer Abgabe des Prozessors, falls der Monitor gerade belegt ist.

Frage 37: In welcher Situation tritt kein Seitenfehler auf?

- ☐ Die Seite wurde im Rahmen einer copy-on-write-Daten Übertragung einem anderen Prozess zur Verfügung gestellt und der empfangende Prozess greift modifizierend auf die Seite zu.
- ☐ Die Seite wurde im Rahmen einer copy-on-write-Daten Übertragung einem anderen Prozess zur Verfügung gestellt und der sendende Prozess greift modifizierend auf die Seite zu.
- ☐ Die Seite wurde auf Hintergrundspeicher geschrieben, wird in einem Freiseitenpuffer gehalten, ist noch in der Seitentabelle des Prozesses eingetragen und der Prozess greift modifizierend zu.
- ☑ Das Referenz-Bit der Seite wurde im Rahmen der Second-Chance-Strategie zurückgesetzt und der Prozess greift danach erneut auf die Seite zu.

Frage 38: In welcher Situation tritt kein Seitenfehler auf?

- ☐ die Seite wurde im Rahmen einer copy-on-write-Daten Übertragung einem anderen Prozess zur Verfügung gestellt und der empfangende Prozess greift modifizierend auf die Seite zu.
- ☑ die Seite wurde im Rahmen einer copy-on-write-Daten Übertragung einem anderen Prozess zur Verfügung gestellt und der sendende Prozess greift lesend auf die Seite zu.
- ☐ die Seite wurde auf Hintergrundspeicher geschrieben, wird in einem Freiseitenpuffer gehalten, ist noch in der Seitentabelle des Prozesses eingetragen und der Prozess greift modifizierend zu.
- ☐ das Anwesenheits-Bit der Seite wurde vom Betriebssystem zurückgesetzt und der Prozess greift danach erneut auf die Seite zu.

Frage 39: In welcher der folgenden Situationen wird ein laufender Prozess in den Zustand blockiert überführt?

- ☐ Ein Kindprozess des Prozesses terminiert.
- ☐ Der Prozess hat einen Seitenfehler für eine Seite, die bereits in den Freiseitenpuffer eingetragen, aber noch im Hauptspeicher vorhanden ist.
- ☑ Der Prozess greift lesend auf eine Datei zu und der entsprechende Datenblock ist noch nicht im Hauptspeicher vorhanden.
- ☐ Der Prozess ruft eine V-Operation auf einen Semaphor auf und der Semaphor hat gerade den Wert 0.

Frage 40: Man unterscheidet Traps und Interrupts. Welche Aussage ist richtig?

- ☐ Bei der mehrfachen Ausführung eines unveränderten Programms mit gleicher Eingabe treten Interrupts immer an den gleichen Stellen auf.
- ☐ Der Zeitgeber (Systemuhr) unterbricht die Programmbearbeitung in regelmäßigen Abständen. Die genaue Stelle der Unterbrechungen ist damit vorhersagbar. Somit sind solche Unterbrechungen in die Kategorie Trap einzuordnen.
- ☑ Der Zugriff auf eine logische Adresse kann zu einem Trap führen.
- ☐ Wenn ein Interrupt einen schwerwiegenden Fehler signalisiert, muss das unterbrochene Programm abgebrochen werden.

Frage 41: Man unterscheidet Traps und Interrupts. Welche Aussage ist richtig?

- ☑ Der Zugriff auf eine virtuelle Adresse kann zu einem Trap führen.
- ☐ Bei der mehrfachen Ausführung eines unveränderten Programms mit gleicher Eingabe treten Interrupts immer an den gleichen Stellen auf.
- ☐ Der Zeitgeber (Systemuhr) unterbricht die Programmbearbeitung in regelmäßigen Abständen. Die genaue Stelle der Unterbrechungen ist damit vorhersagbar. Somit sind solche Unterbrechungen in die Kategorie Trap einzuordnen.
- ☐ Wenn ein Interrupt einen schwerwiegenden Fehler signalisiert, muss das unterbrochene Programm abgebrochen werden.

Frage 42: Man unterscheidet Traps und Interrupts. Welche Aussage ist richtig?

- ☑ Normale Rechenoperationen können zu einem Trap führen.
- ☐ Bei der mehrfachen Ausführung eines unveränderten Programms mit gleicher Eingabe treten Interrupts immer an den gleichen Stellen auf.
- ☐ Der Zeitgeber (Systemuhr) unterbricht die Programmbearbeitung in regelmäßigen Abständen. Die genaue Stelle der Unterbrechungen ist damit vorhersagbar. Somit sind solche Unterbrechungen in die Kategorie Trap einzuordnen.
- ☐ Wenn ein Interrupt einen schwerwiegenden Fehler signalisiert, muss das unterbrochene Programm abgebrochen werden.

Frage 43: Man unterscheidet die Begriffe Programm und Prozess. Welche der folgenden Aussagen zu diesem Themengebiet ist richtig?

- ☐ Wenn ein Programm nur einen aktiven Ablauf enthält, nennt man diesen Prozess, enthält das Programm mehrere Abläufe, nennt man diese Threads.
- ☑ Ein Prozess ist ein Programm in Ausführung - ein Prozess kann aber auch mehrere verschiedene Programme ausführen
- ☐ Das Programm ist der statische Teil (Rechte, Speicher, etc.), der Prozess der aktive Teil (Programmzähler, Register, Stack).
- ☐ Ein Programm kann immer nur von einem Prozess ausgeführt werden

Frage 44: Man unterscheidet kurz- mittel- und langfristige Prozesseinplanung. Welche Aussage hierzu ist richtig?

- ☐ Wenn ein Prozess auf Daten von der Platte wartet, wird er in den Zustand blockiert versetzt. Da die Einlagerung von Daten von der Platte sehr lange dauert (mehrere Millisekunden gegen über einem CPU-Takt im GHz-Bereich), ist diese Situation der mittelfristigen Einplanung zuzuordnen.
- ☐ Wenn der Adressraum eines laufbereiten Prozesses aufgrund von Speichermangel ausgelagert wird (swap-out) wird der Prozess im Rahmen der mittelfristigen Einplanung in den Zustand blockiert überführt bis die Daten wieder eingelagert werden.
- ☑ Wenn ein Prozess auf einen Seitenfehler (page fault) trifft, wird er im Rahmen der kurzfristigen Einplanung in den Zustand blockiert überführt bis die Seite eingelagert wurde.
- ☐ Wenn ein Prozess auf einen Seitenfehler (page fault) trifft, wird er im Rahmen der kurzfristigen Einplanung in den Zustand schwebend bereit überführt, weil er ja unmittelbar nach dem Einlagern der Seite wieder weiterlaufen kann.

Frage 45: Man unterscheidet zwischen privilegierten und nicht-privilegierten Maschinenbefehlen. Welche Aussage ist richtig?

- ☐ Privilegierte Maschinenbefehle dürfen in Anwendungsprogrammen grundsätzlich nicht verwendet werden.
- ☐ Die Benutzung eines privilegierten Maschinenbefehls in einem Anwendungsprogramm führt zu einer asynchronen Programmunterbrechung.
- ☑ Privilegierte Maschinenbefehle können durch Betriebssystemprogramme (partielle Interpretation) implementiert werden.
- ☐ Privilegierte Befehle sind die einzige Möglichkeit, auf Geräteregister zuzugreifen.

Frage 46: Mit logischen Adressräumen kann man mehrere Zwecke erreichen. Was gehört nicht dazu?

- ☐ Sicherheit: man kann unbefugten Zugriff auf Daten verhindern.
- ☑ Schutzmechanismus: man kann die Auswirkungen von Berechnungsfehlern oder technischen Fehlern (wie z. B. Bitkipper) begrenzen.
- ☐ Virtualisierung: man kann in einem Programmlauf mehr Speicher adressieren, als physikalisch vorhanden ist.
- ☐ bessere Verwaltung: man kann Speicherbereiche mit unterschiedlicher Bedeutung voneinander abgrenzen.

Frage 47: Namensräume dienen u. a. der Organisation von Dateisystemen. Welche Aussage ist richtig?

- ☐ Flache Namensräume sind besonders einfach implementierbar und damit vor allem für Mehrbenutzersysteme gut geeignet
- ☐ Flache Namensräume erlauben pro Benutzer nur einen Kontext
- ☐ Der Nachteil von hierarchischen Namensräumen besteht darin, dass das Dateisystem spezielle Funktionen zum Auflösen von Namenskonflikten implementieren muss
- ☑ In einem hierarchisch organisierten Namensraum dürfen gleiche Namen in unterschiedlichen Kontexten enthalten sein

Frage 48: Nehmen Sie an, der Ihnen bekannte Systemaufruf stat(2) wäre analog zu der Funktion readdir(3) mit folgender Schnittstelle implementiert: struct stat *stat(const char *path); Welche Aussage ist richtig?

- ☐ Solch eine Schnittstelle ist nicht schön, da dadurch die aufrufende Funktion auf internen Speicher des Betriebssystems zugreifen könnte.
- ☐ Der Aufrufer muss sicherstellen, dass er den zurückgelieferten Speicher mit free(3) wieder freigibt, wenn er die Dateiattribute nicht mehr weiter benötigt.
- ☐ Der Systemaufruf liefert einen Zeiger zurück, über den die aufrufende Funktion direkt auf eine Datenstruktur zugreifen kann, die die Dateiattribute enthält.
- ☑ Ein Zugriff über den zurückgelieferten Zeiger liefert völlig zufällige Ergebnisse oder einen Segmentation fault.

Frage 49: Sie kennen den Begriff Demand-Paging. Welche Aussage dazu ist richtig?

- ☐ Demand-Paging setzt eine segmentierte Speicherverwaltung voraus.
- ☐ Demand-Paging benötigt keinerlei Hardware-Unterstützung, da sich alle benötigten Mechanismen auch ohne MMU realisieren lassen.
- ☐ Demand-Paging erlaubt es größere logische Adressräume anzulegen, als Hauptspeicher vorhanden ist. Allerdings muss vorausgesetzt werden, dass ein Prozess nicht alle Seiten des logischen Adressraums tatsächlich anspricht.
- ☑ Demand-Paging lädt eine Seite erst dann in den Hauptspeicher, wenn sie tatsächlich angesprochen wird. Nicht benutzte Seiten werden unter Umständen aus dem Hauptspeicher ausgelagert.

Frage 50: Sie kennen den Begriff Seitenflattern (Thrashing). Welche Aussage ist richtig?

- ☐ Als Seitenflattern bezeichnet man das wiederholte Löschen und Neuladen des Translation-Look-Aside-Buffer (TLB), ausgelöst durch häufigen Prozesswechsel.
- ☑ Als Seitenflattern bezeichnet man das wiederholte Einlagern einer erst vor kurzem verdrängten Speicherseite. Die Prozesse verbringen als Folge die meiste Zeit mit dem Warten auf die Behebung von Seitenfehlern.
- ☐ Seitenflattern erkennt man an der starken Geräuschentwicklung der Festplatte, da auf Grund häufiger Seitenzugriffe der Lesekopf ständig neu positioniert wird. Bei Systemen ohne Festplatte (z. B. Thin-Clients) kann das Seitenflattern nicht auftreten.
- ☐ Bei Verwendung der LRU-Seitenersetzungsstrategie kann Seitenflattern prinzipbedingt nicht auftreten.

Frage 51: Sie kennen den Begriff Seitenflattern (thrashing). Welche Aussage ist richtig?

- ☐ Als Seitenflattern bezeichnet man das wiederholte löschen und neu laden des Translation-Look-Aside-Buffer (TLB), ausgelöst durch häufigen Prozesswechsel.
- ☑ Als Seitenflattern bezeichnet man das wiederholte Einlagern einer erst vor kurzem verdrängten Speicherseite. Die Prozesse verbringen als Folge die meiste Zeit mit dem Warten auf die Behebung von Seitenfehlern.
- ☐ Seitenflattern erkennt man an der starken Geräuschentwicklung der Festplatte, da auf Grund häufiger Seitenzugriffe der Lesekopf ständig neu positioniert wird. Bei Systemen ohne Festplatte (Thin-Clients) kann das Seitenflattern nicht auftreten.
- ☐ Bei Systemen ohne Festplatte (Thin-Clients) kann das Seitenflattern nicht auftreten.

Frage 52: Sie kennen den Translation-Look-Aside-Buffer (TLB). Welche Aussage ist richtig?

- ☐ Der TLB puffert die Ergebnisse der Abbildung von physikalische auf logische Adressen, sodass eine erneute Anfrage sofort beantwortet werden kann.
- ☑ Verändert sich die Speicherabbildung von logischen auf physikalische Adressen aufgrund einer Adressraumumschaltung, so werden auch die Daten im TLB ungültig.
- ☐ Der TLB verkürzt die Zugriffszeit auf den physikalischen Speicher, da ein Teil des möglichen Speichers in einem schnellen Pufferspeicher vorgehalten wird.
- ☐ Der TLB puffert Daten bei der Ein-/Ausgabebehandlung und beschleunigt diese damit.

Frage 53: Sie kennen die vier notwendigen/hinreichenden Bedingungen für Verklemmungen. Welche Aussage ist falsch?

- ☐ Verklemmungsvermeidung (Deadlock Avoidance) sorgt zur Laufzeit dafür, dass die 4. Bedingung (zirkuläres Warten) nicht eintreten kann.
- ☑ Bei Verklemmungsvorbeugung (Deadlock Prevention) muss dafür gesorgt werden, dass keine der notwendigen Bedingungen erfüllt ist.
- ☐ Indem man ein Auftreten der 4. Bedingung (zirkuläres Warten) durch Anwendung von Regeln ausschließt, wird Verklemmungsvorbeugung (Deadlock Prevention) erreicht.
- ☐ Bei Verklemmungserkennung wird keine Überprüfung der drei notwendigen Bedingungen durchgeführt.

Frage 54: User- und Kernel-Threads unterscheiden sich in verschiedenen Eigenschaften. Welche Kombination ist richtig?

- ☐ Bei User-Threads können anwendungsabhängig Schedulingstrategien eingesetzt werden; Kernel-Threads können Multiprozessoren nicht ausnutzen.
- ☐ Kernel-Threads werden äußerst effizient umgeschaltet; User-Threads blockieren sich bei blockierenden Systemaufrufen gegenseitig.
- ☐ Bei Kernel-Threads ist die Schedulingstrategie meist vorgegeben; User-Threads können Multiprozessoren ausnutzen.
- ☑ User-Threads werden effizient umgeschaltet; blockierende Systemaufrufe von Kernel-Threads blockieren keine anderen Threads.

Frage 55: User-Level- und Kernel-Level-Threads unterscheiden sich in verschiedenen Eigenschaften. Welche Kombination ist richtig?

- ☐ Bei User-Level-Threads können anwendungsabhängig Schedulingstrategien eingesetzt werden; Kernel-Level-Threads können Multiprozessoren nicht ausnutzen.
- ☐ Kernel-Level-Threads werden sehr effizient umgeschaltet; User-Level-Threads blockieren sich bei blockierenden Systemaufrufen gegenseitig.
- ☐ Bei Kernel-Level-Threads ist die Schedulingstrategie meist vorgegeben; User-Level-Threads können Multiprozessoren ausnutzen.
- ☑ User-Level-Threads werden effizient umgeschaltet; blockierende Systemaufrufe von Kernel-Level-Threads blockieren keine anderen Threads.

Frage 56: Virtualisierung kann als Maßnahme gegen Verklemmungen genutzt werden. Warum?

- ☐ Im Fall einer Verklemmung können zusätzliche virtuelle Betriebsmittel neu erzeugt werden. Diese können dann eingesetzt werden, um die fehlenden physikalischen Betriebsmittel zu ersetzen.
- ☑ Durch Virtualisierung kann man über Abbildungsvorgänge Zyklen, die auf der logischen Ebene vorhanden sind, auf der physikalischen Ebene auflösen.
- ☐ Eine Verklemmungsauflösung ist einfacher, weil virtuelle Betriebsmittel jederzeit ohne Schaden entzogen werden können.
- ☐ Durch Virtualisierung ist ein Entzug von physikalischen Betriebsmitteln möglich, obwohl dies auf der logischen Ebene unmöglich ist.

Frage 57: Was gehört nicht zu den typischen Dateiattributen?

- ☐ die Länge des Dateiinhalts
- ☐ der Zeitpunkt des letzten lesenden Zugriffs
- ☑ die Prozess-Identifikation des Prozesses, der die Datei gerade geöffnet hat
- ☐ die Benutzer-Identifikation des Eigentümers der Datei

Frage 58: Was ist ein Stack-Frame?

- ☐ Der Speicherbereich, in dem der Programmcode einer Funktion abgelegt ist.
- ☐ Ein spezieller Registersatz des Prozessors zur Bearbeitung von Funktionen.
- ☐ Ein Fehler, der bei unberechtigten Zugriffen auf den Stack-Speicher entsteht.
- ☑ Ein Bereich des Speichers, in dem u.a. lokale automatic-Variablen einer Funktion abgelegt sind.

Frage 59: Was muss ein(e) Software-Entwickler(in) unbedingt beachten, damit seine/ihre Programme nicht Opfer eines Hacker-Angriffs werden?

- ☑ Bei Verwendung der Programmiersprache C muss darauf geachtet werden, dass die Stringoperationen strcpy() und strcat() nur eingesetzt werden, wenn die Länge des zu übertragenden Strings noch in das Ziel-Array passt.
- ☐ Es dürfen keine vorgefertigten Bibliotheksfunktionen verwendet werden, weil deren Implementierung als nicht vertrauenswürdig eingestuft werden muss.
- ☐ Der Quellcode darf nicht herausgegeben werden, damit Schwachstellen nicht entdeckt werden können.
- ☐ Der Einsatz der Bibliotheksfunktion fgets() muss verboten werden, da diese Funktion nicht sicher ausgeführt werden kann.

Frage 60: Was passiert, wenn Sie in einem C-Programm über einen ungültigen Zeiger versuchen auf Speicher zuzugreifen?

- ☐ Das Betriebssystem erkennt die ungültige Adresse bei der Weitergabe des Befehls an die CPU (partielle Interpretation) und leitet eine Ausnahmebehandlung ein.
- ☑ Die MMU erkennt die ungültige Adresse bei der Adressumsetzung und löst einen Trap aus.
- ☐ Beim Laden des Programms wird die ungültige Adresse erkannt und der Speicherzugriff durch einen Sprung auf eine Abbruchfunktion ersetzt. Diese Funktion beendet das Programm mit der Meldung "Segmentation fault".
- ☐ Der Compiler erkennt die problematische Code-Stelle und generiert Code, der zur Laufzeit bei dem Zugriff einen entsprechenden Fehler auslöst.

Frage 61: Was passiert, wenn Sie in einem C-Programm über einen ungültigen Zeiger versuchen auf Speicher zuzugreifen?

- ☐ Der Compiler erkennt den ungültigen Zugriff und meldet beim Übersetzen einen Fehler.
- ☑ Beim Zugriff über den Zeiger erkennt die MMU, wenn sie die erforderliche Adressumsetzung vornimmt, die ungültige Adresse und löst einen Trap aus.
- ☐ Der Speicher schickt an die CPU einen Interrupt. Hierdurch wird das Betriebssystem angesprungen, das den gerade laufenden Prozess mit einem Segmentation fault-Signal unterbricht.
- ☐ Das Betriebssystem erkennt die ungültige Adresse bei der Weitergabe des Befehls an die CPU (partielle Interpretation) und leitet eine Ausnahmebehandlung ein.

Frage 62: Was passiert, wenn Sie in einem C-Programm über einen ungültigen Zeiger versuchen auf Speicher zuzugreifen?

- ☐ Der Compiler erkennt die problematische Code-Stelle und generiert Code, der zur Laufzeit bei dem Zugriff einen entsprechenden Fehler auslöst.
- ☐ Der Speicher schickt an die CPU einen Interrupt. Hierdurch wird das Betriebssystem angesprungen, das den gerade laufenden Prozess mit einem Segmentation fault-Signal unterbricht.
- ☑ Beim Zugriff über den Zeiger muss die MMU die erforderliche Adressumsetzung vornehmen, erkennt die ungültige Adresse und löst einen Trap aus.
- ☐ Das Betriebssystem erkennt die ungültige Adresse bei der Weitergabe des Befehls an die CPU (partielle Interpretation) und leitet eine Ausnahmebehandlung ein.

Frage 63: Was passiert, wenn Sie in einem Programm über einen ungültigen Zeiger versuchen auf Speicher zuzugreifen?

- ☐ Der Arbeitsspeicher erkennt, dass es sich um eine ungültige Adresse handelt und schickt an die CPU einen Interrupt. Hierdurch wird das Betriebssystem aktiviert, das den gerade laufenden Prozess mit einem Segmentation fault-Signal unterbricht.
- ☑ Die MMU erkennt die ungültige Adresse bei der Adressumsetzung und löst einen Trap aus.
- ☐ Beim Binden des Programms wird die ungültige Adresse erkannt und ein Sprung auf eine Abbruchfunktion eingefügt. Diese Funktion beendet das Programm mit der MeldungSegmentation fault.
- ☐ Das Betriebssystem erkennt die ungültige Adresse bei der Weitergabe des Befehls an die CPU (partielle Interpretation) und leitet eine Ausnahmebehandlung ein.

Frage 64: Was versteht man unter RAID 0?

- ☐ Datenblöcke werden über mehrere Platten verteilt und repliziert gespeichert.
- ☐ Auf Platte 0 wird Parity-Information der Datenblöcke der Platten 1 - 4 gespeichert.
- ☐ Ein auf Flash-Speicher basierendes, extrem schnelles Speicherverfahren.
- ☑ Datenblöcke eines Dateisystems werden über mehrere Platten verteilt gespeichert.

Frage 65: Was versteht man unter Verklemmungsvorbeugung?

- ☐ Das System überprüft vor dem Belegen von Betriebsmitteln, ob ein unsicherer Zustand eintreten würde.
- ☐ Bei einem Zyklus im Betriebsmittelbelegungsgraph wird einer der Prozesse aus dem Zyklus terminiert.
- ☑ In einem System wird dafür gesorgt, dass eine der vier Voraussetzungen für Verklemmungen nicht eintreten kann.
- ☐ Wurde ein unsicherer Zustand im System erkannt, wird vorbeugend einer der beteiligten Prozesse abgebrochen, bevor die Verklemmung eintritt.

Frage 66: Was versteht man unter der Second-Chance- (oder Clock-) Policy?

- ☐ Eine Seitenersetzungsstrategie, bei der jeweils die älteste Seite ausgelagert wird.
- ☑ Eine Seitenersetzungsstrategie, die mit Hilfe eines Referenz-Bits eine einfacher zu implementierende Annäherung an LRU realisiert.
- ☐ Eine Scheduling-Strategie, bei der Prozesse vor der Verdrängung eine zweite Chance erhalten.
- ☐ Eine Speicherallokationsstrategie, bei der im Fehlerfall ein zweiter Allokationsversuch stattfindet.

Frage 67: Was versteht man unter einem Interrupt?

- ☐ Eine Signalleitung teilt dem Prozessor mit, dass er den aktuellen Prozess anhalten und auf das Ende der Unterbrechung warten soll.
- ☐ Der Prozessor wird veranlasst eine Unterbrechungsbehandlung durchzuführen. Der gerade laufende Prozess kann die Unterbrechungsbehandlung ignorieren.
- ☐ Durch eine Signalleitung wird der Prozessor veranlasst, die gerade bearbeitete Maschineninstruktion abzubrechen.
- ☑ Mit einer Signalleitung wird dem Prozessor eine Unterbrechung angezeigt. Der Prozessor sichert den aktuellen Zustand bestimmter Register, insbesondere des Programmzählers, und springt eine vordefinierte Behandlungsfunktion an.

Frage 68: Was versteht man unter einem Translation-Look-Aside-Buffer (TLB)?

- ☐ Einen Pufferspeicher des Compilers um den Übersetzungsvorgang zu beschleunigen (es werden die Codes der zuletzt übersetzen Statements vorgehalten).
- ☐ Einen speziellen Cache der CPU, der die zuletzt ausgeführten Maschinenbefehle zwischenspeichert (beschleunigt vor allem den Ablauf von Schleifen).
- ☐ Einen speziellen Cache der MMU, der den Inhalt der zuletzt angesprochenen Speicherzellen vorhält.
- ☑ Einen speziellen Cache der MMU, der Informationen aus den zuletzt genutzten Seitendeskriptoren vorhält.

Frage 69: Was versteht man unter einem Translation-Look-Aside-Buffer (TLB)?

- ☐ Einen speziellen Cache der MMU, der den Inhalt der zuletzt angesprochenen Speicherzellen vorhält.
- ☑ Einen speziellen Cache der MMU, der Informationen aus den zuletzt genutzten Seitendeskriptoren vorhält.
- ☐ Einen Pufferspeicher des Compilers um den Übersetzungsvorgang zu beschleunigen (es werden die Codes der zuletzt übersetzten Statements vorgehalten).
- ☐ Einen speziellen Cache der CPU, der die zuletzt ausgeführten Maschinenbefehle zwischenspeichert (beschleunigt vor allem den Ablauf von Schleifen).

Frage 70: Was versteht man unter einem unsicheren Zustand im Rahmen der Verklemmungsvermeidung?

- ☐ Bei einem unsicheren Zustand ist unklar, ob eine P-Operation blockieren wird oder nicht.
- ☐ Wenn sich ein System in einem unsicheren Zustand befindet, ist es verklemmt.
- ☑ Wenn sich ein System in einem unsicheren Zustand befindet, kann es zwar möglich sein, dass ein Teil der beteiligten Prozesse noch weiterläuft - früher oder später wird es aber mit Sicherheit zu einer Verklemmung kommen.
- ☐ Wenn ein System sich in einem unsicheren Zustand befindet, dürfen keine weiteren Ressourcen mehr angefordert werden, weil es sonst zu unkontrollierter Nebenläufigkeit in einem kritischen Abschnitt kommen kann.

Frage 71: Was versteht man unter virtuellem Speicher?

- ☑ Virtueller Speicher kann größer sein als der physikalisch vorhandene Arbeitsspeicher. Gerade nicht benötigte Speicherbereiche können auf Hintergrundspeicher ausgelagert werden.
- ☐ Virtueller Speicher kann größer sein als der physikalisch vorhandene Hintergrundspeicher zusammen mit dem Arbeitsspeicher. Über den tatsächlich existierenden Speicher hinausgehende Speicherbereiche sind dann nur scheinbar vorhanden.
- ☐ Virtueller Speicher sind die nicht vorhandenen Bereiche des physikalischen Adressraums.
- ☐ Virtueller Speicher kann dynamisch zur Laufzeit von einem Programm erzeugt werden (Funktion valloc(3)).

Frage 72: Was versteht man unter virtuellem Speicher?

- ☑ Virtueller Speicher kann größer sein als der physikalisch vorhandene Arbeitsspeicher. Gerade nicht benötigte Speicherbereiche können auf Hintergrundspeicher ausgelagert werden.
- ☐ Virtueller Speicher wird vom Compiler beim Binden angelegt.
- ☐ Virtueller Speicher sind die nicht vorhandenen Bereiche des physikalischen Adressraums.
- ☐ Virtueller Speicher kann dynamisch zur Laufzeit von einem Programm erzeugt werden (Funktion valloc(3)).

Frage 73: Was versteht man unter virtuellem Speicher?

- ☐ Adressierbarer Speicher in dem sich keine Daten speichern lassen, weil er physikalisch nicht vorhanden ist.
- ☑ Speicher der einem Prozess durch entsprechende Hardware (MMU) und durch Ein- und Auslagern von Speicherbereichen vorgespiegelt wird, aber möglicherweise größer als der verfügbare physikalische Hauptspeicher ist.
- ☐ Einen logischen Adressraum.
- ☐ Speicher, der nur im Betriebssystem sichtbar ist, jedoch nicht für einen Anwendungsprozess.

Frage 74: Was versteht man unter virtuellem Speicher?

- ☐ Speicher, in dem sich keine Daten speichern lassen, weil er physikalisch nicht vorhanden ist.
- ☐ Speicher, der nur im Betriebssystem sichtbar ist, jedoch nicht für einen Anwendungsprozess.
- ☑ Speicher, der einem Prozess durch entsprechende Hardware (MMU) und durch Ein- und Auslagern von Speicherbereichen vorgespiegelt wird, aber möglicherweise größer als der verfügbare physikalische Hauptspeicher ist.
- ☐ Unter einem Virtuellen Speicher versteht man einen physikalischen Adressraum, dessen Adressen durch eine MMU vor dem Zugriff auf logische Adressen umgesetzt werden.

Frage 75: Was versteht man unter virtuellem Speicher?

- ☑ Speicher, der einem Prozess durch entsprechende Hardware (MMU) und durch Ein- und Auslagern von Speicherbereichen vorgespiegelt wird, aber möglicherweise größer als der verfügbare physikalische Hauptspeicher ist.
- ☐ Speicher, der nur im Betriebssystem sichtbar ist, jedoch nicht für einen Anwendungsprozess.
- ☐ Unter einem Virtuellen Speicher versteht man einen physikalischen Adressraum, dessen Adressen durch eine MMU vor dem Zugriff auf logische Adressen umgesetzt werden.
- ☐ Virtueller Speicher kann dynamisch zur Laufzeit von einem Programm erzeugt werden (Funktion valloc(3)).

Frage 76: Welche Antwort stellt kein Attribut einer Datei eines UNIX-UFS-Dateisystems dar?

- ☑ Anzahl der symbolischen Links
- ☐ Zugriffsrechte
- ☐ Dateilänge
- ☐ Eigentümer

Frage 77: Welche Antwort trifft für die Eigenschaften eines UNIX/Linux Filedeskriptors zu?

- ☑ Ein Filedeskriptor ist eine prozesslokale Integerzahl, die der Prozess zum Zugriff auf eine Datei benutzen kann.
- ☐ Filedeskriptoren sind Zeiger auf Betriebssystem-interne Strukturen, die von den Systemaufrufen ausgewertet werden, um auf Dateien zuzugreifen.
- ☐ Ein Filedeskriptor ist eine Integerzahl, die über gemeinsamen Speicher an einen anderen Prozess übergeben werden kann, und von letzterem zum Zugriff auf eine geöffnete Datei verwendet werden kann.
- ☐ Beim Öffnen ein und derselben Datei erhält ein Prozess jeweils die gleiche Integerzahl als Filedeskriptor zum Zugriff zurück.

Frage 78: Welche Antwort trifft für die Eigenschaften eines UNIX/Linux-Filedeskriptors zu?

- ☐ Filedeskriptoren sind Zeiger auf Betriebssystemstrukturen, die von den System aufrufen ausgewertet werden, um auf Dateien zuzugreifen.
- ☐ Ein Filedeskriptor ist eine Integerzahl, die über gemeinsamen Speicher an einen anderen Prozess übergeben werden kann, und von letzterem zum Zugriff auf eine geöffnete Datei verwendet werden kann.
- ☑ Ein Filedeskriptor ist eine prozesslokale Integerzahl, die der Prozess zum Zugriff auf eine Datei, ein Gerät, einen Socket oder eine Pipe benutzen kann.
- ☐ Beim Öffnen ein und derselben Datei erhält ein Prozess jeweils die gleiche Integerzahl als Filedeskriptor zum Zugriff zurück.

Frage 79: Welche Art der Kommunikation liegt bei einem write-Aufruf an einem Socket (mit bereits aufgebauter TCP-Verbindung) vor?

- ☐ Synchrone IPC mit Client-seitiger Synchronisation
- ☐ Asynchrone IPC mit pufferblockierendem Senden
- ☑ Synchrone IPC mit sendeseitiger Synchronisation
- ☐ Asynchrone IPC mit unzuverlässigem Senden

Frage 80: Welche Aussage bezüglich der Freispeicherverwaltung mittels einer Bitliste ist falsch?

- ☐ Der zu verwaltende Speicher wird in Speichereinheiten gleicher Größe unterteilt.
- ☐ Zur Suche nach freiem Speicher kann es nötig sein, die gesamte Bitliste zu durchsuchen.
- ☑ Das Zusammenfassen von benachbarten freien Speichereinheiten ist besonders aufwändig.
- ☐ Je kleiner die Speichereinheiten sind, desto Länger ist die Bitliste.

Frage 81: Welche Aussage bezüglich der Freispeicherverwaltung mittels einer Bitliste ist richtig?

- ☐ Der zu verwaltende Speicher wird in Speichereinheiten unterschiedlicher Größe unterteilt.
- ☑ Zur Suche nach freiem Speicher kann es nötig sein, die gesamte Bitliste zu durchsuchen.
- ☐ Das Zusammenfassen von benachbarten freien Speichereinheiten ist besonders aufwändig.
- ☐ Je feiner die Granularität der Speichereinheiten ist, desto kürzer ist die Bitliste.

Frage 82: Welche Aussage bezüglich der Freispeicherverwaltung mittels einer nach Blockgröße sortierten verketteten Liste ist falsch?

- ☐ Der Suchaufwand ist konstant.
- ☑ Eine Verschmelzung benachbarter Freispeicherbereiche ist aufgrund der Sortierung besonders einfach möglich.
- ☐ Bei einer Speicheranforderung muss die Liste u.U. vollständig durchlaufen werden.
- ☐ Die Datenstruktur eignet sich besonders zur Realisierung einer worst-fit Strategie.

Frage 83: Welche Aussage ist bezüglich der Seitenersetzungsstrategie Least Recently Used (LRU) richtig?

- ☑ Als Auswahlkriterium für die Ersetzung einer Seite wird die Zeit seit dem letzten Zugriff auf die Seite verwendet.
- ☐ Zur Implementierung von LRU benötigt man eine sehr genaue Systemuhr.
- ☐ Die LRU-Strategie benötigt ein Referenzbit in jedem Eintrag der Seitenkacheltabelle.
- ☐ Die LRU-Strategie gewährleistet, dass immer die Seiten eingelagert sind, auf die in der Zukunft zugegriffen wird.

Frage 84: Welche Aussage ist in einem Monoprozessor-Betriebssystem falsch? Sept.09

- ☐ Es befindet sich maximal ein Prozess im Zustand laufend und damit in Ausführung auf dem Prozessor.
- ☐ Ist zu einem Zeitpunkt kein Prozess im Zustand laufend, so ist auch kein Prozess im Zustand bereit.
- ☑ Ein Prozess im Zustand blockiert muss warten, bis der laufende Prozess den Prozessor abgibt und kann dann in den Zustand laufend überführt werden.
- ☐ In den Zustand blockiert gelangen Prozesse in der Regel nur, wenn sie vorher den Zustand laufend innehatten.

Frage 85: Welche Aussage zu Fäden (Threads) ist falsch?

- ☐ Kernfäden können als virtuelle Prozessoren zur Ausführung von Benutzerfäden eingesetzt werden.
- ☐ Der Synchronisationsbedarf im Anwendungsprogramm kann von der Ablaufplanung der Kernfäden abhängen.
- ☐ Das Betriebssystem kann bei einem durch einen Benutzerfaden ausgelösten Seitenfehler nicht auf einen anderen Benutzerfaden umschalten.
- ☑ Das Betriebssystem führt Buch über Kernfäden und Benutzerfäden

Frage 86: Welche Aussage zu Schedulingverfahren ist richtig?

- ☐ Bei offline Scheduling werden alle laufenden Prozesse zunächst gestoppt. Dann ermittelt ein anderer Rechner (z. B. in einem Controller-Netzwerk in einem KFZ) einen neuen Ablaufplan nach dem die Prozesse weiter abgearbeitet werden.
- ☑ Bei deterministischem Scheduling kann unabhängig von der aktuellen Systemlast immer vorhergesagt werden, wann welcher Prozess die CPU zugeteilt bekommen wird.
- ☐ Wenn bei deterministischem Scheduling ein Prozess seine Rechenzeit überschreitet erhält er eine niedrigere Priorität.
- ☐ Bei kooperativem Scheduling führen Programme mit Endlosschleifen in jedem Fall dazu, dass kein anderer Prozess mehr von der CPU bedient werden kann.

Frage 87: Welche Aussage zu Semaphoren ist richtig?

- ☐ P- und V-Operationen werden am besten eingesetzt, um Nebenläufigkeit zwischen Signalbehandlungsfunktionen und dem eigentlichen Programmablauf zu synchronisieren.
- ☑ Eine P-Operation kann in einem Anwendungsprogramm unter UNIX durch normale C-Anweisungen nicht implementiert werden.
- ☐ Semaphore werden benutzt um in kritischen Abschnitten Interrupts zu sperren und so den gleichzeitigen Zugriff auf gemeinsame Datenstrukturen zu verhindern.
- ☐ Die V-Operation dekrementiert den Semaphor um 1 und deblockiert andere in einer V-Operation blockierte Prozesse.

Frage 88: Welche Aussage zu Speicherzuteilungsverfahren ist falsch?

- ☐ die worst-fit-Strategie kann einem mit der kürzesten Antwortzeit einen ausreichend großen Speicherbereich liefern
- ☑ best-fit arbeitet mit konstanter Komplexität und ist deshalb das beste Verfahren
- ☐ first-fit hat konstanten Aufwand bei der Verschmelzung von Lücken
- ☐ Beim Buddy-Verfahren gibt es keinen externen Verschnitt

Frage 89: Welche Aussage zu Speicherzuteilungsverfahren ist richtig?

- ☐ Die worst-fit-Strategie ist lediglich theoretisch interessant, da es in der Praxis nie sinnvoll ist, den am schlechtesten passenden Speicherplatz zuzuweisen.
- ☐ Best-fit ist in jedem Fall das beste Verfahren.
- ☑ Bei zunehmenden Blockgrößen nimmt beim Buddy-Verfahren im Mittel auch der interne Verschnitt zu.
- ☐ Buddy-Verfahren sind nur bei sehr leistungsfähigen Rechnern und großem Speicher einsetzbar, weil sie aufwändig in der Berechnung sind.

Frage 90: Welche Aussage zu Symbolic Links ist richtig?

- ☐ Symbolic Links sind nicht über Dateisystemgrenzen hinweg gültig.
- ☐ Wenn eine Datei gelöscht wird, werden auch alle auf sie verweisenden Symbolic Links automatisch mit aufgeräumt.
- ☑ Beim Zugriff auf einen Symbolic Link kann der Fehler No such file or directory auftreten obwohl der Link existiert.
- ☐ Die Anzahl der Symbolic Links, die auf eine Datei verweisen wird in deren Inode gespeichert.

Frage 91: Welche Aussage zu Zeigern ist richtig?

- ☑ Zeiger können verwendet werden, um in C eine call-by-reference Übergabesemantik nachzubilden.
- ☐ Die Übergabesemantik für Zeiger als Funktionsparameter ist call-by-reference.
- ☐ Ein Zeiger kann zur Manipulation von schreibgeschützten Datenbereichen verwendet werden.
- ☐ Zeiger vom Typ void* existieren in C nicht, da solche "Zeiger auf Nichts" keinen sinnvollen Einsatzzweck hätten.

Frage 92: Welche Aussage zu einem RAID-1-Plattensystem ist richtig?

- ☐ Der Schreibzugriff auf ein RAID-1-Plattensystem ist schneller, da mehrere Platten gleichzeitig beauftragt werden können.
- ☐ Die Paritätsinformation wird gleichmäßig über alle beteiligten Platten verteilt.
- ☐ Es müssen mindestens drei Festplatten an einem RAID-1-Verbund beteiligt sein.
- ☑ Ein aus drei Festplatten bestehendes RAID-1-Plattensystem kann den Ausfall zweier Festplatten ohne Datenverlust verkraften.

Frage 93: Welche Aussage zum Aufbau einer Kommunikationsverbindung zwischen einem Client und Server über eine Socket-Schnittstelle ist richtig?

- ☐ Der Server erzeugt einen Socket und ruft anschließend listen auf der Client kann daraufhin mit connect eine Verbindung herstellen und Daten übertragen.
- ☑ Der Server richtet am Socket eine Warteschlange für ankommende Verbindungen ein und kann dann mit accept eine konkrete Verbindung annehmen. Accept blockiert so lange die Warteschlange leer ist bzw. bis der Client seinerseits eine Verbindung mit connect aufbaut.
- ☐ Der Client kann erst connect aufrufen nachdem der Server accept aufgerufen hat - vorher würde der connect-Versuch mit connection rejected abgewiesen werden.
- ☐ Der Server signalisiert durch den Aufruf von connect, dass er zur Annahme von Verbindungen bereit ist, ein Client kann dies durch accept annehmen.

Frage 94: Welche Aussage zum Thema Ablaufplanung ist richtig?

- ☑ Bei der FCFS-Strategie kann es aufgrund des Konvoieffekts zu hohen Antwortzeiten kommen.
- ☐ Offline-Einplanungsverfahren eignen sich vor allem für den Einsatz auf mobilen Geräten, da diese auch ohne Internetverbindung arbeiten können.
- ☐ In Echtzeitsystemen kommt es auf maximalen Durchsatz an, weshalb hier ausschließlich nicht-unterbrechbare Schedulingverfahren verwendet werden.
- ☐ Asymmetrische Einplanungsverfahren zielen auf eine optimale Behandlung von Prozessmengen, die sich in E/A- und CPU-Intensität stark voneinander unterscheiden.

Frage 95: Welche Aussage zum Thema Betriebsarten ist richtig?

- ☐ Beim Stapelbetrieb können keine globalen Variablen existieren, weil alle Daten im Stapel-Segment (Stack) abgelegt sind.
- ☐ Echtzeitsysteme findet man hauptsächlich auf großen Serversystemen, die eine enorme Menge an Anfragen zu bearbeiten haben.
- ☑ Mehrzugangsbetrieb ist nur in Verbindung mit CPU- und Speicherschutz sinnvoll realisierbar.
- ☐ Mehrprogrammbetrieb ermöglicht die simultane Ausführung mehrerer Programme innerhalb desselben Prozesses.

Frage 96: Welche Aussage zum Thema Hard-Links ist falsch?

- ☐ Auf jede Datei existiert mindestens ein Hard-Link
- ☐ Auf jedes Verzeichnis existieren mindestens zwei Hard-Links
- ☐ Jeder Hard-Link besitzt in seinem Kontext einen eindeutigen Namen
- ☑ Hard-Links dürfen nur vom Systemadministrator angelegt werden

Frage 97: Welche Aussage zum Thema Kooperatives Scheduling ist richtig?

- ☐ Kooperatives Scheduling ist in Mehrbenutzersystemen unproblematisch, so lange sich die Benutzer des Systems kennen und bereit sind, sich kooperativ zu verhalten.
- ☑ Bei kooperativem Scheduling kann ein fehlerhaftes Programm zur Blockade des Gesamtsystems führen.
- ☐ Kooperatives Scheduling wird vor allem in Echtzeitsystemen eingesetzt, da durch die explizite Abgabe des Prozessors in den verschiedenen Anwendungen die Umschaltzeitpunkte zur Laufzeit immer vorausberechnet werden können.
- ☐ Kooperatives Scheduling ist in Verbindung mit virtuellem Speicher gut einsetzbar, weil durch die immer wieder auftretenden Seitenfehler automatisch die CPU freigegeben wird.

Frage 98: Welche Aussage zum Thema Kooperatives Scheduling ist richtig?

- ☐ Kooperatives Scheduling ist in Mehrbenutzersystemen unproblematisch, so lange sich die Benutzer des Systems kennen und bereit sind, sich kooperativ zu verhalten.
- ☐ Bei kooperativem Scheduling wird Prozessen die sich nicht kooperativ verhalten (z. B. in einer Endlosschleife rechnen) zwangsweise der Prozessor entzogen.
- ☐ Kooperatives Scheduling ist im Mehrprogrammbetrieb gut einsetzbar, weil das Betriebssystem im Rahmen der Systemaufrufe der beteiligten Prozesse Prozessumschaltungen vornehmen kann und damit eine Monopolisierung der CPU durch einen Prozess in jedem Fall automatisch verhindert wird.
- ☑ Programme, die unter kooperativem Scheduling zum Einsatz kommen, müssen auf diese Situation entsprechend vorbereitet sein (z. B. durch Einstreuen von regelmäßigen Aufrufen an das Betriebssystem), wenn eine Monopolisierung der CPU vermieden werden soll.

Frage 99: Welche Aussage zum Thema Monitor ist falsch?

- ☐ Bei einem Hoareschen Monitor kann die Neuauswertung der Wartebedingung entfallen, weil kein anderer Faden nach der Signalisierung den Monitor betreten konnte.
- ☐ pthread_wait und pthread_broadcast realisieren die wait- und signal-Primitiven Hansen'scher Monitore
- ☑ pthread_wait und pthread_signal realisieren Hoare'schen Monitore
- ☐ bei Hansenschen Monitoren entscheidet ausschließlich der Scheduler wer nach einer Monitorfreigabe den Monitor als nächstes betritt.

Frage 100: Welche Aussage zum Thema Monitor ist richtig?

- ☐ bei einem Hoare'schen Monitor muss die Wartebedingung nach dem wait in jedem Fall neu ausgewertet werden, weil zwischenzeitlich ein anderer Prozess die Bedingung bereits wieder entkräftet haben kann.
- ☑ pthread_wait und pthread_broadcast realisieren die wait- und signal-Primitiven Hansen'scher Monitore
- ☐ bei Hansen'schen Monitoren kann das Anwendungsprogramm gezielt entscheiden, welcher wartende Prozess den Monitor nach der Freigabe betreten darf.
- ☐ wenn in einem Monitor die Wartebedingung nicht erfüllt ist, aber davon ausgegangen werden kann, dass die Wartezeit sehr kurz sein wird, kann auf die Freigabe des Monitors verzichtet werden und kurzzeitig aktiv gewartet werden.

Frage 101: Welche Aussage zum Thema Prozesse/Threads ist richtig?

- ☐ Beim federgewichtigen Prozess bilden Adressraum und Prozess eine Einheit.
- ☑ Leichtgewichtige Prozesse müssen vom Betriebssystem speziell unterstützt werden.
- ☐ Eine gleichzeitige Ausführung mehrerer schwergewichtiger Prozesse auf verschiedenen Prozessoren ist nicht möglich.
- ☐ Die Einlastung eines federgewichtigen Prozesses ist eine privilegierte Operation und erfordert Unterstützung des Betriebssystems.

Frage 102: Welche Aussage zum Thema Prozesseinplanung ist falsch?

- ☑ Bei Stapelbetrieb ist es ein Hauptziel, die Anzahl fertiggestellter Prozesse pro vorgegebene Zeiteinheit zu maximieren.
- ☐ Hohe Systemlast kann bei Echtzeitbetrieb zu einer langsameren Bearbeitung von Prozessen führen.
- ☐ Bei interaktivem Betrieb soll vor allem die Antwortzeit minimiert werden.
- ☐ In einem System mit einer Mischung aus Echtzeitbetrieb und interaktivem Betrieb müssen in jedem Fall alle Echtzeitaufgaben erledigt sein, bevor Prozesse des interaktiven Betriebs den Prozessor zugeteilt bekommen.

Frage 103: Welche Aussage zum Thema Prozesseinplanung ist falsch?

- ☐ Bei Stapelbetrieb ist es ein Hauptziel, die Anzahl fertiggestellter Prozesse pro vorgegebener Zeiteinheit zu maximieren.
- ☐ Bei interaktivem Betrieb soll vor allem die Antwortzeit minimiert werden.
- ☐ Hohe Systemlast kann bei Echtzeitbetrieb zu einer langsameren Bearbeitung von Prozessen führen.
- ☑ Bei allen genannten Betriebsarten wird immer auf eine faire Behandlung aller Prozesse geachtet.

Frage 104: Welche Aussage zum Thema Prozesseinplanung ist richtig?

- ☑ Preemptive Schedulingstrategien bilden die Grundlage zur Implementierung von CPU-Schutz.
- ☐ Kooperative Schedulingstrategien sind vor allem für Mehrbenutzersysteme geeignet.
- ☐ Probabilistische Strategien erlauben eine exakte Vorhersage der CPU-Auslastung
- ☐ Für deterministisches Scheduling ist kein Wissen über das Laufzeitverhalten der beteiligten Prozesse erforderlich.

Frage 105: Welche Aussage zum Thema Prozesszustände ist falsch?

- ☐ Nach seiner Beendigung geht ein Prozess in den Zustand beendet über. In diesem Zustand wird ein Prozess auch als Zombie-Prozess bezeichnet.
- ☐ Es können sich maximal so viele Prozesse im Zustand laufend befinden, wie physikalische Prozessorkerne im System existieren.
- ☑ Nach Wegfall der Blockadebedingung geht ein Prozess direkt in den Zustand laufend über.
- ☐ Ein Prozess kann nur durch eigene Aktivität in den Zustand blockiert gelangen.

Frage 106: Welche Aussage zum Thema Aktives Warten" ist richtig?

- ☐ Aktives Warten vergeudet gegenüber passivem Warten immer CPU-Zeit
- ☐ Auf Mehrprozessorsystemen ist aktives Warten unproblematisch und deshalb dem passiven Warten immer vorzuziehen
- ☐ Aktives Warten sollte bei einer nicht-verdrängenden Scheduling-Strategie auf einem Monoprozessorsystem dem passiven Warten vorgezogen werden
- ☑ Zur Implementierung einer Schlossvariable mit aktivem Warten ist keine Unterstützung durch das Betriebssystem notwendig.

Frage 107: Welche Aussage zum Thema Aktives Warten ist richtig?

- ☐ Aktives Warten vergeudet gegen über passivem Warten immer CPU-Zeit
- ☐ Auf Mehrprozessorsystemen ist aktives Warten unproblematisch und deshalb dem passiven Warten immer vorzuziehen
- ☑ Aktives Warten darf bei nicht-verdrängenden Scheduling-Strategien auf einem Monoprozessorsystem nicht verwendet werden
- ☐ Bei verdrängenden Scheduling-Strategien verzögert aktives Warten nur den betroffenen Prozess, behindert aber nicht andere

Frage 108: Welche Aussage zur Verklemmungsvermeidung (deadlock avoidance) ist richtig?

- ☐ Bei Verklemmungsvermeidung wird dafür gesorgt, dass eine der vier notwendigen Bedingungen für Verklemmungen nicht auftreten kann.
- ☐ Das System überprüft vor dem Freigeben von Betriebsmitteln, ob ein unsicherer Zustand eintreten würde.
- ☑ Mit Hilfe des Bankiers-Algorithmus wird beim Belegen eines Betriebsmittels geprüft, ob mit erfolgreicher Belegung ein unsicherer Zustand eintreten würde.
- ☐ Bei einem Zyklus im erweiterten Betriebsmittelgraph liegt ein unsicherer Zustand vor.

Frage 109: Welche Aussage über das aktuelle Arbeitsverzeichnis (Current Working Directory) trifft zu?

- ☐ Mit dem Systemaufruf chdir() kann das aktuelle Arbeitsverzeichnis durch den Vaterprozess verändert werden.
- ☐ Das aktuelle Arbeitsverzeichnis erbt der Prozess vom aktuellen Benutzer
- ☐ Besitzt ein UNIX-Prozess kein Current Working Directory, so beendet sich der Prozess mit einem Segmentation Fault.
- ☑ Pfadnamen, die nicht mit dem Zeichen / beginnen, werden relativ zu dem aktuellen Arbeitsverzeichnis interpretiert

Frage 110: Welche Aussage über den Rückgabewert von fork() ist richtig?

- ☐ Der Kind-Prozess bekommt die Prozess-ID des Vater-Prozesses.
- ☐ Im Fehlerfall wird im Kind-Prozess -1 zurückgeliefert.
- ☐ Der Rückgabewert ist in jedem Prozess (Kind und Vater) jeweils die eigene Prozess-ID.
- ☑ Dem Vater-Prozess wird die Prozess-ID des Kind-Prozesses zurückgeliefert.

Frage 111: Welche Aussage über den Rückgabewert von fork() ist richtig?

- ☐ Im Fehlerfall wird im Sohn-Prozess -1 zurückgeliefert.
- ☑ Dem Vater-Prozess wird die Prozess-ID des Sohn-Prozesses zurückgeliefert.
- ☐ Der Sohn-Prozess bekommt die Prozess-ID des Vater-Prozesses.
- ☐ Bei erfolgreicher Ausführung kehrt fork() nicht zum Vater-Prozess zurück.

Frage 112: Welche Aussage über den Rückgabewert von wait() ist richtig?

- ☐ Der Systemaufruf wait() wartet auf die Eingabe eines beliebigen Zeichens von der Tastatur, der entsprechende ASCII-Wert wird als Rückgabewert geliefert.
- ☐ Im Fehlerfall wird eine Fehlernummer ungleich -1 zurückgeliefert.
- ☐ Dem Sohn-Prozess wird die Prozess-ID des Vater-Prozesses zurückgeliefert.
- ☑ Beim Beenden eines Sohn-Prozesses wird dessen Prozess-ID zurückgeliefert.

Frage 113: Welche Aussage über die Prozesszustände ist in einem Monoprozessor-Betriebssystem richtig?

- ☑ Es befindet sich zu einem Zeitpunkt maximal ein Prozess im Zustand laufend.
- ☐ Ist zu einem Zeitpunkt kein Prozess im Zustand bereit, so ist auch kein Prozess im Zustand laufend.
- ☐ Ein Prozess im Zustand blockiert muss warten, bis der laufende Prozess den Prozessor abgibt und kann dann in den Zustand laufend überführt werden.
- ☐ In den Zustand blockiert gelangen Prozesse nur aus dem Zustand bereit.

Frage 114: Welche Aussage über die Prozesszustände ist in einem Monoprozessor-Betriebssystem richtig? Sept. 08

- ☐ Ein Prozess im Zustand blockiert muss warten, bis der laufende Prozess den Prozessor abgibt und wird dann unmittelbar in den Zustand laufend überführt.
- ☐ Muss ein Prozess an einer Ein-, Ausgabeoperation warten, wird er vom Zustand laufend in den Zustand bereit überführt.
- ☑ Findet gerade keine Prozessumschaltung statt und ist kein Prozess im Zustand laufend, so ist auch kein Prozess im Zustand bereit.
- ☐ Es befinden sich bis zu zwei Prozesse im Zustand laufend und damit in Ausführung auf dem Prozessor (Vordergrund- und Hintergrundprozess).

Frage 115: Welche Daten werden typischerweise nicht im UNIX-Prozesskontrollblock gespeichert?

- ☐ offenen Dateien
- ☐ aktuelles Arbeitsverzeichnis
- ☐ UID des Eigentümers
- ☑ Homeverzeichnis des Eigentümers

Frage 116: Welche Information wird nicht im Dateikopf (Inode) eines typischen UNIX-Dateisystems gespeichert?

- ☐ Datum und Zeit der letzten Änderung
- ☐ Datum und Zeit des letzten Zugriffs
- ☑ Nummer des Inodes
- ☐ Zugriffsrechte für den Dateibesitzer

Frage 117: Welche Problematik wird durch das Philosophenproblem beschrieben?

- ☐ Ein Erzeuger und ein Verbraucher greifen gleichzeitig auf gemeinsame Datenstrukturen zu.
- ☐ Mehrere Prozesse greifen lesend und schreibend auf gemeinsame Datenstrukturen zu.
- ☐ Exklusive Bearbeitung durch mehrere Bearbeitungsstationen.
- ☑ Gleichzeitiges Belegen mehrerer Betriebsmittel.

Frage 118: Welche Seitennummer und welcher Versatz gehören bei einer Seitengröße von 1024 Bytes zu folgender logischer Adresse: 0xcafe?

- ☑ Seitennummer 0x32, Versatz 0x2fe
- ☐ Seitennummer 0xc, Versatz 0xafe
- ☐ Seitennummer 0x19, Versatz 0x2fe
- ☐ Seitennummer 0xca, Versatz 0xfe

Frage 119: Welche Seitennummer und welcher Versatz gehören bei einer Seitengröße von 2048 Bytes zu folgender logischer Adresse: 0xbabe?

- ☐ Seitennummer 0x11, Versatz 0xabe
- ☐ Seitennummer 0xb, Versatz 0xabe
- ☑ Seitennummer 0x17, Versatz 0x2be
- ☐ Seitennummer 0xba, Versatz 0xbe

Frage 120: Welche der folgenden Aussagen bzgl. Threads ist falsch?

- ☑ Die Einlastung (das Dispatching) eines Threads ist eine privilegierte Operation und kann deshalb grundsätzlich immer nur durch das Betriebssystem vorgenommen werden.
- ☐ Das Betriebssystem ist nicht in der Lage, einen einzelnen User-Level Thread bei einer fehlerhaften Operation (z. B. Segmentation fault) gezielt abzubrechen.
- ☐ Ein Anwendungsprogrammierer kann die Schedulingstrategie für seine User-Level Threads selbst programmieren.
- ☐ Die Erzeugung eines Kernel-Level Threads ist teurer als die Erzeugung eines User-Level Threads.

Frage 121: Welche der folgenden Aussagen bzgl. Threads ist falsch?

- ☐ Wenn ein User-Level Thread im Rahmen einer read-Operation warten muss (blockiert wird), kann das Betriebssystem nicht auf einen anderen User-Level Thread des gleichen Prozesses umzuschalten.
- ☑ Die Einlastung (das Dispatching) eines Threads ist eine privilegierte Operation und kann deshalb grundsätzlich immer nur durch das Betriebssystem vorgenommen werden.
- ☐ Ein Anwendungsprogrammierer kann die Schedulingstrategie für seine User- Level Threads selbst programmieren.
- ☐ Die Erzeugung eines Kernel-Level Threads ist teurer als die Erzeugung eines User-Level Threads.

Frage 122: Welche der folgenden Aussagen bzgl. Threads ist richtig?

- ☐ Die Einlastung (das Dispatching) eines Threads ist eine privilegierte Operation und kann deshalb grundsätzlich immer nur durch das Betriebssystem vorgenommen werden.
- ☐ Wenn ein User-Level Thread eine fehlerhafte Operation ausführt (z. B. Zugriff auf eine ungültige Adresse) wird er vom Betriebssystem abgebrochen. Andere User-Level-Threads der gleichen Anwendung können aber weiterarbeiten (Konzept der Fehlerisolation).
- ☑ Ein Anwendungsprogrammierer kann die Schedulingstrategie für seine User-Level Threads selbst programmieren.
- ☐ Die Erzeugung eines User-Level Threads ist teurer als die Erzeugung eines Kernel-Level Threads.

Frage 123: Welche der folgenden Aussagen zu statischem bzw. dynamischem Binden ist falsch?

- ☐ bei dynamischem Binden werden alle Adressen bis zum Zeitpunkt des Programmstarts aufgelöst
- ☐ bei dynamischem Binden können auch zum Übersetzungszeitpunkt alle bekannten Adressbezüge bereits vollständig aufgelöst werden
- ☐ beim statischen Binden werden alle Adressen zum Bindezeitpunkt aufgelöst
- ☑ statisch gebundene Programme können zum Ladezeitpunkt an beliebige Speicheradressen platziert werden

Frage 124: Welche der folgenden Aussagen zu statischem bzw. dynamischem Binden ist falsch?

- ☐ beim statischen Binden werden alle Adressen zum Bindezeitpunkt aufgelöst
- ☑ statisch gebundene Programme können zum Ladezeitpunkt an beliebige Speicheradressen platziert werden
- ☐ bei dynamischem Binden können auch zum Übersetzungszeitpunkt alle bekannten Adressbezüge bereits vollständig aufgelöst werden
- ☐ bei dynamischem Binden werden alle Adressen bis zum Zeitpunkt des Programmstarts aufgelöst

Frage 125: Welche der folgenden Aussagen zu statischem bzw. dynamischem Binden ist richtig?

- ☑ bei dynamischem Binden werden alle Adressen bis zum Zeitpunkt des Programmstarts aufgelöst
- ☐ bei dynamischem Binden müssen zum Übersetzungszeitpunkt alle Adressbezüge vollständig aufgelöst werden
- ☐ beim statischen Binden werden alle Adressen zum Ladezeitpunkt aufgelöst
- ☐ statisch gebundene Programme können zur Laufzeit durch das Nachladen neuer Programmodule (plug-ins) ergänzt werden

Frage 126: Welche der folgenden Aussagen zu statischem bzw. dynamischem Binden ist richtig?

- ☐ bei dynamischem Binden müssen zum Übersetzungszeitpunkt alle Adressbezüge vollständig aufgelöst werden
- ☐ beim statischen Binden werden alle Adressen zum Ladezeitpunkt aufgelöst
- ☑ dynamisch gebundene Programme können auch noch zur Laufzeit durch das Nachladen neuer Programmodule (plug-ins) ergänzt werden
- ☐ bei statischem Binden werden durch den Compiler alle Adressbezüge vollständig aufgelöst

Frage 127: Welche der folgenden Aussagen zum Thema Threads ist richtig?

- ☐ Zur Umschaltung von User-Threads verschiedener Prozesse ist kein Adressraumwechsel erforderlich.
- ☐ Kern-Threads teilen sich den kompletten Adressraum und verwenden daher denselben Stack.
- ☐ Auf Multiprozessorsystemen kann die Umschaltung von Kern-Threads ohne Mitwirken des Systemkerns erfolgen.
- ☑ Der Synchronisationsbedarf im Anwendungsprogramm kann von der Ablaufplanung der Kernfäden abhängen.

Frage 128: Welche der folgenden Aussagen über Schedulingverfahren ist richtig?

- ☐ Kooperatives Scheduling ist für Steuerungssysteme mit Echtzeitanforderungen völlig ungeeignet.
- ☑ Preemptives Scheduling ist für Mehrbenutzerbetrieb geeignet.
- ☐ Bei kooperativem Scheduling sind Prozessumschaltungen unmöglich wenn ein Prozess in einer Endlosschleife läuft, selbst wenn er bei jedem Schleifendurchlaufeinen Systemaufruf macht.
- ☐ Bei preemptivem Scheduling sind Prozessumschaltungen unmöglich, wenn ein Prozess in einer Endlosschleife läuft.

Frage 129: Welche der folgenden Informationen wird typischerweise in dem Seitendeskriptor einer Seite eines virtuellen Adressraums gehalten?

- ☐ die Position der Seite im virtuellen Adressraum
- ☐ die Identifikation des Prozesses, dem die Seite zugeordnet ist
- ☐ die Zuordnung zu einem Segment (Text, Daten, Stack, ...)
- ☑ Zugriffsrechte (z. B. lesen, schreiben, ausführen)

Frage 130: Welche der genannten Attribute sind in einem Inode eines UNIX-Dateisystems gespeichert?

- ☐ Dateityp, Eigentümer und Dateiname
- ☐ Gruppenzugehörigkeit, Anzahl der Verweise und bei Verzeichnissen zusätzlich die Anzahl der enthaltenen Unterverzeichnisse
- ☑ Eigentümer, Dateigröße und Dateityp
- ☐ Zeitpunkt des letzten Dateizugriffes, Erstellungszeitpunkt und aus Sicherheitsgründen der Zeitpunkt der letzten Rechteänderung.

Frage 131: Welcher UNIX-Systemaufruf wird bei der Verwendung von Sockets auf keinen Fall gebraucht?

- ☐ close()
- ☐ accept()
- ☑ mmap()
- ☐ shutdown()

Frage 132: Welches Attribut ist nicht im Inode eines UNIX-Dateisystems verzeichnet?

- ☐ Eigentümer
- ☑ Dateiname
- ☐ Dateityp (Verzeichnis, normale Datei, Spezialdatei)
- ☐ Zeitpunkt des letzten Dateizugriffes

Frage 133: Welches Problem in der Realität von Betriebssystemen beschreibt das Szenario der speisenden Philosophen?

- ☑ Mehrere Prozesse benötigen zwei wiederverwendbare Betriebsmittel (z. B. zwei Magnetband-Geräte oder einen Eingangs- und einen Ausgangs-Port), die sie aber nur einzeln anfordern können.
- ☐ Ein Benutzer startet ein Programm, das ein anderer Benutzer auch starten möchte.
- ☐ Ein Programm wird fünfmal gestartet und wechselt dann immer wieder zwischen Rechenphasen (= Denken) und Eingabephasen (= Essen).
- ☐ Fünf Prozesse (Philosophen) versuchen mit anderen Prozessen (Gabeln) Daten auszutauschen, wobei jeder Philosoph-Prozess immer nur mit seinen Nachbar-Gabel-Prozessen interagiert.

Frage 134: Welches der folgenden Verfahren ist nicht zur Verklemmungsvorbeugung in dem am Beispiel der speisenden Philosophen beschriebenen Verklemmungsszenario geeignet?

- ☐ Einer der Philosophen nimmt immer nur beide Stäbchen (oder Gabeln) gleichzeitig.
- ☑ Jeder Philosoph nimmt grundsätzlich immer zuerst das rechte und dann das linke Stäbchen. Dadurch wird eine Ordnung im System hergestellt, durch die keine Verklemmungen auftreten können.
- ☐ Alle Philosophen nehmen immer nur beide Stäbchen gleichzeitig.
- ☐ Ein Philosoph kann von seinem Nachbarn die Rückgabe eines Stäbchens fordern.

Frage 135: Welches der folgenden Verfahren ist zur Synchronisation des Zugriffs auf gemeinsame Daten in einem Multiprozessorsystem nicht geeignet?

- ☐ Binäre Semaphore.
- ☐ Spezialbefehle wie cas.
- ☐ Aktives Warten bis die Sperre aufgehoben wird.
- ☑ Sperren der Interrupts.

Frage 136: Welches der folgenden Verfahren trägt in der Praxis am besten dazu bei, Seitenfehler und ihre Auswirkungen zu minimieren?

- ☐ man ermittelt, welche der Seiten eines Prozesses in Zukunft am Längsten nicht angesprochen wird und lagert genau diese aus (OPT-Strategie)
- ☐ man übergibt Prozesse, die einen Seitenfehler verursachen der mittelfristigen Prozesseinplanung, damit sie in nächster Zeit nicht wieder aktiv werden
- ☐ man setzt eine Segmentierung in Kombination mit Seitenadressierung ein
- ☑ man lagert regelmäßig länger nicht genutzte Seiten aus und trägt sie in einem Freiseitenpuffer ein.

Frage 137: Wie funktioniert Adressraumschutz durch Eingrenzung?

- ☐ Der Lader positioniert Programme immer so im Arbeitsspeicher, dass unerlaubte Adressen mit nicht-existierenden physikalischen Speicherbereichen zusammenfallen.
- ☐ Begrenzungsregister legen einen Adressbereich im logischen Adressraum fest, auf den alle Speicherzugriffe beschränkt werden.
- ☑ Begrenzungsregister legen einen Adressbereich im physikalischen Adressraum fest, auf den alle Speicherzugriffe beschränkt werden.
- ☐ Die MMU kennt die Länge eines Segments und verhindert Speicherzugriffe darüber hinaus.

Frage 138: Wie groß ist die Seitenkacheltabelle zur Adressabbildung von logischen auf physikalische Adressen in einem System mit Seitenadressierung wenn ein Eintrag in der Seitenkacheltabelle 32 Bit groß ist, der virtuelle Adressraum 4 GigaByte umfasst und eine Speicherseite 1024 Byte groß ist?

- ☐ 1.048.576 Byte
- ☐ 4.194.304 Byte
- ☐ 512 bis 8.192 Byte
- ☑ 16.777.216 Byte

Frage 139: Wie groß ist die Seitenkacheltabelle zur Adressabbildung von logischen auf physikalische Adressen in einem System mit Seitenadressierung wenn ein Eintrag in der Seitenkacheltabelle 32 Bit groß ist, der virtuelle Adressraum 4 GigaByte umfasst und eine Speicherseite 4096 Byte groß ist?

- ☐ 512 bis 8.192 Byte
- ☐ 1.048.576 Byte
- ☐ 32 Byte
- ☑ 4.194.304 Byte

Frage 140: Wie groß ist die Seitentabelle zur Adressabbildung von logischen auf physikalische Adressen in einem System mit Seitenadressierung wenn ein Eintrag in der Seitentabelle 32 Bit groß ist, der virtuelle Adressraum 4 GiB umfasst und eine Speicherseite 8192 Byte groß ist?

- ☑ 2.097.152 Byte
- ☐ 4.194.304 Byte
- ☐ 512 bis 8.192 Byte
- ☐ 16.777.216 Byte

Frage 141: Wie groß ist typischerweise eine Seite bei Seitenadressierung?

- ☐ bis 16 Bytes
- ☑ 512 bis 8.192 Bytes
- ☐ 32.768 bis 8.388.608 Bytes
- ☐ 256 Bytes

Frage 142: Wie wird erkannt, dass eine Seite eines virtuellen Adressraums gerade ausgelagert ist?

- ☐ Die MMU erkennt bei der Adressumsetzung, dass die physikalische Adresse ungültig ist und löst einen Trap aus.
- ☐ Das Betriebssystem erkennt die ungültige Adresse vor Ausführung eines Maschinenbefehls und lagert die Seite zuerst ein bevor ein Fehler passiert.
- ☐ Im Seitendeskriptor steht bei ausgelagerten Seiten eine Adresse des Hintergrundspeichers und der Speichercontroller leitet den Zugriff auf den Hintergrundspeicher um.
- ☑ Bei ausgelagerten Seiten ist im Seitendeskriptor das present bit nicht gesetzt. Die MMU erkennt dies und löst bei einer Adressauflösung für solch eine Seite einen Trap aus.

Frage 143: Wie wird erkannt, dass eine Seite eines virtuellen Adressraums gerade ausgelagert ist? Sept. 08

- ☐ Bei Programmen, die in virtuellen Adressräumen ausgeführt werden sollen, erzeugt der Compiler speziellen Code, der vor Betreten einer Seite die Anwesenheit überprüft und ggf. die Einlagerung veranlasst.
- ☑ Im Seitendeskriptor wird ein spezielles Bit geführt, das der MMU zeigt, ob eine Seite eingelagert ist oder nicht. Falls die Seite nicht eingelagert ist, löst die MMU einen Trap aus.
- ☐ Bei ausgelagerten Seiten ist im Seitendeskriptor das present bit nicht gesetzt. Das Betriebssystem erkennt dies und löst bei einer Adressauflösung für solch eine Seite einen Trap aus.
- ☐ Im Seitendeskriptor steht bei ausgelagerten Seiten eine Adresse des Hintergrundspeichers und der Speichercontroller leitet den Zugriff auf den Hintergrundspeicher um.

Frage 144: Wodurch kann Nebenläufigkeit in einem System entstehen?

- ☐ durch Seitenflattern
- ☐ durch langfristiges Scheduling
- ☐ durch Compiler-Optimierungen
- ☑ durch Threads auf einem Monoprozessorsystem mit präemptivem Scheduling

Frage 145: Wodurch kann es zu Seitenflattern kommen?

- ☐ durch Programme, die eine Defragmentierung auf der Platte durchführen
- ☐ wenn sich zu viele Prozesse im Zustand blockiert befinden
- ☑ wenn bei einer nicht-verdrängenden Scheduling-Strategie die Zahl der von den Prozessen aktiv genutzten Seiten die Zahl der verfügbaren Seitenrahmen übersteigt
- ☐ wenn zu viele Prozesse im Rahmen der mittelfristigen Einplanung ausgelagert (swap-out) wurden

Frage 146: Wofür wird ein Freiseitenpuffer bei der Speicherverwaltung eingesetzt?

- ☑ Um bei einem Seitenfehler möglichst schnell einen freien Seitenrahmen (page frame) zum Einlagern der fehlenden Seite zur Verfügung zu haben.
- ☐ Um die Zahl der Auslagerungen von Seiten zu minimieren.
- ☐ Um den virtuellen Speicher schnell vergrößern zu können.
- ☐ Um Seiten schneller auslagern zu können.

Frage 147: Wozu benötigt man Bedingungsvariablen (condition variables)?

- ☐ Bei if-Abfragen in C-Programmen.
- ☑ Um in einem kritischen Abschnitt auf ein Ereignis zu warten und den kritischen Abschnitt Während der Wartezeit freizugeben.
- ☐ Zur Implementierung von Spinlocks.
- ☐ Zur Erkennung von Verklemmungen.

Frage 148: Wozu dient der Maschinenbefehl cas (compare-and-swap)?

- ☑ Um auf einem Multiprozessorsystem einfache Modifikationen an Variablen ohne Sperren implementieren zu können.
- ☐ Um bei Monoprozessorsystemen Interrupts zu sperren.
- ☐ Um bei der Implementierung von Schlossvariablen (Locks) aktives Warten zu vermeiden.
- ☐ Zur Realisierung einer effizienten Verdrängungssteuerung bei einseitiger Synchronisation.

2 Multiple-Choice

Frage 1: Gegeben sei folgendes Programmfragment:

```
static int a = 2012;
int f1 (const int *y)
        {
        static int b;
        int c;
        char *d = malloc(2407);
        int (*e)(const int *) = f1;
        y++;
        ...
}
```

Welche der folgenden Aussagen zum obigen Programmfragment sind richtig?

- ☑ a liegt im Datensegment.
- ☐ b liegt im Stacksegment.
- ☐ c ist mit dem Wert 0 initialisiert
- ☑ d ist ein Zeiger, der in den Heap zeigt.
- ☐ Die Speicherstelle, auf die d zeigt, verliert beim Rücksprung aus der Funktion f1() ihre Gültigkeit.
- ☑ e liegt im Stacksegment und zeigt in das Textsegment.
- ☐ Die Anweisung y++ führt zu einem Laufzeitfehler, da y konstant ist.
- ☑ y liegt im Stacksegment.

Frage 2: Welche der folgenden Aussagen zum Thema Einplanung sind richtig?

- ☐ Einplanungsverfahren lassen sich in drei Kategorien einteilen: federgewichtig, leichtgewichtig und schwergewichtig.
- ☑ Ein Prozess, der sich im Zustand laufend befindet, kann nicht direkt in den Zustand schwebend blockiert überführt werden.
- ☑ Prozesse im Zustand gestoppt sind der langfristigen Einplanung zuzuordnen.
- ☐ Ein Prozess kann sich in realen Systemen nie im Zustand beendet befinden, da bei seiner Terminierung sämtliche Betriebsmittel freigegeben werden und damit auch der Prozess selbst verschwindet.
- ☑ Für die mittelfristige Einplanung muss das Betriebssystem die Umlagerung (engl. swapping) von kompletten Programmen bzw. logischen Adressräumen unterstützen.
- ☐ Ein Prozess im Zustand erzeugt kann sich selbst durch die Ausführung des Systemaufrufes exec() in den Zustand bereit überführen.
- ☐ Prozesse im Zustand blockiert oder bereit können unmittelbar in den Zustand gestoppt überführt werden.
- ☐ Verdrängende Prozesseinplanung bedeutet, dass das Eintreten des erwarteten Ereignisses unmittelbar die Einlastung des wartenden Prozesses bewirkt.

Frage 3: Welche der folgenden Aussagen zum Thema Schedulingverfahren sind richtig?

- ☑ Kooperative Schedulingverfahren ermöglichen keinen wirksamen CPU-Schutz.
- ☑ Verdrängende Schedulingverfahren können nur mit Hilfe von Unterbrechungen realisiert werden.
- ☐ Asymmetrische Schedulingverfahren sind nur bei Systemen mit heterogenen Prozessoren (z. B. Grafikprozessoren und CPUs) einsetzbar.
- ☑ Symmetrische Schedulingverfahren sorgen für eine gleichmäßige Lastverteilung auf homogenen Multiprozessorsystemen.
- ☐ Online-Schedulingverfahren sind für den Einsatz in Rechnern ohne Netzwerkschnittstelle ungeeignet.
- ☑ Bei Offline-Schedulingverfahren wird der vollständige Ablaufplan bereits vor dem Start des Systems erstellt.
- ☐ Probabilistische Schedulingverfahren eignen sich für Systeme mit harten Echtzeitanforderungen.
- ☐ Deterministische Schedulingverfahren sind nur in der Theorie relevant, da die genaue Länge der CPU-Stöße nie vorhergesagt werden kann.

Frage 4: Welche der folgenden Aussagen zum Thema Speicherverwaltung sind richtig?

- ☐ Das Buddy-Verfahren verhindert externen Verschnitt.
- ☐ Das Buddy-Verfahren verhindert internen Verschnitt.
- ☑ Die Adressen zweier Buddies unterscheiden sich in genau einem Bit.
- ☑ Die Verschmelzung benachbarter Löcher ist beim Buddy-Verfahren besonders einfach.
- ☐ Benachbarte Löcher gleicher Größe können beim Buddy-Verfahren in jedem Fall verschmolzen werden.
- ☐ Bei einer Speicheranforderung muss bei Worst-Fit u.U. die gesamte Freispeicherliste durchlaufen werden.
- ☐ Der Verschmelzungsaufwand bei Best-Fit ist verglichen mit Worst-Fit erhöht.
- ☑ Ziel von First-Fit ist es, den Verwaltungsaufwand gering zu halten.

Frage 5: Welche der folgenden Aussagen zum Thema Synchronisation sind richtig?

- ☑ Semaphore sind für einseitige Synchronisation geeignet.
- ☑ Semaphore sind für mehrseitige Synchronisation geeignet.
- ☐ Auch nicht-blockierende Synchronisation kann zu Verklemmungen führen.
- ☑ Monitore sind Datentypen mit impliziten Synchronisationseigenschaften.
- ☐ Einseitige Synchronisation erfordert immer Betriebssystem-Unterstützung.
- ☐ Schlossvariablen sind teuer, da lock() immer zu einem Prozesswechsel führt.
- ☐ Die Synchronisation mit einem Signal-Handler unter LINUX erfolgt vorzugsweise über mehrseitige Synchronisationsverfahren.
- ☐ Sperren von Unterbrechungen ist das beste Verfahren zur Synchronisation mit Unterbrechungsbehandlungsfunktionen.

Frage 6: Welche der folgenden Aussagen zum Thema Threads sind richtig?

- ☑ Bei User-Threads ist die Schedulingstrategie keine Funktion des Betriebssystemkerns.
- ☑ User-Threads blockieren sich bei blockierenden Systemaufrufen gegenseitig.
- ☐ Kernel-Threads können Multiprozessoren nicht ausnutzen.
- ☐ Zur Umschaltung von Kernel-Threads ist kein Adressraumwechsel erforderlich.
- ☑ Die Umschaltung von User-Threads ist sehr effizient.
- ☐ Zu jedem Kern-Thread gehört ein eigener Adressraum.
- ☑ Bei Kernel-Threads ist die Schedulingstrategie meist durch das Betriebssystem vorgegeben.
- ☐ Die Umschaltung von Threads muss immer im Systemkern erfolgen (privilegierter Maschinenbefehl).

Frage 7: Welche der folgenden Aussagen zum Thema Threads sind richtig?

- ☑ Bei User-Threads ist die Schedulingstrategie keine Funktion des Betriebssystemkerns.
- ☐ Kern-Threads blockieren sich bei blockierenden Systemaufrufen gegenseitig.
- ☐ Kern-Threads können Multiprozessoren nicht ausnutzen.
- ☐ Zur Umschaltung von User-Threads ist ein Adressraumwechsel erforderlich.
- ☑ Die Umschaltung von User-Threads ist sehr effizient.
- ☐ Zu jedem Kern-Thread gehört ein eigener Adressraum.
- ☑ Bei Kern-Threads ist die Schedulingstrategie meist durch das Betriebssystem vorgegeben.
- ☑ Die Umschaltung von Kern-Threads muss immer im Systemkern erfolgen.

Frage 8: Welche der folgenden Aussagen zum Thema persistenter Datenspeicherung sind richtig?

- ☑ Bei kontinuierlicher Speicherung von Daten ist es unter Umständen mit enormem Aufwand verbunden, eine bestehende Datei zu vergrößern.
- ☐ Bei verketteter Speicherung dauert der wahlfreie Zugriff auf eine bestimmte Dateiposition immer gleich lang, wenn Cachingeffekte außer Acht gelassen werden.
- ☑ Bei indizierter Speicherung von Dateien müssen unter Umständen mehrere Blöcke geladen werden, bevor der Dateiinhalt gelesen werden kann.
- ☐ Extents finden aus Performanzgründen keine Anwendung in modernen Dateisystemen.
- ☑ Journaling-Dateisysteme garantieren, dass auch nach einem Systemausfall alle Metadaten wieder in einen konsistenten Zustand gebracht werden können.
- ☐ Journaling-Dateisysteme sind immun gegen defekte Plattenblöcke.
- ☐ Durch den Einsatz von mehreren Platten wird bei RAID 0 die Ausfallsicherheit des Gesamtsystems so stark erhöht, dass Datensicherungen überflüssig sind.
- ☑ Festplatten eignen sich besser für sequentielle als für wahlfreie Zugriffsmuster.