



7. „Skalierbar hinsichtlich der **Systemlast** ist ein System, wenn es Applikationen mit **neuen Dienstanforderungen** (*service requirements*) befriedigen kann.“

“A system is *scalable* with respect to *system load* if it can satisfy applications with *new service requirements*.”

korrekt

inkorrekt

8. „Das **Betriebsmittelvergabeprotokoll** (*resource allocation protocol*) **Prioritätshürde** (*priority ceiling*) ist **verklemmungsfrei**.“

“The *resource allocation protocol priority ceiling* is *deadlock free*.”

korrekt

inkorrekt

9. „Bei **B\*-Dateien** (*B\*-files*) kann es vorkommen, dass beim **Überlauf** eines Datencontainers **auch innere Knoten überlaufen**.“

“Using *B\*-file* it is possible that *internal nodes overflow as well* if a data container *overflows*.”

korrekt

inkorrekt

## Aufgabe 2 / Question 2

(1 +2 + 4 + 5 Punkte/marks)

1. „Geben Sie ein einfaches Beispiel dafür an, dass sich der **Taskzustand** einer mehrfädigen (*multi threaded*) Applikation vom **Threadzustand** eines ihrer PULTs **unterscheiden** kann!“

“Give a simple example showing that the *task state* of a multithreaded application can be *different* from the *thread state* of one of its PULTs.”

A PULT does a blocking system-call, the PULT is still “running” whereas its task is blocked.

2. „**Aktives Warten** (*busy waiting*) im Rahmen von Synchronisationen kann **schädlich**, manchmal aber auch **nützlich** sein. Nennen Sie **jeweils** ein Synchronisationsbeispiel dafür, dass aktives Warten nützlich bzw. schädlich ist und **begründen** Sie, wie es zu dieser Situation kommen kann!“

“*Busy waiting* within synchronizations can be *harmful*, but sometimes it can also be *useful*. Give an example for *each* situation, i.e. when busy waiting is harmful or useful, and *reason* why this situation can occur.”

Sinnvolles aktives Warten / *useful busy waiting*: Very short CS on a SMP

Begründung/*reason*: If overhead for blocking and unblocking needs so much time, that wasting CPU time due to busy waiting is more efficient

Schädliches aktives Warten / *harmful busy waiting*: Single processor system with strict priority scheduling

Begründung/*reason*: Suppose a low priority thread holds a spin-lock, in its CS it is preempted by a high priority process that also wants to enter the CS  $\Rightarrow$  indefinite busy waiting

3. „Zählen Sie die **vier verschiedenen Formen** von Aktivitätsimplementierungen auf und beschreiben Sie die wesentliche Charakteristik von **einer** dieser vier Formen!“  
*“Enumerate the **four different** activity implementations and describe the major characteristics of **one** of these four.”*

Process                      PULT                      KLT                      KMT (kernel mode thread)

Charakteristik/characteristics: A KMT always runs in kernel mode and it is coded in the KAS

4. „**Analysieren** Sie, ob die folgende Softwarelösung auf **Anwendungsebene** (*application level*) eine **gültige Lösung** eines wechselseitigen Ausschlussproblems darstellt, die **allen Anforderungen** (*requirements*) genügt.“  
*“Analyze whether the following software solution at **application level** is a **valid solution** of a mutual exclusion problem that fulfills **all requirements**.”*

```

/* global data declarations */
var flag:array[0..1] of boolean;      {initially flag[0]=flag[1]=false}
{thread 0}                            {thread 1}
repeat                                repeat
  while flag[1] = true do {nothing};   while flag[0] = true do {nothing};
  flag[0] := true;                     flag[1] := true;
  {critical section}                   {critical section}
  flag[0] := false;                   flag[1] := false;
  {remainder section}                 {remainder section}
forever                               forever

```

Analyse / analysis:

1. Exclusiveness: No, it can happen that both are in the CS
2. Portability: No, on a single processor busy waiting might lead to starvation
3. Progress: yes
4. Bounded Waiting: No (see b))

### Aufgabe 3 / Question 3

(2 + 3 + 1 + 4 + 2 Punkte/marks)

1. „System A besteht aus zwei Standardprozessoren, während System B aus nur einem, allerdings doppelt so schnellen Prozessor besteht. Welches der folgenden **Leistungsmaße** (*performance measures*) wird dabei von System A bzw. System B besser erfüllt, wenn nur zwei gleichlange Prozesse zu bearbeiten sind, die beide gleichzeitig bereit sind?“  
*“System A consists of two standard processors, whereas system B consists of just one processor with double speed. Which of the following **performance measures** will be fulfilled better by System A respectively by system B, when you have to execute only two processes with equal execution time being ready at the same time?”*

**Leistungsmaß / performance measure**

**besseres System / better system**

Mittlere Antwortzeit / average response time

A, both respond at t = 0

Mittlere Verweilzeit / average turnaround time

B, the first process completes earlier

2. „Erläutern Sie die die **Grundidee** des **Betriebsmittelvergabeprotokolls** (*resource allocation protocol*) „**nicht verdrängbarer kritischer Abschnitt**“ (*non-preemptive critical section*)! Welche Eigenschaften hat dieses Protokoll?“  
 “*Explain the **fundamental idea** of the **resource allocation protocol** “**non-preemptive critical section**”. Which characteristics does this protocol have?*”

As soon as the process has allocated its first resource, it gets the maximal system priority, i.e. it will not be pre-empted until it will release its last resource. NPCS is deadlock free, simple to implement, needs no a priori knowledge, but might hinder other higher priority processes that do not interfere with resource demanding ones.

3. „Zählen Sie eine **praktikable Vorschrift** auf, mit der man die **notwendige Verklemmungsbedingung** „**Hold And Wait**“ **verhindern** kann!“  
 “*Enumerate a **practicable rule** how to **prevent** the **necessary deadlock condition** “**Hold And Wait**.”*”

Allocate resource only according to some prescribed order

4. „Erläutern Sie, warum es bei einem System mit Verklemmungsvermeidung (*deadlock avoidance*) zu einer **Situation** kommen kann, in der man einem Prozess ein exklusives Betriebsmittel R **nicht** gibt, obwohl es zur Zeit **frei** wäre!“  
 “*Explain why a situation can occur in a system with deadlock avoidance, where an exclusive resource is **not** given to a process, even though this resource is currently **free**.*”

When a process P requesting a free resource would always get this resource, then P might be the cause that the system enters an unsafe state, i.e. in a worst case scenario, i.e. if all active processes would request all claimed resources, a deadlock would arise.

5. Mit **welchem Algorithmus** kann man **im allgemeinen Fall** Verklemmungsvermeidung implementieren, und **welchen Aufwand** hat dieser Algorithmus?“  
 “*With **which algorithm** can you implement deadlock avoidance in the **general case** and **which complexity** does this algorithm have?*”

Banker,  $O(n^2 \cdot m)$ ,  $n = \#$  processes,  $m = \#$  resource types

#### Aufgabe 4 / Question 4

(2 + 2 + 3 + 2 + 3 Punkte/marks)

1. „Erklären Sie den **Unterschied** zwischen der **Speicherbereinigung** (*garbage collection*) und der **Speicherkompaktifizierung** (*storage compaction*)!“  
 “*Explain the **difference** between **garbage collection** and **storage compaction**.*”

Garbage-collection frees no longer needed memory portions and adds these portion to the free list, reunifying them if possible.

Storage compactions moves all allocated memory pieces to low( or high) addresses in order to provide a maximal contiguous memory portion.

2. „Welche der Systemkomponenten **Adressraumverwaltung** (*address space management*) oder **physische Speicherverwaltung** (*physical memory management*) wird in einem **geschichteten System** (*layered system*) auf der **höheren Schicht** (*layer*) implementiert? **Begründen** Sie Ihre Ansicht!“

*“In a layered system, which of the system components **address space management** or **physical memory management** is implemented at the **higher layer**? **Reason** your opinion.”*

Component on higher layer: [Address space management](#)

Begründung / reason: [ASM needs memory management to provide virtual, logical or real address ranges as well as to establish the corresponding data structures](#)

3. „Zählen Sie mindestens **drei verschiedene Speicherverwaltungsverfahren** auf, bei denen **interner Verschnitt** (*internal fragmentation*) auftreten kann und **erklären Sie jeweils**, warum dabei interner Verschnitt auftritt!“

*“Enumerate at least **three different memory management schemes** that can cause **internal fragmentation** and **explain for each of them**, why there can be internal fragmentation.”*

Buddy system	(only 2 <sup>i</sup> sized memory units)
Slab Allocator	(only a set of predefined, though often used memory units)
Stack	(if 4 bytes aligned)
Paged virtual memory	(page frame = fixed memory unit)

4. „Erklären Sie, wie es in einem System mit **virtuellem Speicher** zum **Seitenflattern** (*page thrashing*) kommen kann.“

*“Explain how **page thrashing** can occur in a **virtual memory system**.”*

[Current multiprogramming degree is too high, i.e. the majority of all working sets is currently growing, e.g. because of concurrent phase transitions per working set.](#)

5. „Ein Prozess P sei hinsichtlich seines Referenzverhaltens in drei Phasen unterteilbar, z.B. Initialisierungs-, Berechnungs- und Terminierungsphase. **Wie** wirken sich diese Phasen auf den **Prozess P selbst**, aber auch **auf das Gesamtsystem** aus, wenn im System eine **lokale Seitenersetzungsstrategie** implementiert ist?“

*“A process P can be divided into three phases with respect to its reference behaviour, e.g. into an initialization-, a computing-, and a termination phase. **How** do these phases affect the **process P itself**, but also the **total system**, when a **local page replacement policy** is implemented?”*

[During a phase transition page faults/process are significantly higher than during a phase. Other processes are affected concerning their file access or paging activities.](#)

#### Aufgabe 5 / Question 5

(3 + 1 + 2 + 3 + 3 Punkte/marks)

1. „Erläutern Sie den Begriff **Metadaten** eines **Dateisystems** und geben Sie mindestens **vier typische Metadaten eines Unix/Linux Dateisystems** an!“

*“Explain the term **meta data** of a **file system** and enumerate at least **four typical meta data** of a **Unix/Linux file system**.”*

*Meta data of a file system:* describe the structure of the FS (file table versus directory tree), the size of the FS, # inode entries, # of data blocks, and other FS related information, i.e. size of data blocks, maximal file size ...

*Meta data of Unix/Linux FS:*

Boot-block, root-directory, inode-table, limit between inodes and data blocks, block size, ...

2. „Bei einer **erweiterbaren Hashdatei** wird stets die **aktuelle Generationszahl**  $gen_{max}$  verwendet. Wie kann man mit diesem  $gen_{max}$  ein Datenelement **finden**, welches zuvor mit einer **kleineren Generationszahl** in einen Datencontainer abgebildet worden war?“  
*“In an extensible hash file, you always use the current generation number  $gen_{max}$ . How can you find a data element that has been mapped previously to a data container with a smaller generation number using this  $gen_{max}$ ?”*

If the corresponding data container is still the same, then the pointer to the data container in the base vector is duplicated for higher gen numbers. If the data container has been split, then each element of this data container will be hashed again with the new gen number.

3. „Welche **Verzeichniseinträge** (*directory entries*) enthält ein **neues** Unix/Linux-Verzeichnis bei seiner Erzeugung?“  
*“Which **directory entries** does a new Unix/Linux directory contain upon creation?”*

. = directory it self

.. = parent directory

4. „Beschreiben Sie die **drei wesentlichen zeitintensiven Phasen** eines Plattenzugriffs und **ordnen** Sie diese Phasen der **Zeitdauer** nach, wenn die Blockgröße 0,5 KB beträgt!“  
*“Describe the **three major time intensive phases** of a disk access, and arrange them according to their **time length**, when the block size is 0.5 KB.”*

Disk arm movement  
4 – 14 ms

Rotational waiting time  
0.1 ms

Data transfer time (small)  
0.01 ms

5. „Entwerfen Sie eine **effiziente** Implementierung der Funktion **bzero()** (siehe unten)! Der Anwender soll in der Lage sein, einen sehr großen Puffer **buffer** mit Nullen zu füllen, von dem er aber nicht weiß, ob er vollständig benutzt wird, u.U. wird nur ein kleiner Bruchteil davon wirklich benötigt! (Sie können hierzu Pseudocode, eine detaillierte Beschreibung oder eine Kombination aus beiden angeben.)“  
*“Design an efficient implementation of **bzero()**. A user wants to be able to zero-fill a very large buffer **buffer**, however, he does not know, whether the complete buffer or only a small amount of this buffer will be used actually. (You can use pseudocode, a detailed description, or a combination.)”*

```
//fill this memory buffer with zeroes
void bzero(void* buffer, long buffersize)
```

```
// Use the zero page (often page frame 0) and
// use copy on write for each buffer-page
```