Sample solution of the written examination in

Operating Systems

February 12th 2024

Last name: _____

First name: _____

Student number: _____

Mit dem Bearbeiten dieser schriftlichen Prüfung (Klausur) bestätigen Sie, dass Sie diese alleine bearbeiten und dass Sie sich gesund und prüfungsfähig fühlen. Mit dem Erhalt der Aufgabenstellung gilt die Klausur als angetreten und wird bewertet.

By attending this written exam, you confirm that you are working on it alone and feel healthy and capable to participate. Once you have received the examination paper, you are considered to have participated in the exam, and it will be graded.

- Use the provided sheets. Do *not* use own paper.
- You are allowed to use a *self prepared*, *single sided DIN-A4 sheet* in the exam. Only *hand-written originals* are allowed, but no copies.
- Do *not* use a red pen.
- Time limit: *90 minutes*
- Turn off your mobile phones!

Grade: _____

Questions:	1	2	3	4	5	6	7	8	9	10	11	12	13	**Σ**
Maximum Points:	10	6	8	7	7	7	8	8	4	7	6	7	5	90
Achieved Points:														

1.0: 90.0-85.5, **1.3**: 85.0-81.0, **1.7**: 80.5-76.5, **2.0**: 76.0-72.0, **2.3**: 71.5-67.5,
2.7: 67.0-63.0, **3.0**: 62.5-58.5, **3.3**: 58.0-54.0, **3.7**: 53.5-49.5, **4.0**: 49.0-45.0, **5.0**: <45

Question 1)

Points: ..... of 10

1 Point (1) Describe what swapping is.
Process of storing and removing data to/from main memory from/into background memory (HDDs/SSDs).

1 Point (2) Explain what singletasking is.
At any given moment, only a single program can be executed.

1 Point (3) Describe what half multi-user operating systems are.
Different users can work with the system only one after the other, but the data and processes of the different users are protected from each other.

1 Point (4) Describe the difference between 8 bit, 16 bit, 32 bit, and 64 bit operating systems.
The bit number indicates the memory address length, with which the operating system works internally.

½ Point (5) Give the maximum amount of memory, a 32-bit architecture can address.
With 32-bit architectures, 2^{32} *memory addresses and therefore up to* $4,294,967,296$ *Bytes = 4 GB can be addressed.*

2 Points (6) Explain why multi-level paging and not single-level paging is used in 32-bit and 64-bit systems.
In 32-bit operating systems with a page size of 4 kB, the page table of each process can be 4 MB in size. For 64 bit operating systems, the page tables can be considerably larger. Multi-level paging reduces main memory usage as individual pages of the different levels can be moved to swap memory to free up memory capacity in main memory.

1 Point (7) Explain the event that causes a page fault exception.
A process tries to access a page, which is not located in the physical main memory.

1 Point (8) Give the name of the best page replacement strategy and describe how it works.
The optimal strategy is the best page replacement strategy. It replaces the page that will not be accessed for the longest time in the future. Sadly, it cannot be implemented.

1 Point (9) Describe the key message of Laszlo Belady's anomaly.
FIFO produces worse results for certain access patterns with increased memory.

½ Point (10) Give the name of the page replacement strategy that is implemented by most modern operating systems (Hint: It is not OPT and not random).
Clock / Second Chance.

Question 2)

Points: ..... of 6

Give a command that can be used to...

½ Point (1) modify the permissions of files or directories.
`chmod`

½ Point (2) print out the path of the present working directory in the shell.
`pwd`

½ Point (3) create a new directory.
`mkdir`

½ Point (4) create an empty file.
`touch`

½ Point (5) concatenate the content of different files or print out the content of a file.
`cat`

½ Point (6) print out lines from the end of a file in the shell.
`tail`

½ Point (7) print out lines from the beginning of a file in the shell.
`head`

½ Point (8) delete files or directories.
`rm`

½ Point (9) place a string in the shell.
`echo`

½ Point (10) create a link.
`ln`

½ Point (11) search a file for lines, which contain a search pattern.
`grep`

½ Point (12) terminate a process.
`kill`

Question 3)

Points: ..... of 8

½ Point (1) Name one persistent data storage.

HDD, SSD, USB drive, CF/SD card, Magnetic tape, Floppy, CD/DVD,...

½ Point (2) Name one non-persistent data storage.

Cache and Registers (SRAM), Main memory (DRAM).

½ Point (3) The storage of computer systems is distinguished into the categories primary, secondary, and tertiary storage. Give the name of the category or categories the CPU can access directly.

Primary storage.

1 Point (4) Give the name of the category or categories of subtask (3) the CPU can only access via a controller.

Secondary storage and tertiary storage.

1½ Points (5) Name one example for each category of subtask (3).

Primary storage: Register, Cache, Main memory.
Secondary storage: HDD, SSD, CF/SD flash storage card.
Tertiary storage: CD/DVD drive, magnetic tape.

1 Point (6) Describe what near-line storage is.

Near-line storage is tertiary storage that is automatically and without human intervention connected to the system (e.g. tape library).

1 Point (7) Describe what off-line storage is.

Off-line storage is tertiary storage that is stored in cabinets or storage rooms and must be connected manually to the system.

2 Points (8) Name one advantage and one drawback of NAND memory compared with NOR memory.

Benefits:

- *Lesser data lines $\Longrightarrow$ requires less surface area as NOR memory*
- *Lower manufacturing costs compared with NOR flash memory*
- *Available with bigger storage capacity than NOR flash memory*
- *Lesser power consumption compared with NOR memory*

Drawbacks:

- *No random access $\Longrightarrow$ Poorer latency compared with NOR memory*
- *Read and write operations can only be carried out for entire pages*
- *Erase operations can only be carried out for entire blocks*

Question 4)

Points: ..... of 7

1 Point (1) Explain the effect when executing this command in the command-line shell:

```
$ chmod 777 script.sh
```

The user (owner), the members of the assigned group, and all other users that have access to the system get permissions to read, write and execute the file `script.sh`.

```
-rwxrwxrwx
```

1 Point (2) Explain the effect when executing this command in the command-line shell:

```
$ chmod 544 script.sh
```

The user (owner) gets permissions to read and execute, and the members of the assigned group, and all other users that have access to the system get the permission to read the file `script.sh`.

```
-r-xr--r--
```

1 Point (3) Explain the effect when executing this command in the command-line shell:

```
$ chmod 000 script.sh
```

The user (owner), the members of the assigned group, and all other users that have access to the system get all permissions to read, write and execute the file `script.sh` *taken away.*

```
----------
```

1 Point (4) Explain the effect when executing this command in the command-line shell:

```
$ chmod u-x folder
```

The user (owner) gets the permission to execute the directory `folder` *taken away. As consequence, the user cannot make* `folder` *its new present working directory.*

```
drw-r-x---
```

½ Point (5) For executing a program written in the language C one requires a(n)...

☐ Booster	☐ Interpreter	☐ All of them
☒ Compiler	☐ Mixer	☐ None of them

½ Point (6) For executing a program written in the language Python one requires a(n)...

☐ Booster	☒ Interpreter	☐ All of them
☐ Compiler	☐ Mixer	☐ None of them

1 Point (7) Explain the purpose of the Page-Table Base Register (PTBS).

It stores the address where the page table of the current process starts.

1 Point (8) Explain the purpose of the Page-Table Length Register (PTLR).

It stores the length of the page table of the current process.

Question 5)

Points: ..... of 7

½ Point (1) Local variables of functions reside inside the...

☐ Heap ☒ Stack ☐ Text Segment

½ Point (2) Call parameters and return addresses of functions reside inside the...

☐ Heap ☒ Stack ☐ Text Segment

½ Point (3) Constants and variables which get values assigned in global declarations (outside of functions) reside inside the...

☒ Heap ☐ Stack ☐ Text Segment

½ Point (4) Environment variables of a process reside inside the...

☐ Heap ☒ Stack ☐ Text Segment

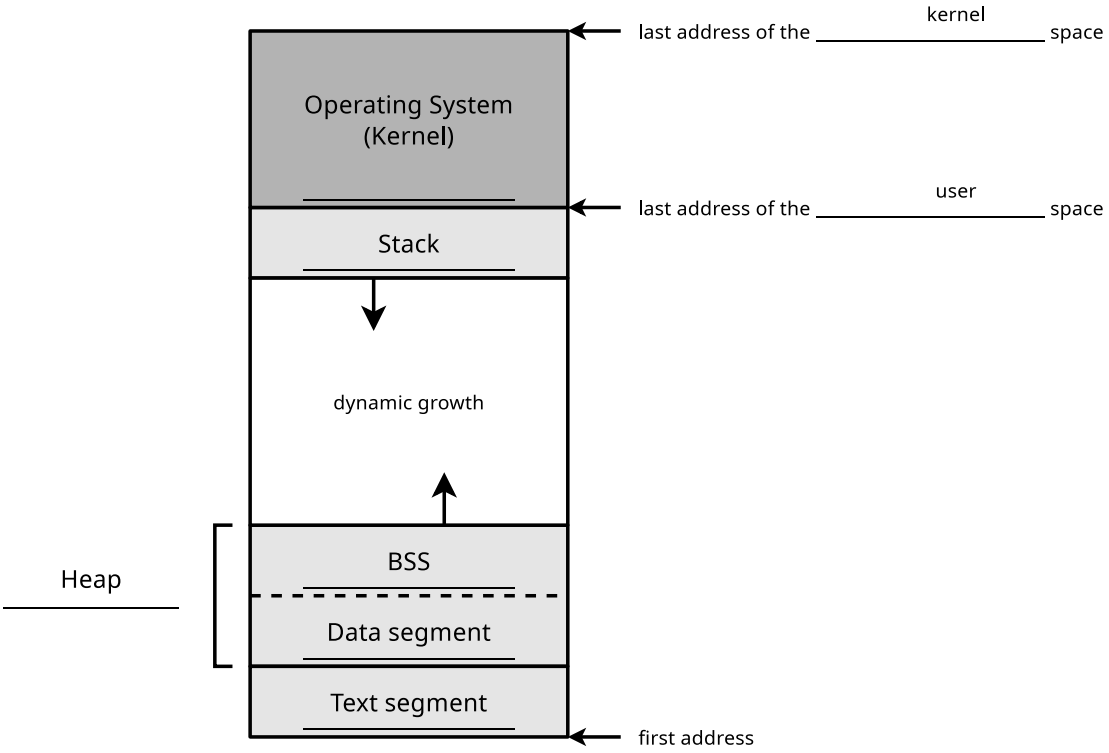
½ Point (5) The machine code of a process resides inside the...

☐ Heap ☐ Stack ☒ Text Segment

½ Point (6) Command line arguments of a process reside inside the...

☐ Heap ☒ Stack ☐ Text Segment

4 Points (7) The figure shows the structure of a UNIX process in memory. Fill in the missing labels (technical terms) of the process-related data and the missing information about the content of this data.



Question 6)

Points: ..... of 7

1 Point (1) Describe which information inodes store.
An inode stores a file's metadata, except the file name.

1 Point (2) Describe what a cluster in the file system is.
File systems address clusters and not blocks of the storage device. Each file occupies an integer number of clusters.

½ Point (3) Give one example for an absolute path name.
Every absolute path name begins with the root symbol (/).
Example: `/var/log/messages`

½ Point (4) Name one Linux file system that implements block addressing.
Minix, ext2, ext3

½ Point (5) Name one Linux file system that implements journaling.
ext3, ext4, ReiserFS, XFS, JFS

½ Point (6) Name one Linux file system that implements extents.
ext4, JFS, XFS, btrfs

½ Point (7) Name one Windows file system that implements the file allocation table.
FAT12, FAT16, FAT32, exFAT

½ Point (8) Name one Windows file system that implements journaling.
NTFS

½ Point (9) Name one Windows file system that implements extents.
NTFS

½ Point (10) Name one file system that implements copy-on-write.
ZFS, btrfs, ReFS

1 Point (11) Describe what the master file table is.
The file system NTFS contains a master file table (MFT). It contains the references of the files to the clusters. It also contains the metadata of the files like file size, file type, date of creation, date of last modification and possibly the file content, etc. The content of small files ≤ 900 Bytes is stored directly in the MFT.

Question 7)

Points: ..... of 8

1 Point

(1) Explain what a zombie process is.

A zombie process has completed execution (via the system call `exit`) but its entry in the process table exists until the parent process has fetched (via the system call `wait`) the exit status (return code). Its PID can not yet be assigned to a new process.

3 Points

(2) The following C source code creates a child process.

```
1  #include <stdio.h>
2  #include <unistd.h>
3  #include <stdlib.h>
4
5  void main() {
6    int returnvalue = fork();
7
8    if (returnvalue < 0) {
9      printf("Error.\n");
10     exit(1);
11   }
12   else if (returnvalue > 0) {
13     printf("Parent.\n");
14     exit(0);
15   }
16   else {
17     printf("Child.\n");
18     exit(0);
19   }
20 }
```

Give the value of the `returnvalue` variable for the child process and for the parent process. In your answer, explain the importance of the return value in the parent process.

In the child process, `fork()` has the return value 0.

In the parent process `fork()` has a positive return value that is equal to the PID of the newly created child process.

The return value of `fork()` in the parent process allows the parent process to identify the child process.

2 Points

(3) Name <u>two</u> differences of a child process from the parent process shortly after its creation.

The PID, the PPID, and the memory areas.

2 Points

(4) Describe the consequences if a parent process is terminated before the child process.

If a parent process terminates before the child process, it gets `init` or `systemd` as the new parent process assigned. Orphaned processes are always adopted by `init` or `systemd`. The PPID of the cild process then becomes value 1.

Question 8)

Points: ..... of 8

1 Point (1) Explain why fairness is a relevant criteria in scheduling.

If a scheduling algorithm is not fair, low-priority processes may starve.

2 Points (2) Explain the difference between preemptive and non-preemptive scheduling.

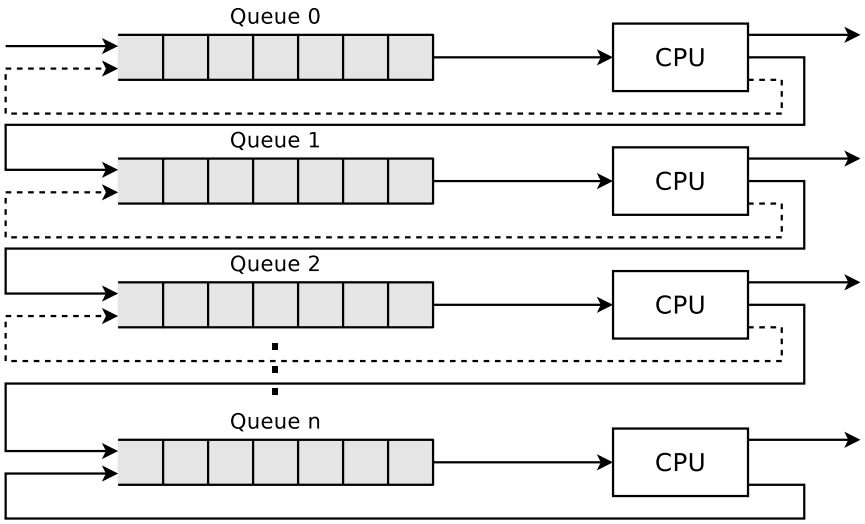
When using preemptive scheduling, the CPU may be removed from a process before its execution is completed.

When using non-preemptive scheduling, a process, which gets the CPU assigned by the scheduler, remains control over the CPU until its execution is finished or it gives the control back on a voluntary basis.

1 Point (3) Name the scheduling method that Windows operating systems implement.

Multilevel feedback scheduling.

4 Points (4) Explain how the scheduling method of Windows operating systems works. (Hint: A schematic diagram may help here!)



Multilevel feedback scheduling works with multiple queues. Each queue has a different priority or time multiplex (e.g. 70%:15%:10%:5%). Each new process is added to the top queue. This way it has the highest priority. Each queue uses Round Robin. If a process returns the CPU on voluntary basis, it is added to the same queue. If a process utilized its entire time slice, it is inserted in the next lower queue, with has a lower priority. The priorities are therefore dynamically assigned with this method.

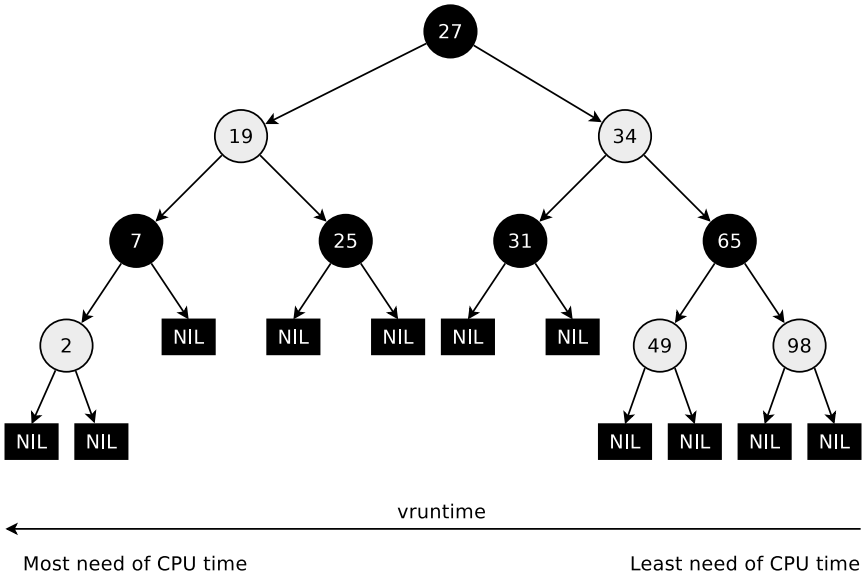
No complicated estimations. Processes with many Input and output operations are preferred because they are inserted in the original queue again when they resigns the CPU on voluntary basis $\Longrightarrow$ This way they keep their priority value. Older, longer-running processes are delayed.

Question 9)

Points: ..... of 4

4 Points

(1) Explain how the Completely Fair Scheduler of the Linux kernel (Kernel 2.6.23 until Kernel 6.5.13) works.
(Hint: A schematic diagram may help here!)



The kernel implements a CFS scheduler for every CPU core and maintains a variable `vruntime` *(virtual runtime) for every* `SCHED_OTHER` *process. The value represents a virtual processor runtime in nanoseconds.* `vruntime` *indicates how long the particular process has already used the CPU core. The process with the lowest* `vruntime` *gets access to the CPU core next. The management of the processes is done using a red-black tree (self-balancing binary search tree). The processes are sorted in the tree by their* `vruntime` *values.*

Aim: All processes should get a similar (fair) share of computing time of the CPU core they are assigned to. For n processes, each process should get $1/n$ of the CPU time. If a process got the CPU core assigned, it can run until its `vruntime` *value has reached the targeted portion of $1/n$ of the available CPU time. The scheduler aims for an equal* `vruntime` *value for all processes.*

The values are the keys of the inner nodes. leaf nodes (NIL nodes) have no keys and contain no data. NIL stands for none, nothing, null, which means it is a null value or null pointer. For fairness reasons, the scheduler assigns the CPU core next to the leftmost process in the tree. If a process gets replaced from the CPU core, the `vruntime` *value is increased by the time the process did run on the CPU core.*

The nodes (processes) in the tree move continuously from right to left $\Longrightarrow$ fair distribution of CPU resources.

The scheduler takes into account the static process priorities (`nice` *values) of the processes. The* `vruntime` *values are weighted differently depending on the* `nice` *value. In other words: The virtual clock can run at different speeds.*

Question 10)

Points: ..... of 7

1 Point (1) Describe what a critical section is.

Processes carry out read and write operations on common data. Critical sections may not be processed by multiple processes at the same time.

1 Point (2) Describe what a race condition is.

It is an unintended race condition of two processes, which want to modify the value of the same record.

1 Point (3) Describe why race conditions are hard to locate and fix.

The result of a process depends on the order or timing of other events. The occurrence of the symptoms depends on different events. The symptoms may be different or disappear with each test run.

1 Point (4) Explain what a system call is.

If a user-mode process must carry out a higher privileged task (e.g. access hardware), it can tell this the kernel via a system call. A system call is a function call in the operating system, which triggers a switch from user mode to kernel mode ($\Longrightarrow$ context switch).

1 Point (5) Explain what the standard library is and its purpose.

The standard library is logically located between the user mode processes and the kernel. It handles the communication between user mode processes and kernel and takes care about the context switching between user mode and kernel mode. It implements wrapper functions for the system calls for improving portability and security.

1 Point (6) Explain what a semaphore is.

A semaphore is a counter lock. It is used to protect (lock) critical sections.

1 Point (7) Explain what a mutex is.

If the Semaphore-feature of counting is not required, a simplified alternative, the mutex can be used instead. Mutexes (derived from Mutual Exclusion) are used to protect critical sections, which are allowed to be accessed by only a single process at any given moment. Mutexes can only have 2 states: occupied and not occupied. Mutexes have the same functionality as binary semaphores.

Question 11)

Points: ..... of 6

6 Points

(1) Perform the deadlock detection with matrices and check if a deadlock occurs.

$$\text{Existing resource vector} = \begin{pmatrix} 10 & 5 & 7 \end{pmatrix}$$

$$\text{Current allocation matrix} = \begin{bmatrix} 0 & 1 & 0 \\ 2 & 0 & 0 \\ 3 & 0 & 2 \\ 2 & 1 & 1 \\ 0 & 0 & 2 \end{bmatrix} \qquad \text{Request matrix} = \begin{bmatrix} 7 & 4 & 3 \\ 1 & 2 & 2 \\ 6 & 5 & 0 \\ 4 & 1 & 1 \\ 4 & 3 & 5 \end{bmatrix}$$

The existing resource vector and the current allocation matrix are used to calculate the available resource vector.

$$\textit{Available resource vector} = \begin{pmatrix} 3 & 3 & 2 \end{pmatrix}$$

Only process 2 can run with this available resource vector. The following available resource vector results when process 2 has finished execution and deallocates its resources.

$$\textit{Available resource vector} = \begin{pmatrix} 5 & 3 & 2 \end{pmatrix}$$

Only process 4 can run with this available resource vector. The following available resource vector results when process 4 has finished execution and deallocates its resources.

$$\textit{Available resource vector} = \begin{pmatrix} 7 & 4 & 3 \end{pmatrix}$$

Only process 1 can run with this available resource vector. The following available resource vector results when process 1 has finished execution and deallocates its resources.

$$\textit{Available resource vector} = \begin{pmatrix} 7 & 5 & 3 \end{pmatrix}$$

Only process 3 can run with this available resource vector. The following available resource vector results when process 3 has finished execution and deallocates its resources.

$$\textit{Available resource vector} = \begin{pmatrix} 10 & 5 & 5 \end{pmatrix}$$

Process 5 is not blocked.

No deadlock occurs.

Question 12)

Points: ..... of 7

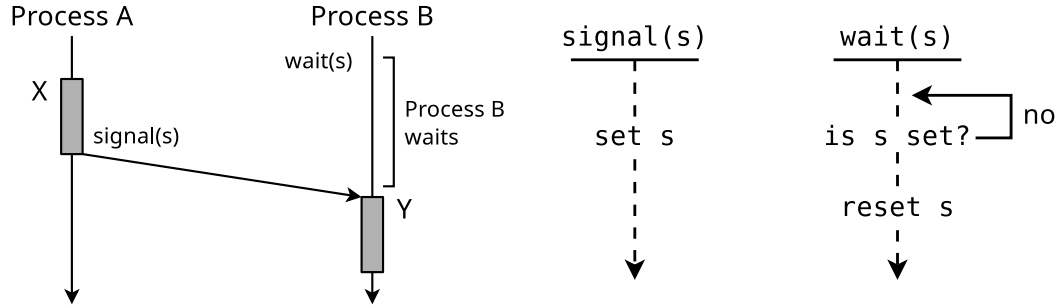
½ Point (1) Name one sort of inter-process communication that can only be used for processes that are closely related to each other.

Anonymous pipes.

½ Point (2) Name one sort of inter-process communication that allows communication over computer system boundaries.

Sockets.

3 Points (3) The figure shows the working principle of signaling, a technique that is used to specify an execution order of critical sections of processes.



Describe where you see room for improvement in terms of CPU utilization.

The figure shows busy waiting at the signal variable `s`*. The technique is also called active waiting, spinlock, or polling. CPU resources are wasted, because the wait operation occupies the processor at regular intervals.*

2 Points (4) Explain one possible way of implementing the signaling technique shown in subtask (3) in Linux.

One way to specify in Linux an execution order with passive waiting, is by using the function `sigsuspend`*. Thereby a process blocks itself until another process sends it an appropriate signal (usually* `SIGUSR1` *or* `SIGUSR2`*) with the command* `kill` *(or the system call of the same name) and in this way signals that it should continue working.*

Alternative system calls and function calls by which a process can block itself until it is woken up again by a system call are `pause` *and* `sleep`*.*

1 Point (5) Name a technique for process synchronisation, which has less drawbacks than signaling shown in subtask (3).

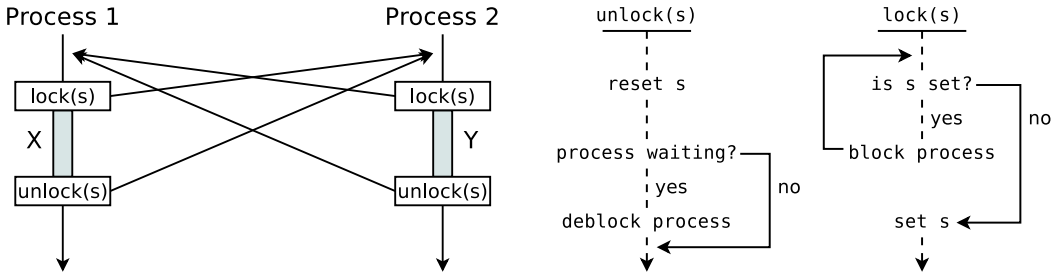
Passive waiting, (blocking) Sockets, Semaphore concept, Message Queues, Shared Memory with Semaphores,...

Question 13)

Points: ..... of 5

2 Points

(1) The figure shows the working principle of a synchronisation technique that ensures that the execution of critical sections does not overlap and does not specify the execution order of the critical sections.



Explain one possible way of implementing the signaling technique shown in this subtask in Linux.

Alternative 1: Implementation of locking with the signals `SIGSTOP` *(No. 19) and* `SIGCONT` *(No. 18). With* `SIGSTOP` *another process can be stopped. With* `SIGCONT` *another process can be reactivated.*

Alternative 2: A local file serves as a locking mechanism for mutual exclusion. Each process verifies before entering its critical section whether it can open the file exclusively. e.g. with the system call `open` *or standard library function* `fopen`*. If this is not the case, it must pause for a certain time (e.g. with the system call* `sleep`*) and then try again (busy waiting). Alternatively, it can pause itself with* `sleep` *or* `pause` *and hope that the process that has already opened the file unblocks it with a signal at the end of its critical section (passive waiting).*

½ Point

(2) Name one sort of inter-process communication that operates bidirectional.

Shared memory segments, Message queues, Sockets.

½ Point

(3) Name one sort of inter-process communication where the operating system does not guarantee the synchronization.

Shared memory segments.

2 Points

(4) Explain the meaning of the lines and columns in the file `/proc/buddyinfo`.

```
$ cat /proc/buddyinfo
Node 0, zone      DMA      1      1      1      0      2      1      1      0      1      1      3
Node 0, zone    DMA32    208    124   1646    566    347    116    139    115     17      4    212
Node 0, zone   Normal     43     62    747    433    273    300    254    190     20      8    287
```

The `DMA` *row shows the first 16 MB of the system.*
The `DMA32` *row shows all memory > 16 MB and < 4 GB of the system.*
The `Normal` *row shows all memory > 4 GB of the system.*
Column 1: number of free memory chunks („buddies“) of size $2^0 * PAGESIZE$
Column 2: number of free memory chunks („buddies“) of size $2^1 * PAGESIZE$
...
Column 11: number of free memory chunks („buddies“) of size $2^{10} * PAGESIZE$