Sample solution of the written examination in

Operating Systems

February 25th 2022

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: _____

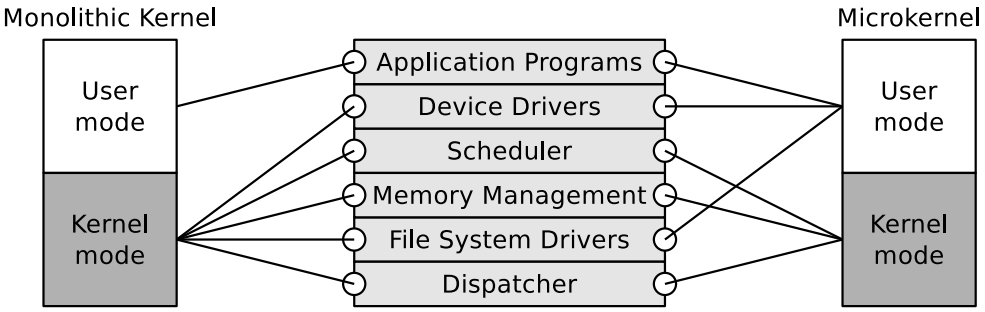
Questions:	1	2	3	4	5	6	7	8	9	**Σ**
Maximum points:	17	11	8	8	11	11	7	8	9	90
Achieved Points:										

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

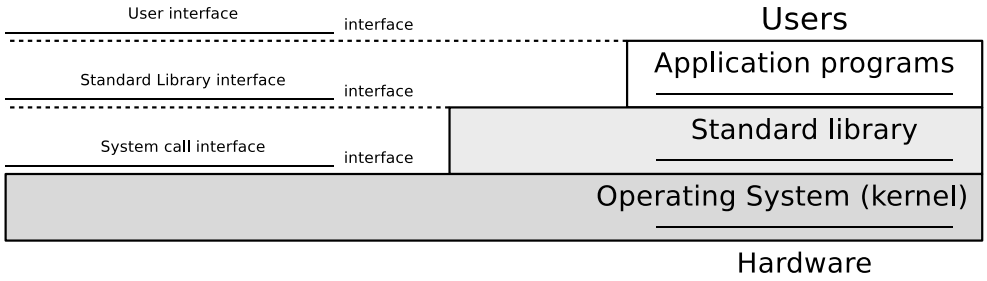
Question 1)

Points: .....

6 Points (1) The center column of the image contains operating system functions. Draw a line from each component to the left and one line to the right to indicate for monolithic kernels and microkernels both whether the component belongs to the kernel mode or user mode.



6 Points (2) The users cannot communicate directly with the hardware. Three layers can be identified between the hardware and the users. Each of these layers implements an interface. Name the layers and the interfaces in the figure.



1 Point (3) Explain why records in the upper layers of the memory hierarchy are continuously replaced.

Since the uppermost storage levels practically always lack free capacity, records must be replaced.

1 Point (4) Mark the concept that does not require any hardware support:

☐ Direct Memory Access ☐ Interrupt driven ☒ Busy waiting

3 Points (5) Mark the six steps of the CPU's instruction cycle:

☐ Swap
☒ Decode
☒ Fetch Operands
☒ Execute
☐ Main
☒ Update Program Counter
☒ Write Back
☐ Add
☐ Cross
☐ Set Stack Pointer
☒ Fetch
☐ Arrange

Question 2)

Points: .....

1 Point (1) Name the type of computer memory that benefits from using wear leveling algorithms.

Flash memory linke USB flash memory drives, solid state drives (SSD), Compact Flash Cards, ...

3 Points (2) Describe the purpose of wear leveling algorithms.

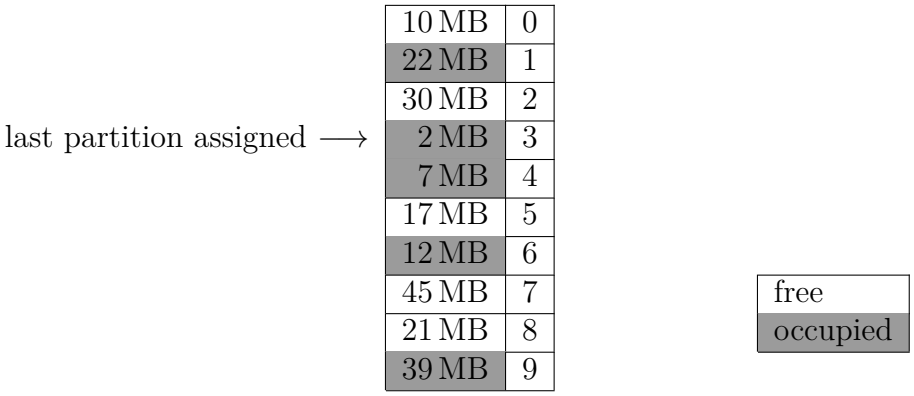
Wear leveling algorithms evenly distribute write operations. It is useful to use a wear leveling algorithm with SSD and other flash memory drives because flash memory cells have a limited number of program/erase cycles and without evenly distributing the write operations, some flash memory cells would break faster than others.

1 Point (3) Name the two aims (characteristics) that can be enhanced by a RAID.

Read-/Write performance and reliability.

3 Points (4) The following memory area belongs to a memory with dynamic partitioning. For each of the three algorithms, First Fit, Next Fit, and Best Fit, specify the number of the free partition that the corresponding algorithm uses to insert a process that requires 21 MB of memory.

a) First Fit: *2* b) Next Fit: *7* c) Best Fit: *8*



1 Point (5) Give the maximum number of memory addresses that can be addressed with a 32-bit computer system.

2^{32} *addresses.*

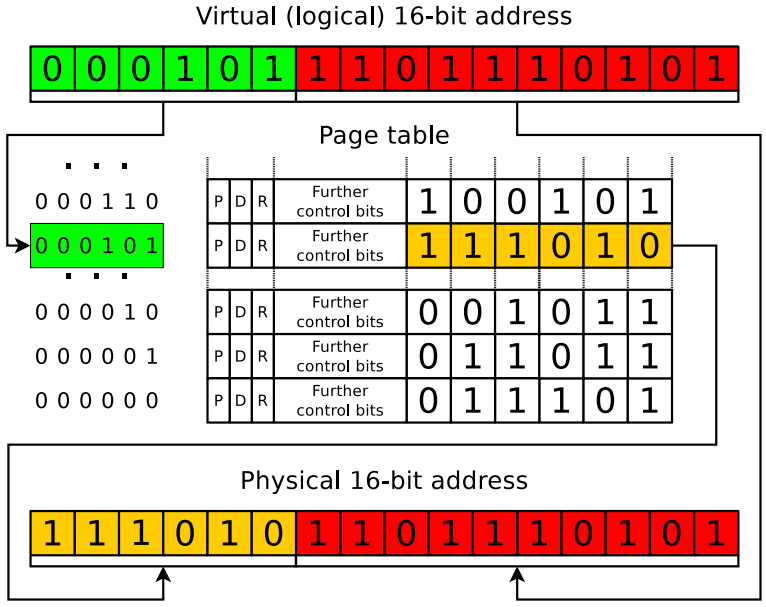
2 Points (6) Explain why multi-level paging is used in 32-bit and 64-bit systems and not single-level paging.

In 32 bit operating systems with 4 kB page length, the page table of each process can be 4 MB in size. In 64 bit operating systems, the page tables can be much larger. Multi-level paging reduces the main memory usage because individual pages of the different levels can be relocated to the swap storage to free up storage capacity in the main memory.

Question 3)

Points: .....

4 Points (1) Calculate the physical 16-bit memory address using address translation with single level paging. Fill in the single bits in the physical 16-bit address.



1 Point (2) Explain the purpose of the Page-Table Base Register (PTBR).

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

1 Point (3) The best page replacement strategy is the optimal strategy. Describe how it works.

It replaces the page, which is not requested for the longest time in the future.

1 Point (4) Explain why modern operating systems do not implement the optimal page replacement strategy.

It is impossible to implement because the future cannot be predicted.

1 Point (5) Explain a scenario where the optimal strategy is useful in practice.

OPT is used to evaluate the efficiency of other replacement strategies.

Question 4)

Points: .....

1 Point (1) Explain why using the real address mode would not be a good idea in a modern operating system.

In real mode, there is no memory protection. Each process can access the entire memory address space. One process could read or overwrite a memory address that is use by another process.

2 Points (2) Imagine a file system with an endlessly large (or at least really large) block storage device. Name and explain a limiting factor that prevents you from creating an infinite number of files.

(The storage capacity of the block storage device is not the limiting factor here!)

There can only be as many files as there are inodes, and the number of possible inodes depends on the size of the inode ID.

2 Points (3) Explain two reasons why defragmentation is not recommended when using modern storage devices and modern operating systems.

The objective of defragmentation is reducing the latency. Modern storage devices are SSDs (flash memory). For SSDs the position of the clusters is irrelevant for the latency.

As multiple programs are executed at the same time, applications can almost ever read large amounts of data in a row, without another application reading/writing in between. Therefore, it makes no strong difference if the file system is fragmented or not.

1 Point (4) Some file systems use a concept called Copy-on-write (COW). Mark the two answers that apply to such file systems.

When a file is modified, the old clusters in the file system that need to be modified...

☒ are preserved (not changed).
☐ are overwritten with the new modifications.
☐ are erased, by removing the cluster address in the inode.
☒ are copied into new clusters, where the modifications are made.

2 Points (5) You tried to run `script.sh` but the following happens:

```
$ ./script.sh
permission denied: ./script.sh
```

Give a solution for the command-line shell that allows you executing `script.sh`.

Some examples (each one will do the job)...

```
$ chmod 777 script.sh
$ chmod 700 script.sh
$ chmod 500 script.sh
$ chmod u+rwx script.sh
$ chmod u+rx script.sh
```

Question 5)

Points: .....

1 Point (1) In a very simple process model, two process states are enough. Name the two states in such a model.

idle: for processes that wait for the allocation of a CPU core.
running: for process that are allocated to a CPU core.

2 Points (2) Processes constantly alternate in multi-program mode. Explain how it is possible for a process to continue the execution in the same state as it was interrupted.

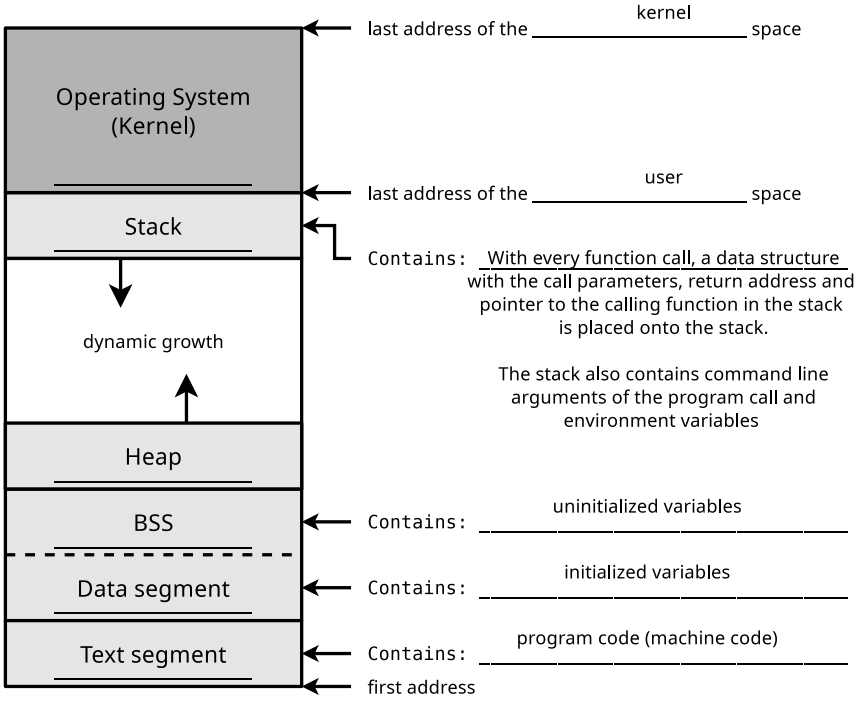
The operating system implements a process control block for each process. When an interrupt or a system call occurs, the hardware context and the system context are stored in the process control block. As soon as the process is assigned the CPU again, the hardware context and the system context from its process control block are restored.

2 Points (3) Describe what happens if you execute this program:

```
while(true){
    fork()
}
```

It is a fork bomb. The program creates copies of the process in an endless loop. The program create copies of the process until there is no more free memory and the system becomes unusable.

6 Points (4) The image shows the memory structure of a UNIX process. Add the missing labels (technical terms) of the process-related data and missing information about the content of these data.



Question 6)

Points: .....

1 Point (1) Explain what can go wrong when using (static) priority-driven scheduling.

Processes with low priority may starve.

1 Point (2) Some systems implement one or more idle process. Explain what idle processes are good for.

The system idle process is always active and has the lowest priority. It gets the CPU assigned when no other process is ready. Due to the system idle process, the scheduler must never consider the case that no active process exists.

1 Point (3) How many idle processes exist in a modern Linux system?

For each CPU core (in hyperthreading systems for each logical CPU), exists a system idle process.

6 Points (4) The two processes P_A (4 ms CPU time) and P_B (26 ms CPU time) are both in state `ready` at time point 0 and are to be executed one after the other.

Fill the table with correct values.

(Hint: Runtime = Lifetime)

Execution order	**Runtime** P_A	P_B	**Average runtime**	**Waiting time** P_A	P_B	**Average waiting time**
P_A, P_B	4	30	17	0	4	2
P_B, P_A	30	26	28	26	0	13

2 Points (5) Explain what can be observed from the values you filled into the table in (4).

If a short-running process runs before a long-running process, the runtime and waiting time of the long process process get slightly worse. If a long-running process runs before a short-running process, the runtime and waiting time of the short process get significantly worse Therefore, it is favorable to run short processes first, if a low average waiting time and average runtime is desired.

Question 7)

Points: .....

1 Point (1) If two processes access common resources (e.g. data), their relationship is characterized as...

- ☐ allocation
- ☒ cooperation
- ☐ communication
- ☐ virtual
- ☐ All of them
- ☐ None of them

(*Hint: A single answer is correct.*)

1 Point (2) If a process sends a copy of its data to a second process, their relationship is characterized as...

- ☐ allocation
- ☐ cooperation
- ☒ communication
- ☐ virtual
- ☐ All of them
- ☐ None of them

(*Hint: A single answer is correct.*)

1 Point (3) Mark the concept that is essential for the answers from (1) and (2).

- ☐ orchestration
- ☐ serialization
- ☐ highlighting
- ☐ bypassing
- ☐ parallelization
- ☒ synchronization

(*Hint: A single answer is correct.*)

1 Point (4) A drawback of deadlock detection with resource graphs is that it cannot be used...

- ☐ because it can only represent a maximum of three processes.
- ☐ when a process is starved.
- ☐ because it can only represent the resources at a single point in time.
- ☒ when multiple copies (instances) of a resource exist.

(*Hint: A single answer is correct.*)

1 Point (5) Mark the form of inter-process communication where processes need to coordinate access operations by themselves.

- ☐ Sockets
- ☐ Pipes
- ☐ Message queues
- ☒ Shared memory
- ☐ All of them
- ☐ None of them

1 Point (6) Name the scheduling method that modern Windows operating systems implement.

Multilevel feedback scheduling

1 Point (7) Name the scheduling method that modern Linux operating systems implement.

Completely Fair Scheduling (CFS)

Question 8)

Points: .....

Take a look at the given file system tree.

```
/
├── bin
├── boot
├── dev
...
├── img
│   ├── logo
│   └── thumb
├── src                          <--- (2)
│   ├── factories                <--- (3)
│   ├── adapters
│   ├── main
│   │   ├── worker.py
│   │   └── app.py               <--- (2)
│   └── util
└── test
    ├── main
    │   ├── test_factory.py      <--- (3)
    │   └── test_save.py         <--- (1)
    └── misc
```

2 Points (1) Give the absolute path to `test_save.py`:
`/test/main/test_save.py`

2 Points (2) Give the relative path from `src` to `app.py`:
`main/app.py`

2 Points (3) Give the relative path from `factories` to `test_factory.py`:
`../../test/main/test_factory.py`

1 Point (4) Give the command that can be used to print out the absolute path to your current working directory in the command-line shell.
`pwd`

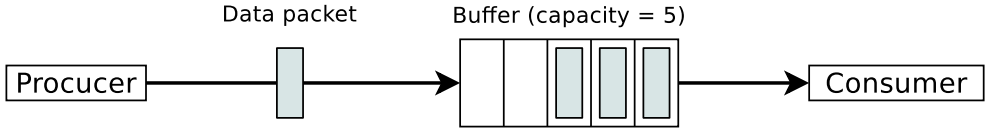
1 Point (5) The Bash command-line shell is a...

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

Question 9)

Points: .....

A producer process writes data into a buffer and a consumer process removes it. Mutual exclusion is necessary to avoid inconsistencies. If the buffer has no more free capacity, the producer must be blocked. If the buffer is empty, the consumer must be blocked.



The scenario includes three semaphores. `filled` indicates how many buffer slots are occupied. `empty` indicates how many buffer slots are empty. `mutex` is used for mutual exclusion of critical sections.

```
1     typedef int semaphore;
2     semaphore filled = 0;
3     semaphore empty  = 5;
4     semaphore mutex  = 0;     <== Bug! mutex must be initialized with 1
5
6     void producer(void) {
7         int data;
8         while (1) {
9             createData(data);
10            V(empty);         <== Bug! This must be: P(empty);
11            P(mutex);
12            insertData(data);
13            V(mutex);
14            P(filled);        <== Bug! This must be: V(filled);
15        }
16    }
17
18    void consumer(void) {
19        int data;
20        while (1) {
21            P(filled);
22            P(mutex);
23            removeData(data);
24            V(mutex);
25            V(empty);
26            consumeData(data);
27        }
28    }
```

The source code includes three mistakes (bugs).

3 Points (1) Give the line of each mistake.

3 Points (2) Explain each mistake.

3 Points (3) Propose a solution for each mistake.

Additional page for your solution of Question 9)

Line 4:

Mutex is initialized with value 0. Therefore, neither the producer nor the consumer will ever be able to access the critical section and both will be blocked forever right from the start.

Solution: `semaphore mutex = 1;`

Line 10:

Empty is increased by calling V. But since the producer creates data and puts it into the buffer, empty needs to be reduced.

Solution: `P(empty);`

Line 14:

Filled is reduced by calling P. But since the producer creates data and puts it into the buffer, filled needs to be increased.

Solution: `V(filled);`