# Exam Revision

Edward Zhang

SOFTENG 370 T8

## Exam Info

Your exam will be short answer, not MCQ. That means the exam from 2012 - 2017 aren't very useful. 2018 had a different lecturer for the first $\frac{1}{4}$ so it's not super helpful either.

## Which of the following is not a necessary component of a monitor?

- Publicly accessible entry points
- A readers/writers lock
- A scheduler
- A shared resource which is protected by the monitor

## Which of the following is not a necessary component of a monitor?

- Publicly accessible entry points
- A readers/writers lock
- A scheduler
- A shared resource which is protected by the monitor

### Explanation

Reader/Writers lock can enhance performance, but is not required.

## Which of the following best explains what happens when a damaged C program comes to an end but doesn't call the exit routine?

- The damaged program can corrupt memory used by other processes and cause them to crash or perform illegal instructions.
- The operating system takes control when the program tries to execute an illegal instruction or attempts to access unallocated memory.
- The C standard library takes control when the program fails to return to the code which called the main function.
- The operating system creates a new process and restarts the damaged program in that process so that it gets another chance to complete.

## Which of the following best explains what happens when a damaged C program comes to an end but doesn't call the exit routine?

- The damaged program can corrupt memory used by other processes and cause them to crash or perform illegal instructions.
- The operating system takes control when the program tries to execute an illegal instruction or attempts to access unallocated memory.
- The C standard library takes control when the program fails to return to the code which called the main function.
- The operating system creates a new process and restarts the damaged program in that process so that it gets another chance to complete.

## The code below uses a compare and swap function “cas”. What is the code doing?

```
add_to_balance(increase):
  previous_amount = balance
  while (!cas(&balance,
    previous_amount,
    previous_amount + increase)):
  previous_amount = balance
```

- It repeatedly increments balance by increase until balance overflows.
- It increments balance by increase using a condition variable.
- It safely swaps the values of balance with previous_amount + balance using a wait-free algorithm.
- It safely increments balance by increase using a lock-free algorithm.

## The code below uses a compare and swap function “cas”. What is the code doing?

```
add_to_balance(increase):
  previous_amount = balance
  while (!cas(&balance,
    previous_amount,
    previous_amount + increase)):
  previous_amount = balance
```

- It repeatedly increments balance by increase until balance overflows.
- It increments balance by increase using a condition variable.
- It safely swaps the values of balance with previous_amount + balance using a wait-free algorithm.
- It safely increments balance by increase using a lock-free algorithm.

## Which of the following does NOT happen in a context switch between threads in the same process?

- The processor registers for the currently running thread are saved.
- The processor registers are loaded with the saved values for the new thread.
- The page table is switched from the old thread to the new thread.
- The thread states for the two threads may be changed.
- The stack is changed from the old thread to the new thread.

## Which of the following does NOT happen in a context switch between threads in the same process?

- The processor registers for the currently running thread are saved.
- The processor registers are loaded with the saved values for the new thread.
- The page table is switched from the old thread to the new thread.
- The thread states for the two threads may be changed.
- The stack is changed from the old thread to the new thread.

### Explanation

Memory is shared between threads, so same page table.

## Which of the following is False?

- FUSE works by redirecting file operations through the FUSE module to a process running in user mode.
- To use a FUSE file system we mount the file system over an existing directory.
- To use FUSE requires root privileges.
- If the FUSE process is killed the files and directories contained within it will not be accessible.
- There has to be a FUSE kernel module in order for FUSE to work on Linux.

## Which of the following is False?

- FUSE works by redirecting file operations through the FUSE module to a process running in user mode.
- To use a FUSE file system we mount the file system over an existing directory.
- To use FUSE requires root privileges.
- If the FUSE process is killed the files and directories contained within it will not be accessible.
- There has to be a FUSE kernel module in order for FUSE to work on Linux.

### Explanation

You probably used FUSE w/o root in your assignment.

## Which of the following disk scheduling algorithms are commonly used for scheduling SSDs?

- Shortest Seek time First
- SCAN
- First come, first served
- Circular SCAN
- None of the above

## Which of the following disk scheduling algorithms are commonly used for scheduling SSDs?

- Shortest Seek time First
- SCAN
- First come, first served
- Circular SCAN
- None of the above

### Explanation

SSDs have no Seek time, and no head/platter so SCAN is irrelevant. FCFS makes sense since no special handling is required.

## What causes thrashing?

- When the foreground process has completely used up the number of frames it has been allocated.
- When the sum of the pages of the working-sets exceeds the number of frames.
- When there is not enough contiguous memory to be allocated for all current working sets.
- When all frames are currently being used.
- When all processes have filled up their page tables.

## What causes thrashing?

- When the foreground process has completely used up the number of frames it has been allocated.
- When the sum of the pages of the working-sets exceeds the number of frames.
- When there is not enough contiguous memory to be allocated for all current working sets.
- When all frames are currently being used.
- When all processes have filled up their page tables.

### Explanation

Recall that thrashing is when the virtual memory system is overused, and is thus stuck in a constant state of paging / pagefaults.

## Which of the following statements about user level device drivers is FALSE?

- User level drivers cannot deal with device interrupts.
- User level drivers can communicate with memory mapped devices.
- Most problems with user level drivers do not affect the kernel.
- Because of mode transitions user level drivers are sometimes not used for fast devices.
- User level drivers can communicate with IO ports

## Which of the following statements about user level device drivers is FALSE?

- User level drivers cannot deal with device interrupts.
- User level drivers can communicate with memory mapped devices.
- Most problems with user level drivers do not affect the kernel.
- Because of mode transitions user level drivers are sometimes not used for fast devices.
- User level drivers can communicate with IO ports

## Which of the following conditions have a direct influence on the effective access time of memory?

- The time to retrieve page table information for the page.
- The page fault rate.
- The time taken to provide a free frame for the page if required.
- All of the above.

## Which of the following conditions have a direct influence on the effective access time of memory?

- The time to retrieve page table information for the page.
- The page fault rate.
- The time taken to provide a free frame for the page if required.
- All of the above.

### Explanation

Recall that effective means we take into account the average time.

## Which of the following best explains what happens concerning the current instruction when a page fault occurs?

- The instruction is completed and then the required page is brought into a frame so that the program can continue correctly.
- The current thread changes to another thread and the offending instruction and its thread are removed from the system.
- After the required page has been brought into a frame the instruction must usually be restarted from its initial state before the page fault occurred.
- The current instruction is only restarted if it caused the page fault. If it didn't cause the page fault it continues normally.

## Which of the following best explains what happens concerning the current instruction when a page fault occurs?

- The instruction is completed and then the required page is brought into a frame so that the program can continue correctly.
- The current thread changes to another thread and the offending instruction and its thread are removed from the system.
- After the required page has been brought into a frame the instruction must usually be restarted from its initial state before the page fault occurred.
- The current instruction is only restarted if it caused the page fault. If it didn't cause the page fault it continues normally.

## Question 9b

One protection against the Meltdown exploit that has been implemented in operating systems is kernel page-table isolation (KPTI). Explain what kernel page-table isolation is.

## Question 9b

One protection against the Meltdown exploit that has been implemented in operating systems is kernel page-table isolation (KPTI). Explain what kernel page-table isolation is.

### Answer

Separate page tables are kept for a process when it is running in kernel and user mode. The user mode page tables do not have most of the kernel pages mapped into them.

## Question 9c/d

What effect could KPTI have on efficiency, and how does it prevent Meltdown?

## Question 9c/d

What effect could KPTI have on efficiency, and how does it prevent Meltdown?

### Efficiency

Anything that switches to kernel mode, like syscalls, requires a new page table to be loaded. This will also likely flush the TLB.

## Question 9c/d

What effect could KPTI have on efficiency, and how does it prevent Meltdown?

### Efficiency

Anything that switches to kernel mode, like syscalls, requires a new page table to be loaded. This will also likely flush the TLB.

### Preventing Meltdown

Meltdown's timing attack relies on attempts to access values in kernel address space (even if these are eventually rejected by a privledge check). However, with KPTI, those addresses are not accessible in the current address space, so the attempt cannot be made.

## Question 10

What is a user-level device driver?

## Question 10

What is a user-level device driver?

### Answer

A device driver implemented in code to be run in user mode rather than kernel mode.

## Question 4c

Explain what a Least Slack Time schedule is and why it is a suitable choice for real-time scheduling.

## Question 4c

Explain what a Least Slack Time schedule is and why it is a suitable choice for real-time scheduling.

### Answer

The slack time is the amount of time from the current time to the end of the deadline for a process minus the amount of compute time remaining for that process. A Least Slack Time schedule chooses the process with the smallest slack time at any point to be the scheduled process. This is appropriate for a real-time system because processes have to complete before their deadlines or the real-time requirement has not been met. When the slack time of a process gets to zero it has to run or it will pass its deadline.

## Question 9b

One protection against the Meltdown exploit that has been implemented in operating systems is kernel page-table isolation (KPTI). Explain what kernel page-table isolation is.

## Question 9b

One protection against the Meltdown exploit that has been implemented in operating systems is kernel page-table isolation (KPTI). Explain what kernel page-table isolation is.

### Answer

Separate page tables are kept for a process when it is running in kernel and user mode. The user mode page tables do not have most of the kernel pages mapped into them.

## Question 2b/c

Give an example of when a hard link is preferable to a soft link

## Question 2b/c

Give an example of when a hard link is preferable to a soft link

### Answer

When we want to avoid the problem of a broken link when the original file gets moved.

## Question 2b/c

Give an example of when a hard link is preferable to a soft link

**Answer**

When we want to avoid the problem of a broken link when the original file gets moved.

Give an example of when a soft link is preferable to a hard link

## Question 2b/c

Give an example of when a hard link is preferable to a soft link

**Answer**

When we want to avoid the problem of a broken link when the original file gets moved.

Give an example of when a soft link is preferable to a hard link

**Answer**

When the link is to a file on a different device or volume, or to a directory.

## Question 4a

Using virtual memory slows down the effective access time, EAT, of memory for our processes. In lectures we saw two different ways in which access to a particular address in memory can be slowed down compared to a non-virtual memory system. Describe the two different ways and relate both of them to the page table information.

### Hint

Consider what extra steps are added when using virtual memory.

## Question 4a Answer

1. The page table information needs to be accessed forevery memory access. This either leads to two (or more)memory reads for every memory access or else requires acache (the TLB) to hold the page table information forrecently accessed pages.
2. The page may not be resident in RAM but on disk. Inorder to access data on the page it has to be brought intoa frame of real memory. The information about whether thepage is in RAM or on disk is stored in the page tableentry.