ⓘ Students have either already taken or started taking this quiz, so take care when editing it. If you change any quiz questions in a significant way, you might want to consider re-grading students' quizzes who took the old version of the quiz.

Points 8 Published

Details Questions

Show question details

Question 1 pts

Which of the following is NOT true for Hybrid approach for creating smaller page tables i.e. the approach that combines segmentation and pages tables.

- It increases the number of time physical memory needs to be accessed for translation compared to one large page tables
- It requires the use of base and bound registers for each segment of each process.
- It divides the address space into fixed size portions for each segment
- The size of the page table becomes variable requiring free-space management for page tables

Question 1 pts

In designing a multi-level page table for:

Address space = 8KB
Page size = 64 bytes
Page table entry size = 4 bytes

The linear page table will be divided into how many pages?

- 8
- 16
- 4
- 32

Question 1 pts

When a computer's virtual memory exceeds the available physical memory (RAM), the operating system employs a technique called swapping to temporarily move inactive data from RAM to a dedicated storage area on the hard disk known as swap space. Which of the following statements about swap space is most accurate?

- Utilizing swap space can lead to a noticeable decrease in system responsiveness due to slower data access compared to RAM.
- Swapping significantly enhances system performance by improving memory access speeds.
- Swap space is a volatile storage area, meaning data stored there gets erased when the computer is turned off.
- Swapping involves transferring entire processes, including code and data segments, to swap space.

Question 1 pts

Which of the following statements correctly describes how the OS handles a page fault when the present bit is set to 0?

Background (you can skip): A page fault occurs when a program attempts to access a memory address that is not currently loaded into physical memory. When a page fault occurs, the operating system (OS) is notified and it takes steps to resolve the issue. One of the key pieces of information the OS uses to handle page faults is the present bit in the page table entry (PTE). The present bit indicates whether the corresponding memory page is currently loaded into physical memory. If the present bit is set to 0, it means the page is not in physical memory and a page fault has occurred.

- The OS switches to kernel mode and loads the missing page from secondary storage into physical memory.
- The OS immediately terminates the program that caused the page fault.
- The OS marks the page as invalid and prevents further access to the memory address.
- The OS ignores the page fault and continues executing the program.

Question 1 pts

Suppose:

The cost of accessing memory = 10 nanoseconds
The cost of accessing disk = 10 milliseconds
The miss-rate = 10%

Then the average memory access time (AMAT) is:

- about 1ms
- about 0.1ms
- about 0.01ms
- about 0.001ms

Question 1 pts

On 80-20 workload which of the following replacement policies perform best?

- LRU
- FIFO
- Random
- MRU

Question 1 pts

Which of the following statements is NOT true about condition variables?

iswer

- Condition variables can be used to signal threads without waking them up.
- Condition variables are used to wait for specific conditions to occur before proceeding.
- Condition variables are used to wake up threads that are waiting on a condition.
- Condition variables are owned by mutexes.

Question

1 pts

Consider a scenario where a producer thread is responsible for generating data and placing it in a shared buffer, while a consumer thread retrieves data from the buffer for processing. To ensure synchronized access to the shared buffer, condition variables are employed. Which of the following sequences of pthread function calls correctly represents the producer thread's actions?

iswer

- pthread_mutex_lock();
produce data;
pthread_cond_signal();
pthread_mutex_unlock();
- pthread_mutex_lock();
produce data;
pthread_cond_wait();
pthread_mutex_unlock();
- pthread_cond_signal();
produce data;
pthread_mutex_lock();
pthread_cond_wait();
- pthread_cond_wait();
produce data;
pthread_mutex_lock();
pthread_cond_signal();

+ New question | + New question group | Find questions

Notify users this quiz has changed

Cancel | Save