

# Final Exam - Fall 2020

**Due** Mar 24 at 8pm**Points** 60**Questions** 60**Available** Mar 24 at 9am - Mar 24 at 8pm about 11 hours**Time Limit** 90 Minutes

## Instructions

There are 60 questions (24 from CLO1, 18 from CLO2 and 18 from CLO3) in this exam.

The total duration of the final exam is 1.5 hours = 90 minutes.

**Please remember that there is no back button so once you have answered a question, you cannot go back.**

You are required to do this on your own without consultation with anyone else.

Good luck.

This quiz is no longer available as the course has been concluded.

## Attempt History

	Attempt	Time	Score
LATEST	<a href="#">Attempt 1</a>	100 minutes	39 out of 60

⚠ Correct answers are hidden.

Score for this quiz: **39** out of 60

Submitted Dec 18, 2020 at 12pm

This attempt took 100 minutes.

### Question 1

**1 / 1 pts**

The *cooperating approach* of sharing CPU among multiple processes would use the

- ☐ timer interrupt
- ☐ interrupt handlers
- ☒ yield() system call
- ☐ atomic instructions

**Question 2****1 / 1 pts**

The operating system provides abstraction of memory through virtualization of memory. What do you understand from virtualization of memory?

- ☐ It means providing access to non-volatile memory.
- ☐ It means providing access to memory in a highly optimized way
- ☒ It means providing access to memory locations through logical addresses which are later mapped to a physical address through address translation.
- ☐ It means providing access to volatile memory.

**Incorrect****Question 3****0 / 1 pts**

Once a program terminates, any memory that has not been free'd will result in memory leaks.

☒ True☐ False**Question 4****1 / 1 pts**

Which of the following scheduling policies is more suitable for interactive systems:

☐ First In First Out☒ Round Robin☐ Shortest Time to Completion First☐ Shortest Job First**Question 5****1 / 1 pts**

A process in running state may go into ready state when one of the following happens:

☐ none of these other options is correct☐ An event happens for e.g. an I/O event☒ An interrupt is raised.☐ A new process is admitted

**Question 6****1 / 1 pts**

What will be response time of a process with an arrival time of 20msecs and a first run time of 10msecs?

☐ 0 msecs

☐ -10 msecs

☐ 10 msecs

☒

This is impossible as the first run time must always be greater than or equal to the arrival time.

**Incorrect****Question 7****0 / 1 pts**

The following data is given to you.

- Total CPU cores: 4
- Time slice: 1 msec
- Total processes: 4 (P1, P2, P3, P4)
- Each process takes 2 msecs to finish.
- P2 can start only after P1 finishes execution
- P4 can start only after P3 finishes execution

Assuming that the processes can be multithreaded, what is the least amount of time that the 4 processes can take to finish execution completely (ignoring the context switch time and assuming no I/O)?

☐ 1 msecs

☐ 4 msecs☐ 2 msecs☒ 8 msecs

Incorrect

**Question 8****0 / 1 pts***Paging without page tables*☐ is exactly the same as segmentation☒ cannot implement virtual memory☐ will have external fragmentation☐ will make program execution extremely slow

Incorrect

**Question 9****0 / 1 pts**

The following data is given to you.

- Total CPU cores: 1
- Time slice: 1 msec
- Total processes: 4
- Each process takes 2 msecs to finish.

Assuming that the processes can be multithreaded, what is the most amount of time that the 4 processes can take to finish execution completely (ignoring the context switch time and assuming no I/O)?

☒ 4 msecs☐ 8 msecs☐ 2 msecs☐ 1 msecs**Question 10****1 / 1 pts**

The operating system provides virtualization of CPU. What does this mean?

☐ It means allowing one program to always run on the same CPU core.☒ It means mapping each program to a process. Then each process is provided access to a virtual CPU which is later mapped to a physical CPU.☐ It means giving each program access to physical CPU directly without abstraction indefinitely.☐ It means allowing a single program to access multiple CPUs.**Question 11****1 / 1 pts**

The following data is given to you.

- Total CPU cores: 4
- Time slice: 1 msec

- Total processes: 4
- Each process takes 2 msecs to finish.

Assuming that the processes can be multithreaded, what is the least amount of time that the 4 processes can take to finish execution completely (ignoring the context switch time and assuming no I/O)?

☐ 4 msecs

☒ 2 msecs

☐ 8 msecs

☐ 1 msecs

Incorrect

### Question 12

0 / 1 pts

In *free space management*, the *coalescing* operation can only be performed on those members of the free list which are next to each other:

☐ in the address space

☐ in the linked list as well as in the address space

☒ in the free list

☐ in the programmer's mind

### Question 13

1 / 1 pts

What do you understand from process starvation when MLFQ scheduling is used?



A process which does not finish within its stipulated time slice is demoted starting from the top most queue all the way to the lowest priority queue. This makes this process unable to execute and hence starve for CPU.



None of the given options.



A process which requires memory but is unable to get it and hence starve.



A process which want to acquire an I/O device but it is unable to do so.

### Question 14

1 / 1 pts

```
#include <stdio.h>
int main() {
    for (int i=0; i<3; i++)
        fork();

    printf ("hello\n");
}
```

How many times will the above program print the string "hello"?



8



4



1



2



**Question 15****1 / 1 pts**

Which of the following is *not* an example of *fair share scheduling*:

- ☒ Multi-Level Feedback Queue
- ☐ Lottery scheduling
- ☐ Completely Fair Scheduler
- ☐ Stride scheduling

**Question 16****1 / 1 pts**

Single queue multiprocessor scheduling (SQMS) is not scalable because (pick the best answer)

- ☒ Needs locking to be implemented for synchronized access to the single shared queue between multiple CPUs.
- ☐ None of these other options is correct
- ☐ Requires extra overhead of managing memory.
- ☐ Requires extra overhead to manage I/O.

**Question 17****1 / 1 pts**

Priority boost in the multi-level feedback queue (MLFQ) policy serves to:

- ☐ prioritize the newly arrived jobs
- ☒ prevent starvation of long CPU intensive jobs
- ☐ boost the priority of smaller jobs
- ☐ improve the performance of I/O intensive jobs

**Question 18****1 / 1 pts**

In *free space management*, the *magic number* is used to:

- ☐ store the pointer returned by the function malloc()
- ☐ store the size of the allocated block
- ☐ optimize the free space management
- ☒ verify the integrity of the pointer being passed to the function free()

**Question 19****1 / 1 pts**

What is the use of the wait(NULL) call if it is given in the context/scope of the parent process?

- ☐ It ensures that the parent process may finish execution immediately.
- ☐ It ensures that both child and parent may finish together.
- ☒ It ensures that the parent process waits until the child process finishes.
- ☐ It ensures that the child process waits until the parent process finishes.

**Question 20****1 / 1 pts**

Translation Look-aside Buffers (TLBs) store:

- ☒ Recently used entries of page tables
- ☐ Data of the running process
- ☐ interrupt handlers
- ☐ Page tables

**Incorrect****Question 21****0 / 1 pts**

There are two processes: P1 and P2. P1 runs for 4 msec and P2 runs for 2 msec. Arrival time of P1 is 0 msec while the arrival time of P2 is 2 msec. Assuming a time slice value of 1 msec, what will be the average turn around time using round robin scheduling with pre-emption. At time 2 msec when P2 arrives the Scheduler has a choice of scheduling P1 or P2; assume it schedules P2 first.

- ☐ Avg. turnaround time: 3.5 msec

- ☐ Avg. turnaround time: 4.5 msec
- ☒ Cannot be determined as insufficient data given.
- ☐ Avg. turnaround time: 2.5 msec

**Question 22****1 / 1 pts**

The fork() call differs from the exec() call because (choose all that apply)



exec does not create a new process but it modifies the currently running process into a different program.



fork does not create a new process but it modifies the currently running process into a different program.



fork creates a new process which is an exact replica of the currently running program



exec creates a new process which is an exact replica of the currently running program

**Question 23****1 / 1 pts**

Which of the following techniques does not suffer from *internal* fragmentation?

- ☒ Segmentation
- ☐ Allocating one chunk of RAM for the whole address space of a process
- ☐ Paging
- ☐ the buddy allocator

### Question 24

1 / 1 pts

The disadvantage of multi-queue multiprocessor scheduling (MQMS) compared to single-queue multiprocessor scheduling (SQMS) is:

- ☐ it doesn't handle cache affinity well
- ☒ has trouble due to load imbalance
- ☐ it is simpler to implement
- ☐ it doesn't scale better

Incorrect

### Question 25

0 / 1 pts

In a multi-threaded program sharing global variables between threads, race conditions can occur in

- ☐ both single core as well as multi-core CPU environments
- ☐ neither single-core nor in multi-core CPU environments
- ☒ only a multi-core CPU environment
- ☐ only a single-core CPU environment

**Partial****Question 26****0.5 / 1 pts**

If atomicity is violated, what problems may arise in a multi-threaded program. (choose all that apply)

- ☐ no mutual exclusion
- ☒ non-deadlock bugs
- ☐ mutual exclusion
- ☐ deadlock bugs

**Incorrect****Question 27****0 / 1 pts**

Condition variables prevent \_\_\_\_\_ which is unavoidable when locks or mutex are used. (choose all that apply). Assuming the lock/mutex implementation does not allow the waiting threads to sleep.

- ☐ encryption
- ☐ wasting of CPU cycles

☐ spin wait or busy waiting☒ mutual exclusion

Incorrect

**Question 28****0 / 1 pts**

In the reader-writer locking scheme, if we are given information about a reader thread (R1) that it is currently in its critical section. From this information, which of the following statements can we deduce about the other threads:

☐ There is no other reader thread in its critical section at this moment☒ All of these other options are correct☐ There is no other thread in its critical section at this moment☐ There is no other writer thread in its critical section at this moment**Question 29****1 / 1 pts**

Semaphores can mimic the behaviour of

☐ locks only☒ both locks and condition variables☐ neither locks nor condition variables☐ condition variables only

Partial

**Question 30****0.5 / 1 pts**

Which of the following might happen if multiple threads try to access the same shared resource (choose all that apply)

☐

the shared resource remains with only one thread and all other threads cannot access the shared resource in their life time.

☐

everything works as normal nothing needs to be done.

☐

data may become inconsistent

☒

race condition

**Question 31****1 / 1 pts**

Mutual exclusion helps us avoid

☐

Concurrency

☐

Memory leaks

☒

Race conditions

☐

Deadlocks



Incorrect

**Question 32****0 / 1 pts**

Threads can share address space which allows

- ☐ each thread to execute on a different CPU core.
- ☐ each thread to do separate I/O request.
- ☒ none of these other options is correct
- ☐ multiple tasks to be performed simultaneously while sharing resources (code, data, and other resources).

Incorrect

**Question 33****0 / 1 pts**

Assume, in a multi-threaded program, the main thread initializes a semaphore with a value of -1, creates other threads, and then calls *sem\_wait()* on this semaphore. This thread will sleep until

- ☐ one of the child threads calls *thread\_exit()*
- ☐ when *sem\_post()* is called once on this same semaphore
- ☐ when *sem\_post()* is called twice on this same semaphore
- ☒ the end of times

Incorrect

**Question 34****0 / 1 pts**

In the *producer-consumer* problem

- ☐ a consumer waits when the buffer is empty
- ☒ the consumer waits when the buffer is full
- ☐ a producer waits when the buffer is not full
- ☐ the producer waits when the buffer is empty

Incorrect

### Question 35

0 / 1 pts

When using multiple locks between threads, there is a good probability of entering into a *deadlock* if we are not careful. Which of the following *does not* help in preventing or resolving deadlocks?

- ☐ After waiting for a certain amount of time for a lock, a thread should abandon waiting and release all the locks that it already holds
- ☐ All the threads should follow the same order when acquiring locks
- ☐ Always use an odd number of locks in your program
- ☒ all of these other options

Incorrect

### Question 36

0 / 1 pts

The `pthread_join` function is declared as follows:

```
int pthread_join(pthread_t thread, void** value_ptr);
```

What is the use of the second parameter value\_ptr?

- ☐ It is used to return values from the thread function.
- ☐ It is used to pass a value to the thread function.
- ☒ None of the options given are correct.
- ☐ It is used to capture error codes.

### Question 37

1 / 1 pts

Which of the following conditions should happen for a deadlock to occur?

- ☐ Circular wait
- ☐ Hold and wait
- ☒ All of these other options
- ☐ Mutual exclusion

### Question 38

1 / 1 pts

Why is it important to provide a thread join call after the thread has been created?



It allows the main thread to wait until the child thread we are creating finishes execution.



It allows the child thread to wait until the main thread finishes execution.



It allows the child thread to perform I/O.



It allows the main thread to perform I/O.

### Question 39

1 / 1 pts

What does the fairness property of locks ensure?



It ensures that all threads get a chance to terminate when they wish..



It ensures that all thread get a chance to perform I/O.



It ensures that all threads get a chance to acquire the lock



It ensures that all threads get a chance to perform memory transactions.

### Question 40

1 / 1 pts

Having multiple threads in a program is useful in



only a multi-core CPU environment

- ☒ both single core as well as multi-core CPU environments
- ☐ only a single-core CPU environment
- ☐ neither single-core nor in multi-core CPU environments

Unanswered

**Question 41****0 / 1 pts**

Why is it so that we can return address of a variable allocated dynamically through malloc from a thread function but we cannot return address of a variable allocated on stack? (choose all that apply)

- ☐ each thread has its own stack so once a thread function finishes execution, its stack memory is reclaimed.
- ☐ all threads of a process share the same heap memory hence a variable allocated dynamically (through malloc) remains visible to all other threads.
- ☐ all threads of a process share the same stack so once a thread function finishes execution, its stack memory remains visible to all other threads.
- ☐ each thread has its own heap memory so once a thread function finishes execution, its heap memory is reclaimed.

**Question 42****1 / 1 pts**

Why are locks used when a shared resource is accessed? (choose all that apply)

☐ locks improve I/O performance.

☒ locks prevent race conditions.

☐ locks protect data from viruses.

☐ locks prevent data over and under flow



locks provide mutual exclusion that is only one thread can access a shared resource.

### Question 43

1 / 1 pts

In the Very Simple File System (VSFS) discussed in the book, which of the following operations may modify a data block?

☐ creating a new file

☐ deleting a file

☐ writing to an opened file

☒ all of these other options

Incorrect

### Question 44

0 / 1 pts

In disk scheduling, the Shortest Seek Time First (SSTF) algorithm had a flaw, eventually corrected in the SCAN algorithm, that the

- ☐ jobs accessing a far away track risked starvation
- ☒ newly arrived jobs risked starvation
- ☐ small jobs risked starvation
- ☐ jobs accessing a nearby track risked starvation

Incorrect

### Question 45

0 / 1 pts

Why is polling bad?

- ☐ because it repeatedly accesses the I/O device.
- ☒ none of these other options is correct
- ☐ because it hinders operating system execution.
- ☐ because it wastes CPU cycles as CPU is not doing any useful work.

### Question 46

1 / 1 pts

Which block is used when the file system is mounted?

- ☐ data block

- ☐ inode block
- ☒ super block
- ☐ file descriptor block

**Question 47****1 / 1 pts**

Why is track skew given in hard disk drives (HDDs)?

- ☐ It provides data security.
- ☐ It helps provide additional storage.
- ☐ It maintains data integrity.
- ☒ It helps in transferring of head when data across track boundaries is involved.

**Question 48****1 / 1 pts**

Why can't hard links be created on directories?

- ☐ Directories can create hard links to other directories.
- ☐ Directories can create hard links to other directories on a different partition.





Directories have special characteristics which are not supported by hard links.



Directories might create links to themselves creating dependency cycles which would be difficult to deal with.

### Question 49

1 / 1 pts

What is the significance of the inode data structure?



It stores file descriptors of all directories in the current directory.



It stores data associated with a file.



it contains information about all aspect of file which helps in reading/writing of data on file as well as provides access to all properties of a file or directory.



It stores file descriptors of all files in the current directory.

### Question 50

1 / 1 pts

Why can't hard links be created on directories?



Directories might create links to themselves creating dependency cycles which would be difficult to deal with.



Directories can create hard links to other directories on a different partition.



Directories can create hard links to other directories.



Directories have special characteristics which are not supported by hard links.

### Question 51

1 / 1 pts

In the Very Simple File System (VSFS) discussed in the book, which of the following operations may access the data bitmap?



reading from an opened file



writing to an opened file



creating a new file



opening an existing file

### Question 52

1 / 1 pts

Assuming the following permission bits information, what is the access level for group members.

-r--rw-rw-

☐ read/write/execute

☐ write only

☒ read/write

☐ read only

Incorrect

### Question 53

0 / 1 pts

What is the use of the file descriptor that is returned or given to most file system API functions?

☒ It stores information about the file format.

☐ It helps identify the file uniquely with in the same process

☐ Its a redundant number which is not useful.

☐ It stores information about the file like file size.

### Question 54

1 / 1 pts

In communicating with slow-speed I/O device we'd prefer using which of the following two techniques:

☒ interrupts

☐ polling

### Question 55

1 / 1 pts

In the Very Simple File System (VSFS) discussed in the book, which of the following operations may access the inode bitmap?

☐ opening an existing file

☐ writing to an opened file

☒ creating a new file

☐ reading from an opened file

Incorrect

### Question 56

0 / 1 pts

The purpose of the DMA is to

☒ increase the CPU clock frequency

☐ liberate the CPU from doing data transfers to and from the I/O device

- ☐ help the CPU execute instructions atomically
- ☐ help the processor do efficient scheduling

**Incorrect****Question 57****0 / 1 pts**

What is the role of DMA (Direct Memory Access) during I/O?

- ☐ Provide buffering support for I/O
- ☐ offload I/O from CPU so that CPU has no involvement in data movement.
- ☒ Improve accuracy of I/O
- ☐ Provide encryption to data during I/O

**Question 58****1 / 1 pts**

Assume you have two disks A and B. All other things are equal but:

- the average seek time for A is 10% greater than B
- the max transfer rate for A is 10% greater than B

For a workload which continuously does many small transfers at random locations on hard disk, which of these disks will you prefer?

- ☒ B

☐ A

### Question 59

1 / 1 pts

Assuming the following permission bits information, what is the access level for owner.

-r--r--r--

☐ read/write/execute

☒ read only

☐ read/write

☐ None of the options.

### Question 60

1 / 1 pts

In the Very Simple File System (VSFS) discussed in the book, which of the following operations may modify the *inode table*?

☐ writing to an opened file

☒ all of these options

☐ opening an existing file

☐ reading from an opened file

Quiz Score: **39** out of 60