Unit Exam: Concurrency

Total points 110/110

Take this exam alone. It is closed book, closed notes, but feel free to use a single 8.5" x 11" sheet of paper (both sides) of notes. Take the exam only once.

The respondent's email (**parth2@pdx.edu**) was recorded on submission of this form.

✓ All concurrent programs are parallel programs * 5/5

True [✓ False] ✓

✓ While one thread holds a mutex, other threads may lock or unlock that mutex * 5/5

True [✓ False] ✓

✓ A race condition is undesirable, it is a bug * 5/5

[✓ True] ✓ False

Which of the following are examples of concurrent execution? *

	Concurrent Execution	Not Necessarily Concurrent Execution	Score	
a C program initializes the pthreads library	○	●	1/1	✓
a program calls functions recursively	○	●	1/1	✓
a DBMS executes multiple queries at the same time	●	○	1/1	✓
while a process is executing, the operating system de-schedules the running process to handle an I/O interrupt	●	○	1/1	✓
a timer interrupt occurs while a user-level process executes within a loop	●	○	1/1	✓
a java method is declared as "synchronized"	○	●	1/1	✓
a web server handles multiple HTTP requests at the same time	●	○	1/1	✓

✓ What do we call a concurrent read (or write) of an unprotected shared data value? * 5/5

- race condition
- deadlock
- [✓ data race] ✓
- prioritized
- concurrent
- order violation

✓ the C language statement "num++;" is an atomic operation * 5/5

- True
- [✓ False] ✓

✓ Which fundamental operation is often used to implement mutex locks on Intel CPUs? * 5/5

- ○ TestAndTestAndSet
- ○ FetchAndAdd
- ○ TestAndSet
- ○ CompareAndSwap
- ○ LoadLink
- [✓ xchg] ✓

✓ When pthread code calls signal() on a condition variable, which of the threads waiting on that condition variable will wake up and run as a result of the signal() * 5/5

- ○ none of the threads waiting on the condition variable
- ○ only the thread that most recently began to wait on the condition variable
- [✓ one or more threads waiting on the condition variable] ✓
- ○ only the thread that least recently began to wait on the condition variable

✓ According to the lecture/slides, how many condition variables are needed for a bug-free implementation of the Producer/Consumer pattern? * 5/5

- ○ as many as the total number of threads in the system
- ● 2 ✓
- ○ 1
- ○ none
- ○ as many as the number of slots in the shared buffer

✓ If multiple threads are waiting on a condition variable, and one other thread calls signal() on that condition variable, then which of the waiting thread(s) will be awoken? * 5/5

- ○ at least one, maybe more
- ○ the thread that started to wait most recently
- ● one will be chosen randomly ✓
- ○ the thread that started to wait earliest

✓ When you need your threads to take turns then you should use condition variables * 5/5

- ● True ✓
- ○ False

✓ Use of mutex is optional when using condition variables * 5/5

True [✓ False] ✓

✓ When a thread returns from a wait() call it will be holding the associated mutex * 5/5

[✓ True] ✓ False

✓ Data races are the most common type of concurrency bug * 5/5

[✓ True] ✓ False

✓ Fix times for concurrency bugs tend to vary more than for other types of bugs. * 5/5

[✓ True] ✓ False

✓ Atomicity Violation bugs generally can be fixed with proper use of mutex locks. * 5/5

[✓ True] ✓
False

Which of the following types of bugs are types of Concurrency Bug? *

	Concurrency Bug	Other type of bug	Score	
Order Violation	●	○	1/1	✓
Segmentation Fault	○	●	1/1	✓
Data Race	●	○	1/1	✓
Illegal Instruction	○	●	1/1	✓
beetle	○	●	1/1	✓
Atomicity Violation	●	○	1/1	✓
Starvation	●	○	1/1	✓
Infinite Recursion	○	●	1/1	✓
Deadlock	●	○	1/1	✓
Null Pointer Dereference	○	●	1/1	✓

Which of the following are essential for deadlock to occur? *

	Essential for deadlock	Not essential for deadlock	Score	
circular chain of locks	●	○	1/1	✓
paging	○	●	1/1	✓
condition variables	○	●	1/1	✓
mutual exclusion	●	○	1/1	✓
no pre-emption	●	○	1/1	✓
offending Bruce's dog	○	●	1/1	✓
recursion	○	●	1/1	✓
hold one lock while waiting for another	●	○	1/1	✓

✓ A program fails to make progress because a thread is holding something (a lock) while trying to acquire something else (another lock). This is an example of what type of concurrency bug? * 5/5

- ○ Order Violation
- ○ Atomicity Violation
- ○ Starvation
- [✓ Deadlock] ✓

✓ Database Servers sometimes use ______________ to reduce the negative effects of deadlocks. * 5/5

- [✓ deadlock detection] ✓
- random ordering of locks
- non-preemptible locks
- mutual exclusion locks
- hold-and-wait