CS 544: Intro to Big Data Systems

Fall 2025 Worksheets

Worksheet 1: Linux Notes

As each tool or operator is introduced in lecture, write a brief note describing what it does in your own words.

1. ssh: _____
2. pwd: _____
3. ls: _____
4. touch: _____
5. nano/vim/emacs: _____
6. apt: _____
7. wget: _____
8. mv: _____
9. cp: _____
10. scp: _____
11. cat: _____
12. head/tail: _____
13. mkdir: _____
14. man: _____
15. cd: _____
16. sudo/su: _____
17. chmod: _____
18. python3: _____
19. which: _____
20. export: _____
21. echo: _____
22. |: _____
23. >: _____
24. >>: _____

25. &>: _____
26. wc: _____
27. grep: _____
28. find: _____
29. &: _____
30. ps: _____
31. kill: _____
32. pkill: _____
33. htop: _____
34. df: _____
35. du: _____
36. ss: _____

Worksheet 2: Docker Notes

As each Docker command or directive is introduced in lecture, write a brief note describing what it does in your own words.

docker COMMAND

pull: _____

images: _____

tag: _____

run: _____

ps: _____

rm: _____

rmi: _____

system df: _____

system prune: _____

logs: _____

exec: _____

stats: _____

kill: _____

stop: _____

build: _____

Dockerfile INSTRUCTIONS

FROM: _____

RUN: _____

COPY: _____

CMD: _____

Worksheet 3: Cache Policy

Problem 1

FIFO, size 2:

Data	Hit?
1	
2	
1	
3	
1	

Hit Rate: _____

Problem 2

LRU, size 2:

Data	Hit?
A	
B	
A	
C	
A	

Hit Rate: _____

Problem 3

FIFO, size 3:

Data	Hit?
W	
W	
X	
Y	
Y	
Z	
Y	
X	

Hit Rate: _____

Miss latency: 20 ms

Hit latency: 0.1 ms

Average latency: _____

Problem 4 LRU, size 4:

Data	Hit?
3	
4	
5	
6	
7	
3	
4	
5	
6	
7	

Hit Rate: _____

Problem 5 LRU, size 5:

Data	Hit?
3	
4	
5	
6	
7	
3	
4	
5	
6	
7	

Hit Rate: _____

Worksheet 4: Race Conditions

```
thread 1
if x >= 20:
    y += 20
    x -= 20
```

```
thread 2
if x >= 10:
    z += 10
    x -= 10
```

thread 1	thread 2	x	y	z
---	---	25	100	200
if x >= 20				
y += 20			[ ]	
	if x >= 10			
	z += 10			[ ]
x -= 20		[ ]		
	x -= 10	[ ]		

time

Problem 1: Fill in the above boxes to indicate the variable changes.

thread 1	thread 2	x	y	z
---	---	25	100	200

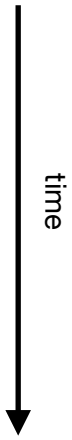
time

Problem 2: Fill in a timeline above so that x is always >= 0 and z ends at 210.

```
thread 1 (T1)
load total
load 1
add
store total
```

```
thread 2 (T2)
load total
load 1
add
store total
```

thread 1	thread 2	total	T1 vals	T2 vals
---	---	5		



Problem 3: Choose a bytecode-level interleaving above to finish with total=6.

```
thread 1 (T1)
load total
load 2
add
store total
```

```
thread 2 (T2)
load total
load 3
add
store total
```

Problem 4: Assume any bytecode-level interleaving is possible, and total starts at 0. What is the SMALLEST possible final value for total?

Worksheet 5: Locks

```
thread 1
lock.acquire()
L.append(3)
x += 1
lock.release()
```

```
thread 2
y += 1
y += 2
lock.acquire()
diff = len(L) - x
lock.release()
```

thread 1	thread 2	x	L	diff	y
---	---	2	[5,4]	None	4
	y += 1				5
lock.aquire()					
	y += 2				7
L.append(3)			[5,4,3]		
	lock.acquire()				
	diff = len(L) - x			1	
	lock.release()				
x += 1		3			
lock.release()					

Problem 1: thanks to locking, the correct timeline is IMPOSSIBLE. Circle the FIRST statement executed in the timeline that could not possibly be executed at that time given locking rules. Then cross out everything that occurs after that.

```
thread 1
if q != 0:          #A
    lock.acquire()  #B
    r = 1/q         #C
    lock.release()  #D
```

```
thread 2
lock.acquire() #X
q = 0          #Y
lock.release() #Z
```

Problem 2: assume q is 2 before the threads start running. Write out an interleaving (for example, something like A, B, C, ...) that leads to an ZeroDivisionError.

```
lock = threading.Lock()
x = 1

def task():
     global x
     with lock:
          x = 2


t = threading.Thread(target=task)
a = x
t.start()
with lock:
     b = x
t.join()
c = x
```

a = _____

b = _____

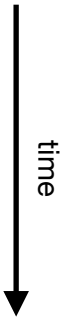
c = _____

Problem 3: how do a, b, and c end? Write "?" if it is impossible to know.

```
thread 1
lockA.acquire()
lockB.acquire()
A += 1
B -= 1
lockA.release()
lockB.release()
```

```
thread 2
lockB.acquire()
lockA.acquire()
B += 2
A -= 2
lockB.release()
lockA.release()
```

thread 1	thread 2	A	B
---	---	30	40



Problem 4: write an interleaving that leads to "deadlock" (both threads blocked).

Worksheet 6: Cassandra

Token Map:
token(n1) = {-2, 4} token(n2) = {-6, 0} token(n3) = {-4, 2, 5}

Problem 1: how many *nodes* are there? How many *vnodes*?

Problem 2: which node likely has greater resources (compute, memory, etc.)?

Problem 3: one of the vnode positions of n2 is drawn in the ring below. Draw the rest.

```
        n2
-8|-7|-6|-5|-4|-3|-2|-1| 0| 1| 2| 3| 4| 5| 6| 7
```

Problem 4: what ring positions are in the *wrapping range*? Draw the region above.

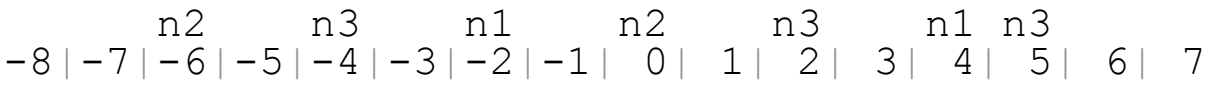
Problem 5: what node is responsible for each of the following tokens?

4: _____, 1: _____, 6: _____

Problem 6: a row's *primary key* is ("A", "B"). The primary key consists of one partition column followed by one cluster column. Which node owns this row? Assume token("A") = -3, token("B") = -6, and token(("A", "B")) = 3.

Problem 7: assume a new node n4 joins the cluster with vnodes -3 and -1. Which existing nodes will pass off some data to this new node?

Ring (this is the same as the previous page, filled in for you):



Problem 8: assuming 2x replication, what are the positions of the vnodes responsible for a row with token -1?

Problem 9: assuming 3x replication, what are the positions of the vnodes responsible for a row with token 1?

Problem 10: assume R=2, W=2, and RF=3. Assume the token of a row being written is -3. To which nodes will the coordinator attempt to write the data?

Problem 11: assume R=2, W=2, and RF=3. Assume the token of a row being written is -3. The timeline is as follows:

1. n1 is down
2. the row is written
3. n1 recovers, but n3 crashes
4. the row is read

Which nodes perform reads?

Which nodes perform writes?

Is the data that was written read back?

Problem 12: W=3 and RF=4. What should R be to make sure readers see successful writes?