Task 1: [14pts]

1. A transaction might **access** or **update** after completing all its actions.

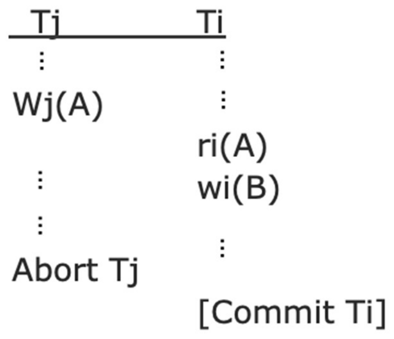
2. What are Atomic transactions?(describe in 1 sentence)
   Either all operations of the transaction are properly reflected in the database or none are.

3. Multiple users can submit transactions. [True/False]
   True

4. Which transaction property in ACID is ensured by adding concurrency control to transactions.
   Isolation

5. Schedule is conflict serializable if and only if its dependency graph is acyclic. [True/False]
   True

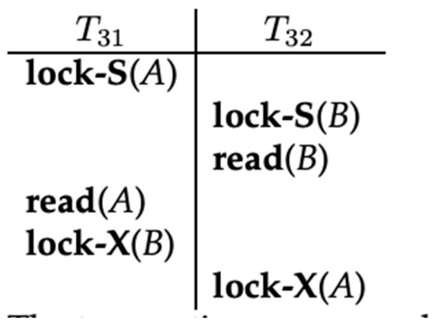
6. What is the risk in the following schedule of transactions T_i and T_j?



Tj	Ti
⋮	⋮
Wj(A)	⋮
	ri(A)
⋮	wi(B)
⋮	
Abort Tj	⋮
	[Commit Ti]

   Cascading Rollback

7. What is the risk in the following schedule:



T_{31}	T_{32}
lock-S(A)	
	lock-S(B)
	read(B)
read(A)	
lock-X(B)	
	lock-X(A)

   Deadlock

Task 2 [6 pts]

Transactions T1 and T2 execute in interleaved fashion. What anomalies do the following interleaved executions have?

Schedule 1:

```
T1:      R(A), W(A)                    R(B), W(B), Abort
T2:                    R(A), W(A), C
```

Dirty Read: T1 reads and writes to A, but it hasn't committed yet when T2 reads it.

Schedule 2:

```
T1:      W(A)                   W(B), C
T2:                W(A), W(B), C
```

Lost update problem: T2 writes to A immediately after T1 which didn't commit yet.

Task 3 [10 pts]

1. Is this schedule conflict serializable? No, A is being read by T2 immediately after it is written to by T1.

2. Show the dependency graph of T1 and T2.

```
T1:      R(A), W(A),                                  R(B), W(B)
T2:                          R(A), W(A), R(B), W(B)
```

$$\begin{tikzcd} T1 \arrow[r, bend left] & T2 \arrow[l, bend left] \end{tikzcd}$$

Task 4: [10 pts]

Consider the following schedule of transactions:

Transaction   Operations

T1	read(A), write(A)
T2	read(B), write(B)
T3	read(C), write(C)

Is this schedule view-serializable? Yes, all of the reads and writes are independent from one another.

Task 5 10pts

Consider the following two transactions:

```
$T_{31}$: read($A$);
      read($B$);
      **if** $A = 0$ **then** $B := B + 1$;
      write($B$).

$T_{32}$: read($B$);
      read($A$);
      **if** $B = 0$ **then** $A := A + 1$;
      write($A$).
```

No question is given here.

Task 6 10pts

1. Add lock and unlock instructions to transactions T31 and T32, so that they observe the two-phase locking protocol. Can the execution of these transactions result in a deadlock?

   Yes, for instance in the following instructions: lock-S(A), lock-S(B), read (B), read (A), lock-X(B), lock-X(A), T32 ends up holding lock-S(B) and won't release it.

2. What are the benefits of strict two-phase locking? What are its disadvantages?

   Some benefits are serializability and preventing deadlock. Some disadvantages are increased lock-hold times and overall high overhead.

Task 7 10 pts

Is this schedule conflict serializable? RA represents read on object A and WA represents write on object A

T1	T2	T3
RA		
	RB	
WA		
		RB
	WB	
		WB
	RA	
	WA	
COMMIT	COMMIT	COMMIT

No, because it is impossible to swap the WB of T2 with the WB of T3.

Task 8 5pts:

How do you check view serializability in a schedule of transactions?

1. If in schedule S, transaction Ti, reads the initial value of Q, then in schedule S' also transaction Ti must read the initial value of Q
2. If in schedule S transaction T, executes read(Q), and that value was produced by transaction Tk, then in schedule S' also transaction T, must read the value of Q that was produced by the same write (Q) operation of transaction Tj
3. The transaction (if any) that perform the final write(Q) operation in schedule S must also perform the final write(Q) operation in schedule S'