Part 3: Two-Phase Locking (20 points)

A) Now modify the above schedule by adding locks, which may block some transactions from doing their operations until the lock is released. You'll need to **rewrite** the above schedule in a table form. (The lecture slides show how to represent blocking in your schedules.)

Use two-phase locking in your modified schedule to ensure a conflict-serializable schedule for the transactions above.

Use the notation L(A) to indicate that the transaction acquires the lock on element A and U(A) to indicate that the transaction releases its lock on.

T1	T2	T3
L(A), L(B)	L(A) Blocked.. L(B) Blocked..	L(A) Blocked.. L(B) Blocked..
R(A) W(A)		
U(A)		..Granted L(A)
R(B) W(B)		R(A) W(A)
U(B), commit	..Granted L(A)	...Granted L(B), U(A)
	R(A)	R(B) W(B)
	...Granted L(B), U(A)	U(B), commit
	R(B)	
	U(B), commit	

B) If 2PL ensures conflict-serializability, why do we need strict 2PL? Explain briefly.

Strict 2PL ensures recoverability and also conflict-serializability while 2PL only ensures conflict-serializability. There sometimes may be deadlock occurs in 2PL while in strict 2PL, no such instance occurs, To add on, when a given transaction is happening, no other transaction can edit the database and rollback function is thus enabled when using strict 2PL.