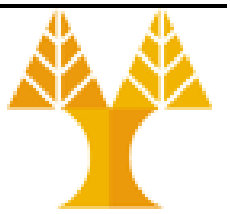

ΕΡΓΑΣΤΗΡΙΟ #9

ΜΕΡΟΣ Α΄: Concurrency control with Locking

1. Prove the following:
   a. Strict two-phase locking ensures strict schedules
   b. Cautious waiting avoids deadlocks.

Answer:

a. Since no other transaction can read or write an element that is written from a transaction T until T is completed, the requirement for strict schedules is met.

b. In cautious waiting, a transaction Ti can wait on a transaction Tj (and hence Ti becomes blocked) only if Tj is not blocked at that time, say time b(Ti), when Ti waits. Later, at some time b(Tj) > b(Ti), Tj can be blocked and wait on another transaction Tk only if Tk is not blocked at that time. However, Tj cannot be blocked by waiting on an already blocked transaction since this is not allowed by the protocol. Hence, the wait-for graph among the blocked transactions in this system will follow the blocking times and will never have a cycle, and so deadlock cannot occur.

2. For each of the following locking protocols specify the kind of schedules) that they (*conflict-serializable, recoverable, avoids-cascading-aborts, strict).*
   a. Get an exclusive lock before writing. Keep all exclusive locks until the end of the transaction. No shared locks are kept.
   b. In addition to (a), get a shared lock before reading. Shared locks can be released at any time.
   c. In addition to (b), we have two-phase locking.
   d. In addition to (b), all locks are kept until the end of the transaction.

Answer:

	Conflict-serializable	Recoverable	Avoid cascading aborts
a	No	No	No
b	No	Yes	Yes
c	Yes	Yes	Yes
d	Yes	Yes	Yes

ΜΕΡΟΣ Β΄: Concurrency control with timestamp

1. Apply the timestamp ordering algorithm to the following schedules and determine whether the algorithm will allow the execution of the schedules.

   a. Schedule E

Transaction T_1	Transaction T_2	Transaction T_3
	read_item(Z);	
	read_item(Y);	
	write_item(Y);	
		read_item(Y);
		read_item(Z);
read_item(X);		
write_item(X);		
		write_item(Y);
		write_item(Z);
	read_item(X);	
read_item(Y);		
write_item(Y);		
	write_item(X);	

   b. Schedule F

Transaction T_1	Transaction T_2	Transaction T_3
		read_item(Y);
		read_item(Z);
read_item(X);		
write_item(X);		
		write_item(Y);
		write_item(Z);
	read_item(Z);	
read_item(Y);		
write_item(Y);		
	read_item(Y);	
	write_item(Y);	
	read_item(X);	
	write_item(X);	

Answer:

a. Schedule E can be written as follows (assuming a linear clock and that read and write timestamps are initially 0):

$r_2(Z)$; $r_2(Y)$; $w_2(Y)$; $r_3(Y)$; $r_3(Z)$; $r_1(X)$; $w_1(X)$; $w_3(Y)$; $w_3(Z)$; $r_2(X)$; $r_1(Y)$; $w_1(Y)$; $w_2(X)$;
1 2 3 4 5 6 7 8 9 10 11 12 13

Initial Values

Schedule E
TS(T1) = 6
TS(T2) = 1
TS(T3) = 4

	Read TS	Write TS
X	0	0
Y	0	0
Z	0	0

T2: read_item(Z)
TS(T2) > write_TS(Z)
Εκτέλεσε το read_item(Z)
read_TS(Z) <- max(read_TS(Z),TS(T2)) = 1

	Read TS	Write TS
X	0	0
Y	0	0
Z	~~**0**~~**, 1**	0

T2: read_item(Y)
TS(T2) > write_TS(Y)
Εκτέλεσε το read_item(Y)
read_TS(Y) <- max(read_TS(Y),TS(T2)) = 1

	Read TS	Write TS
X	0	0
Y	~~**0**~~**, 1**	0
Z	~~**0**~~**, 1**	0

T2: write_item(Y)
TS(T2) = read_TS(Y) and TS(T2) > write_TS(Y)
Εκτέλεσε το write_item(Y)
write_TS(Y) <- max(write_TS(Y),TS(T2)) = 1

	Read TS	Write TS
X	0	0
Y	~~**0**~~**, 1**	~~**0**~~**, 1**
Z	~~**0**~~**, 1**	0

T3: read_item(Y)
TS(T3) > write_TS(Y)
Εκτέλεσε το read_item(Y)
read_TS(Y) <- max(read_TS(Y),TS(T3)) = 4

	Read TS	Write TS
X	0	0
Y	~~**0**~~**,** ~~**1**~~**, 4**	~~**0**~~**, 1**
Z	~~**0**~~**, 1**	0

T3: read_item(Z)
TS(T3) > write_TS(Z)
Εκτέλεσε το read_item(Z)
read_TS(Z) <- max(read_TS(Z),TS(T3)) = 4

	Read TS	Write TS
X	0	0
Y	~~**0**~~**,** ~~**1**~~**, 4**	~~**0**~~**, 1**
Z	~~**0**~~**,** ~~**1**~~**, 4**	0

T1: read_item(X)
TS(T1) > write_TS(X)
Εκτέλεσε το read_item(X)
read_TS(X) <- max(read_TS(X),TS(T1)) = 6

	Read TS	Write TS
X	~~**0**~~**, 6**	0
Y	~~**0**~~**,** ~~**1**~~**, 4**	~~**0**~~**, 1**
Z	~~**0**~~**,** ~~**1**~~**, 4**	0

T1: write_item(X)
TS(T1) = read_TS(X) and TS(T1) > write_TS(X)
Εκτέλεσε το write_item(X)
write_TS(X) <- max(write_TS(X),TS(T1)) = 6

	Read TS	Write TS
X	**~~0~~, 6**	**~~0~~, 6**
Y	**~~0~~, ~~1~~, 4**	**~~0~~, 1**
Z	**~~0~~, ~~1~~, 4**	0

T3: write_item(Y)
TS(T3) = read_TS(Y) and TS(T3) > write_TS(Y)
Εκτέλεσε το write_item(Y)
write_TS(Y) <- max(write_TS(Y),TS(T3)) = 4

	Read TS	Write TS
X	**~~0~~, 6**	**~~0~~, 6**
Y	**~~0~~, ~~1~~, 4**	**~~0~~, ~~1~~, 4**
Z	**~~0~~, ~~1~~, 4**	0

T3: write_item(Z)
TS(T3) = read_TS(Z) and TS(T3) > write_TS(Z)
Εκτέλεσε το write_item(Z)
write_TS(Z) <- max(write_TS(Z),TS(T3)) = 4

	Read TS	Write TS
X	**~~0~~, 6**	**~~0~~, 6**
Y	**~~0~~, ~~1~~, 4**	**~~0~~, ~~1~~, 4**
Z	**~~0~~, ~~1~~, 4**	**~~0~~, 4**

T2: read_item(X)
TS(T2) < write_TS(X)
Απόρριψε το read_item(X), Ακύρωσε και επανέφερε (Rollback) το T2

	Read TS	Write TS
X	**~~0~~, 6**	**~~0~~, 6**
Y	**~~0~~, ~~1~~, 4**	**~~0~~, ~~1~~, 4**
Z	**~~0~~, ~~1~~, 4**	**~~0~~, 4**

Since T3 read the value of Y written by T2, T3 must also be rollbacked from the recovery technique due to cascading rollback. Therefore all acts of T2, T3 will be deleted and only the T1 will execute.

b. Schedule F can be written as follows (assuming a linear clock and that read and write timestamps are initially 0):

r3(Y); r3(Z); r1(X); w1(X); w3(Y); w3(Z); r2(Z); r1(Y); w1(Y); r2(Y); w2(Y); r2(X); w2(X);
1 2 3 4 5 6 7 8 9 10 11 12 13

Initial Values

Schedule F
TS(T1) = 3
TS(T2) = 7
TS(T3) = 1

	Read TS	Write TS
X	0	0
Y	0	0
Z	0	0

T3: read_item(Y)
TS(T3) > write_TS(Y)
Εκτέλεσε το read_item(Y)
read_TS(Y) <- max(read_TS(Y),TS(T3)) = 1

	Read TS	Write TS
X	0	0
Y	**~~0~~, 1**	0
Z	0	0

T3: read_item(Z)
TS(T3) > write_TS(Z)
Εκτέλεσε το read_item(Z)
read_TS(Z) <- max(read_TS(Z),TS(T3)) = 1

	Read TS	Write TS
X	0	0
Y	**~~0~~, 1**	0
Z	**~~0~~, 1**	0

T1: read_item(X)
TS(T1) > write_TS(X)
Εκτέλεσε το read_item(X)
read_TS(X) <- max(read_TS(X),TS(T1)) = 3

	Read TS	Write TS
X	~~0~~, 3	0
Y	~~0~~, 1	0
Z	~~0~~, 1	0

T1: write_item(X)
TS(T1) = read_TS(X) and TS(T1) > write_TS(X)
Εκτέλεσε το write_item(X)
write_TS(X) <- max(write_TS(X),TS(T1)) = 3

	Read TS	Write TS
X	~~0~~, 3	~~0~~, 3
Y	~~0~~, 1	0
Z	~~0~~, 1	0

T3: write_item(Y)
TS(T3) = read_TS(Y) and TS(T3) > write_TS(Y)
Εκτέλεσε το write_item(Y)
write_TS(Y) <- max(write_TS(Y),TS(T3)) = 1

	Read TS	Write TS
X	~~0~~, 3	~~0~~, 3
Y	~~0~~, 1	~~0~~, 1
Z	~~0~~, 1	0

T3: write_item(Z)
TS(T3) = read_TS(Z) and TS(T3) > write_TS(Z)
Εκτέλεσε το write_item(Z)
write_TS(Z) <- max(write_TS(Z),TS(T3)) = 1

	Read TS	Write TS
X	~~0~~, 3	~~0~~, 3
Y	~~0~~, 1	~~0~~, 1
Z	~~0~~, 1	~~0~~, 1

T2: read_item(Z)
TS(T2) > write_TS(Z)
Εκτέλεσε το read_item(Z)
read_TS(Z) <- max(read_TS(Z),TS(T2)) = 7

	Read TS	Write TS
X	~~0~~, 3	~~0~~, 3
Y	~~0~~, 1	~~0~~, 1
Z	~~0~~, ~~1~~, 7	~~0~~, 1

T1: read_item(Y)
TS(T1) > write_TS(Y)
Εκτέλεσε το read_item(Y)
Set read_TS(Y) <- max(read_TS(Y),TS(T1)) = 3

	Read TS	Write TS
X	~~0~~, 3	~~0~~, 3
Y	~~0~~, ~~1~~, 3	~~0~~, 1
Z	~~0~~, ~~1~~, 7	~~0~~, 1

T1: write_item(Y)
TS(T1) = read_TS(Y) and TS(T1) > write_TS(Y)
Εκτέλεσε το write_item(Y)
write_TS(Y) <- max(read_TS(Y),TS(T1)) = 3

	Read TS	Write TS
X	~~0~~, 3	~~0~~, 3
Y	~~0~~, ~~1~~, 3	~~0~~, ~~1~~, 3
Z	~~0~~, ~~1~~, 7	~~0~~, 1

T2: read_item(Y)
TS(T2) > write_TS(Y)
Εκτέλεσε το read_item(Y)
Set read_TS(Y) <- max(read_TS(Y),TS(T2)) = 7

	Read TS	Write TS
X	~~0~~, 3	~~0~~, 3
Y	~~0~~, ~~1~~, ~~3~~, 7	~~0~~, ~~1~~, 3
Z	~~0~~, ~~1~~, 7	~~0~~, 1

T2: write_item(Y)
TS(T2) = read_TS(Y) and TS(T2) > write_TS(Y)
Εκτέλεσε το write_item(Y)
write_TS(Y) <- max(write_TS(Y),TS(T2)) = 7

	Read TS	Write TS
X	~~0~~, 3	~~0~~, 3
Y	~~0~~, ~~1~~, ~~3~~, 7	~~0~~, ~~1~~, ~~3~~, 7
Z	~~0~~, ~~1~~, 7	~~0~~, 1

T2: read_item(X)
TS(T2) > write_TS(X)
Εκτέλεσε το read_item(X)
Set read_TS(X) <- max(read_TS(X),TS(T2)) = 7

	Read TS	Write TS
X	~~**0**~~, ~~**3**~~, **7**	~~**0**~~, **3**
Y	~~**0**~~, ~~**1**~~, ~~**3**~~, **7**	~~**0**~~, ~~**1**~~, ~~**3**~~, **7**
Z	~~**0**~~, ~~**1**~~, **7**	~~**0**~~, **1**

T2: write_item(X)
TS(T2) = read_TS(X) and TS(T2) > write_TS(X)
Εκτέλεσε το write_item(X)
write_TS(X) <- max(write_TS(X),TS(T2)) = 7

	Read TS	Write TS
X	~~**0**~~, ~~**3**~~, **7**	~~**0**~~, ~~**3**~~, **7**
Y	~~**0**~~, ~~**1**~~, ~~**3**~~, **7**	~~**0**~~, ~~**1**~~, ~~**3**~~, **7**
Z	~~**0**~~, ~~**1**~~, **7**	~~**0**~~, **1**

Schedule F executes normally