Question 1:

i: Composite attributes can be subdivided into smaller parts while simple attributes themselves are already minimal attributes; Composite attributes search and store data efficiently.

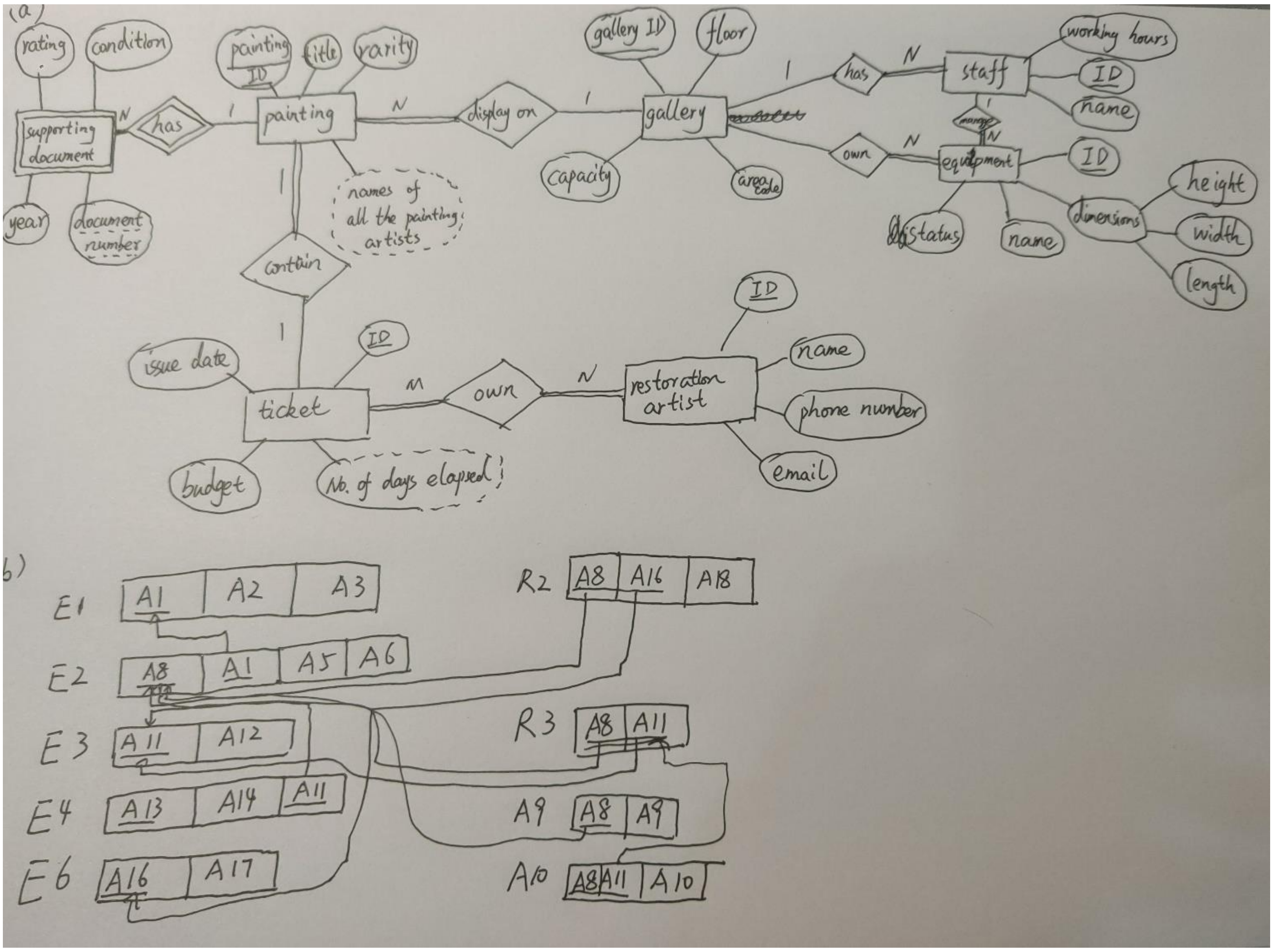
ii: If attributes in R1(R2) contain or intersect on that in another relation, the result will be different; if attributes in R1(R2) disjoint to that in another relation, the result will be the same.

iii: example: A transferred 50 Australian dollars to B, first A's account -50 then B's account +50, but the second step error, so the system will rollback(A's account +50 to the initial state) counterexample : A transferred 50 Australian dollars to B, first A's account -50 then B's account +50, but the second step error, so the system will stop which make 50 Australian dollars missed.

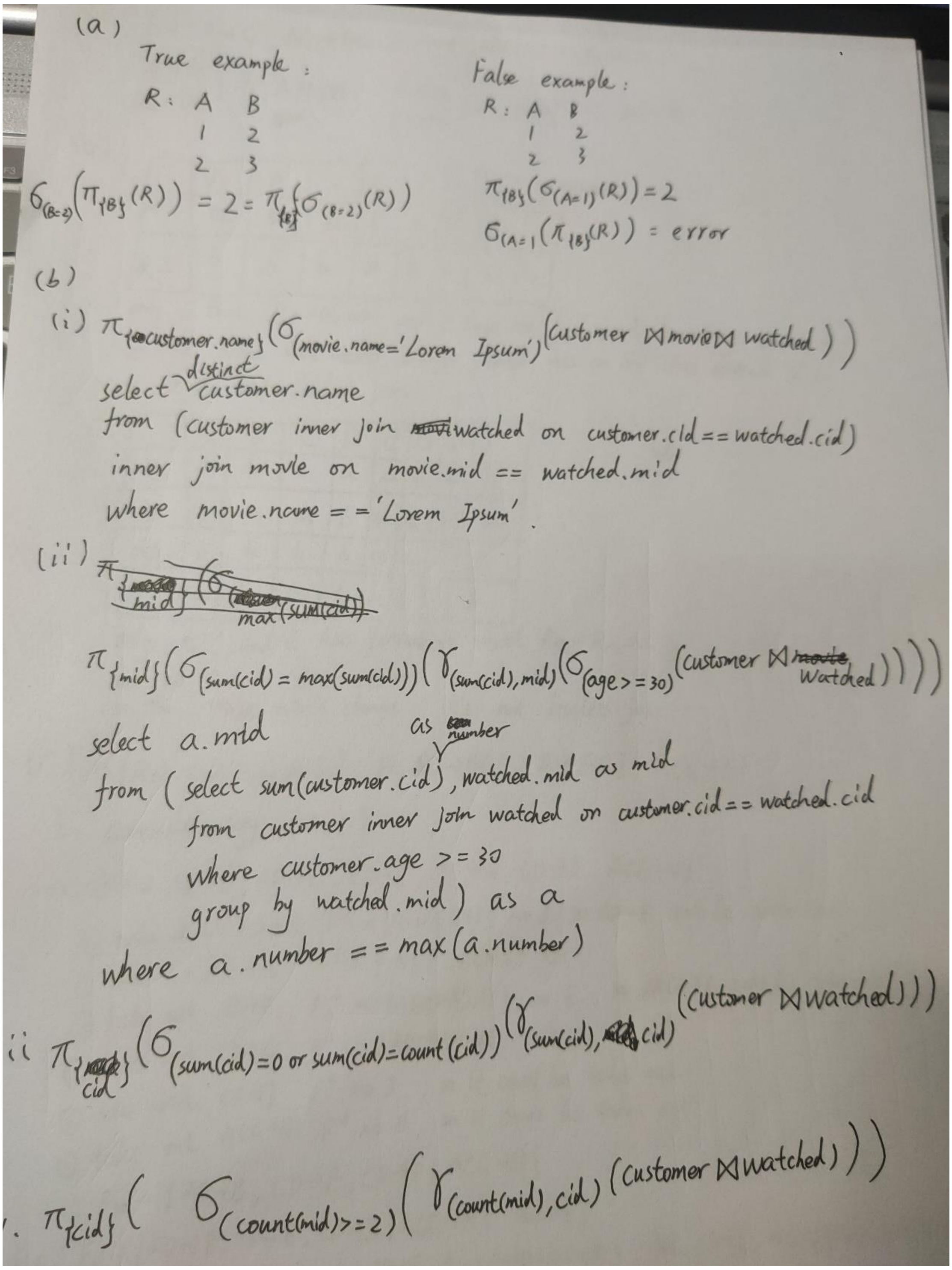
iv: 3NF has no transfer function dependencies. There are no properties in the relationship that depend on non-primary keys. This means that all non-key attributes are completely function dependent on the key which satisfied with 2NF.

v: (b)the latter. Because the former will get not only the maximum value of A but also other values of A, only the first row of the result is needed.

Question2:


(a)
rating
condition
supporting document
has
N
1
year
document number
painting ID
title
rarity
painting
N
display on
1
names of all the painting artists
1
contain
1
gallery ID
floor
gallery
capacity
area code
1
has
N
staff
working hours
ID
name
1
manage
N
own
N
equipment
ID
status
name
dimensions
height
width
length
ID
issue date
ticket
M
own
N
restoration artist
ID
name
phone number
email
budget
No. of days elapsed
b)
E1
A1
A2
A3
E2
A8
A1
A5
A6
E3
A11
A12
E4
A13
A14
A11
E6
A16
A17
R2
A8
A16
A18
R3
A8
A11
A9
A8
A9
A10
A8
A11
A10

Question3:



(a)

True example:

R:

A	B
1	2
2	3

$\sigma_{(B=2)}(\pi_{\{B\}}(R)) = 2 = \pi_{\{B\}}(\sigma_{(B=2)}(R))$

False example:

R:

A	B
1	2
2	3

$\pi_{\{B\}}(\sigma_{(A=1)}(R)) = 2$

$\sigma_{(A=1}(\pi_{\{B\}}(R)) = error$

(b)

(i) $\pi_{\{customer.name\}}(\sigma_{(movie.name='Lorem\ Ipsum')}(Customer \bowtie movie \bowtie watched))$

```
select distinct customer.name
from (customer inner join watched on customer.cid == watched.cid)
 inner join movie on movie.mid == watched.mid
 where movie.name == 'Lorem Ipsum' .
```

(ii) ~~$\pi_{\{mid\}}(\sigma_{max(sum(cid))}$~~

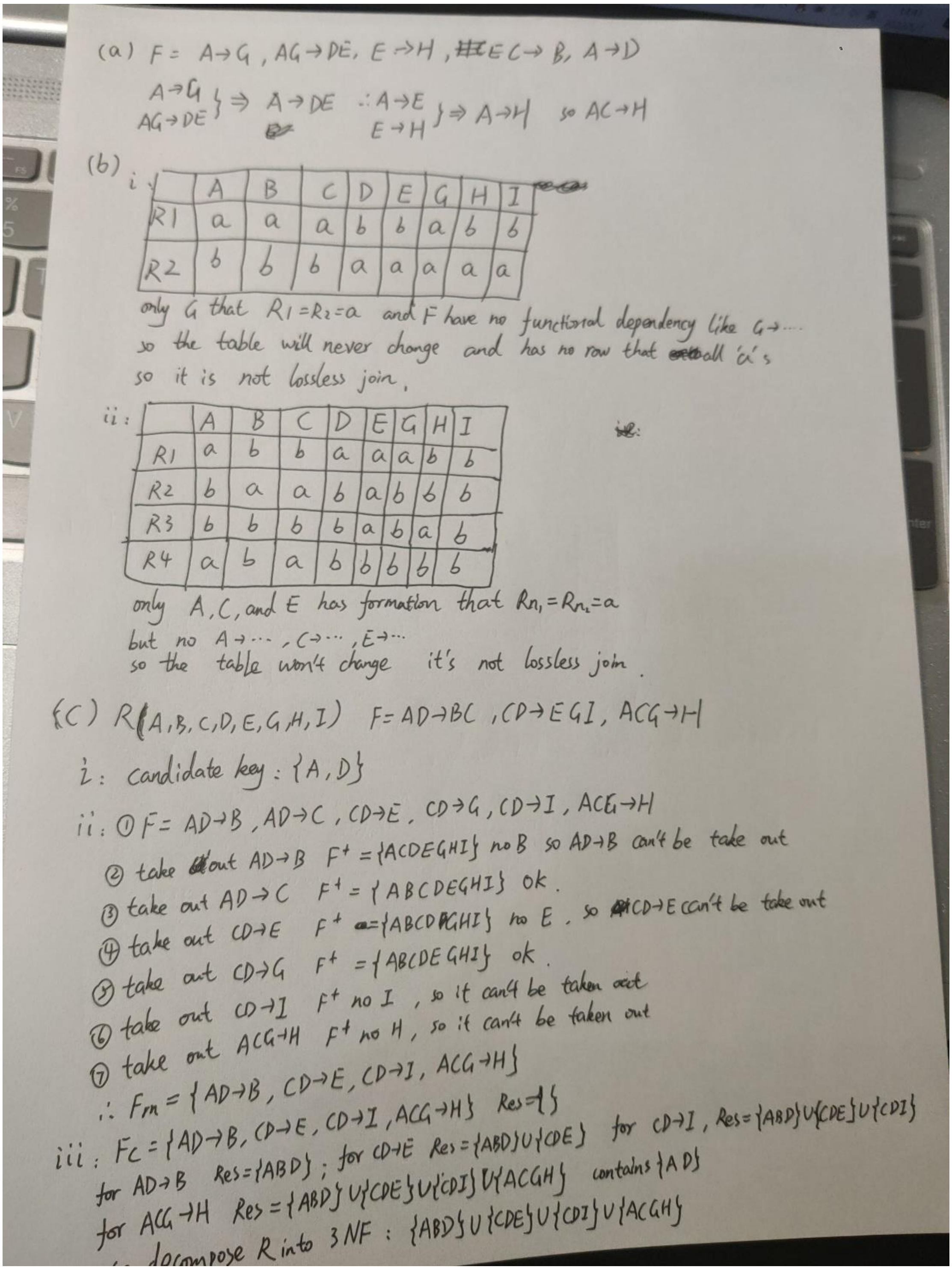
$\pi_{\{mid\}}(\sigma_{(sum(cid) = max(sum(cid)))}(\gamma_{(sum(cid), mid)}(\sigma_{(age>=30)}(customer \bowtie ~~movie~~ Watched))))$

```
select a.mid
from ( select sum(customer.cid) as number, watched.mid as mid
       from customer inner join watched on customer.cid == watched.cid
       where customer.age >= 30
       group by watched.mid) as a
 where a.number == max(a.number)
```

ii $\pi_{\{cid\}}(\sigma_{(sum(cid)=0\ or\ sum(cid)=count(cid))}(\gamma_{(sum(cid), cid)}(custoner \bowtie watched)))$

. $\pi_{\{cid\}}(\sigma_{(count(mid)>=2)}(\gamma_{(count(mid), cid)}(customer \bowtie watched)))$

Question 4:



(a) F = A→G , AG→DE, E→H , EC→ B, A→D

A→G, AG→DE } ⇒ A→DE ∴ A→E, E→H } ⇒ A→H so AC→H

(b) i:

	A	B	C	D	E	G	H	I
R1	a	a	a	b	b	a	b	b
R2	b	b	b	a	a	a	a	a

only G that $R_1 = R_2 = a$ and F have no functional dependency like G→....
so the table will never change and has no row that all 'a's
so it is not lossless join.

ii:

	A	B	C	D	E	G	H	I
R1	a	b	b	a	a	a	b	b
R2	b	a	a	b	a	b	b	b
R3	b	b	b	b	a	b	a	b
R4	a	b	a	b	b	b	b	b

only A, C, and E has formation that $R_{n_1} = R_{n_2} = a$
but no A→··· , C→··· , E→···
so the table won't change it's not lossless join.

(c) R(A,B,C,D,E,G,H,I) F= AD→BC , CD→EGI, ACG→H

i: candidate key : {A, D}

ii: ① F = AD→B , AD→C , CD→E , CD→G , CD→I , ACG→H

② take out AD→B F^+ = {ACDEGHI} no B so AD→B can't be take out

③ take out AD→C F^+ = {ABCDEGHI} ok.

④ take out CD→E F^+ = {ABCDGHI} no E . so CD→E can't be take out

⑤ take out CD→G F^+ = {ABCDEGHI} ok.

⑥ take out CD→I F^+ no I , so it can't be taken out

⑦ take out ACG→H F^+ no H, so it can't be taken out

∴ F_m = {AD→B, CD→E, CD→I, ACG→H} Res={}

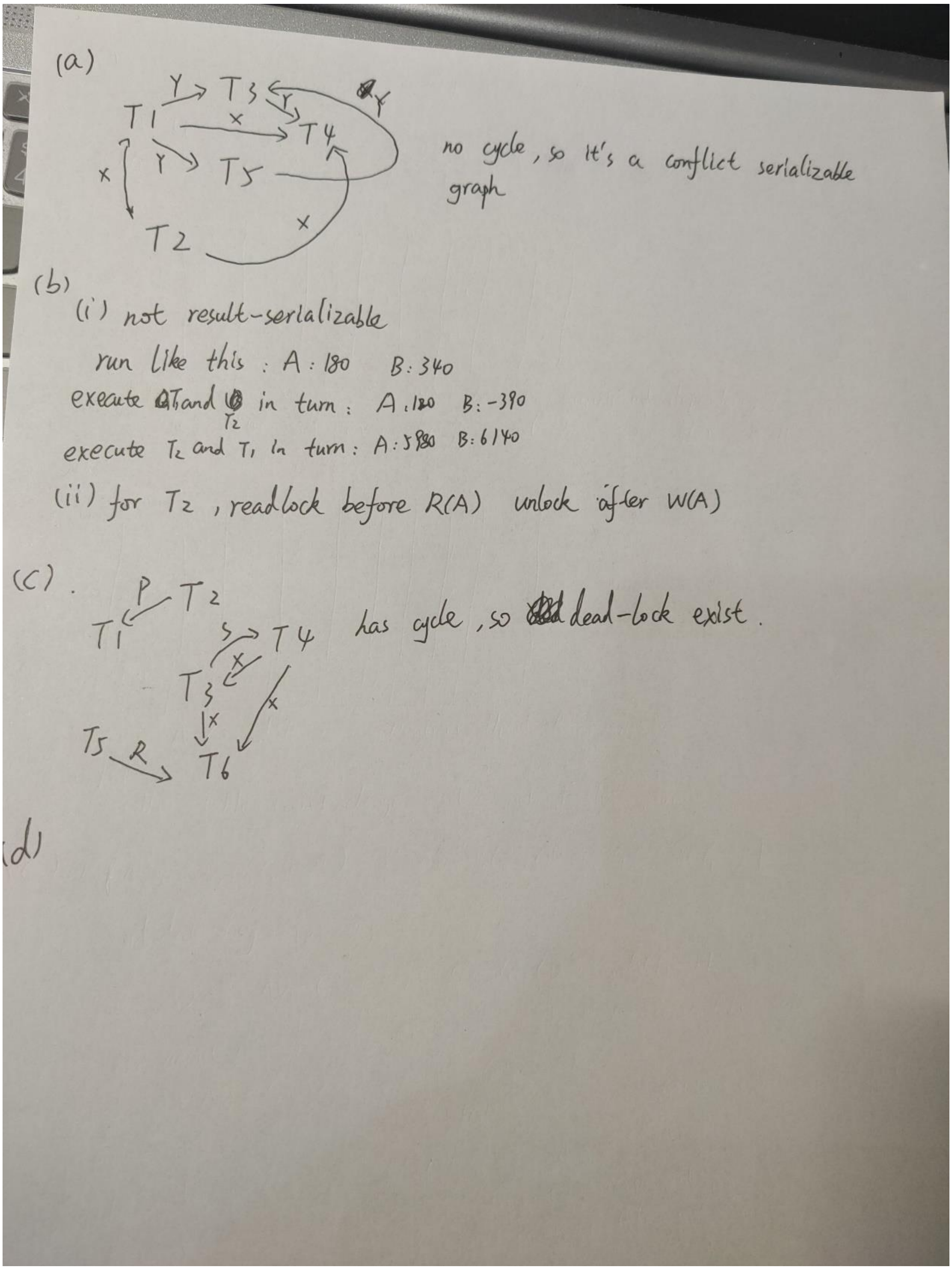
iii: F_c = {AD→B, CD→E, CD→I, ACG→H} for CD→E Res={ABD}U{CDE} for CD→I, Res={ABD}U{CDE}U{CDI}

for AD→B Res={ABD};

for ACG→H Res={ABD}U{CDE}U{CDI}U{ACGH} contains {AD}

∴ decompose R into 3NF : {ABD}U{CDE}U{CDI}U{ACGH}

Question 5:



(a)

T1
T2
T3
T4
T5
Y
X
Y
X
Y
Y
X

no cycle, so It's a conflict serlalizable graph

(b)

(i) not result-serlalizable

run like this : A: 180 B: 340

execute T_1 and T_2 in turn : A: 180 B: -390

execute T_2 and T_1 In turn : A: 5980 B: 6140

(ii) for T_2, readlock before R(A) unlock after W(A)

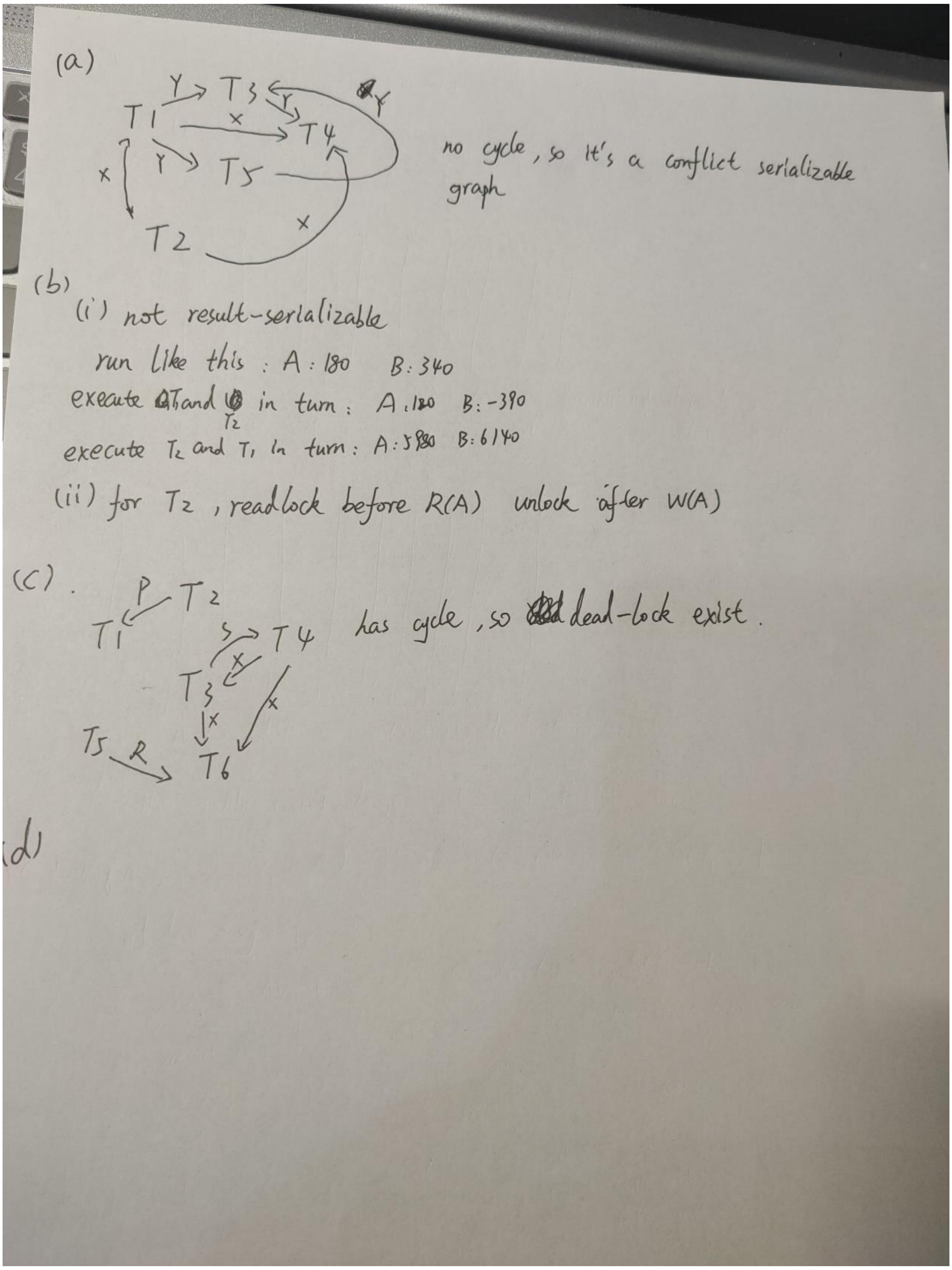
(c).

T1
T2
T3
T4
T5
T6
P
S
X
X
X
R

has cycle, so dead-lock exist.

(d)

Question 6:



(a)

T1
T2
T3
T4
T5
Y
X
Y
X
Y
X
Y

no cycle, so it's a conflict serlalizable graph

(b)

(i) not result-serlalizable

run like this : A: 180 B: 340

execute T1 and T2 in turn : A: 180 B: -390

execute T2 and T1 in turn : A: 5980 B: 6140

(ii) for T2 , readlock before R(A) unlock after W(A)

(c).

P
T1
T2
S
T4
X
T3
X
X
T5
R
T6

has cycle, so dead-lock exist.

(d)