C Instructions DBII

Uppsala University

Department of Information Technology

Database Design II (1DL400)

Instructions: Read through the complete exam and note below any unclear directives before you start solving the questions. Answer **all** questions.

The paper has two types of questions:

- If a **question is marked with ♥** you must **select ALL correct choices**. If you do not select all correct choices or you include any incorrect choice, your answer will be marked as incorrect.
- For all **other questions** you must **select only one choice** even if there are several correct choices. Your answer will be marked as correct if you select any of the correct choices. If you select an incorrect choice or select more than one choice, your answer will be marked as incorrect.

Please also answer questions: ♣ Q1, Q2 and Q3 which can be useful to us.

Grading. For each correct answer, you gain 1 point. A wrong answer does not generate negative points.

To achieve a grade of 3, you must gain at least 14 points in the whole exam. To achieve a grade of 4, you must gain at least 17 points in the whole exam. To achieve a grade of 5, you must collect at least 21 points in the whole exam.

If you find any unclear directives, please note the question below and explain what you think is unclear.

1 ♣ C Question G1: When

General questions (useful for us)

When have you attended the course?

Select one alternative (no points awarded for this question):

- ◯ 2022
- ◯ 2021
- ◯ 2020
- ◯ Before 2020

Maximum marks: 0

2 ♣ C Question G2: How many

General questions (useful for us)

How many lectures have you attended?

Select one alternative (no points awarded for this question):

- ◯ None or very few
- ◯ Around 25%
- ◯ Around 50%
- ◯ Around 75%
- ◯ Almost all

Maximum marks: 0

3 ♣ C Question G3: Study program

General questions (useful for us)

What is your study program?
Select one alternative (no points awarded for this question):

- ○ F
- ○ STS
- ○ CS
- ○ X
- ○ IT
- ○ None of the previous answers

Maximum marks: 0

4 C Joins size

Consider the relations ***r*1(<u>*A*</u>, *B, C*)**, ***r*2(<u>C</u>, D, E)** and ***r*3(<u>E</u>, F),** with primary keys ***A***, ***C***, ***E*** respectively. Assume that ***r*1** has **4,000** tuples, ***r*2** has **2,500** tuples, ***r*3** has **500** tuples.

What is the estimated size of **r1 ⋈ r2 ⋈ r3**?

Select one alternative:

- ○ 4,000
- ○ None of the other choices is a correct answer!
- ○ 3,000*500
- ○ 500
- ○ 2,500

Maximum marks: 1

5 ♥ C Joins

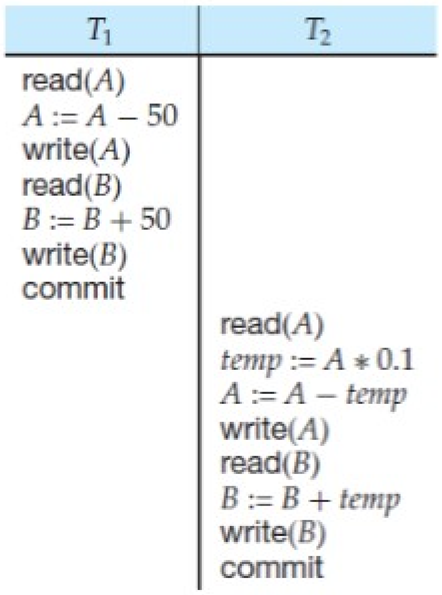
Let relations **r1(A, B, C)** and **r2(C, D, E)** have the following properties: **r1** has 20,000 tuples, **r2** has 45,000 tuples; 25 tuples of **r1** fit on one block, and 30 tuples of **r2** fit on one block. Consider that we want to join **r1** and **r2** using the **nested-loop join** algorithm, which of the following is/are true:

Select one or more alternatives:

- ☐ In the case where at least one relation can fit in memory, the most efficient application of the algorithm will require 800 +1,500 block transfers.
- ☐ In the worst case (e.g., the two relations cannot fit in memory), the most efficient application of the algorithm will require 1,500*800 +800 block transfers.
- ☐ In the worst case (e.g., the two relations cannot fit in memory), the most efficient application of the algorithm will require 800 +20,000*1,500 block transfers.
- ☐ In the best case (e.g. both relations can fit in memory), the most efficient application of the algorithm will require 800 +1,500 block transfers.
- ☐ In the best case (e.g. both relations can fit in memory), the most efficient application of the algorithm will require 1,500*800 +800 block transfers.
- ☐ In the worst case (e.g., the two relations cannot fit in memory), the most efficient application of the algorithm will require 1,500+800 block transfers.

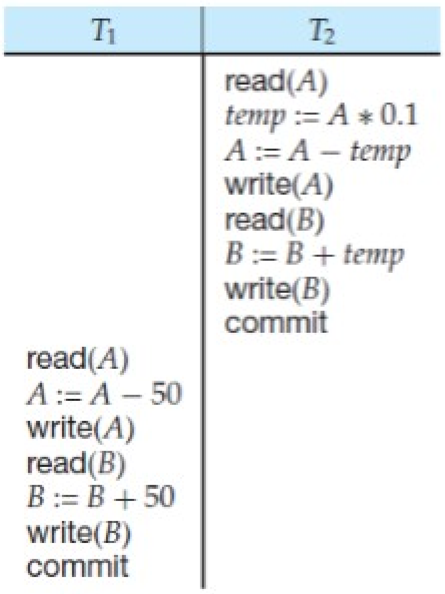
Maximum marks: 1

Considering the following transactions and schedules, answer the following questions.



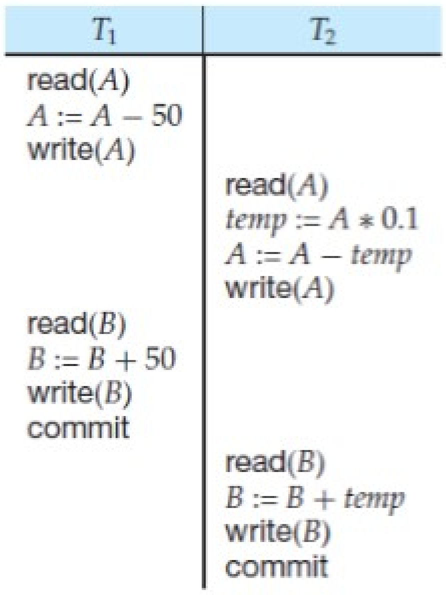
T_1	T_2
read(A)	
$A := A - 50$	
write(A)	
read(B)	
$B := B + 50$	
write(B)	
commit	
	read(A)
	$temp := A * 0.1$
	$A := A - temp$
	write(A)
	read(B)
	$B := B + temp$
	write(B)
	commit

Schedule A



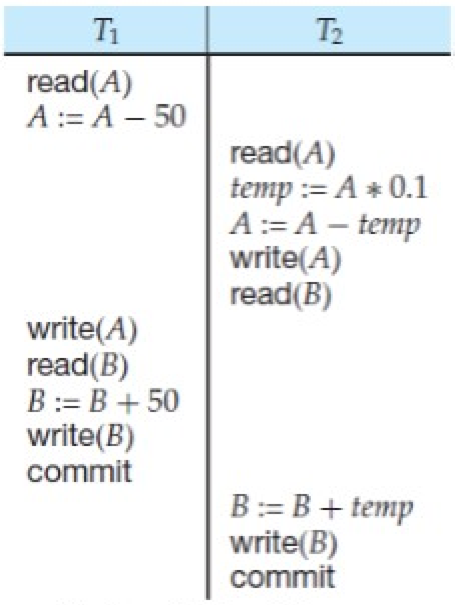
T_1	T_2
	read(A)
	$temp := A * 0.1$
	$A := A - temp$
	write(A)
	read(B)
	$B := B + temp$
	write(B)
	commit
read(A)	
$A := A - 50$	
write(A)	
read(B)	
$B := B + 50$	
write(B)	
commit	

Schedule B



T_1	T_2
read(A)	
$A := A - 50$	
write(A)	
	read(A)
	$temp := A * 0.1$
	$A := A - temp$
	write(A)
read(B)	
$B := B + 50$	
write(B)	
commit	
	read(B)
	$B := B + temp$
	write(B)
	commit

Schedule C

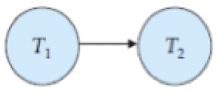


T_1	T_2
read(A)	
$A := A - 50$	
	read(A)
	$temp := A * 0.1$
	$A := A - temp$
	write(A)
	read(B)
write(A)	
read(B)	
$B := B + 50$	
write(B)	
commit	
	$B := B + temp$
	write(B)
	commit

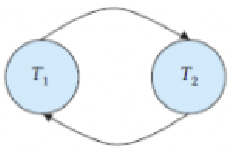
Schedule D

6 C Transactions

Considering the following Precedence graphs, which of the following is true?


$$\begin{tikzcd} T_1 \arrow[r] & T_2 \end{tikzcd}$$

Precedence Graph A1


$$\begin{tikzcd} T_1 \arrow[r, bend left] & T_2 \arrow[l, bend left] \end{tikzcd}$$

Precedence Graph B1

Select one alternative:

- ○ Precedence graph A1 corresponds to Schedule A, B and C.
- ○ Precedence graph A1 corresponds to Schedule D.
- ○ Precedence graph B1 corresponds to Schedule D.
- ○ None of the other answers is correct!

Maximum marks: 1

7 ♥ C Transactions

Consider the two serial schedules A and B. Which of the following are correct?

Select one or more alternatives:

- ☐ Schedule C is serializable to Schedule A
- ☐ Schedule D suffers from the lost update problem.
- ☐ Schedule D is serializable to Schedule A
- ☐ Schedule C suffers from the lost update problem.
- ☐ Schedule D is serializable to Schedule B
- ☐ Schedule C is serializable to Schedule B

Maximum marks: 1

8 ♥ C Serializability

For the following sets of transactions T_1, T_2, and T_3, which of the schedules are (conflict) serializable?:

♥ Select one or more alternatives:

T1	T2	T3
	read_item(A);	
	A := A + 10	
	write_item(A);	
read_item(A);		
A := A * 3;		
write_item(A);		
read_item(B);		
B := B – 5;		
write_item(B);		
		read_item(C);
		C := C / 5;
		write_item(C)
		read_item(A)
		A := A / 3;
		write_item(A)

☐

T1	T2	T3
	read_item(A);	
	A := A + 10	
		read_item(C);
		C := C / 5;
		write_item(C)
		read_item(A)
read_item(A);		
A := A * 3;		
write_item(A);		
read_item(B);		
B := B – 5;		
write_item(B);		
	write_item(A);	
		A := A / 3;
		write_item(A)

☐

T1	T2	T3
read_item(A);		
A := A * 5;		
write_item(A);		
	read_item(A);	
read_item(B);		
B := B – 10;		
write_item(B);		
	A := A + 10	
		read_item(C);
		C := C / 10;
		write_item(C)
	write_item(A);	
		read_item(A)
		A := A / 5;
		write_item(A)

☐

T1	T2	T3
read_item(A);		
		read_item(C);
	read_item(A);	
A := A * 5;		
write_item(A);		
read_item(B);		
B := B – 10;		
write_item(B);		
	A := A + 10	
		C := C / 10;
		write_item(C)
	write_item(A);	
		read_item(A)
		A := A / 5;
		write_item(A)

☐

Maximum marks: 1

Consider the file ***BRANCH(branch-name, city, assets)***, where the primary key is underlined. Suppose that the file is sequential on the primary key. Also, suppose that we have a B+-tree and a hash index on the *city* attribute, and that no other index is available.

9 C B+-Tree and Hashing Opt.

What is the best approach among the following methods for implementing:

$\sigma_{((\text{city}=\text{“Lan Kwai Fong”})\ \text{AND}\ (\text{assets}<5000))}\ \textbf{(branch)}$

Select one alternative:

- Apply the binary search algorithm on the city field of the file to find the tuple with city=“Lan Kwai Fong” and then check if assets<5000.
- Scan the file sequentially and select all tuples with city=“Lan Kwai Fong” and assets<5000.
- Use the B+-tree index to find the tuple with city=“Lan Kwai Fong” and then check if assets<5000.
- Use the hash index to find the tuple with city=“Lan Kwai Fong” and then check if assets<5000.

Maximum marks: 1

10 C B+-Tree and Hashing Opt.

What is the best approach among the following methods for implementing:

$$\sigma_{((\text{branch-name}=\text{“HKmain”})\ \text{AND}\ (\text{assets}<5000))}\ (\text{branch})$$

Select one alternative:

- ○ Apply the binary search algorithm on the branch-name field of the file to find the tuple with branch-name=“HKmain” and then check if assets<5000.
- ○ Use the B+-tree index to find the tuple with branch-name=“HKmain” and then check if assets<5000.
- ○ Use the hash index to find the tuple with branch-name=“HKmain” and then check if assets<5000.
- ○ Scan the file sequentially and select all tuples with branch-name=“HKmain” and assets<5000.

Maximum marks: 1

11 C B+-Tree and Hashing Opt.

What is the best approach among the following methods for implementing:

$\sigma_{((\text{city} \geq \text{“Wan Chai”})\ \text{OR}\ (\text{assets} < 5000))}$ **(branch)**

Select one alternative:

- ○ Use the B+-Tree index on city to find the first tuple of the file according to the city field value. From the first tuple follow the pointer chain till the end of the file and apply the criteria (city≥“Wan Chai”) OR (assets<5000). All the tuples satisfying the criteria form the result.
- ○ Use the hash index to find the tuple with city=“Wan Chai”. Then continue a sequential access of the file until the end of the file and at the same time check if assets<5000.
- ○ We can scan the file sequentially and select all tuples with city≥“Wan Chai” or assets<5000.
- ○ Apply the binary search algorithm on the city field of the file to find the tuple with city=“Wan Chai”. Then access the file sequentially after city “Wan Chai” and at the same check if assets<5000.
- ○ Using the B+-Tree city index, we can retrieve all tuples with city value greater than or equal to “Wan Chai” by following the pointer chains from the first “Wan Chai” tuple. Then, we apply the additional criterion of assets<5000 on every tuple.

Maximum marks: 1

12 C B+-Tree and Hashing Opt.

What is the best approach among the following methods for implementing:

$\sigma_{((city<\text{“Wan Chai”})\ AND\ (city\geq\text{“Hang Hau”})\ AND\ (assets<5000))}(branch)$

Select one alternative:

- ○ We can scan the file sequentially and select all tuples that satisfy the criteria, i.e. (city<“Wan Chai”) AND (city≥“Hang Hau”) AND (assets<5000).

- ○ Using the city B+-Tree index, we can retrieve all tuples with city value greater than or equal to “Hang Hau” and less than “Wan Chai”. We can achieve this by following the pointer chains from the first “Hang Hau” tuple for as long as city is less than “Wan Chai”. Then for each tuple, we apply the additional criterion of assets<5000.

- ○ Use the B+-Tree index to find the first tuple of the file according to the city field value. From the first tuple follow the pointer chain till the end of file. For each tuple, we apply the criteria, i.e. (city<“Wan Chai”) AND (city≥“Hang Hau”) AND (assets<5000) .

- ○ Using the city B+-Tree index, we can retrieve all tuples with city value smaller than “Wan Chai” by following the pointer chains from the first “Wan Chai” tuple. Then, we apply the additional criterion of assets<5000 on every tuple.

- ○ Using the city Hash index, we can retrieve all tuples with city value smaller than “Wan Chai” by following the pointer chains from the first “Wan Chai” tuple. Then, we apply the additional criterion of assets<5000 on every tuple.

Maximum marks: 1

13 C B+-Tree and Hashing Opt.

What is the best approach among the following methods for implementing:

$\sigma_{((\text{branch-name}=\text{“HKmain”})\ \text{OR}\ (\text{assets}<5000))}$ **(branch)**

Select one alternative:

- ○ Use the B+-tree index to find the tuple with branch-name=“HKmain” and then check if assets<5000.
- ○ Apply the binary search algorithm on the branch-name field of the file to find the tuple with branch-name=“HKmain” and then check if assets<5000.
- ○ Use the hash index to find the tuple with branch-name=“HKmain” and then check if assets<5000.
- ○ Scan the file sequentially and select all tuples with branch-name=“HKmain” or assets<5000.

Maximum marks: 1

14 C B+-Tree and Hashing Opt.

What is the best approach among the following methods for implementing:

$\sigma_{((\text{city}=\text{“Lan Kwai Fong”})\ \text{OR}\ (\text{branch-name}=\text{“HKmain”}))}(\text{branch})$

Select one alternative:

- ○ Scan the file sequentially and select all tuples with city=“Lan Kwai Fong” or branch-name=“HKmain”.
- ○ Apply the binary search algorithm on the branch-name to find the tuple with branch-name=“HKmain”. Then, use the hash index on the city field to find the tuple with city=“Lan Kwai Fong”.
- ○ Use the B+-tree index on city to find the tuple with city=“Lan Kwai Fong” and then check if branch-name=“HKmain”.
- ○ Use the hash index on city to find the tuple with city=“Lan Kwai Fong” and then check if branch-name=“HKmain”.

Maximum marks: 1

15 ♥ C Equivalence Rules

Which of the following choices are correct?

Select one or more alternatives:

- ☐ Natural join operations are NOT associative, i.e. $(E_1 \bowtie E_2) \bowtie E_3 \neq E_1 \bowtie (E_2 \bowtie E_3)$
- ☐ $\Pi_{t1}(\Pi_{t2}(E)) = \Pi_{t2}(\Pi_{t1}(E))$
- ☐ $\sigma_{\theta1 \wedge \theta2\ \wedge \theta3}(E) = \sigma_{\theta3}(\sigma_{\theta2}(\sigma_{\theta1}(E)))$
- ☐ $\sigma_{\theta1}(\sigma_{\theta2}(E)) = \sigma_{\theta2 \wedge}\sigma_{\theta1}(E)$

Maximum marks: 1

16 C Hashing Files

A STUDENTS file with StudentID as hash key includes records with the following StudentID values: **800, 199, 178, 201, 206, 123, 102, 106, 189, 202, 301, 108, 200, 987, 999, 307, 400**. The file uses 8 buckets, numbered 0 to 7. Each bucket is one disk block and holds two records. Load these records into the file in the given order using the hash function **h(K)=K mod 8**.

Select one alternative:

Bucket 0: {800}
Bucket 1: {201}
Bucket 2: {202}
Bucket 3: {307}
Bucket 4: {108}
Bucket 5: {189}
Bucket 6: {206}
Bucket 7: {199}

Overflow buckets {102, 106, 178, 301, 200, 987, 999, 123}

None of the other answers is correct!

Bucket 0: {800, 200, 400}
Bucket 1: {201}
Bucket 2: {178, 202, 106}
Bucket 3: {123, 307, 987}
Bucket 4: {108}
Bucket 5: {189, 301}
Bucket 6: {206, 102}
Bucket 7: {199, 999}

Bucket 0: {800, 400}
Bucket 1: {201}
Bucket 2: {178, 202}
Bucket 3: {123 307}
Bucket 4: {108}
Bucket 5: {189, 301}
Bucket 6: {206, 102}
Bucket 7: {199, 999}

Overflow buckets {106, 987, 200}

Maximum marks: 1

17 ♥ C Sequential files

Consider Sequential files (i.e. where records are sorted by an ordering field). Which of the following statements are correct.

Select one or more alternatives:

- ☐ Every time we need to insert a new record, we need also to move many records one position, which can be very expensive in time.
- ☐ We need to reorganize the file periodically to restore sequential order.
- ☐ Sequential files are suitable for applications that require sequential processing of the entire file (since reading records in order of ordering key value is extremely efficient).
- ☐ Finding the next record is very efficient.
- ☐ Sequential files are suitable for applications that require very fast search of records, as we can find any record with constant time 1.

Maximum marks: 1

18 ♥ C NOSQL v SQL

Comparing NOSQL and SQL systems, which of the following statements are correct?

Select one or more alternatives:

- ☐ For applications with vast amount of data that support many users, NOSQL can be more efficient than traditional relational models (SQL).
- ☐ SQL systems are more preferable than NOSQL systems as a solution for social network systems managing users' activities, such as photos uploads, shares, "likes", etc.
- ☐ In NOSQL, data must be normalised up to the 3rd normal form.
- ☐ A structured data model such as the traditional relational model may be too restrictive. NOSQL is more flexible in modelling data and can support semi structure, self descriptive data models.
- ☐ SQL systems offer too many services (powerful query language, concurrency control, etc.), which can be demanding with respect to CPU and memory resources. Some applications such as email systems may not need such services, thus NOSQL is more efficient and thus more preferable.

Maximum marks: 1

19 ♥ C Heap Files

Consider Heap files. Which of the following statements are correct.

Select one or more alternatives:

- ☐ Deletion techniques: use a deletion marker.
- ☐ Records are placed in the file in the order of insertion.
- ☐ We can apply the binary search algorithm to search for a record which is reasonably fast.
- ☐ Inserting a new record is efficient.

Maximum marks: 1

20 ♥ C Consistency

Which of the following statements regarding ***consistency*** are correct.

Select one or more alternatives:

- ☐ In ACID, the term consistency refers to the fact that a transaction will not violate the integrity constraints specified on the database schema.
- ☐ The consistency in CAP and in ACID refers to the same identical concept.
- ☐ In CAP, the term consistency refers to the conistency of measurement techniques.
- ☐ In CAP, the term consistency refers to the consistency of the values in different copies of the same data item in a replicated distributed system.

Maximum marks: 1

21 C Fast Access

We need a fast search of records from a heap file. Also, the sequential access of records is needed. Which of the following is the most appropriate choice?

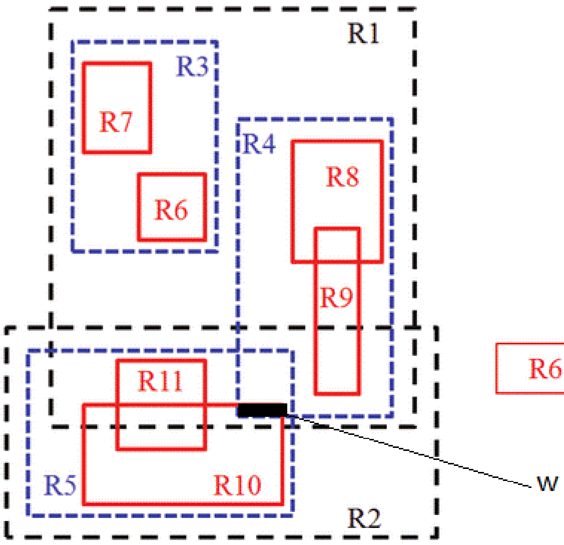
Select one alternative:

- ○ Linear Search
- ○ Hashing
- ○ B+-Tree
- ○ Binary Search

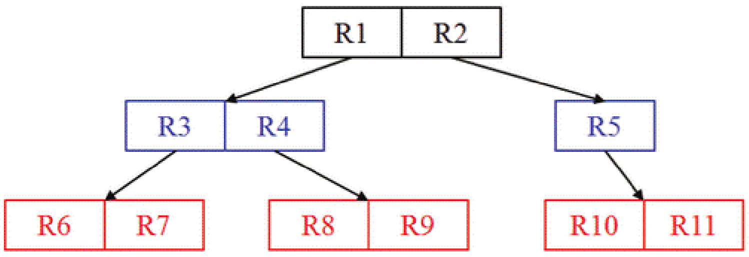
Maximum marks: 1

22 C R-Tree

Consider the following regions (rectangles) and the respective R-Tree. Which R-Tree nodes will be visited while searching for the query window ***w***?



(a)



(b)

Select one alternative:

- ○ R1, R4, R2, R5, R10
- ○ R1, R2, R5, R10
- ○ The R-Tree is wrong, thus all other answers are wrong!
- ○ R2, R5, R10

Maximum marks: 1

23 C Selection Cardinality

Consider the file **CITIZEN*(<u>CID</u>, Sex, City, Assets)***, where the primary key is underlined. Suppose that the file is sequential on the primary key. Also, suppose that the relation has 1000 citizens, where 10 of the citizens are from the Ayia Napa City and half of the citizens are females. Suppose, we have two B+-Trees indexes, i.e. one on Sex and one on City attributes and that no other index is available.

What is the best approach among the following methods for implementing:

$\sigma_{\text{((city="Ayia Napa") AND (Sex="Female"))}}\text{(CITIZEN)}$

Select one alternative:

- ○ Using the B+-Tree on City, first select all citizens from Ayia Napa. Then check if they are female.
- ○ Scan the file sequentially and select all tuples with female citizens from Ayia Napa.
- ○ Apply the binary search algorithm on the City field of the file to find citizens from Ayia Napa and then check if they are female.
- ○ Using the B+-Tree on Sex, first select female citizens and then check if they are from Ayia Napa. Then, using the B+-Tree on City select all citizens from Ayia Napa and then check if they are females.
- ○ Using the B+-Tree on Sex, first select all female citizens and then check if they are from Ayia Napa.

Maximum marks: 1

24 ♥ C R-Tree Th

In the context of R-tree select all correct statements.

(Where MBR is Minimum Bounding Rectangle.)

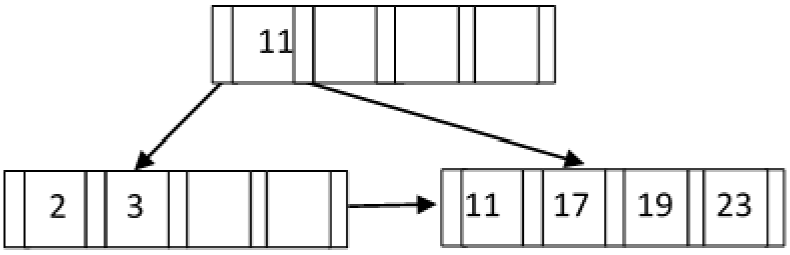
Select one or more alternatives:

- ☐ Due to space savings, we do not allow MBR overlaps.
- ☐ Minimising dead space inside an MBR improves R-tree efficiency.
- ☐ The parent nodes will hold child nodes where child nodes completely overlap the region of parent nodes
- ☐ For a range search, we need h operations, where h is the height of the tree.
- ☐ R-tree is always a balanced tree

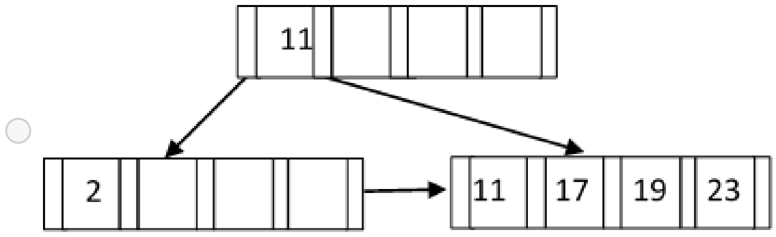
Maximum marks: 1

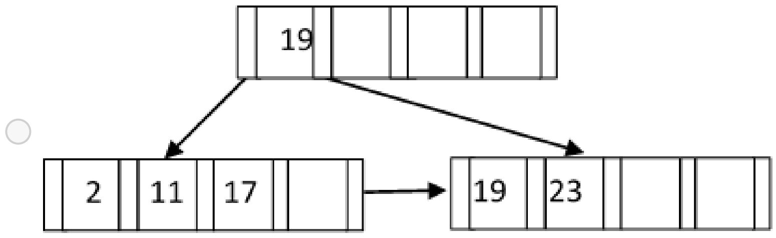
25 C B+-Tree

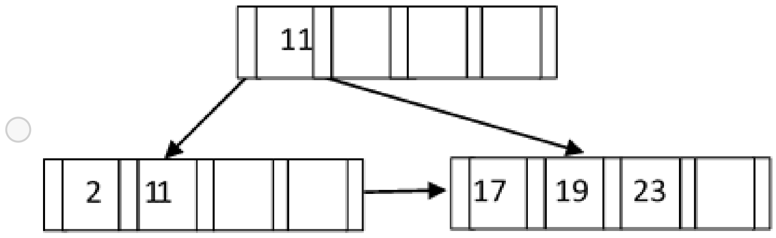
For the B+-tree below, what is the form of the tree after deleting 3?

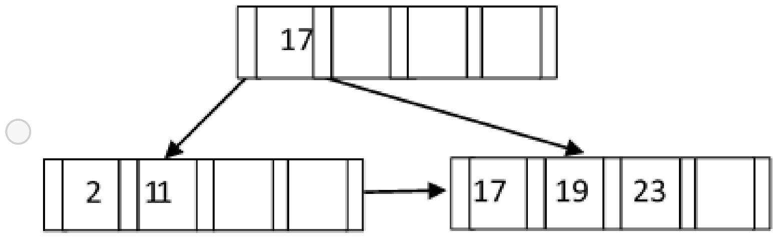


Select one alternative:





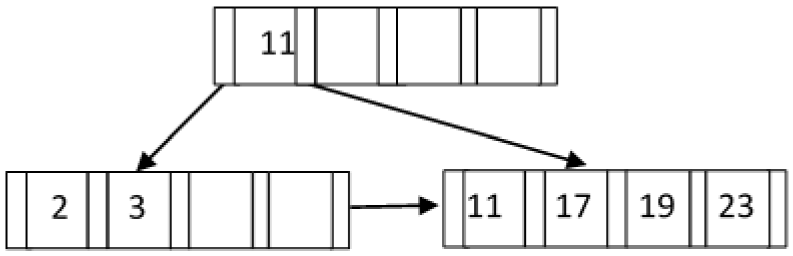




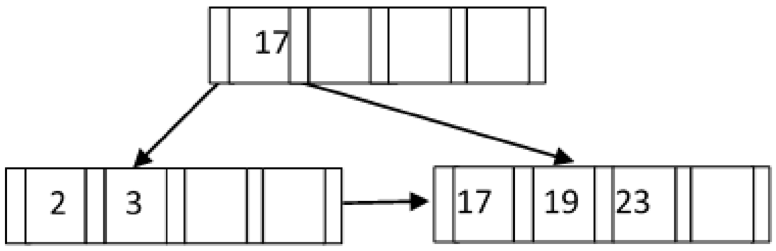
Maximum marks: 1

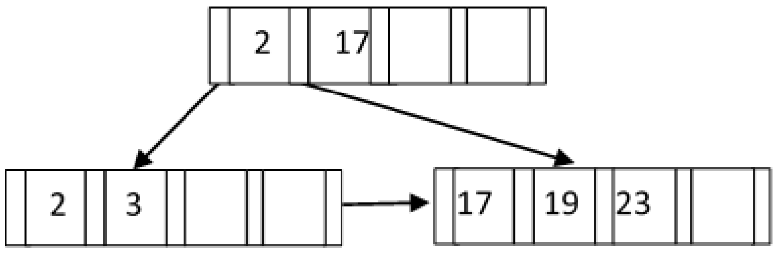
26 C B+-Tree

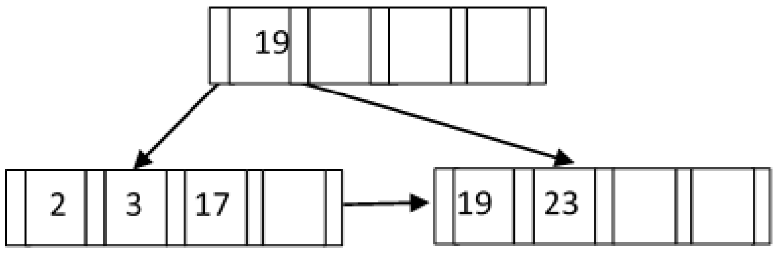
For the B+-tree below, what is the form of the tree after deleting 11?

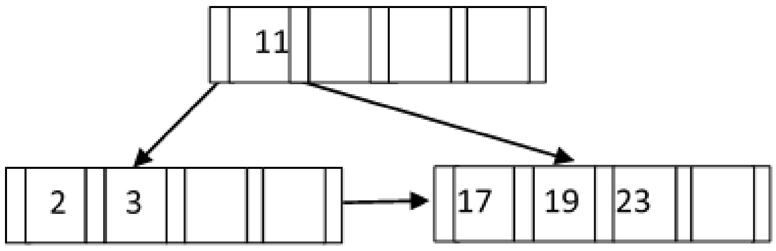


Select one alternative:

- 

- 

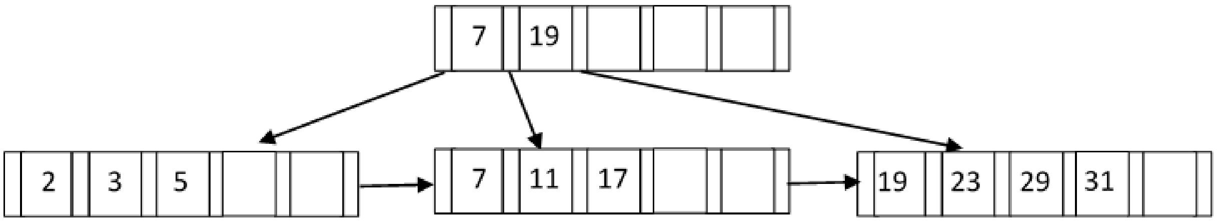
- 

- 

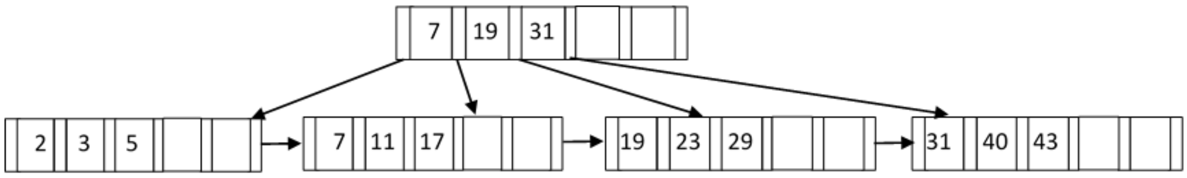
Maximum marks: 1

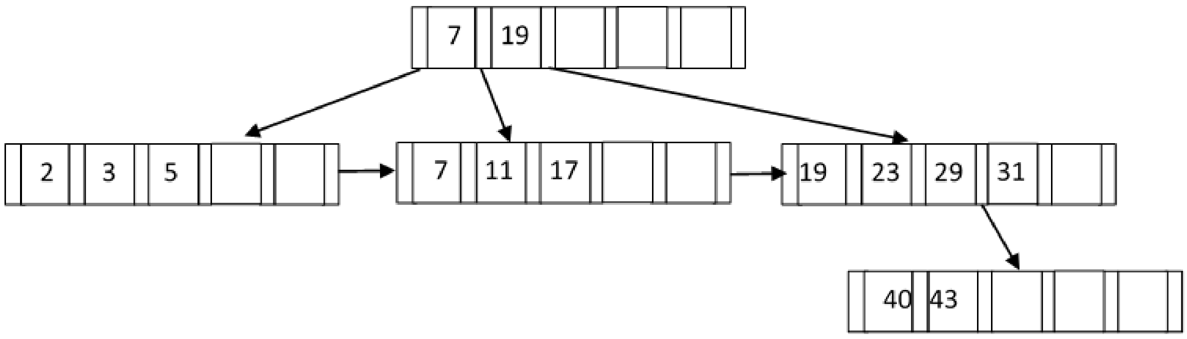
27 C B+-Tree

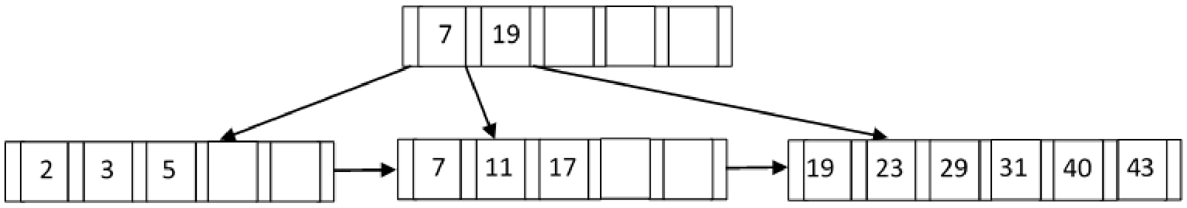
For the B+-tree below, what is the form of the tree after adding 40 and then 43?



Select one alternative:







Maximum marks: 1