Name

CWID

Exam
2

May 4th, 2023

CS525 - Final Exam
Grading Guidelines

Please leave this empty! 1 2 3 4 Sum

Instructions

- The exam is **closed books and closed notes, no calculators allowed**
- For your convenience the number of points for each part and questions are shown in parenthesis.
- There are 4 parts in this exam (100 points total)
  1. SQL (35)
  2. Index Structures (27)
  3. I/O Estimation (18)
  4. Schedules (20)

Part 1 SQL (Total: 32 Points)

Consider the following database storing information about a company's warehouses, orders, pricing, and stock.

person

name	affiliation	field	since
Goris Blavic	IIT	CS	2012-08-15
Bustafa Milgic	IIT	CS	2011-08-15
Hyle Kale	IIT	CS	2016-08-15
Kulu Lang	IIT	MATH	2015-08-15
Fichael Mranklin	UC	CS	2016-01-01

journal

jname	field	impactfactor
Journal of statistical nonsense	MATH	3.5
Journal of database nonsense	CS	1.3
International journal of chat bots	cS	12.5
Journal of kernel hacking	CS	10.6

article

journal	title	issue	numb
International journal of chat bots	Will chatGPT rule the world?	1	1
Journal of database nonsense	Log-structured read-only indexes	2	3
Journal of database nonsense	Why windows is the best OS for DBs	2	4
Journal of statistical nonsense	Green gummy bears cause cancer	14	2
Journal of database nonsense	Spark and Spark and Spark and Spark	1	3

author

author	journal	title
Goris Blavic	Journal of database nonsense	Log-structured read-only indexes
Goris Blavic	Journal of database nonsense	Why windows is the best OS for DBs
Hyle Kale	Journal of database nonsense	Why windows is the best OS for DBs
Kulu Lang	Journal of statistical nonsense	Green gummy bears cause cancer
Bustafa Milgic	International journal of chat bots	Will chatGPT rule the world?
Fichael Mranklin	Journal of database nonsense	Spark and Spark and Spark and Spark

Hints:

- When writing queries do only take the schema into account and **not** the example data given here. That is your queries should return correct results for all potential instances of this schema.
- Attribute **journal** of relation **article** are foreign keys to relation **journal**.
- Attributes **journal** and **title** of relation **author** form a foreign key to relation **article**.
- Attribute **author** of relation **author** is a foreign key to relation **person**.

Question 1.1 (7 Points)

Write a SQL query that returns for each person the average impact factor of their publications (the impact factor of an article is the impact factor of the journal it is published in). Order the result in decreasing order of average impact factor.

Solution

```
SELECT name, avg(impactfactor) AS avgif
  FROM person p, author a, article c, journal j
 WHERE p.name = a.author
       AND j.jname = c.journal
       AND (a.journal, a.title) = (c.journal, c.title)
GROUP BY name
ORDER BY avgif DESC
```

Correction Guideline

- 2 Points for joins
- 3 Points for aggregation
- 2 Points for ordering

Question 1.2 (7 Points)

Write a SQL query that returns the names of researchers that have published in every venue in their field.

Solution

```
SELECT name
  FROM person p
 WHERE NOT EXISTS (SELECT *
                     FROM journal j
                    WHERE j.field = p.field
                      AND NOT EXISTS (SELECT *
                                        FROM article c, author a
                                       WHERE p.name = a.author
                                         AND (a.journal, a.title) = (c.journal, c.title)
                                         AND j.jname = a.journal))
```

Correction Guideline

- 3 Points for each level of negation (6 in total)
- 1 Point for outer query

Question 1.3 (7 Points)

Write a SQL query that returns the three institutions (affiliations of persons) with the highest number of publications (total amount of publication for all authors affiliated with an institution). **Make sure not to double count publications, as a single publication may have more than one author from the same institution.** Ties between institutions can be resolved arbitrarily.

Solution

```
WITH apubl AS (
  SELECT DISTINCT affiliation, a.journal, a.title
    FROM person p, author a, article c
   WHERE p.name = a.author AND (a.journal, a.title) = (c.journal, c.title))

SELECT affiliation, count(*) AS num_publ
FROM apubl
GROUP BY affiliation
ORDER BY num_publ DESC
LIMIT 3;
```

Correction Guideline

- 2 Points for distinct publ per affiliation
- 2.5 Points for group by aggregation
- 2.5 Points for order limit

Question 1.4 (7 Points)

Write a SQL query that returns for each field of study (attribute field of table person) the university (affiliation) with the most publications in this field.

Solution

```
SELECT affiliation, field
FROM (SELECT affiliation, field, row_number() OVER (PARTITION BY field
                                                    ORDER BY n DESC) AS r
          FROM (SELECT affiliation, field, count(*) AS n
                  FROM person p, author a, article c
                 WHERE p.name = a.author
                       AND (a.journal, a.title) = (c.journal, c.title)
                 GROUP BY affiliation, field) nump) ranked
 WHERE r = 1;
```

Correction Guideline

- 3 Points for counting publications per field and person
- 3 Points for computing ranks
- 1 Point for filtering based on rank

Question 1.5 (7 Points)

Write a SQL query that calculates for each university (affiliation) a rank (higher is better) based on a score that is calculated as follows: the rank of a university is the number of persons affiliated with the university that have published at least 2 articles in journals with an impact factor higher than 2.

Solution

```
WITH numhighpubl AS (
  SELECT name, affiliation
    FROM person p, author a, article c, journal j
   WHERE p.name = a.author
     AND (a.journal, a.title) = (c.journal, c.title)
     AND j.jname = c.journal
     AND impactfactor > 2.0
   GROUP BY name, affiliation
  HAVING count(*) >= 2),
  scores AS (SELECT affiliation, count(*) AS score
               FROM numhighpubl
              GROUP BY affiliation)
SELECT affiliation, ROW_NUMBER() OVER (ORDER BY score DESC) AS rank
  FROM scores
```

Correction Guideline

- 3 Points for calculating person scores
- 2 Points for counting high profile persons
- 2 for final ranking

Part 2 Index Structures (Total: 27 Points)

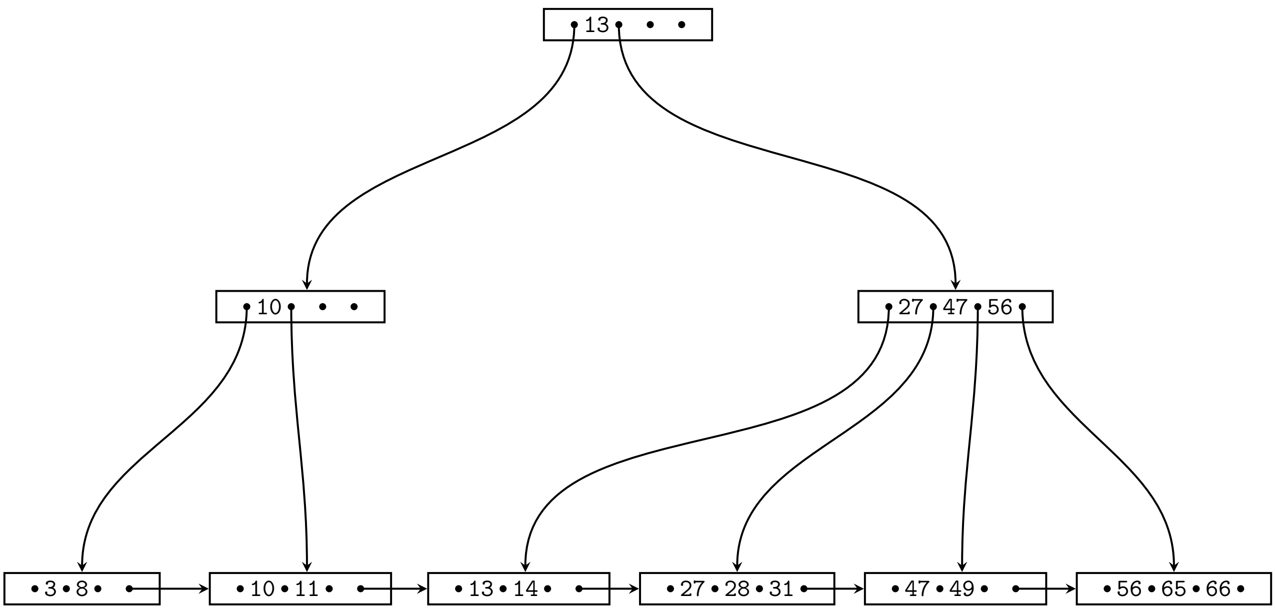
Question 2.1 Operations (27 Points)

Given is the B+-tree shown below ($n = 3$). Execute the following operations and write down the resulting B+-tree after each operation:

delete(65),insert(29),insert(4),insert(44),delete(49),insert(92),insert(51),delete(3),delete(8)

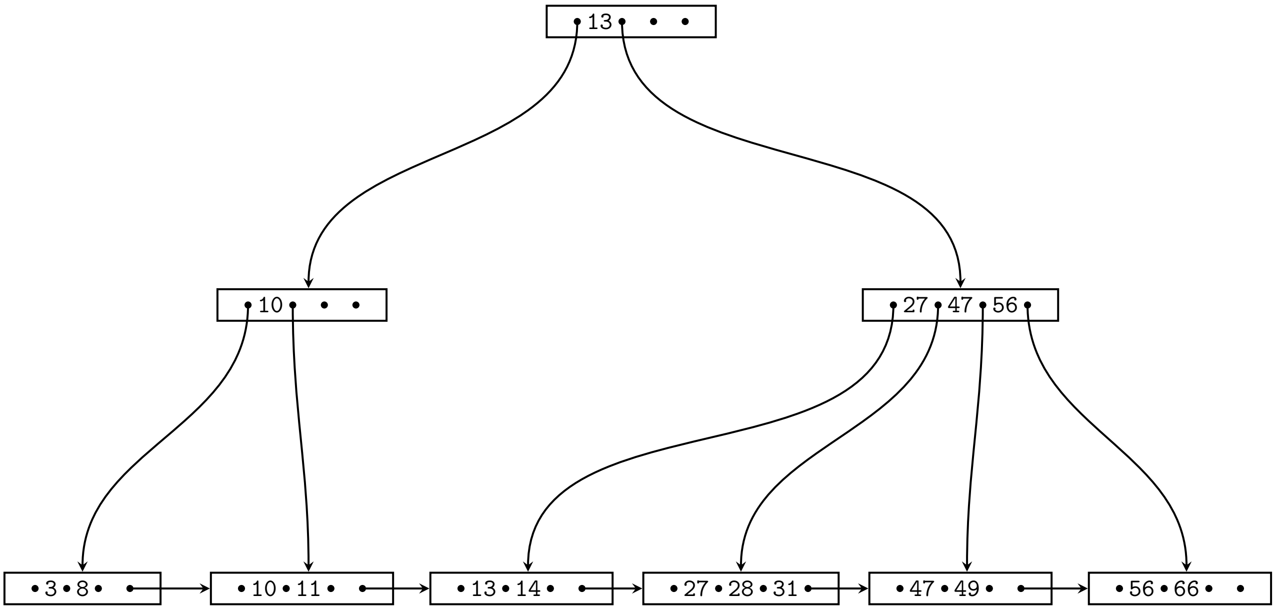
When splitting or merging nodes follow these conventions:

- **Leaf Split**: In case a leaf node needs to be split, the left node should get the extra key if the keys cannot be split evenly.
- **Non-Leaf Split**: In case a non-leaf node is split evenly, the "middle" value should be taken from the right node.
- **Node Underflow**: In case of a node underflow you should first try to redistribute and only if this fails merge. Both approaches should prefer the left sibling.

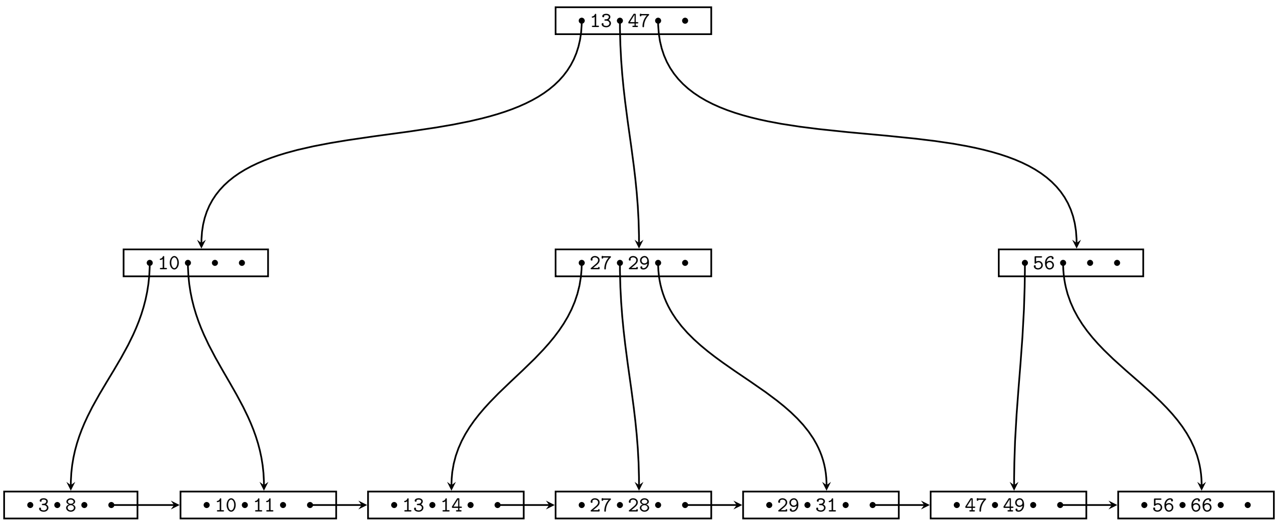


Solution

delete(65)

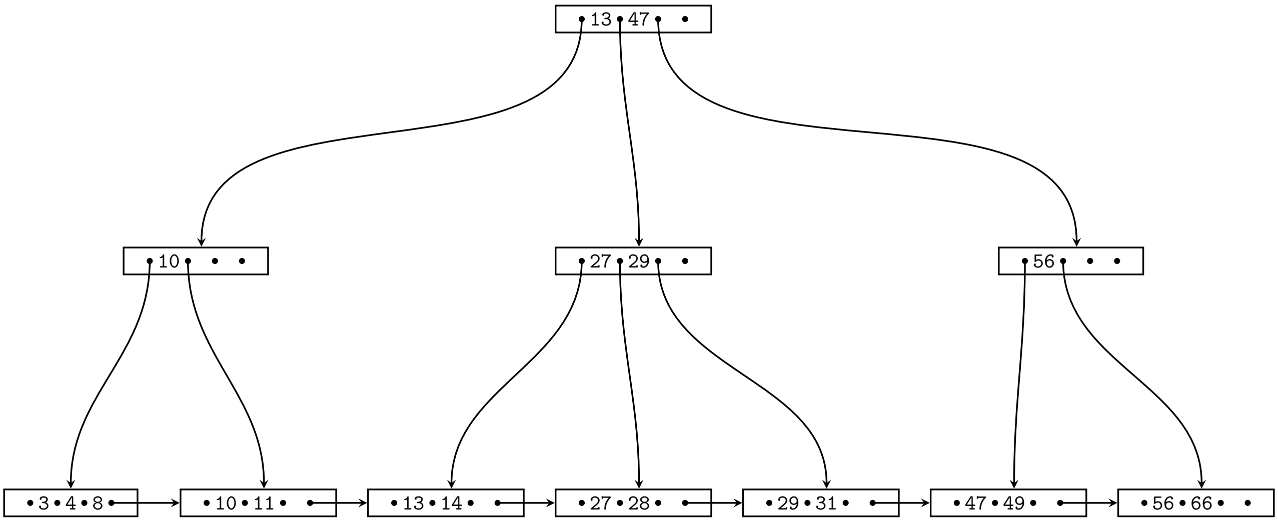


Solution
insert(29)



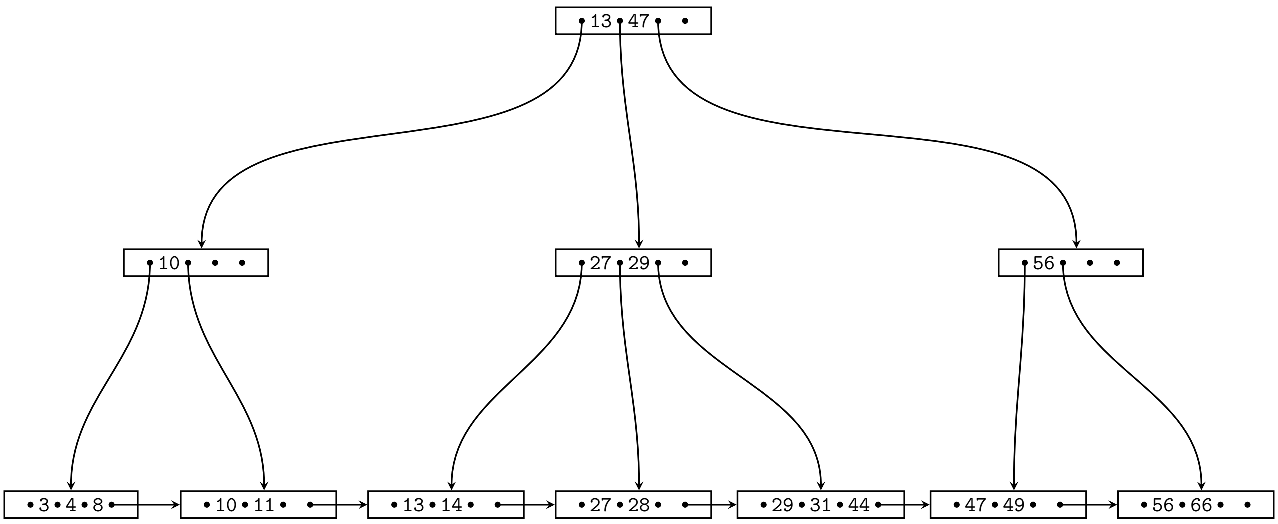
Solution

insert(4)



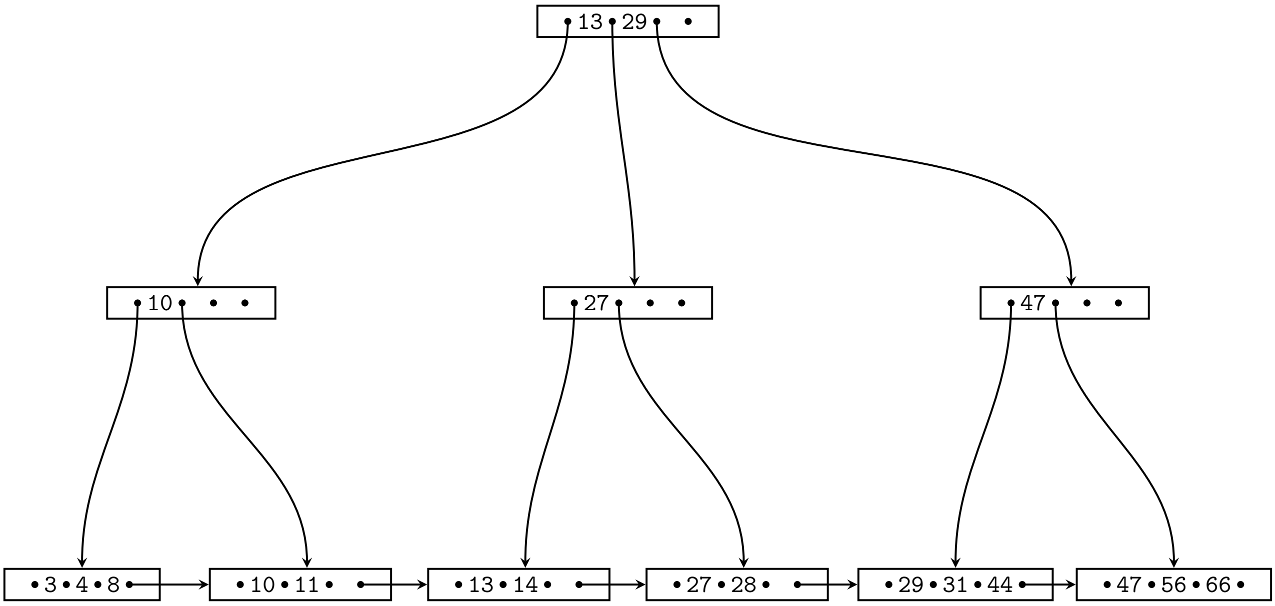
Solution

insert(44)



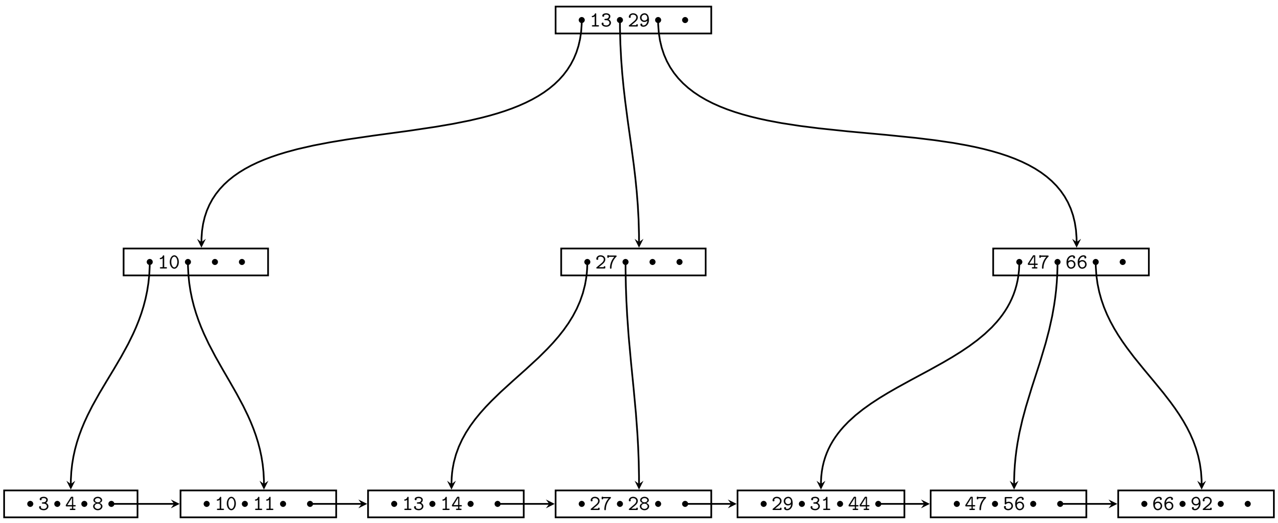
Solution

delete(49)



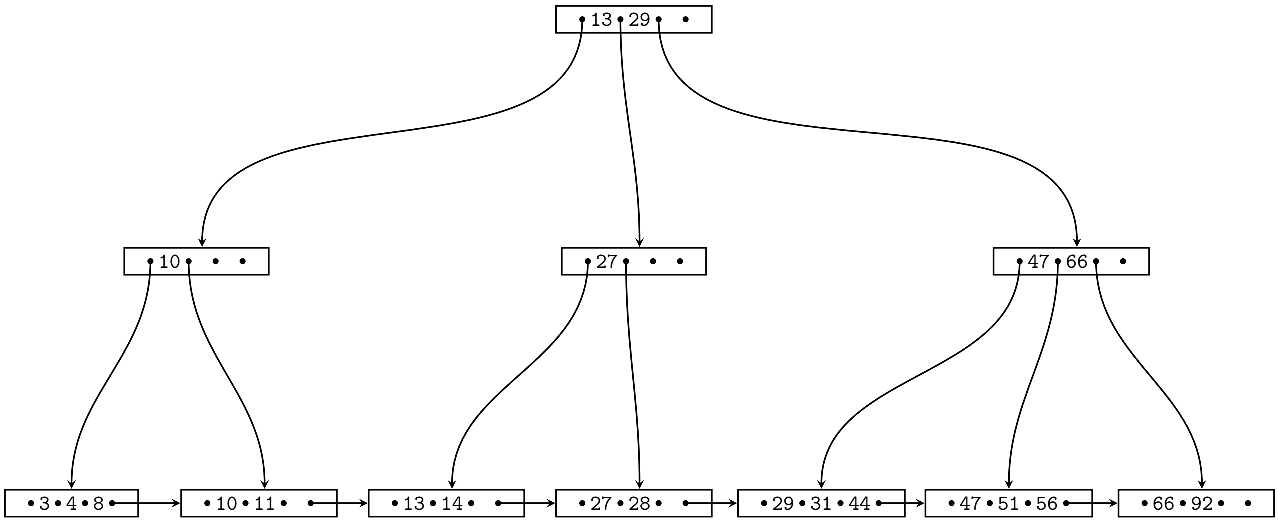
Solution

insert(92)



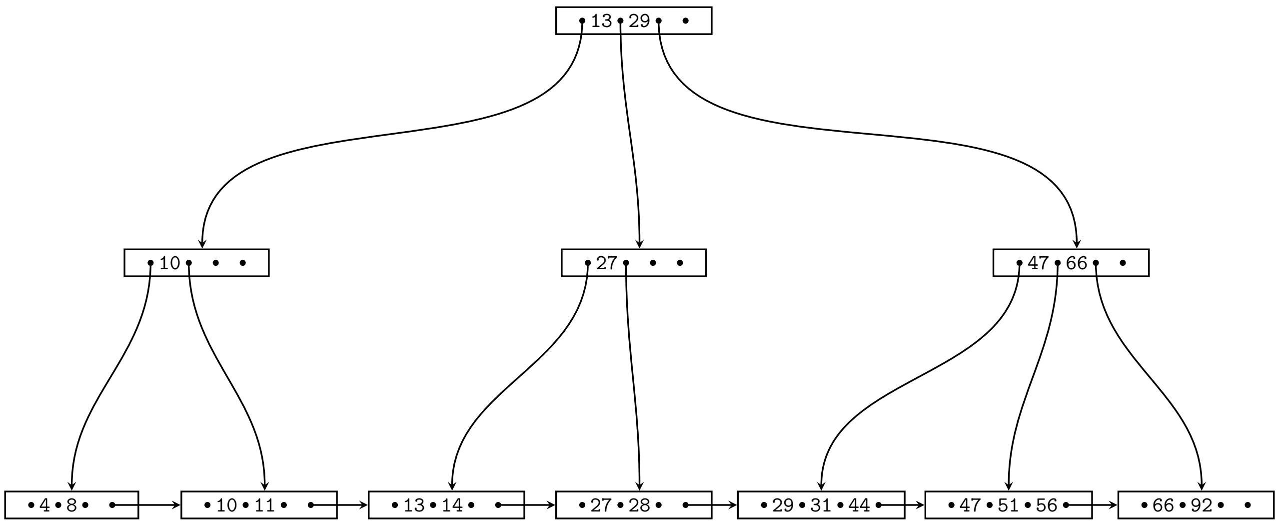
Solution

insert(51)



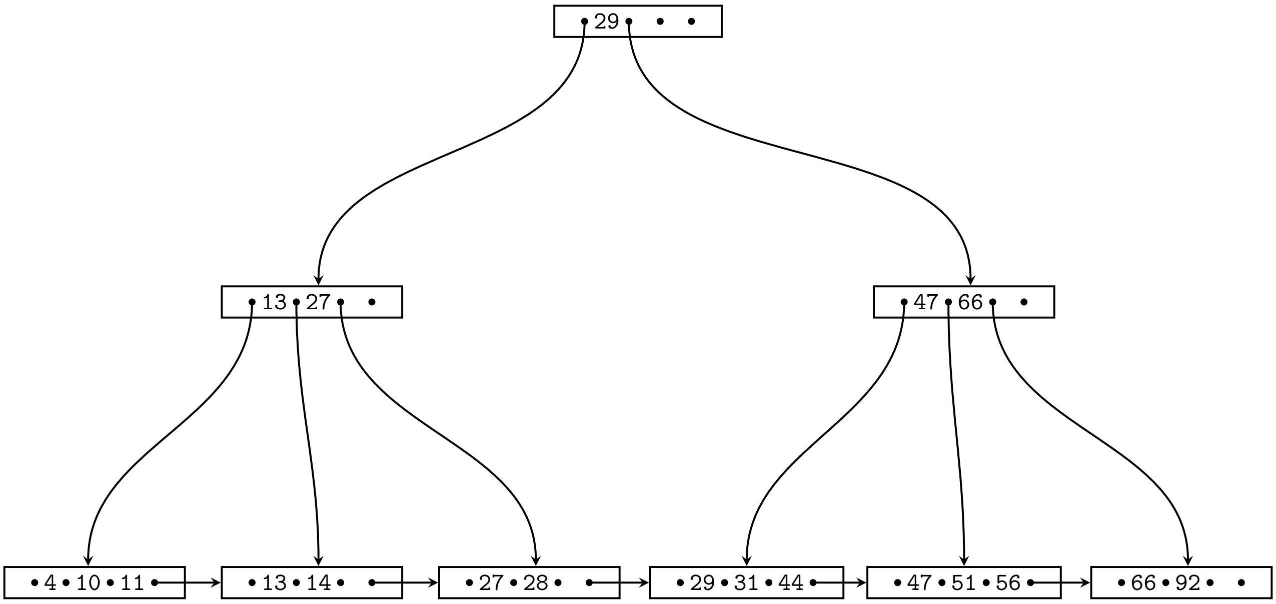
Solution

delete(3)



Solution

delete(8)



Correction Guideline

3 point for each operation

Part 3 I/O Cost Estimation (Total: 18 Points)

Question 3.1 External Sorting (3 Points)

You have $M = 129$ memory pages available and should sort a relation R with $B(R) = 64,000,000$ blocks. Estimate the number of I/Os necessary to sort R using the external merge sort algorithm introduced in class.

Solution

$$
\begin{aligned}
IO &= 2 \cdot B(R) \cdot \left(1 + \left\lceil log_{M-1}\left(\frac{B(R)}{M}\right)\right\rceil\right) \\
&= 2 \cdot 64,000,000 \cdot (1 + 3) \\
&= 512,000,000
\end{aligned}
$$

Correction Guideline

2 Points for the correct solution
1 Points if the formula is correct, but the result is wrong

Question 3.2 External Sorting (3 Points)

You have $M = 9$ memory pages available and should sort a relation R with $B(R) = 120,000$ blocks. Estimate the number of I/Os necessary to sort R using the external merge sort algorithm introduced in class.

Solution

$$
\begin{aligned}
IO &= 2 \cdot B(R) \cdot \left(1 + \left\lceil log_{M-1}\left(\frac{B(R)}{M}\right)\right\rceil\right) \\
&= 2 \cdot 120,000 \cdot (1 + 11) \\
&= 1,440,000
\end{aligned}
$$

Correction Guideline

2 Points for the correct solution
1 Points if the formula is correct, but the result is wrong

Question 3.3 I/O Cost Estimation (12 = 4 + 4 + 4 Points)

Consider two relations R and S with $B(R) = 70,000$ and $B(S) = 60,000$. You have $M = 101$ memory pages available. Compute the minimum number of I/O operations needed to join these two relations using **block-nested-loop join**, **merge-join** (the inputs are not sorted), and **hash-join**. You can assume that the hash function evenly distributes keys across buckets. Justify your result by showing the I/O cost estimation for each join method.

Solution

- **BNL**: S is the smaller relation.
  $\lceil \frac{B(S)}{M-1} \rceil \cdot [B(R) + min(B(S), (M-1))] = 600 \cdot [70,000 + 101] = 42,060,000$ I/Os

- **MJ**: We can generate sorted runs of size 101 that means we need 2 merge pass(es) for R and 2 merge passes for S. The number of runs in the last phase of sorting is 7 for R and 6 for S. The optimization is applicable, because $7+6 < M$. Thus, the total cost is $5*B(R)+5*B(S) = 5*70,000+5*60,000 = 650,000$ I/Os.

- **HJ**: After 2 partition phases the size of the partitions for S (60 pages) is small enough to fit one partition into memory, build an in-memory hash table of each partition of S, and stream a partition of R probing the hash table. $(2 \cdot 2 + 1) \cdot (B(R) + B(S)) = 5 \cdot (70,000 + 60,000) = 650,000$ I/Os.

Correction Guideline

4 Points for each subquestion. 2 Point(s) if they write down the correct formula or reasoning, but the result is wrong.

Part 4 Schedules (Total: 20 Points)

Question 4.1 Schedule Classes (20 = 5 + 5 + 5 + 5 Points)

Indicate which of the following schedules belong to which class. **Every correct answer is worth 1 point. Every incorrect answer results in 1 point being deducted. You are allowed to skip questions (0 points). For each schedule you will get at least 0 points.** Recall transaction operations are modeled as follows:

$w_1(A)$	transaction 1 wrote item A	$r_1(A)$	transaction 1 read item A
c_1	transaction 1 commits	a_1	transaction 1 aborts

$S_1 = w_1(A), r_2(A), w_2(B), r_3(B), w_1(B), r_4(B), c_1, c_2, c_3, c_4$

$S_2 = w_1(A), r_2(B), w_2(C), r_3(D), w_4(E), r_4(C), w_5(B), r_6(F), w_7(G), r_7(D), w_8(H), c_1, c_2, c_3, c_4, c_5, c_6, c_7, c_8$

$S_3 = r_1(A), r_1(B), w_1(C), r_2(A), w_2(B), w_2(C), w_2(D), c_2, w_3(D), c_3, r_4(B), w_4(E), w_4(F), c_4, r_5(E), w_5(G), c_5, c_1, c_3$

$S_4 = w_1(A), w_1(B), c_1, r_2(B), w_2(C), c_2, r_3(C), w_3(D), c_3, r_4(D), w_4(A), c_4, r_5(B), w_5(E), c_5, r_6(E), w_6(D), c_6$

■ yes ☐ no S_1 is recoverable

☐ yes ■ no S_1 is cascade-less

☐ yes ■ no S_1 is strict

☐ yes ■ no S_1 is conflict-serializable

☐ yes ■ no S_1 is 2PL

■ yes ☐ no S_2 is recoverable

☐ yes ■ no S_2 is cascade-less

☐ yes ■ no S_2 is strict

■ yes ☐ no S_2 is conflict-serializable

■ yes ☐ no S_2 is 2PL

■ yes ☐ no S_3 is recoverable

■ yes ☐ no S_3 is cascade-less

☐ yes ■ no S_3 is strict

■ yes ☐ no S_3 is conflict-serializable

■ yes ☐ no S_3 is 2PL

■ yes ☐ no S_4 is recoverable

■ yes ☐ no S_4 is cascade-less

■ yes ☐ no S_4 is strict

■ yes ☐ no S_4 is conflict-serializable

■ yes ☐ no S_4 is 2PL

Correction Guideline

- 1 points per correct answer
- -1 points per each wrong answer
- 0 points if no answer checked
- grade the subquestion for each schedule individually (negative points do not propagate across subquestions)