Concurrency Control 3

R&G Chapter 17

(slides adapted from content by J.Gehrke, J.Shanmugasundaram, and/or C.Koch)

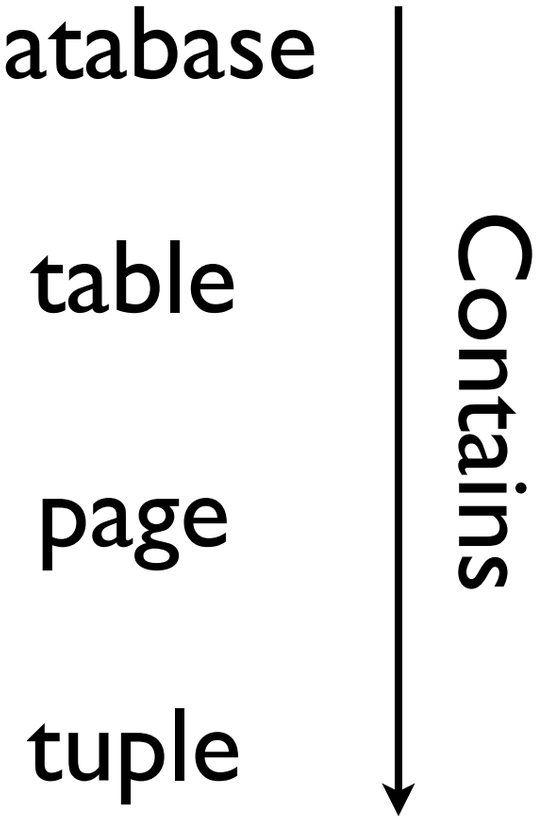
Reminders

- Homework due Monday after break (Posted tonight)

Recap: New Lock Modes

- ‘Intent’ to lock (a child)
  - Intent-to-Lock Shared (IS)
  - Intent-to-Lock Exclusive (IX)
- Actual lock (on the object)
  - Lock-Shared (S)
  - Lock-Exclusive (x)
- Lock Shared + Intent-to-Lock Exclusive (SIX)

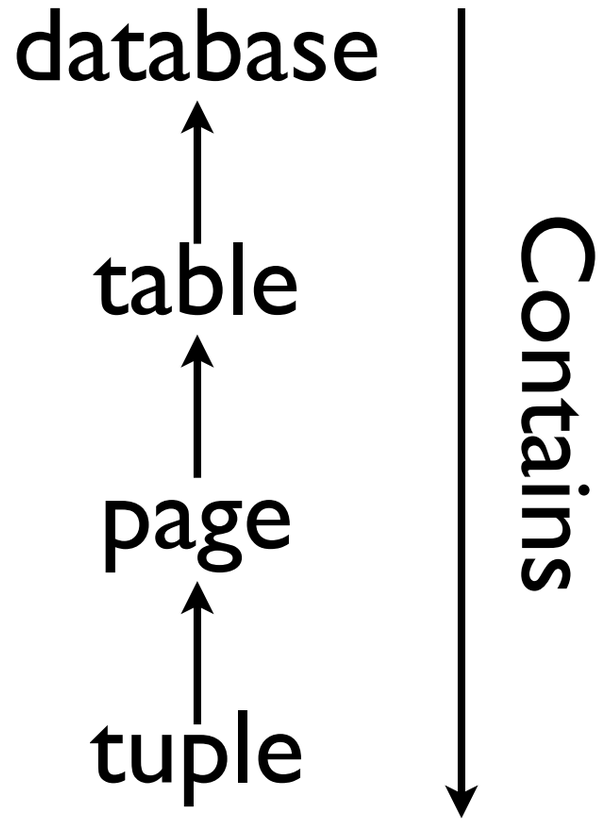
What do we lock?



What do we lock?

- Objects Locked Top-Down

- Before acquiring a lock on an object, an xact must have at least an intention lock on its parent!



Recap: New Lock Modes

Lock Mode(s) Currently Held By Other Xacts

Lock Mode Desired

	None	IS	IX	S	X
None	valid	valid	valid	valid	valid
IS	valid	valid	valid	valid	fail
IX	valid	valid	valid	fail	fail
S	valid	valid	fail	valid	fail
X	valid	fail	fail	fail	fail

Problem:

Locking assumes that we can lock all objects!

(What happens if we insert objects?)

T1

T2

```
DELETE FROM Officers
WHERE rank = 1
   AND age =
    (SELECT MAX(age)
     FROM Officers WHERE rank=1)
LIMIT 1;
```

Time

T1

T2

```
DELETE FROM Officers
WHERE rank = 1
   AND age = (71)
    (SELECT MAX(age)
     FROM Officers WHERE rank=1)
LIMIT 1;
```

Time

T1

```
DELETE FROM Officers
WHERE rank = 1
   AND age = (71)
    (SELECT MAX(age)
     FROM Officers WHERE rank=1)
LIMIT 1;
```

T2

```
INSERT INTO Officers(rank,age)
                VALUES (1, 96);
```

Time

Time

T1

```
DELETE FROM Officers
WHERE rank = 1
   AND age = (71)
    (SELECT MAX(age)
     FROM Officers WHERE rank=1)
LIMIT 1;
```

T2

```
INSERT INTO Officers(rank,age)
                VALUES (1, 96);
DELETE FROM Officers
WHERE rank = 2
   AND age = (80)
    (SELECT MAX(age)
     FROM Officers WHERE rank=2)
LIMIT 1;
```

Time

T1

```
DELETE FROM Officers
WHERE rank = 1
   AND age = (71)
    (SELECT MAX(age)
     FROM Officers WHERE rank=1)
LIMIT 1;
```

```
SELECT MAX(age)
FROM Officers(rank,age) (63)
WHERE rank = 2
```

T2

```
INSERT INTO Officers(rank,age)
                VALUES (1, 96);
```

```
DELETE FROM Officers
WHERE rank = 2
   AND age = (80)
    (SELECT MAX(age)
     FROM Officers WHERE rank=2)
LIMIT 1;
```

Time

T1

```
DELETE FROM Officers
WHERE rank = 1
   AND age = (71)
    (SELECT MAX(age)
     FROM Officers WHERE rank=1)
LIMIT 1;
```

T2

```
INSERT INTO Officers(rank,age)
                VALUES (1, 96);
DELETE FROM Officers
 WHERE rank = 2
    AND age = (80)
     (SELECT MAX(age)
      FROM Officers WHERE rank=2)
LIMIT 1;
```

[INCONSISTENT]

```
SELECT MAX(age)
FROM Officers(rank,age)
WHERE rank = 2
```

(63)

The Problem

- T1 assumes that is has locked all sailor records with rating = 1
  - Solution 1: Lock entire table (expensive!)
  - Solution 2: Lock a **predicate**.

    (e.g., `rank = 1`)
  - Solution 3: Lock **index page(s)**.

    (equivalent to a range/predicate lock)

Index Locking

- If there is an index on `rank`, T1 locks the index page(s) with `rank = 1`.
- If no such entries exist, lock the page where an entry would go.
- Need to stop other indexes from being locked
  - 1) Have a single primary index at any time.
  - 2) Conflict detection a'la predicate locking.

Predicate Locking

- Grant a lock on all records that satisfy some logical predicate (e.g., age > 2 * rank)
- Need a way to compute predicate intersections: Is it possible for a record to satisfy both predicates?
  - No analytical solution in the worst case.
  - Use a conservative approximation.

Predicate Locking

- Index locking is a special case of predicate locking when an index exists for the predicate.
- Can easily identify (superset) of records that satisfy multiple predicates based on pages.
- Easy to lock pages. Harder to lock predicates.
  - Predicate locking is rarely used.

Locking in B+ Trees

- How can we efficiently lock specific leaf pages?
  - **Note:** This is not quite multiple granularity locking
    - We may need to modify tree nodes (IX wrong).
    - Modifications to leaves may not affect the tree.
- **Solution 1**: Ignore the tree and lock **every** page accessed (S) or modified (X) (both tree and leaf).
  - Horrible performance! Why?

Locking in B+ Trees

- Higher levels of the tree are often only used to direct searches for leaf pages.
  - We only need an X on a tree page if a child of the page could possibly split/ merge on an insert/delete.
- We can exploit these observations to design a locking protocol that guarantees serializability, even though it violates 2PL.

Simple Tree Locking Algorithm

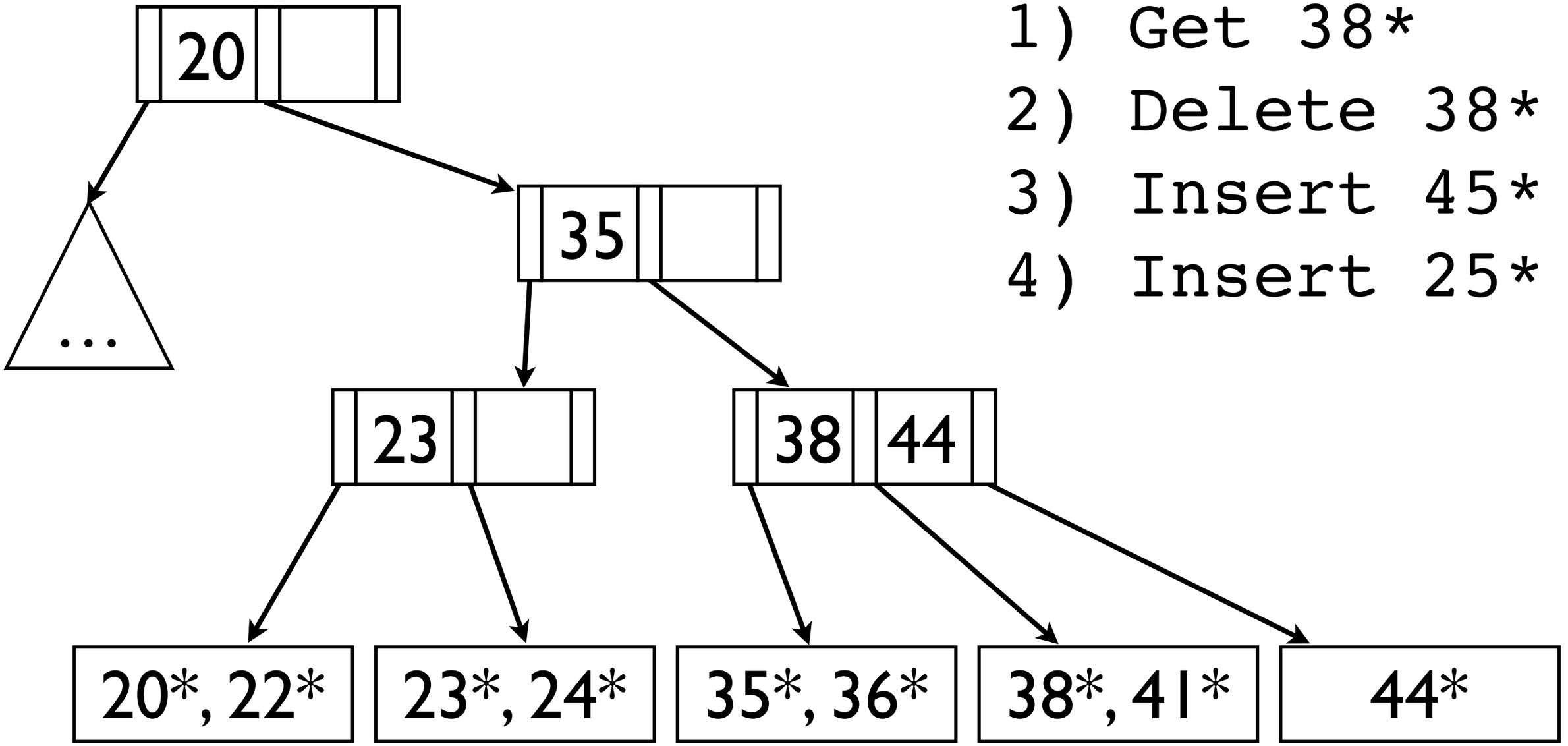
- **Scan:** Start at the root and descend.
  - Repeatedly S lock node, then unlock parent.
- **Update**: Start at the root and descend
  - Repeatedly X lock node.
  - If all children of a node are locked and safe, release the parent lock
  - Safe node: A node that will not propagate changes.

Simple Tree Locking Algorithm

- **Scan:** Start at the root and descend.
  - Repeatedly S lock node, then unlock parent.
- **Update**: Start at the root and descend
  - Repeatedly X lock node.
  - If all children of a node are locked and safe, release the parent lock
  - Safe node: A node that will not propagate changes.

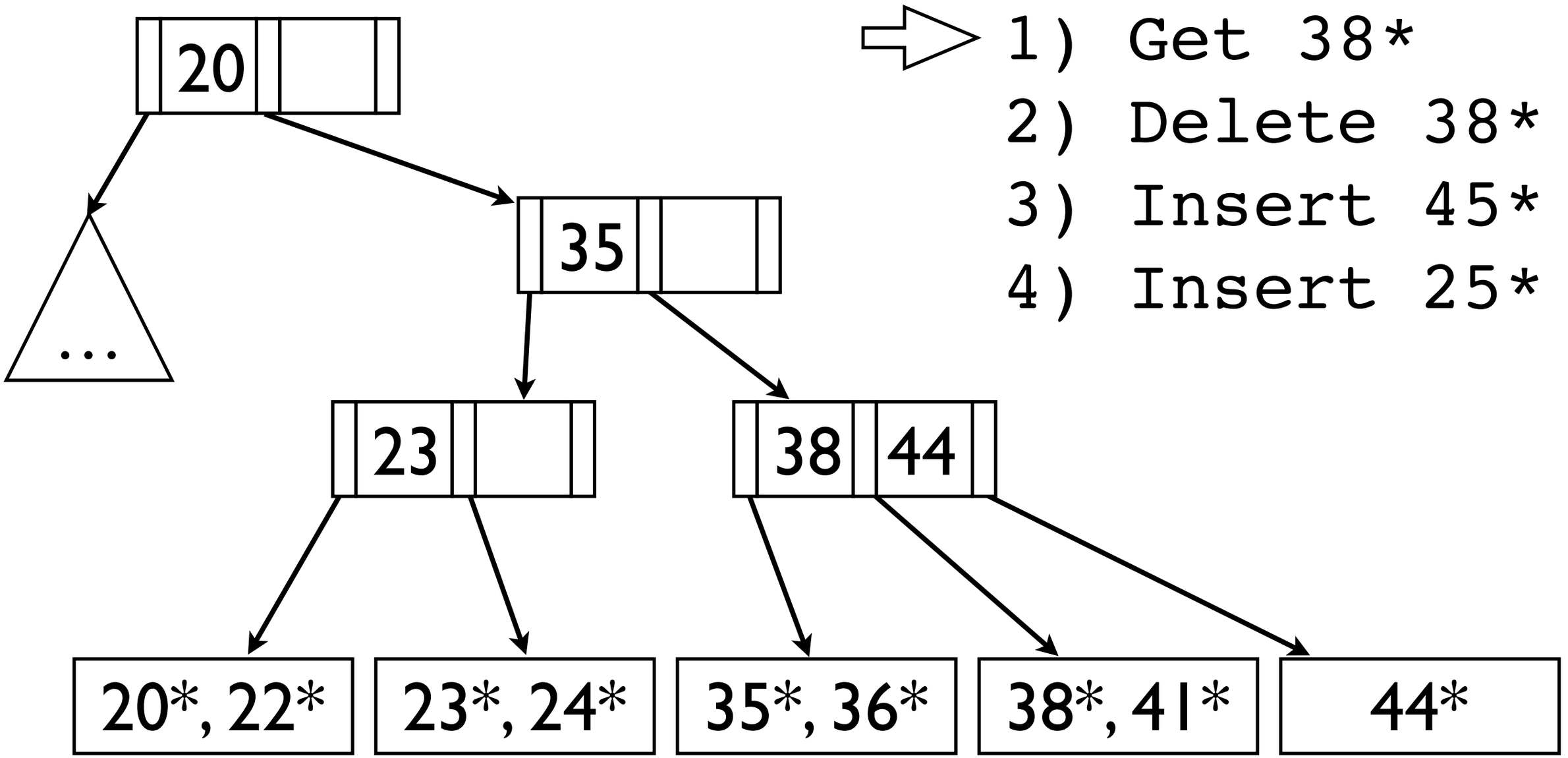
When is a node safe for single inserts? single deletes?

Examples



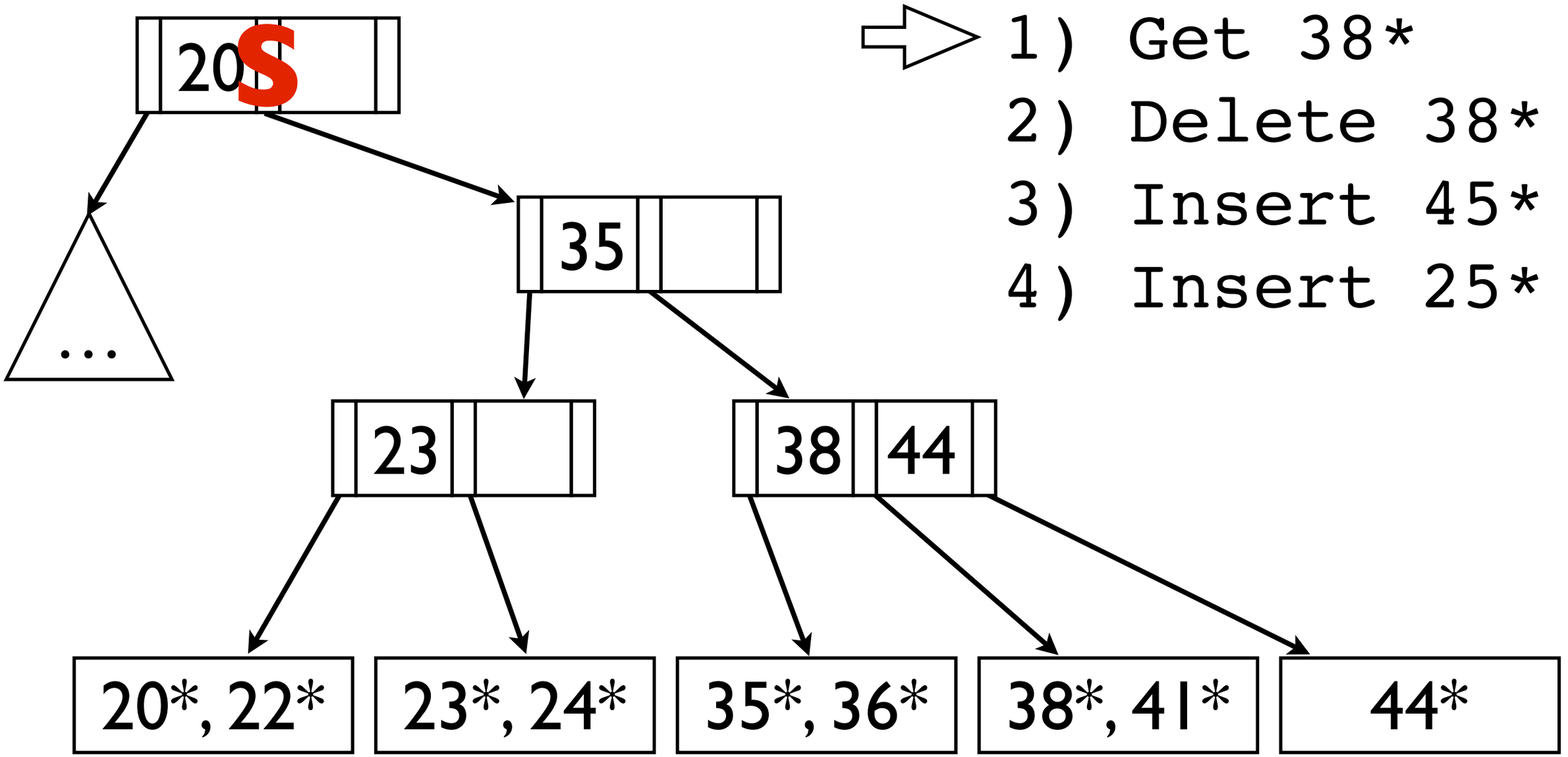
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



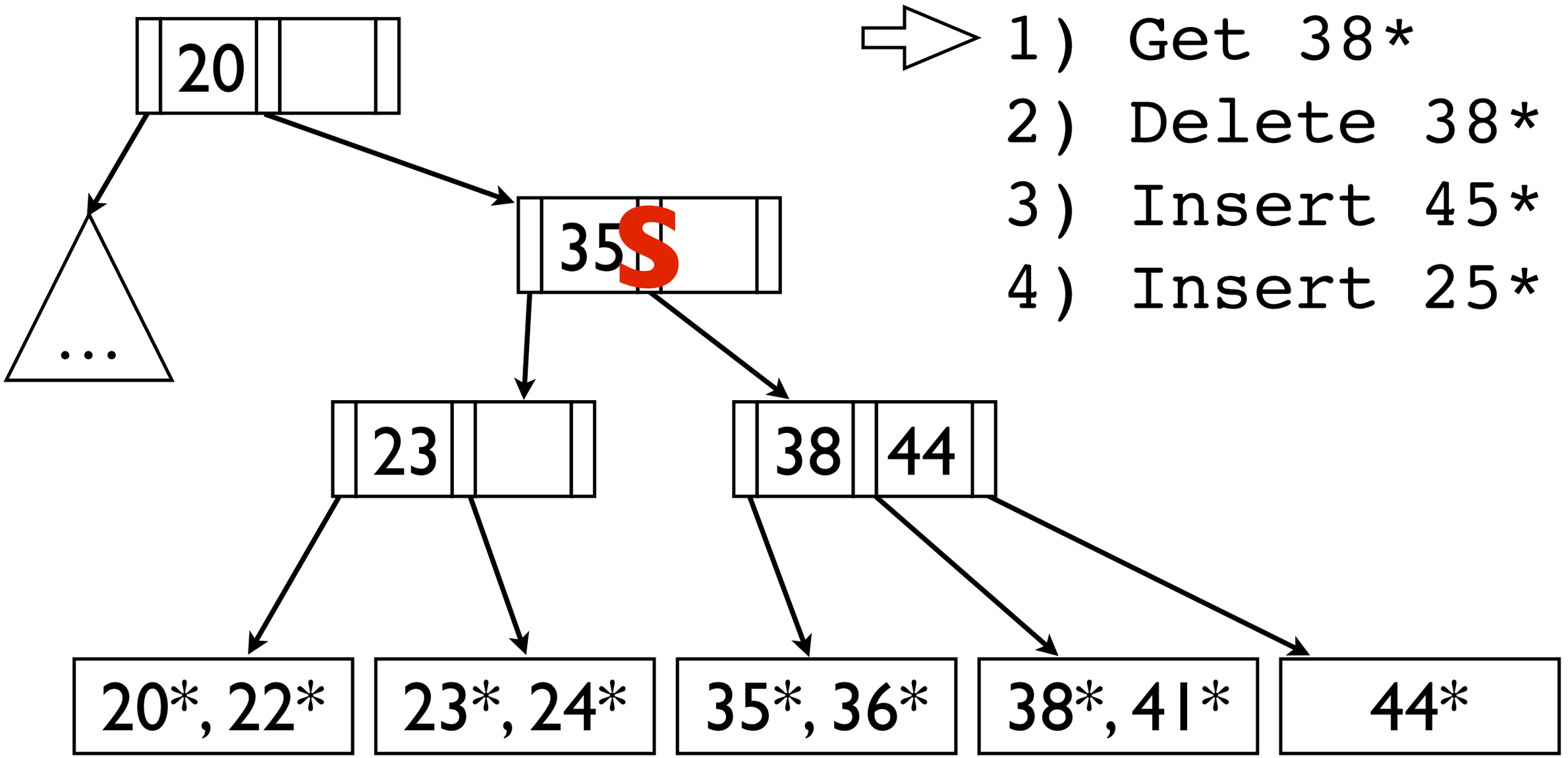
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



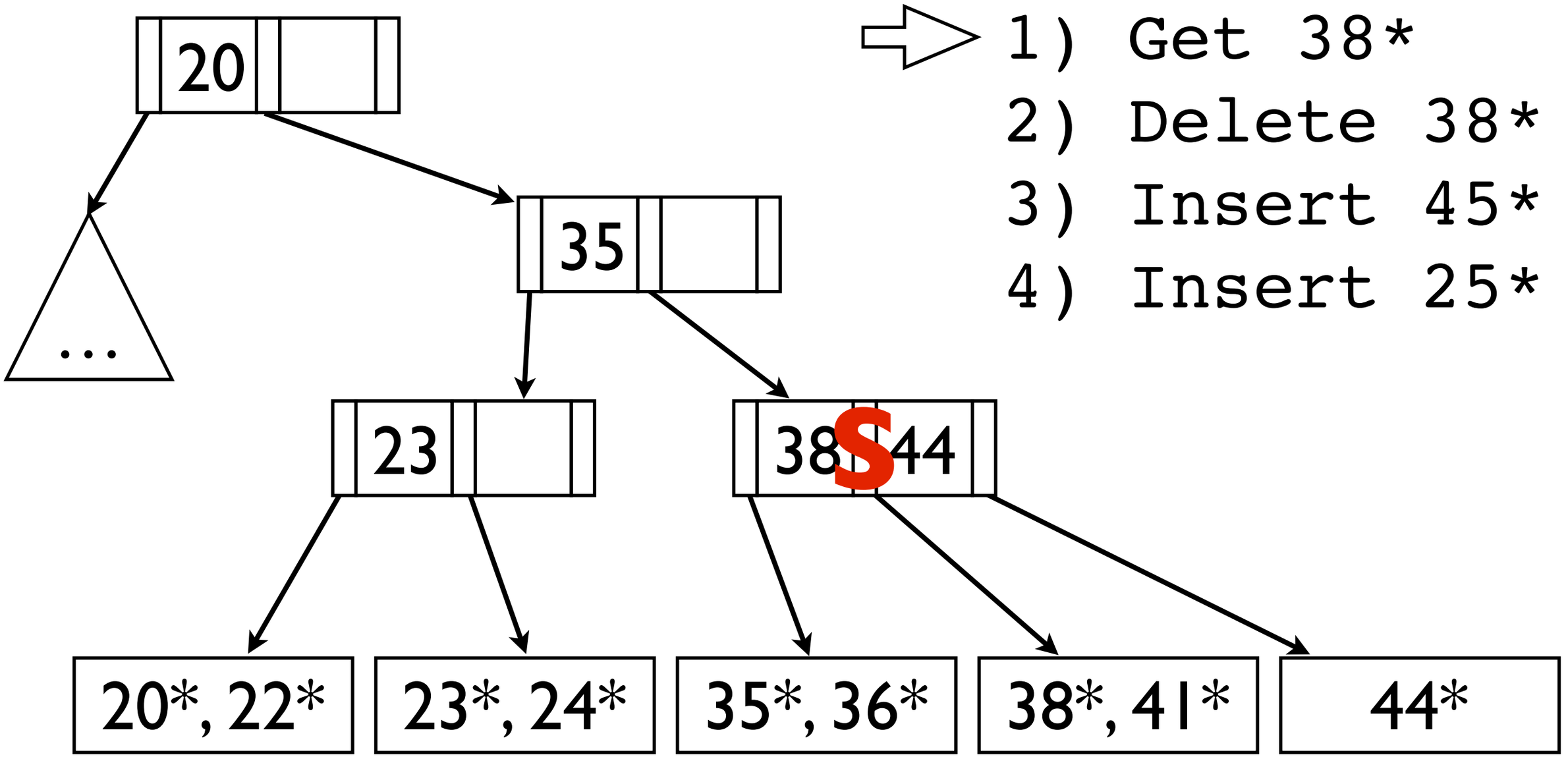
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



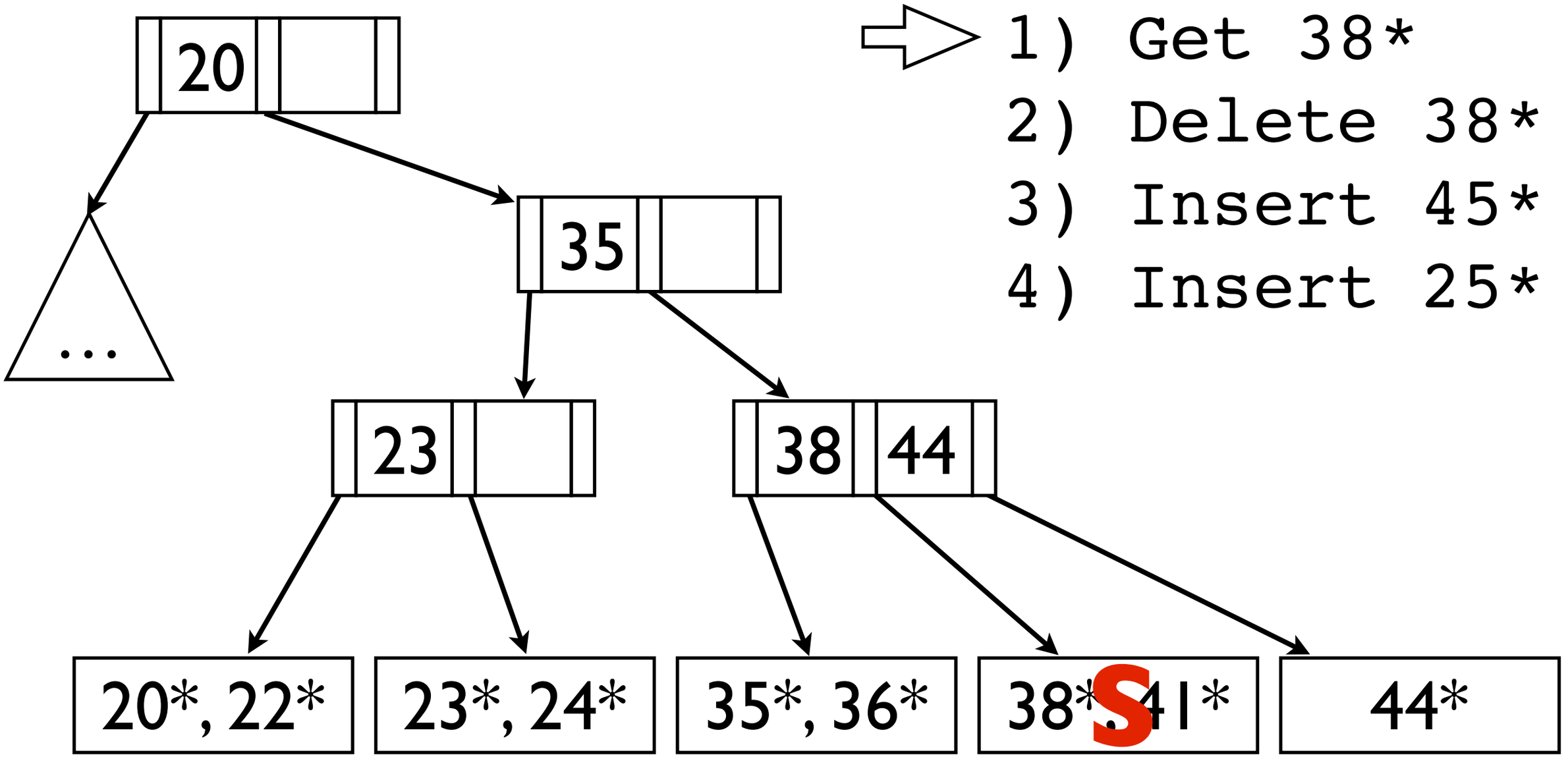
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



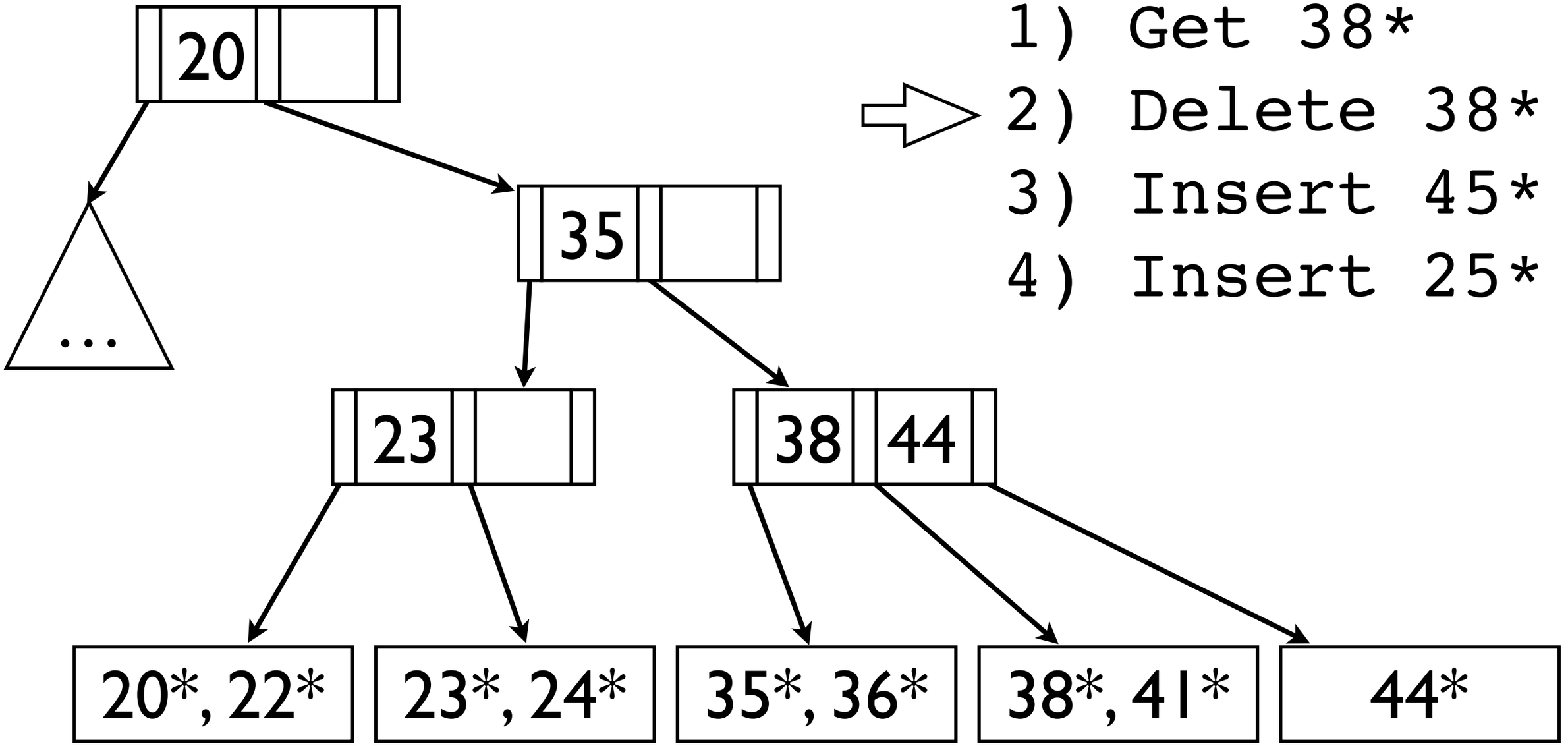
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



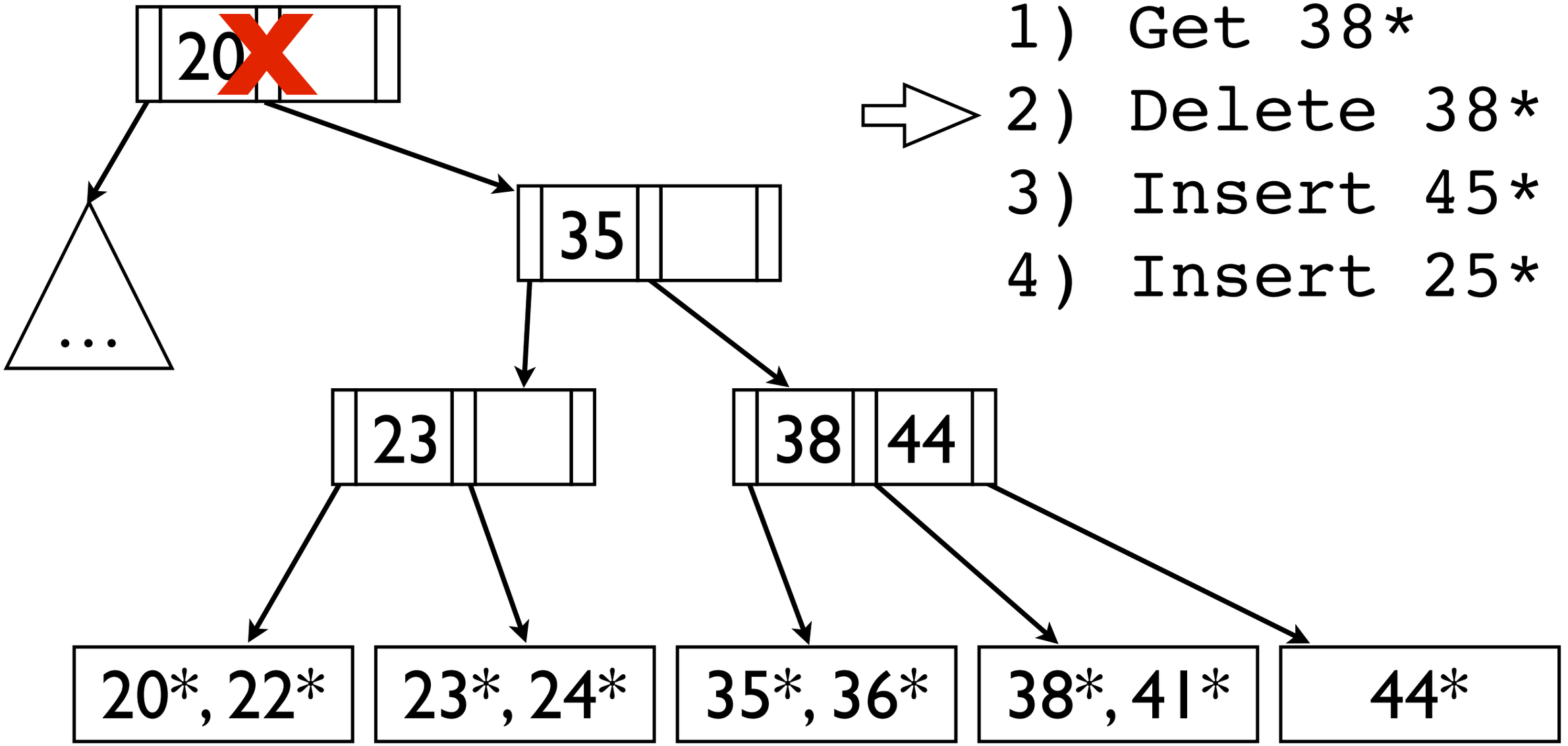
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



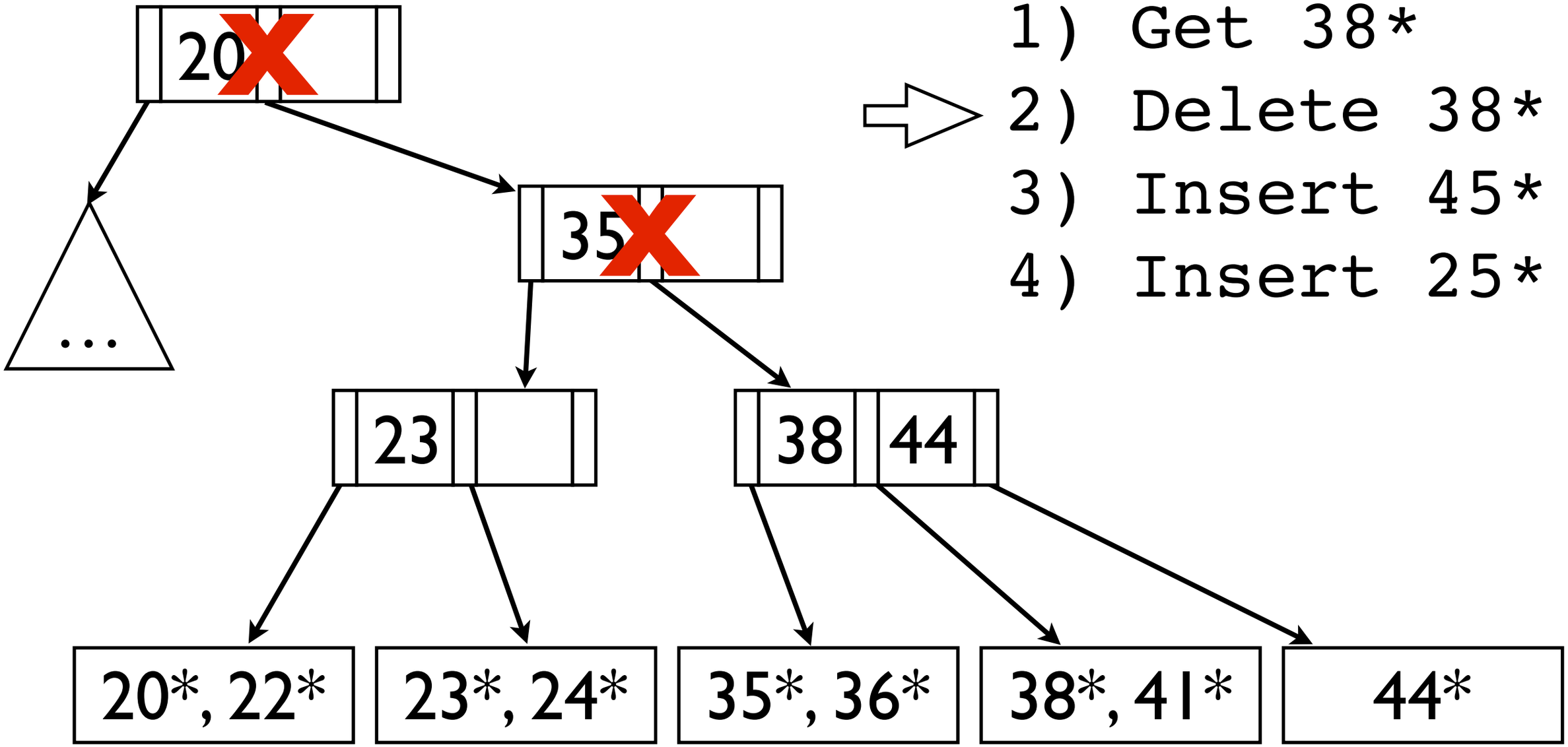
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



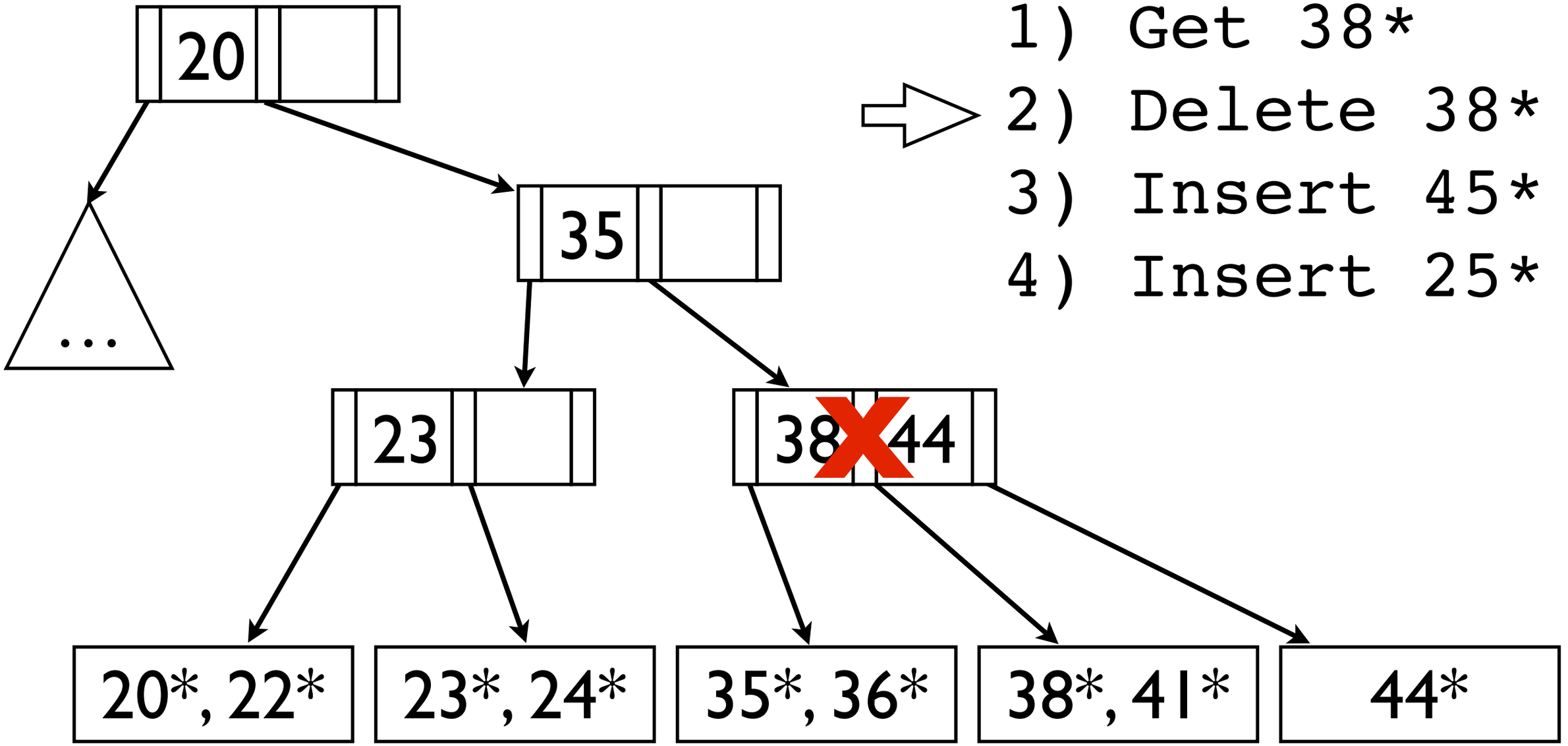
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



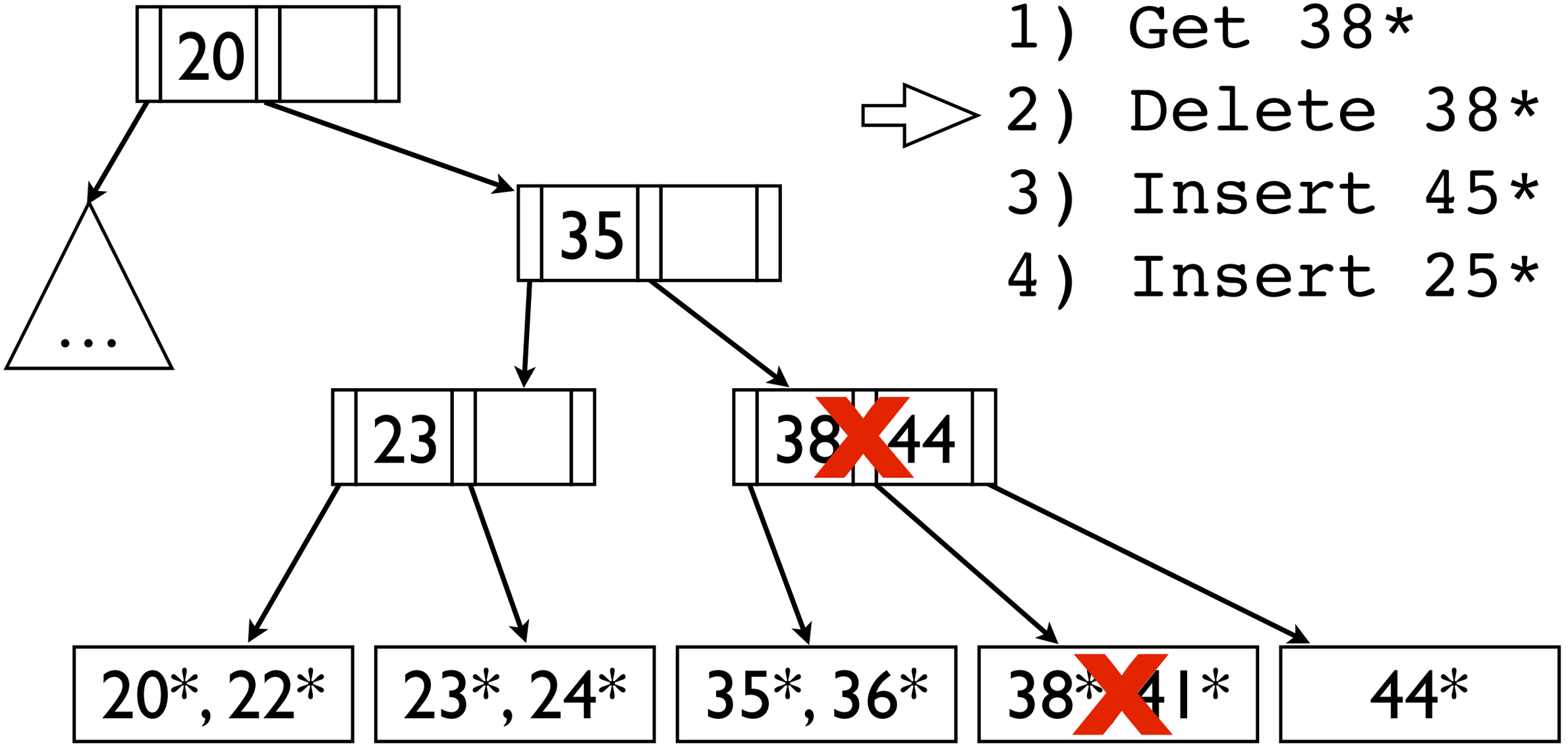
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



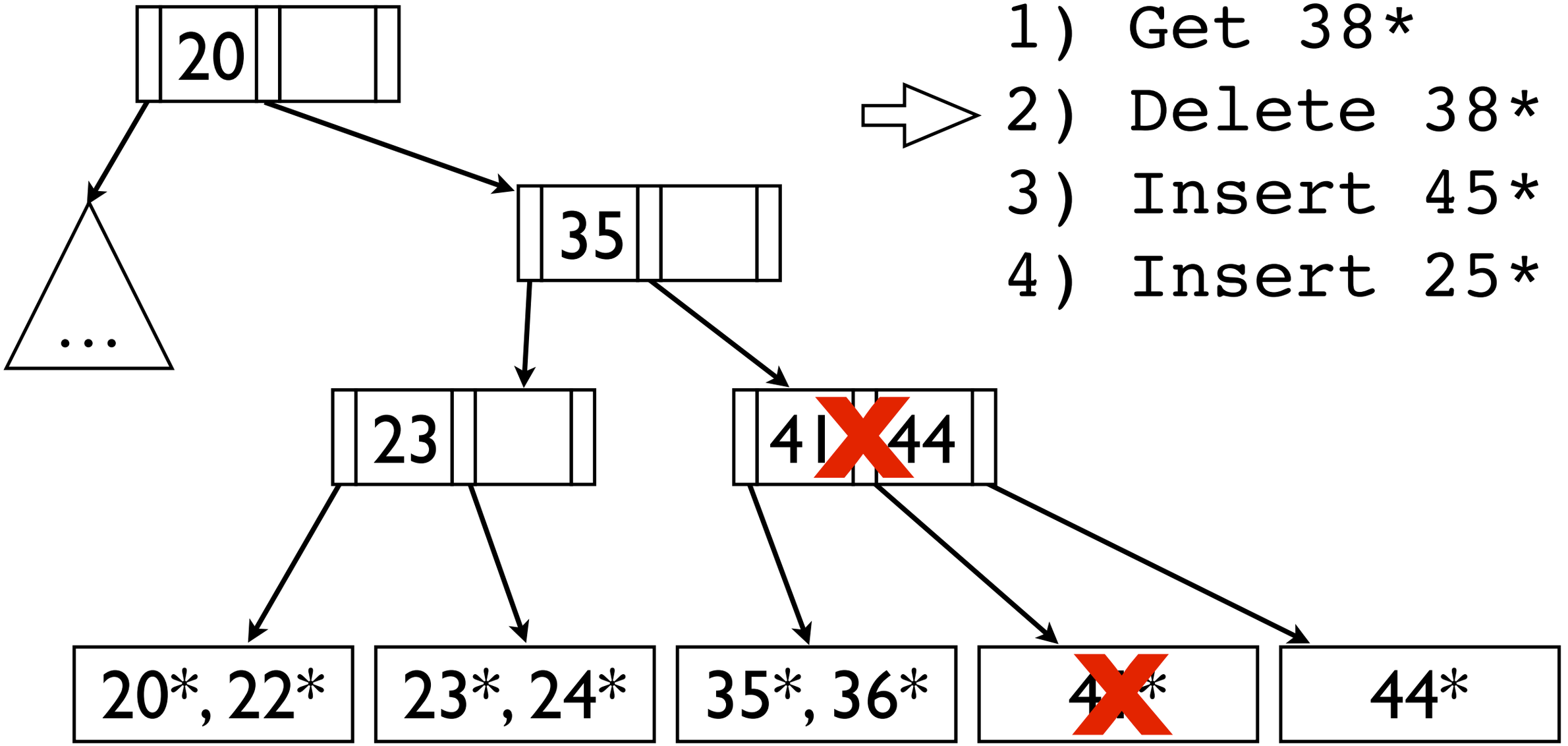
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



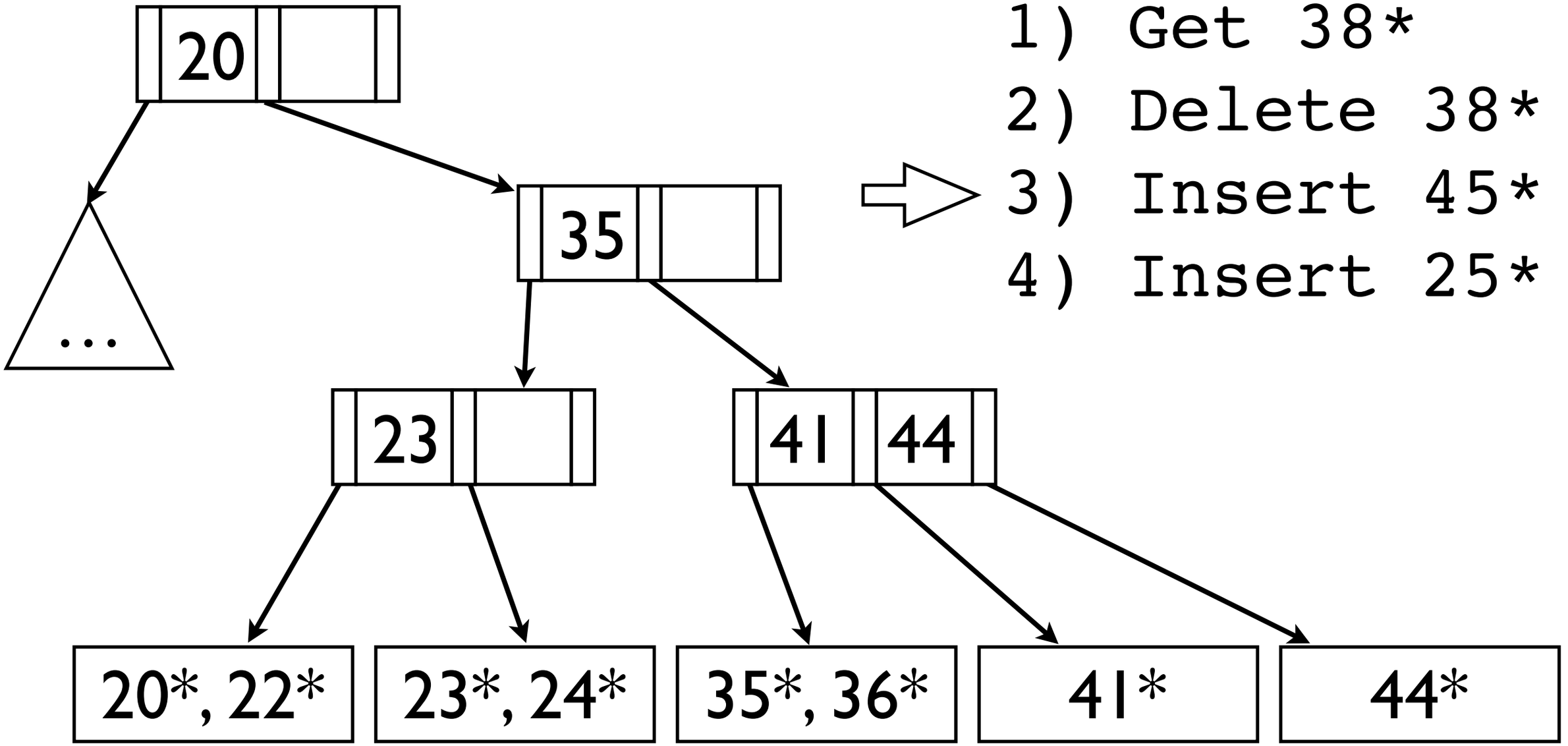
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



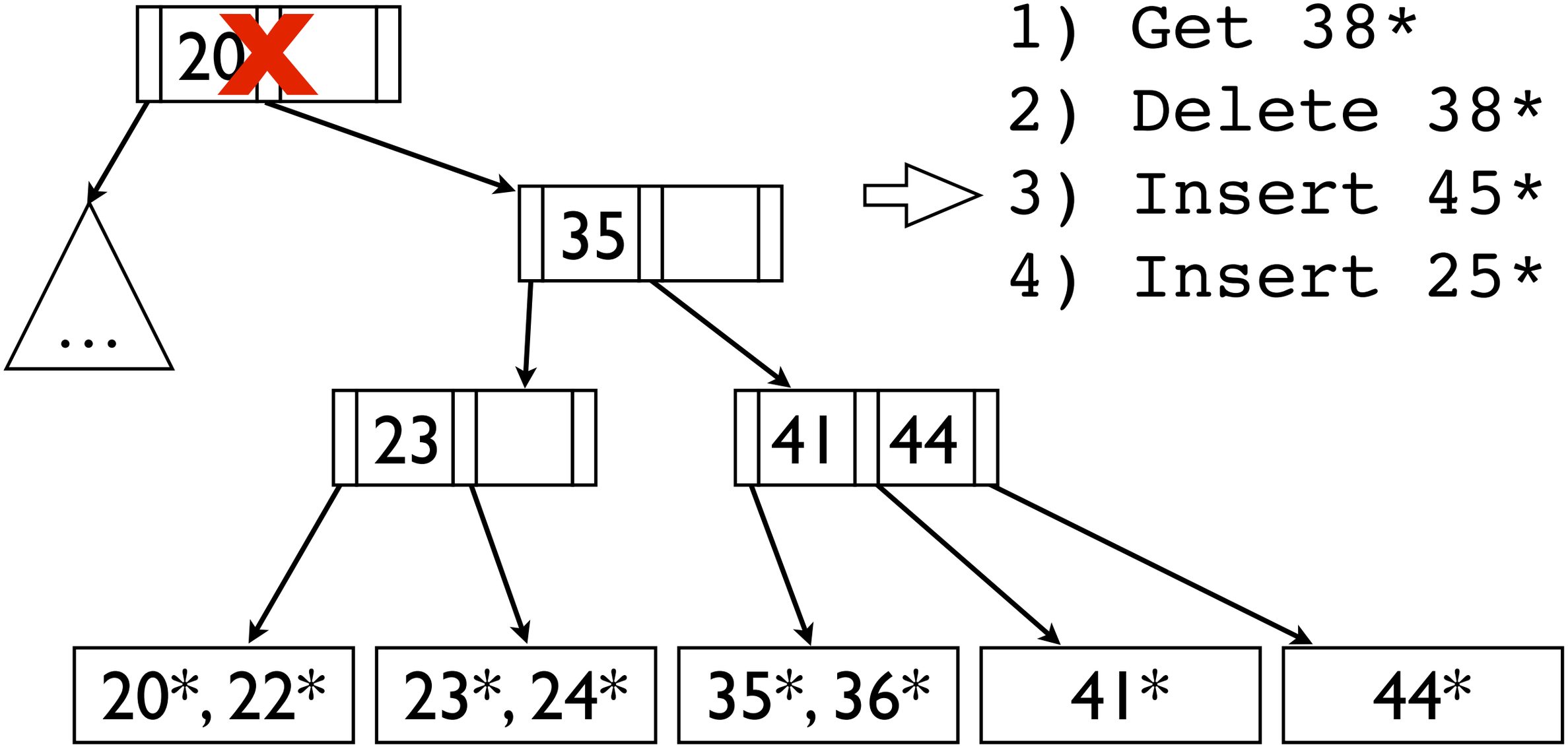
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



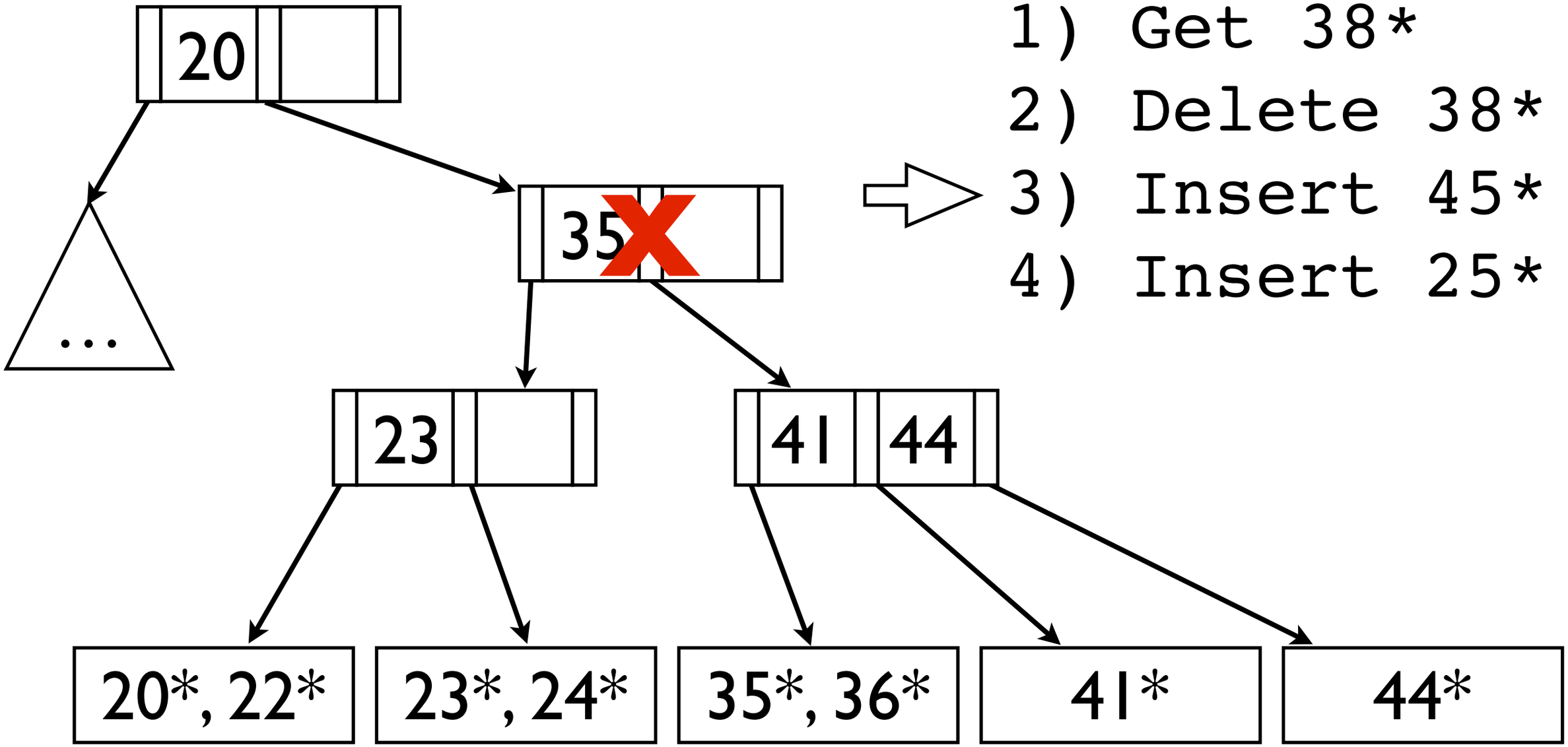
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



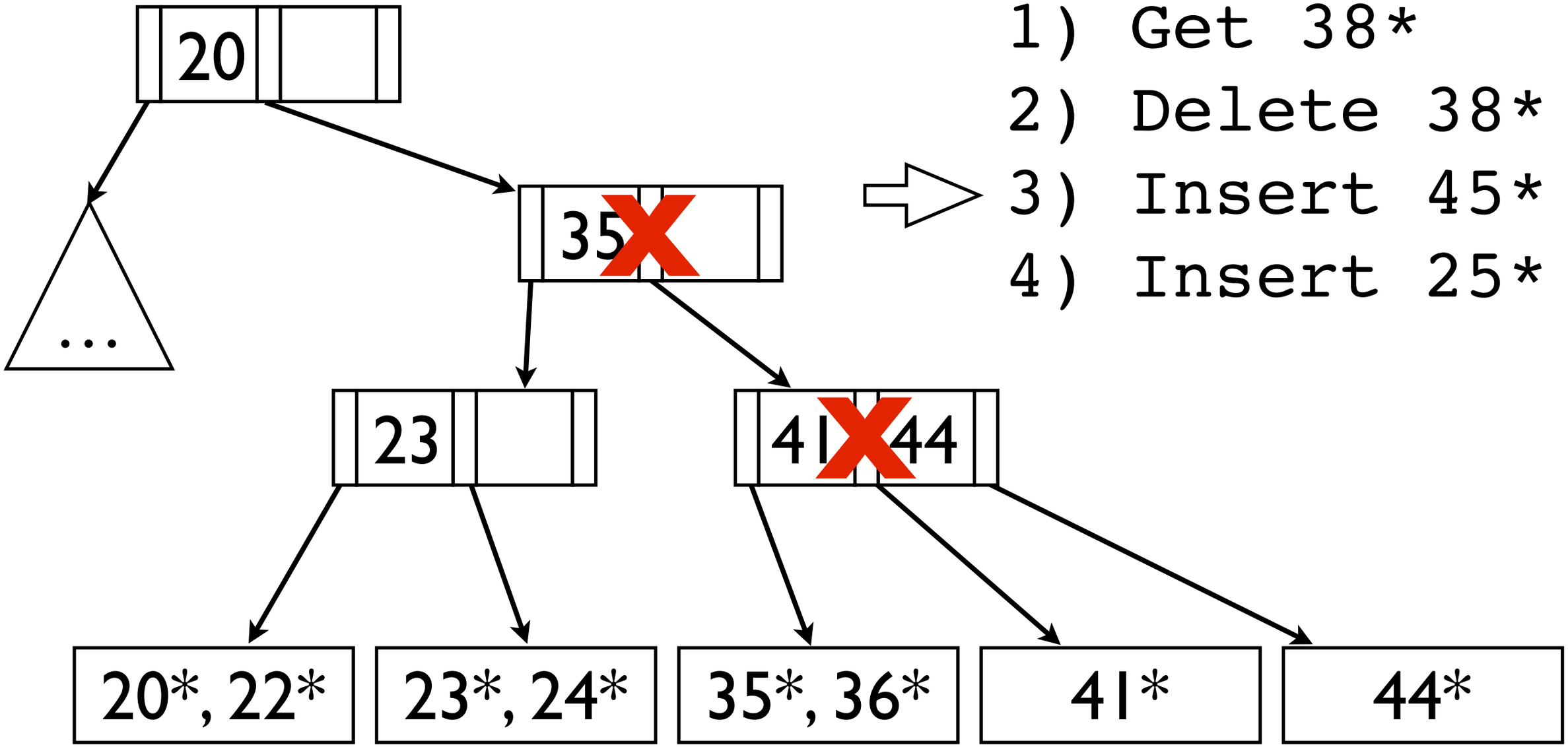
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



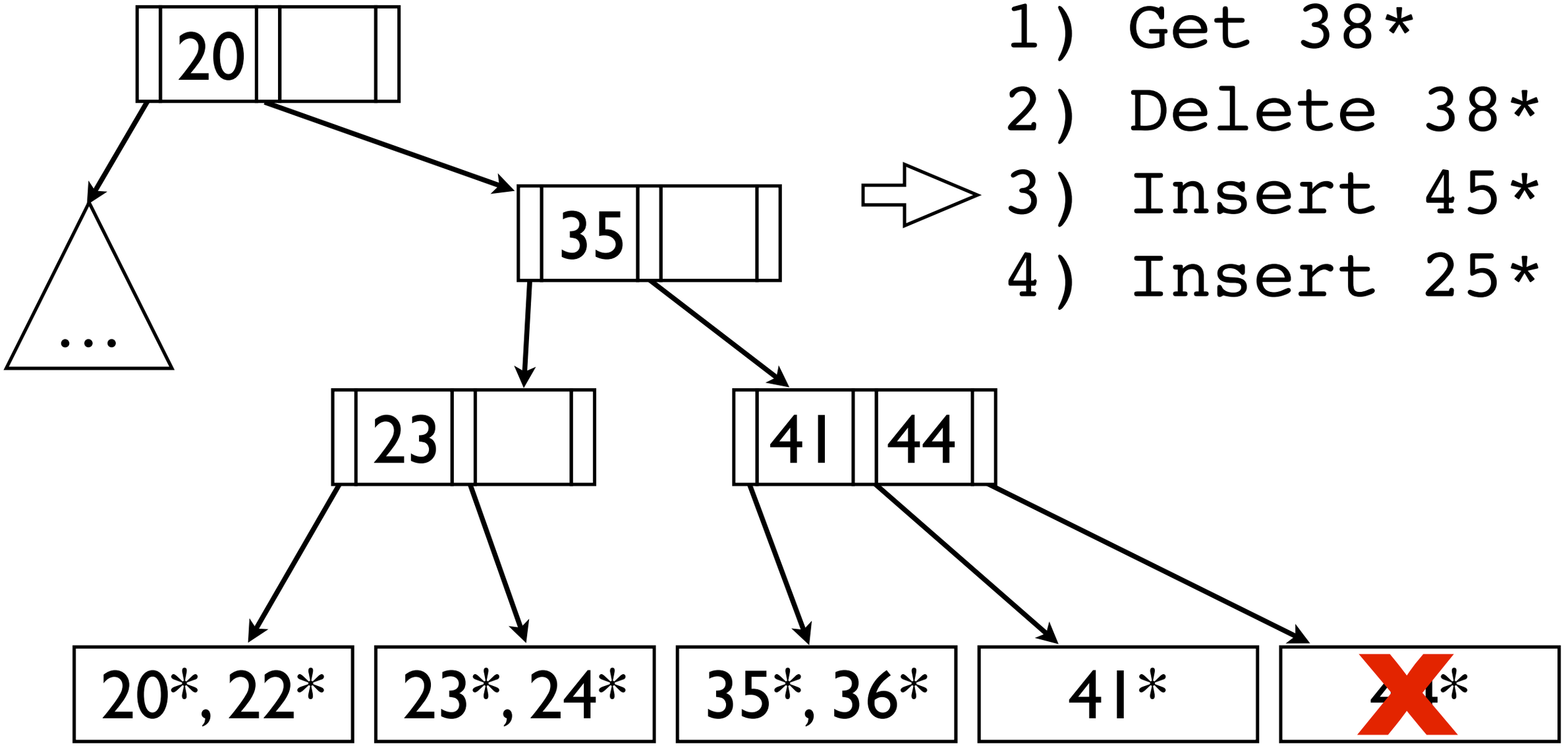
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



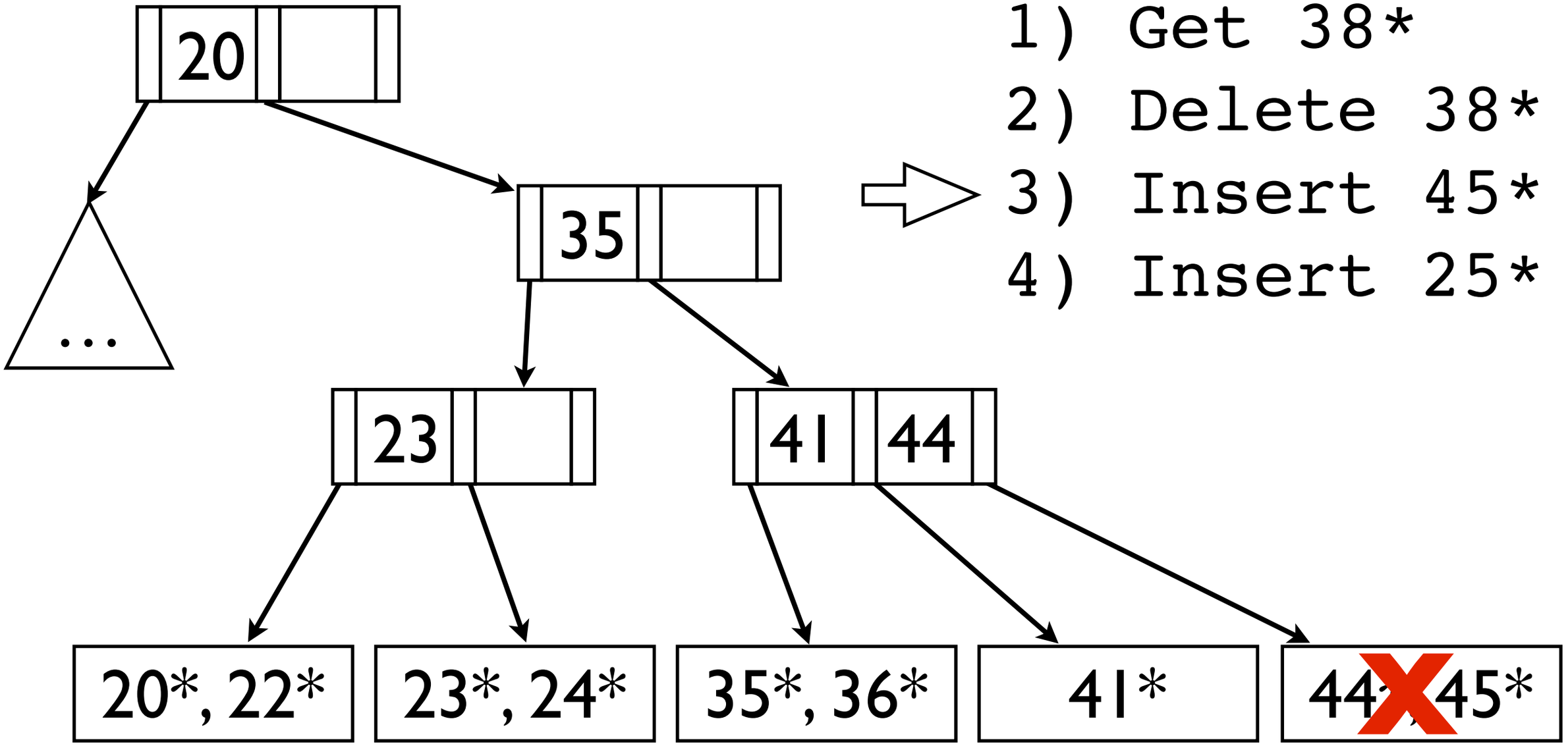
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



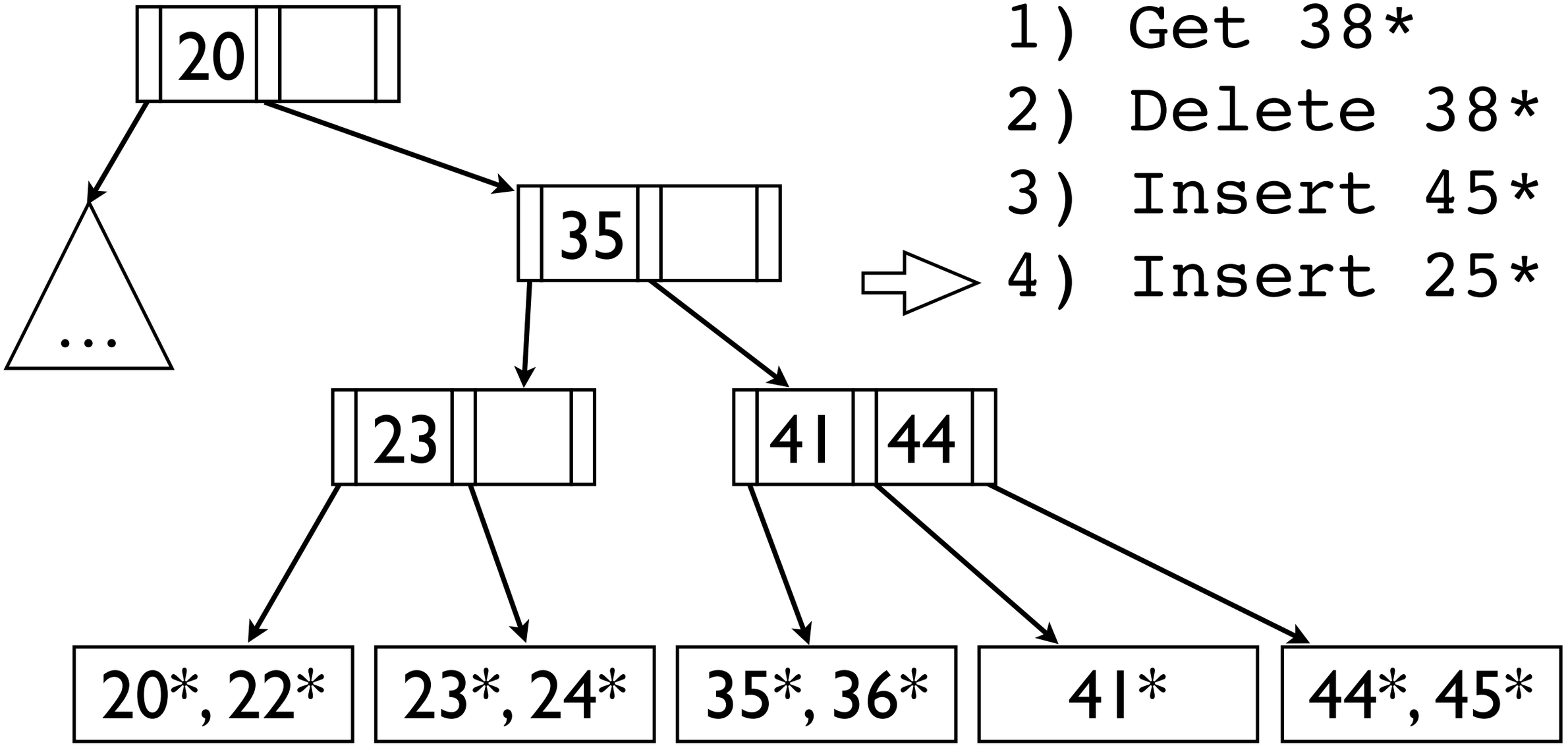
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



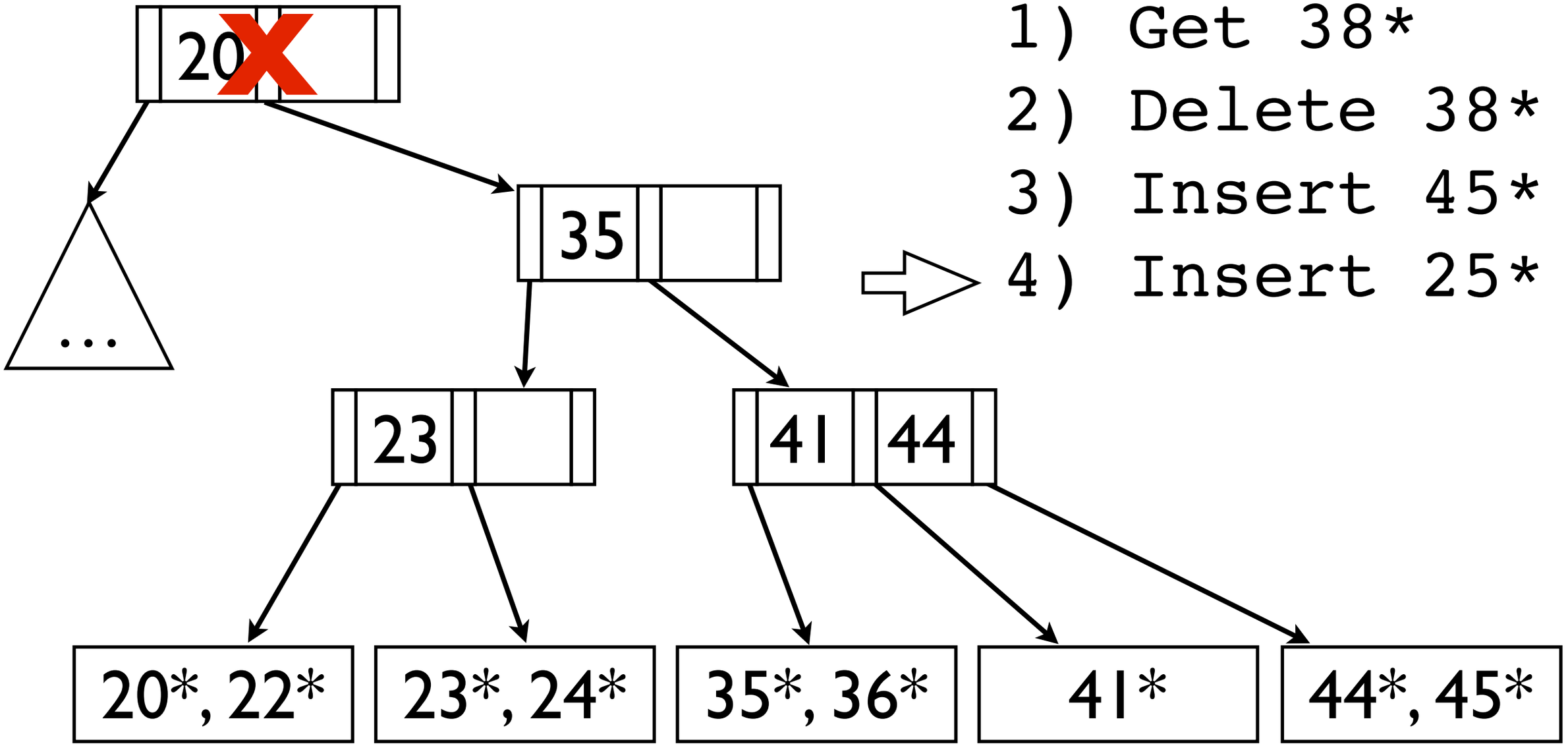
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



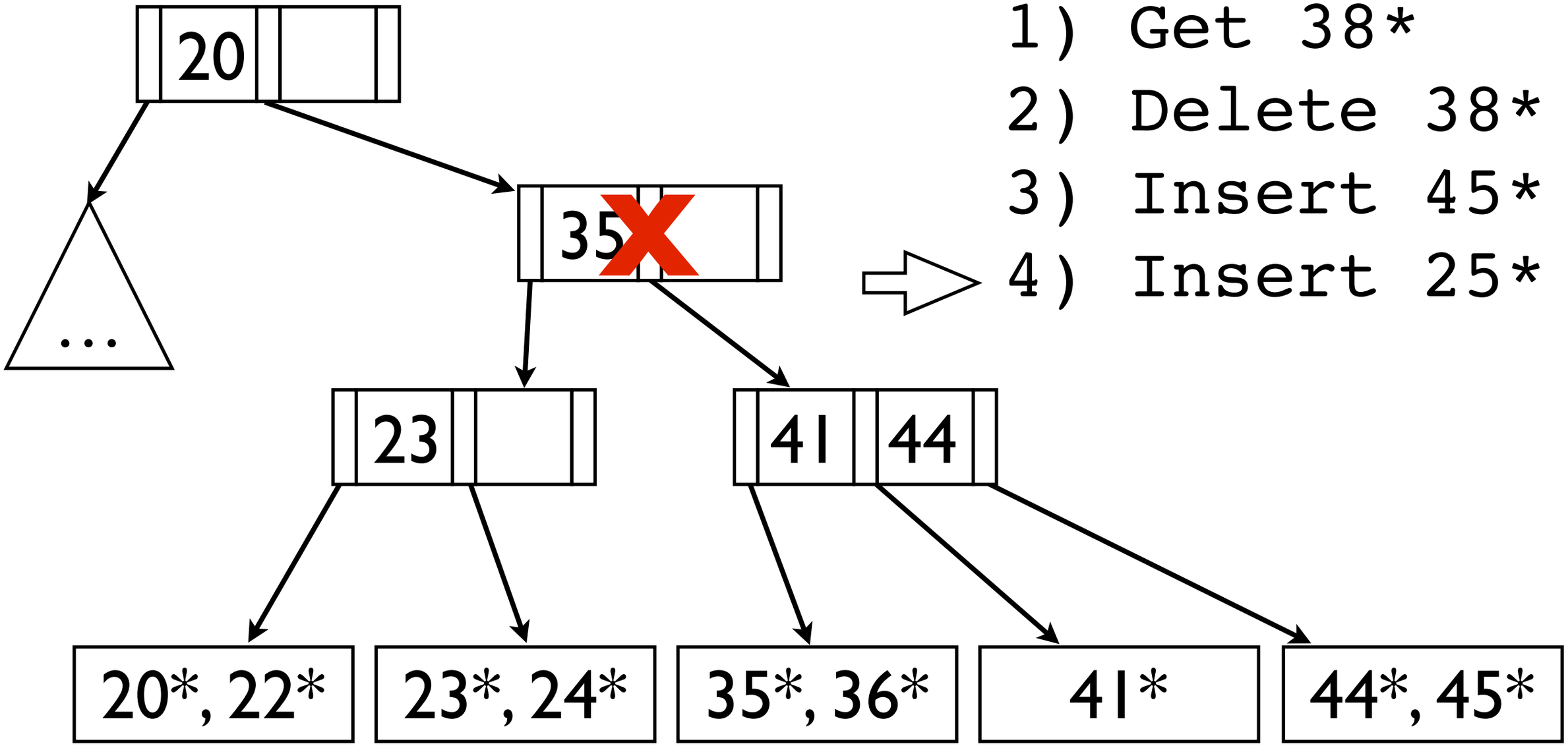
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



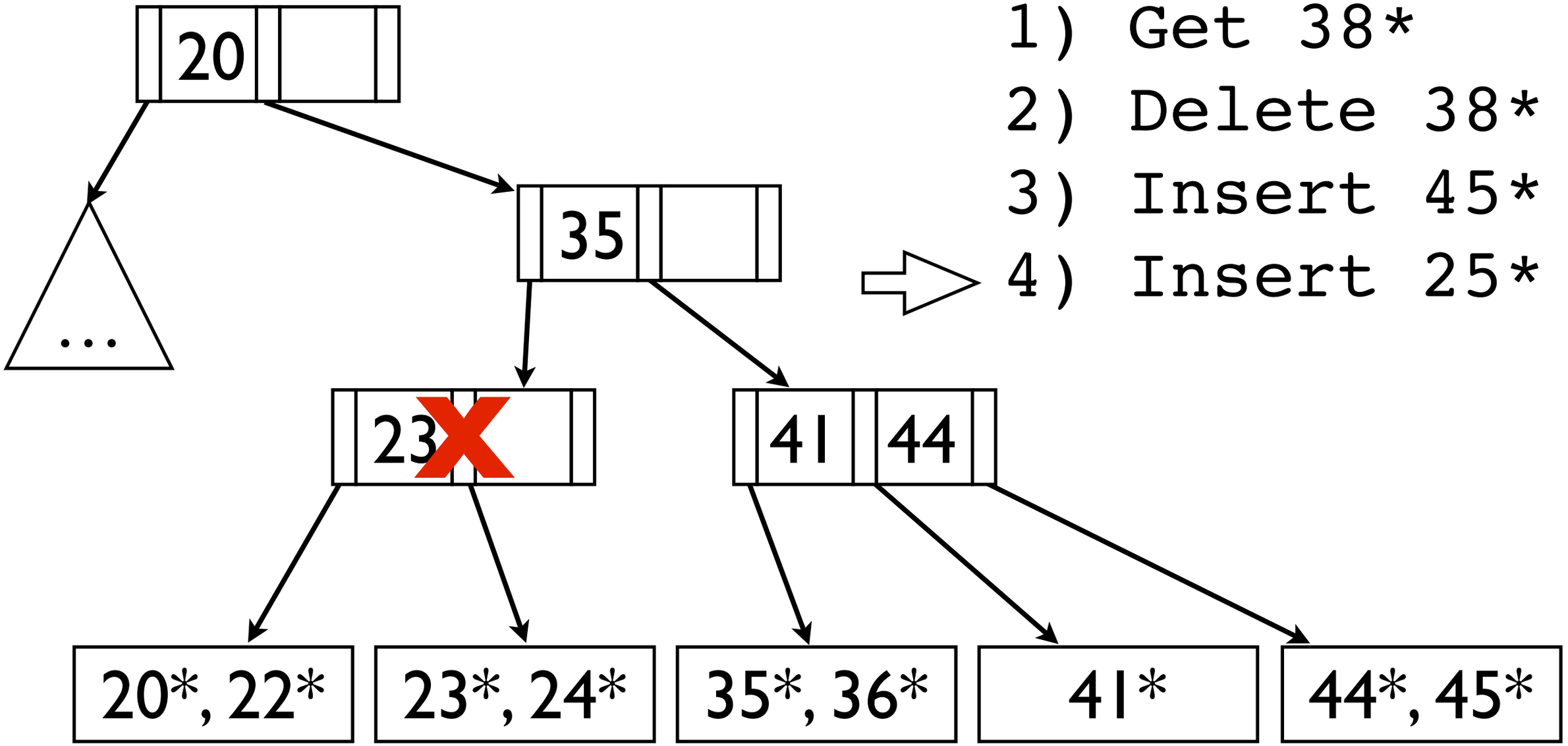
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



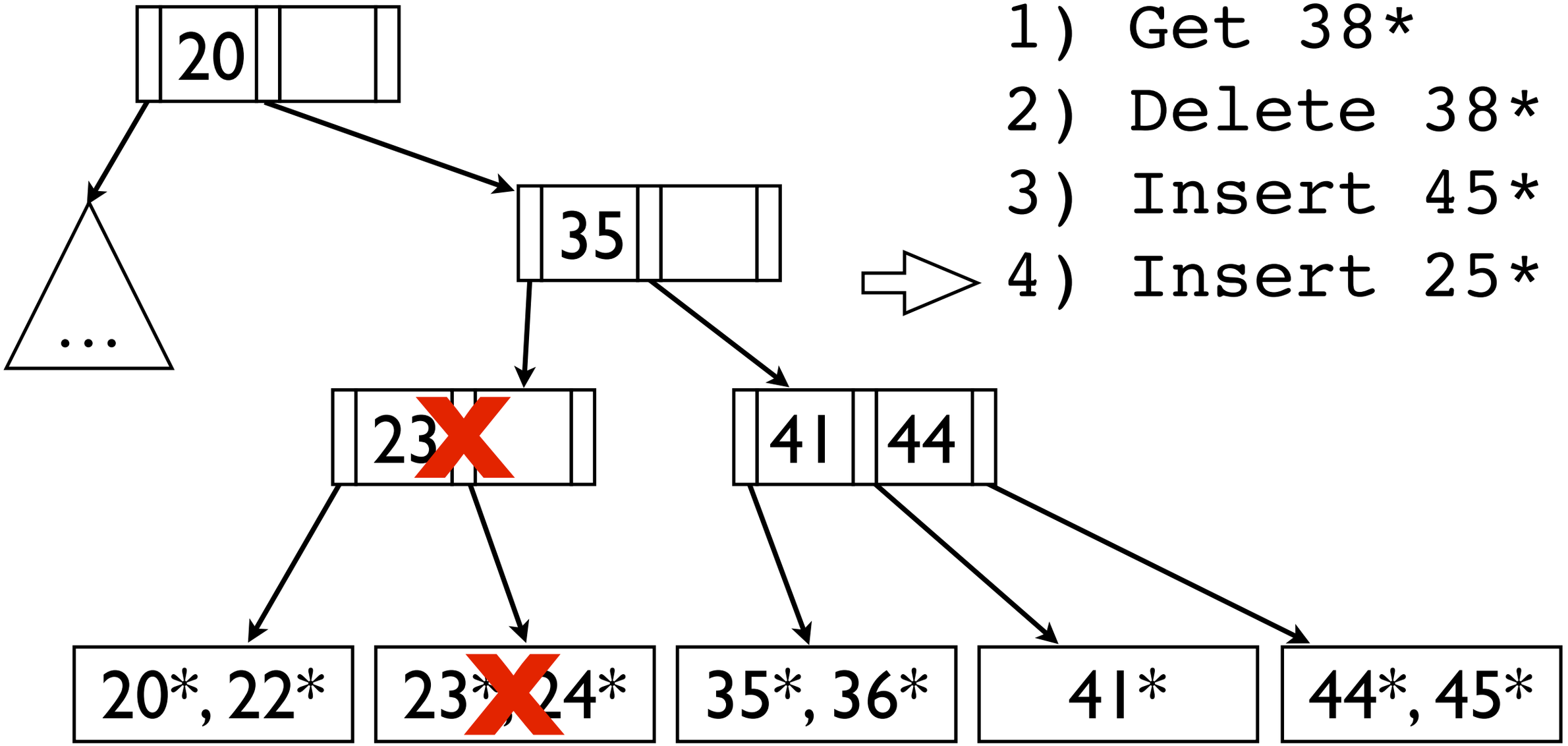
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



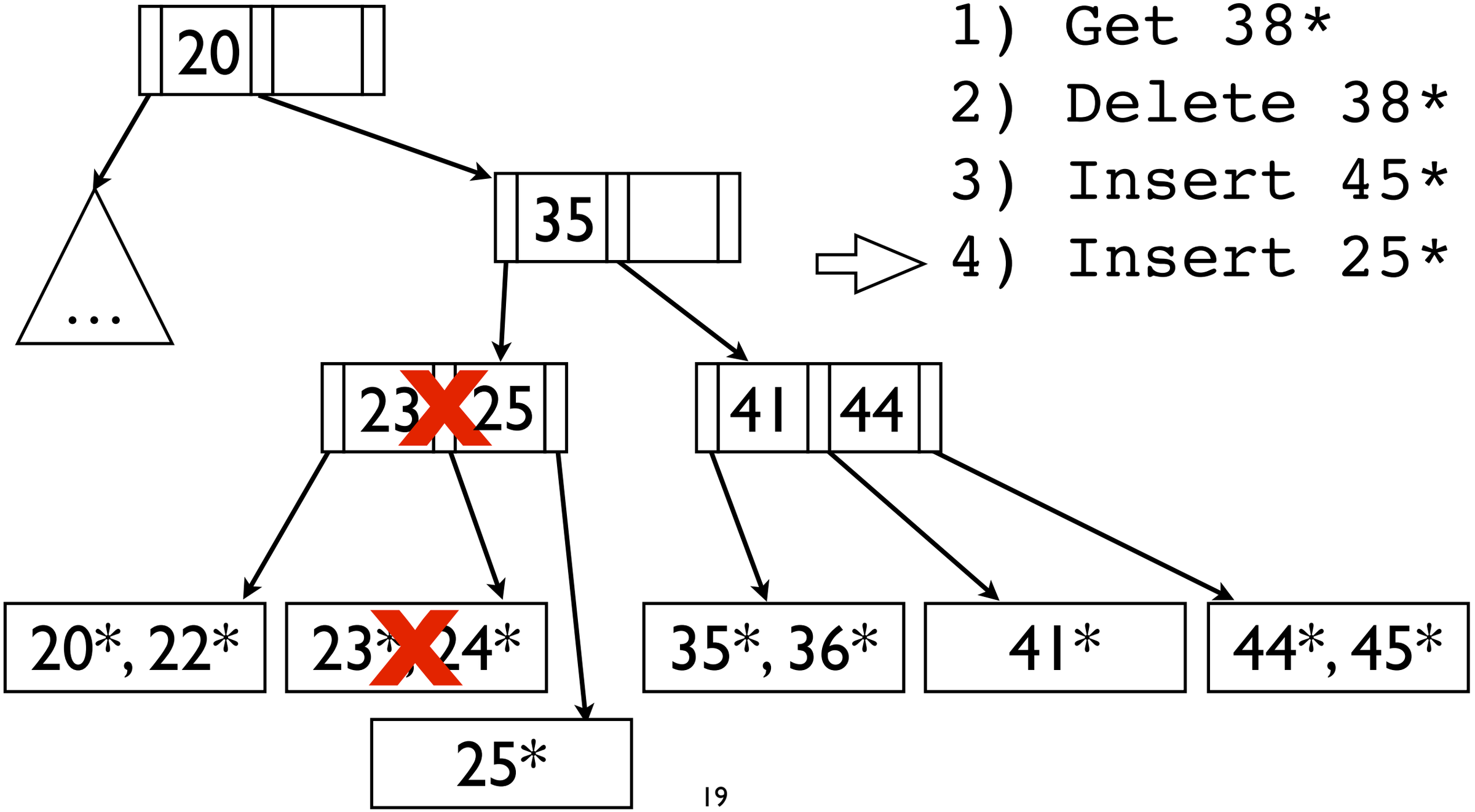
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples

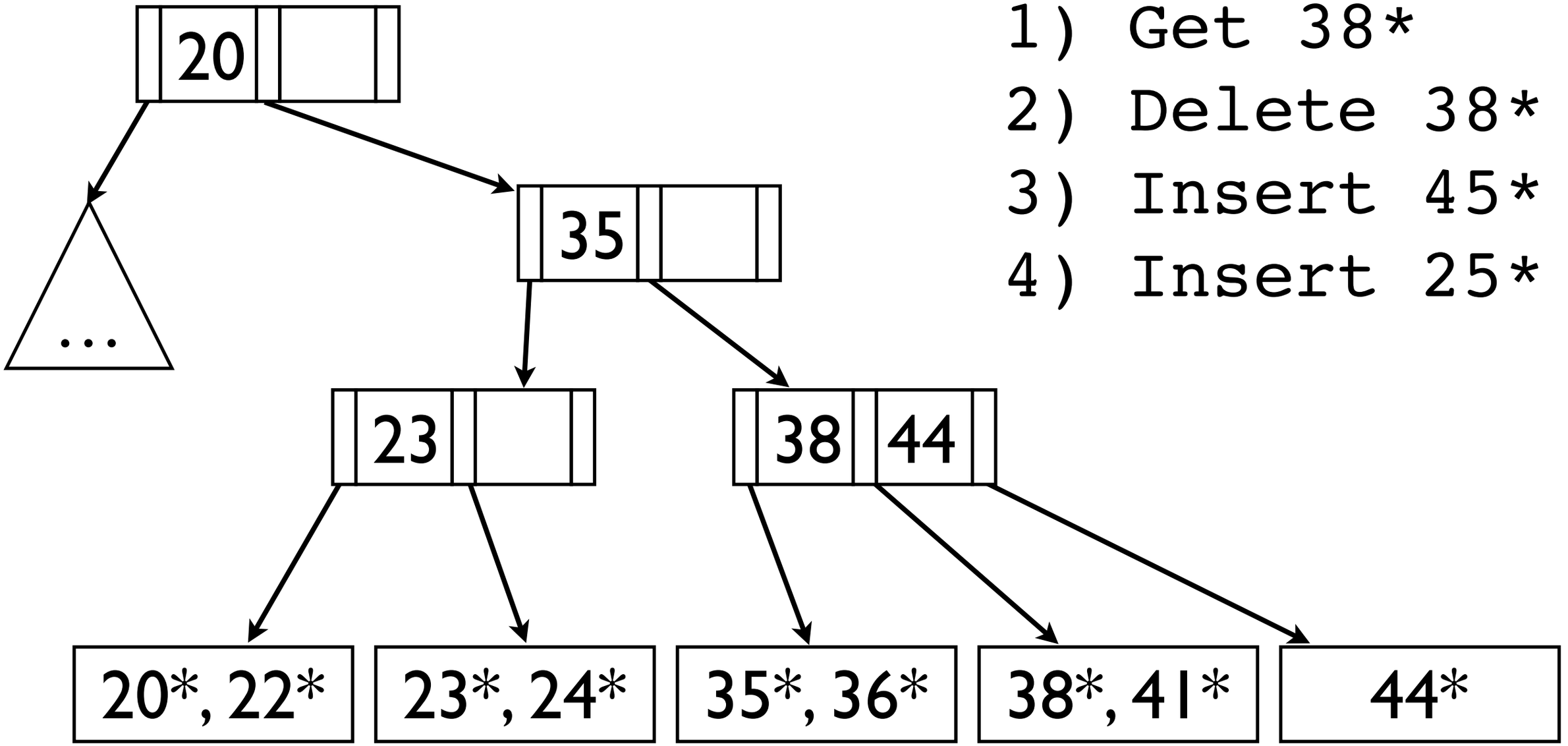


1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Better Tree Locking Algorithm (Bayer Schkolnick)

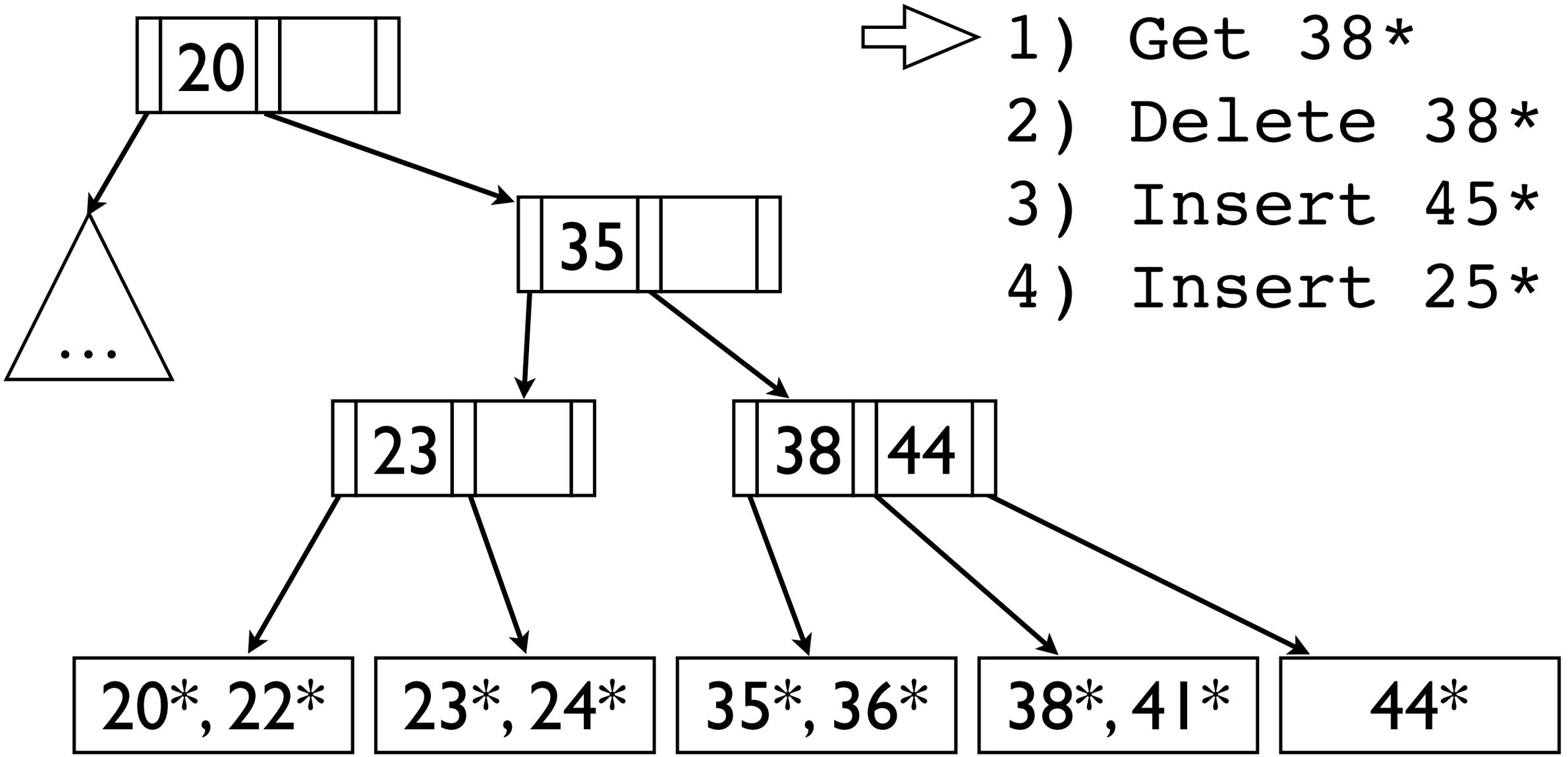
- **Scan:** As before
- **Update**: Set locks as if for search (using S)
  - Acquire X lock on the leaf.
  - If leaf is not safe, release all locks and restart using the simple algorithm.
- Gambles that only the leaf node will be modified.
  - S locks set on first pass are wasted otherwise.
  - In practice, better than the simple algorithm.

Examples



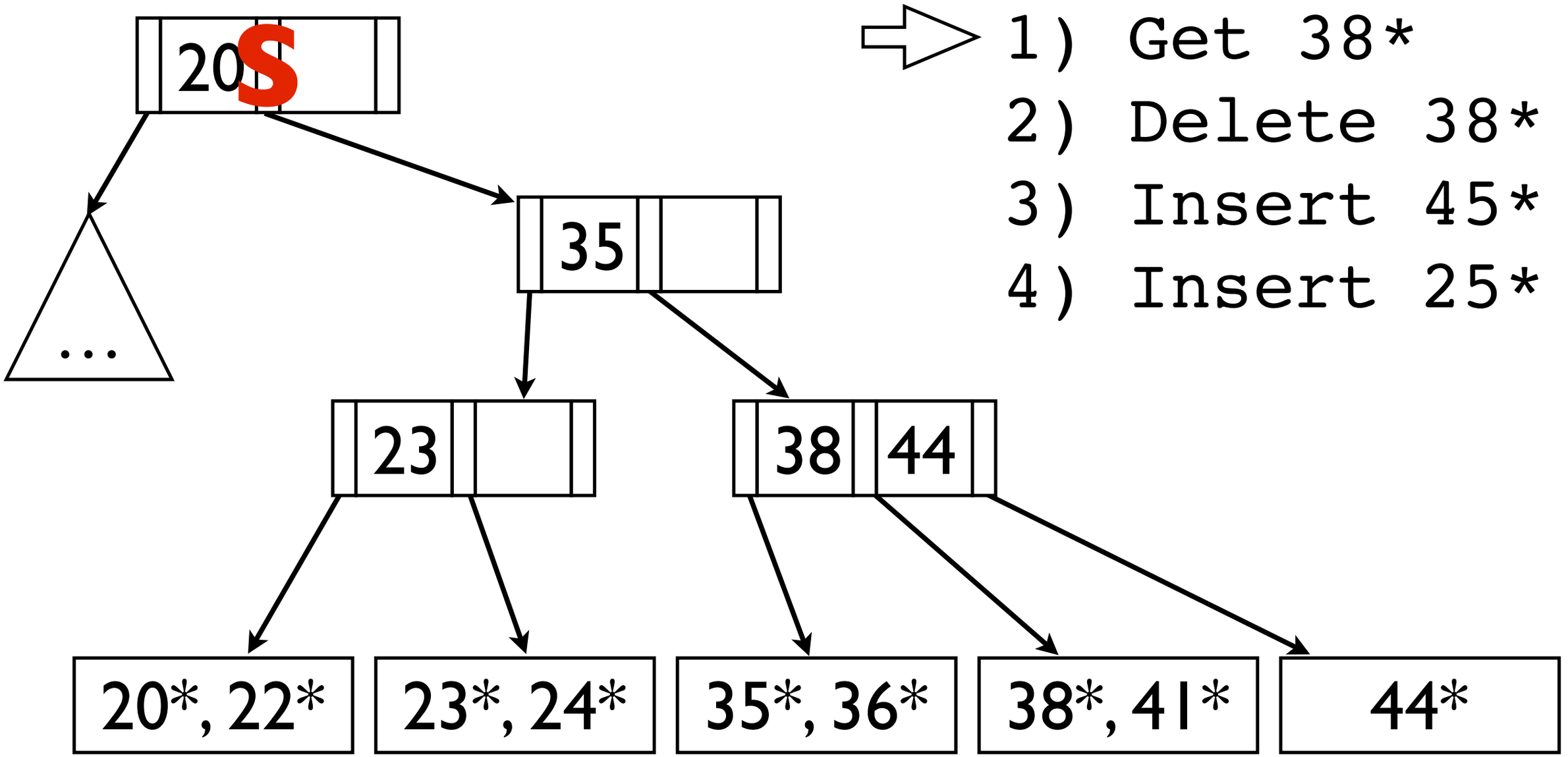
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



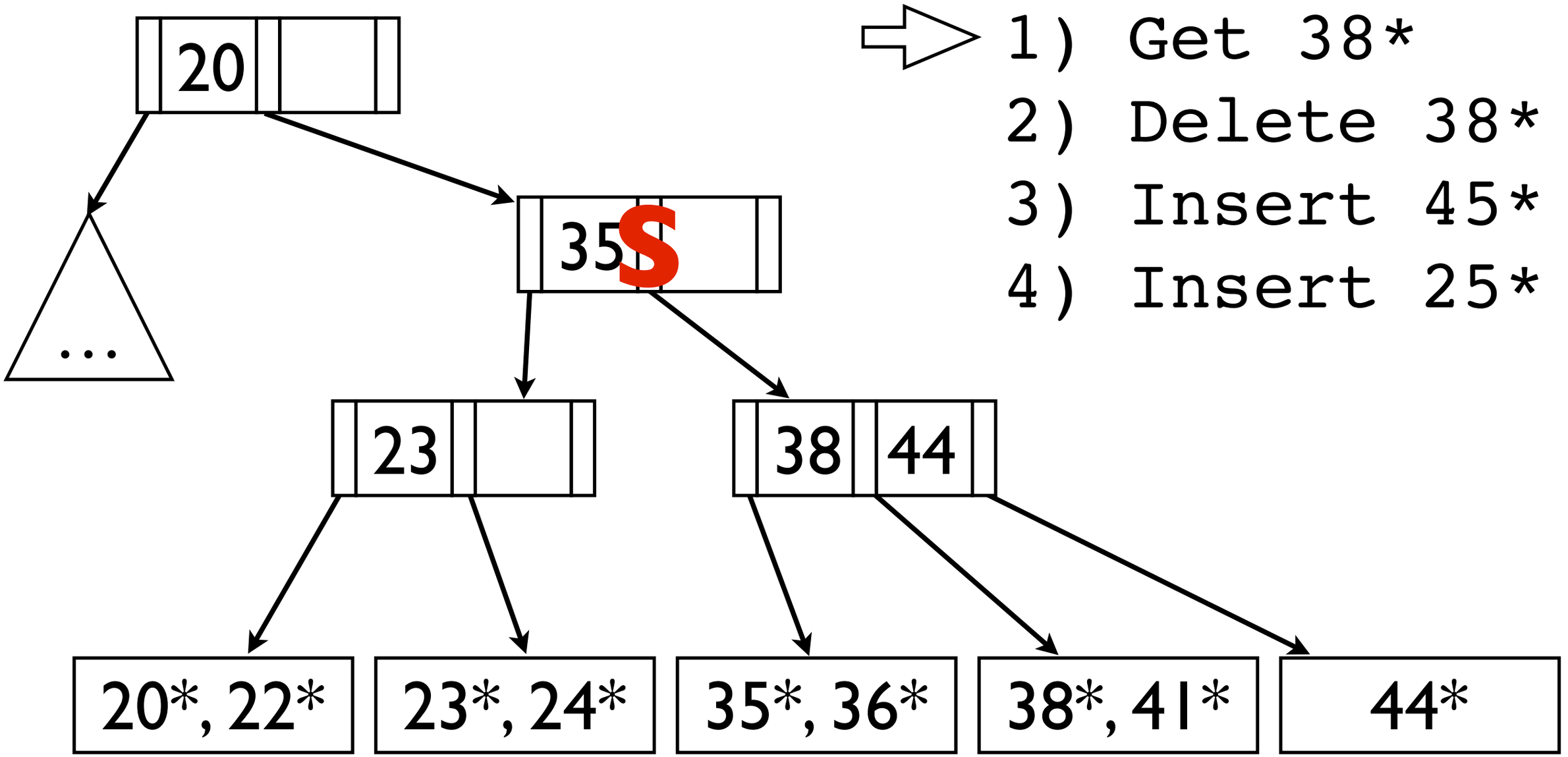
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



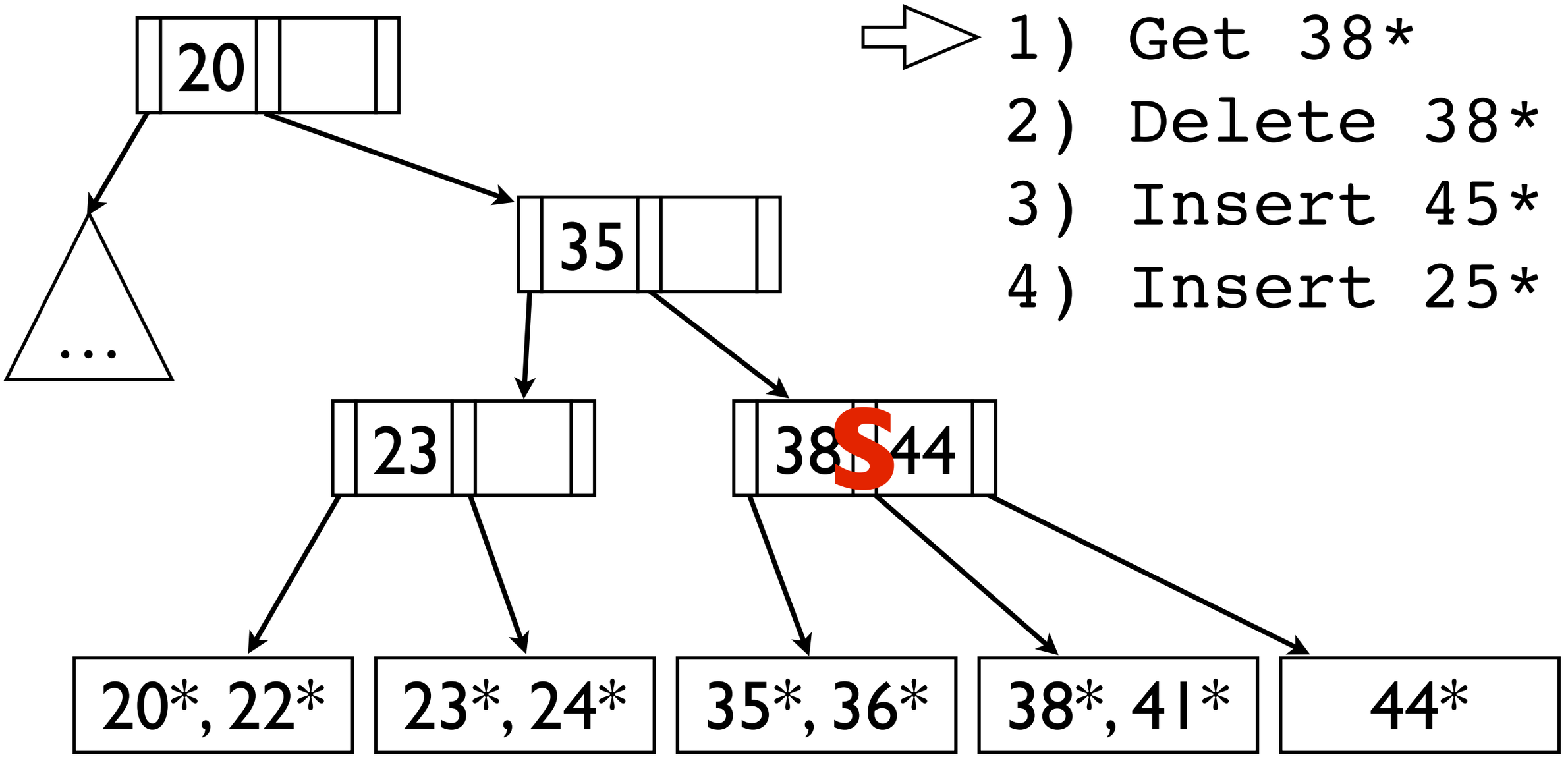
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



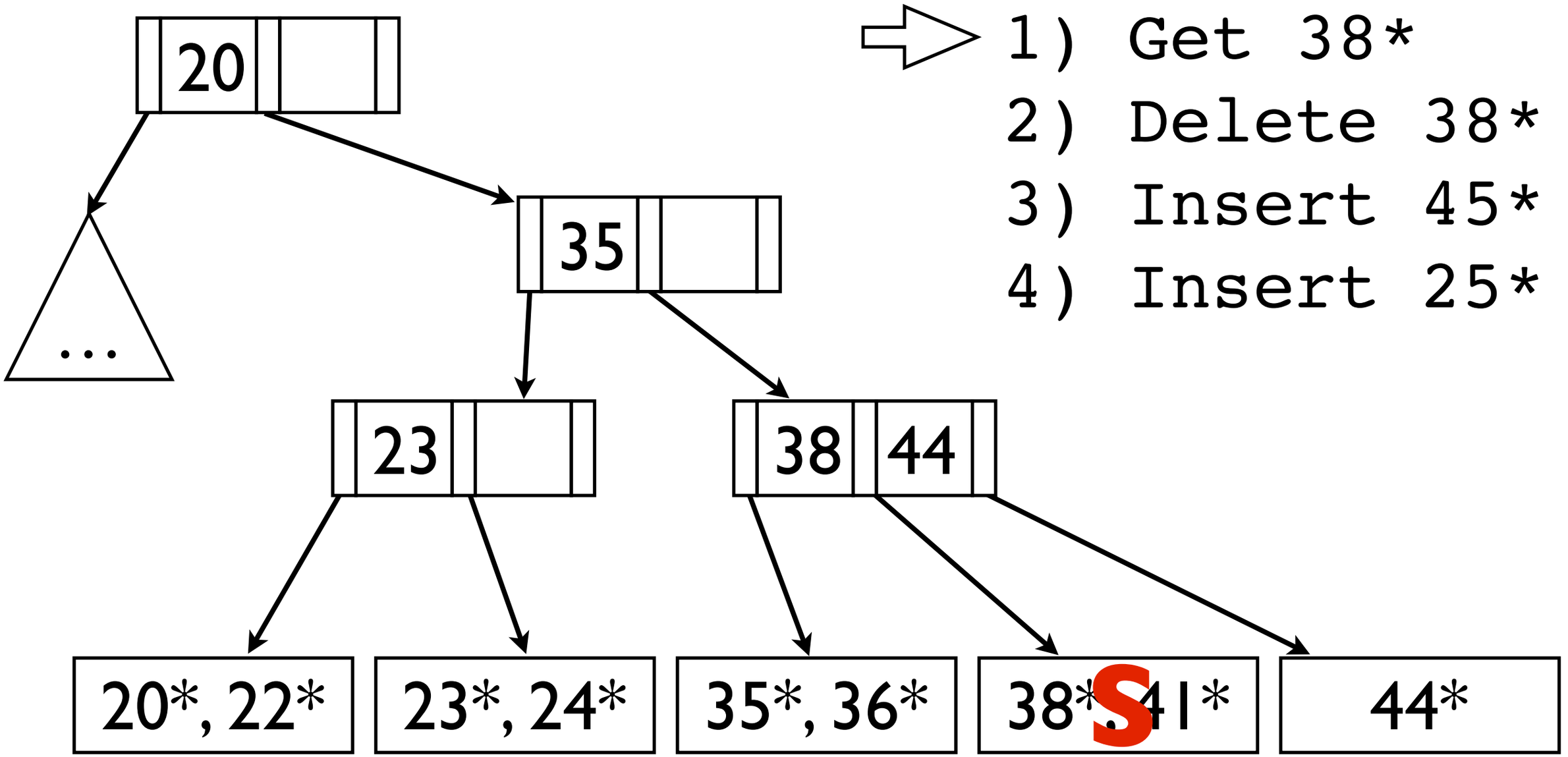
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



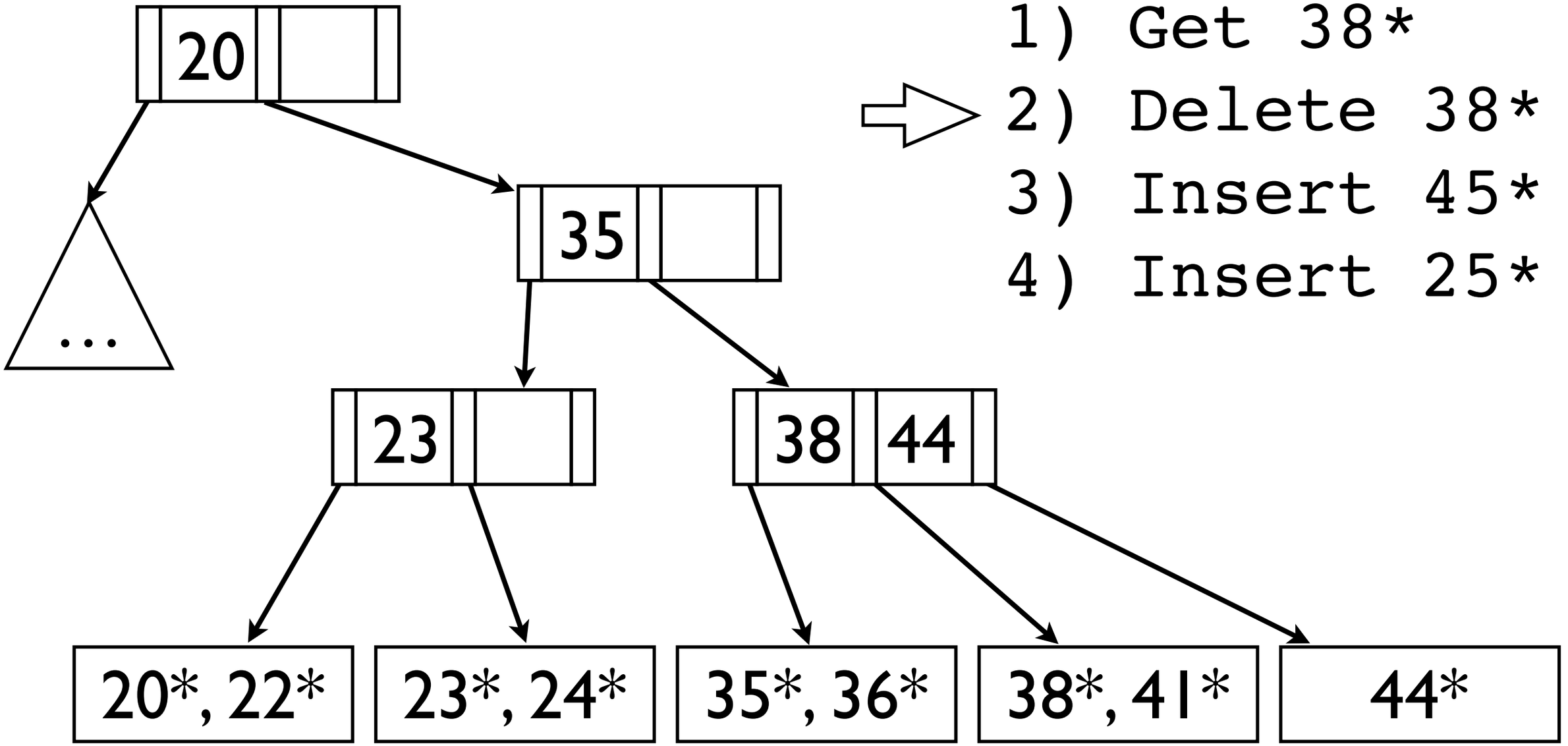
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



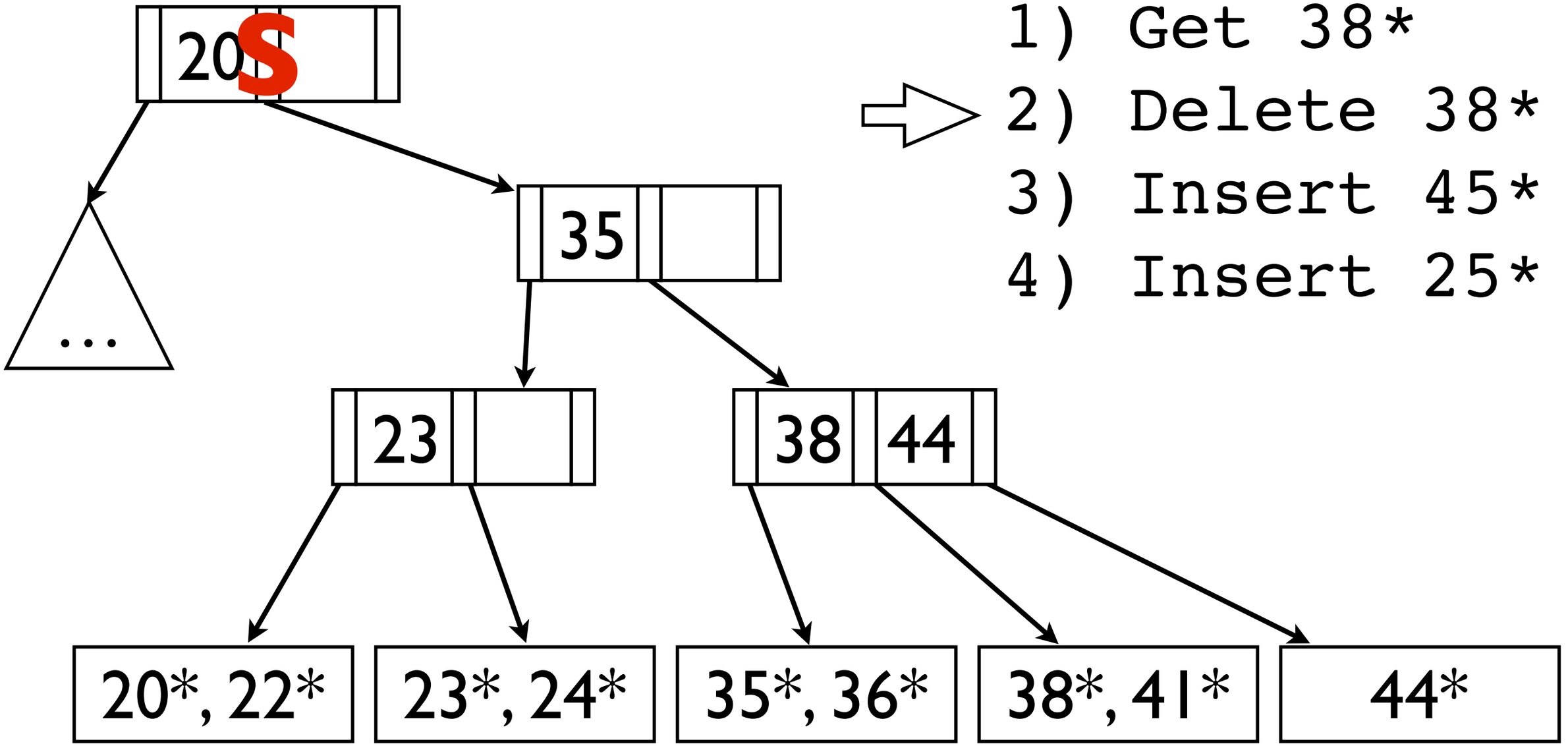
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



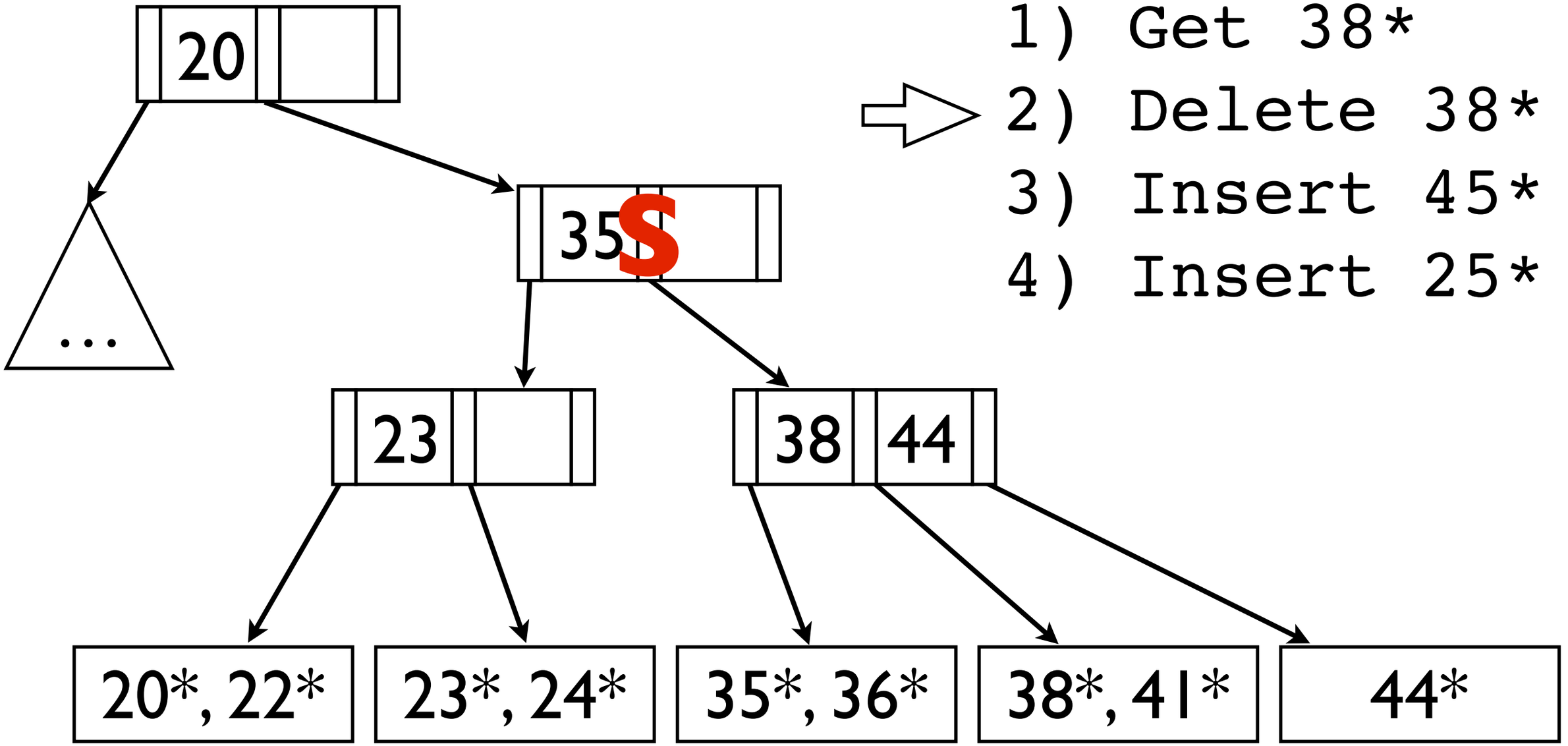
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



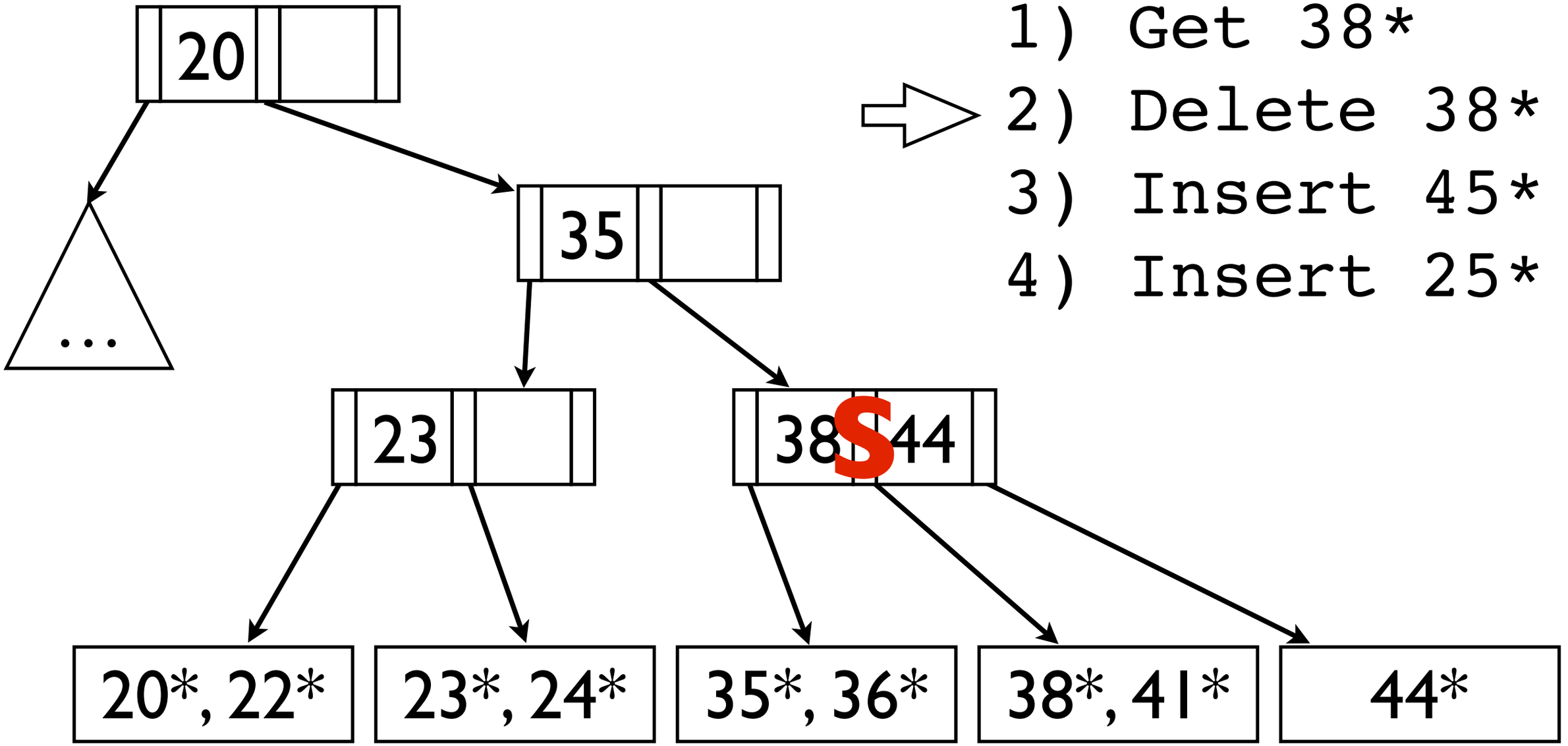
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



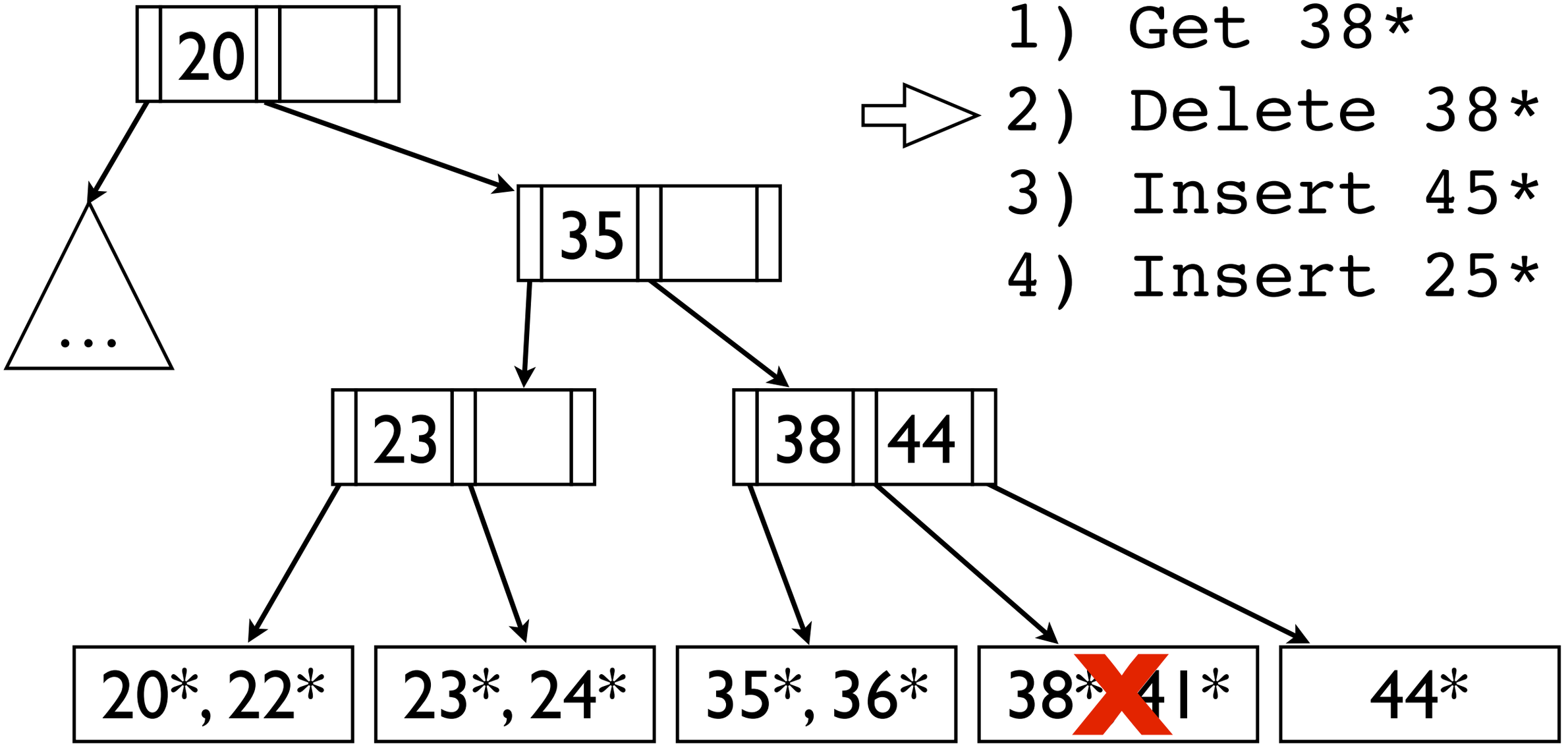
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



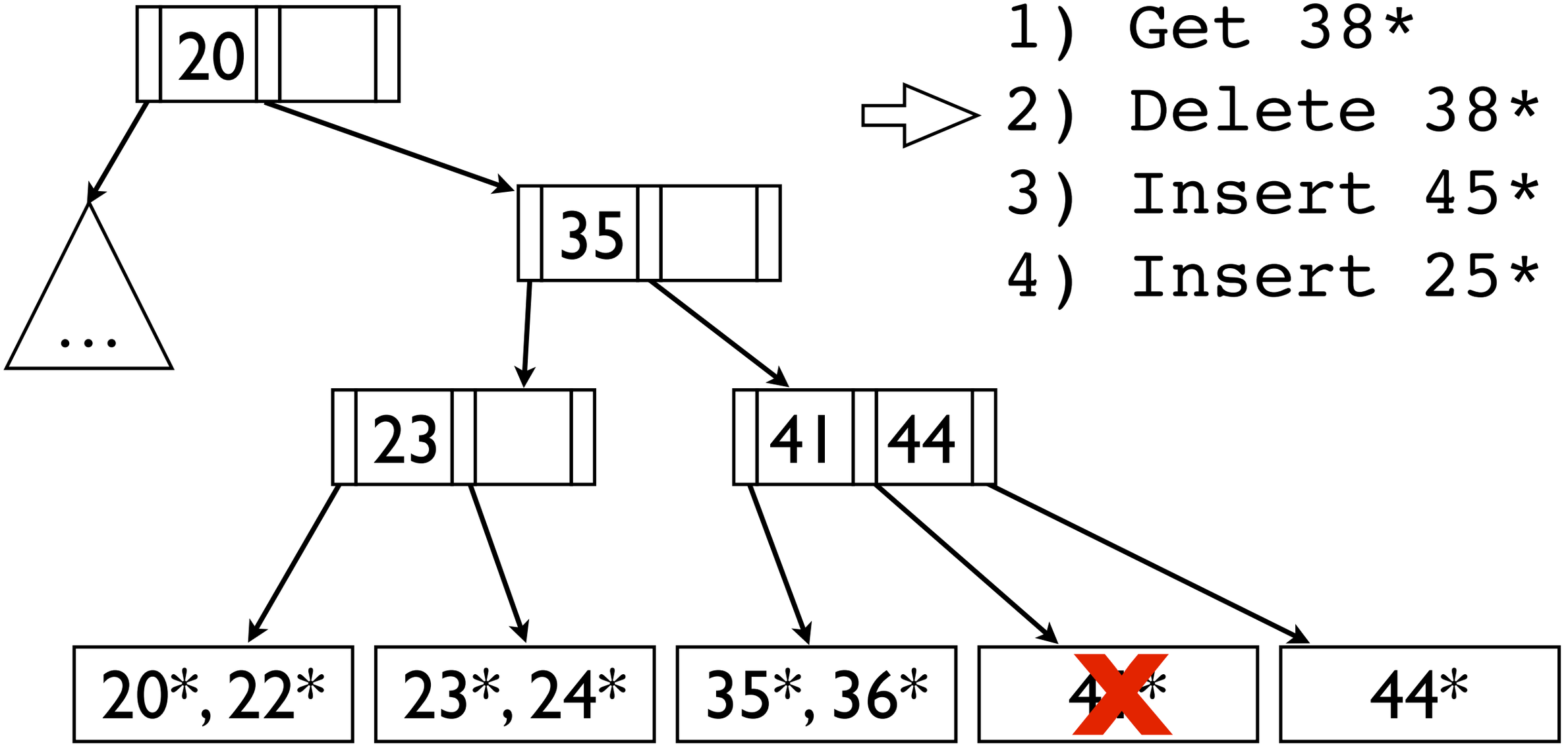
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



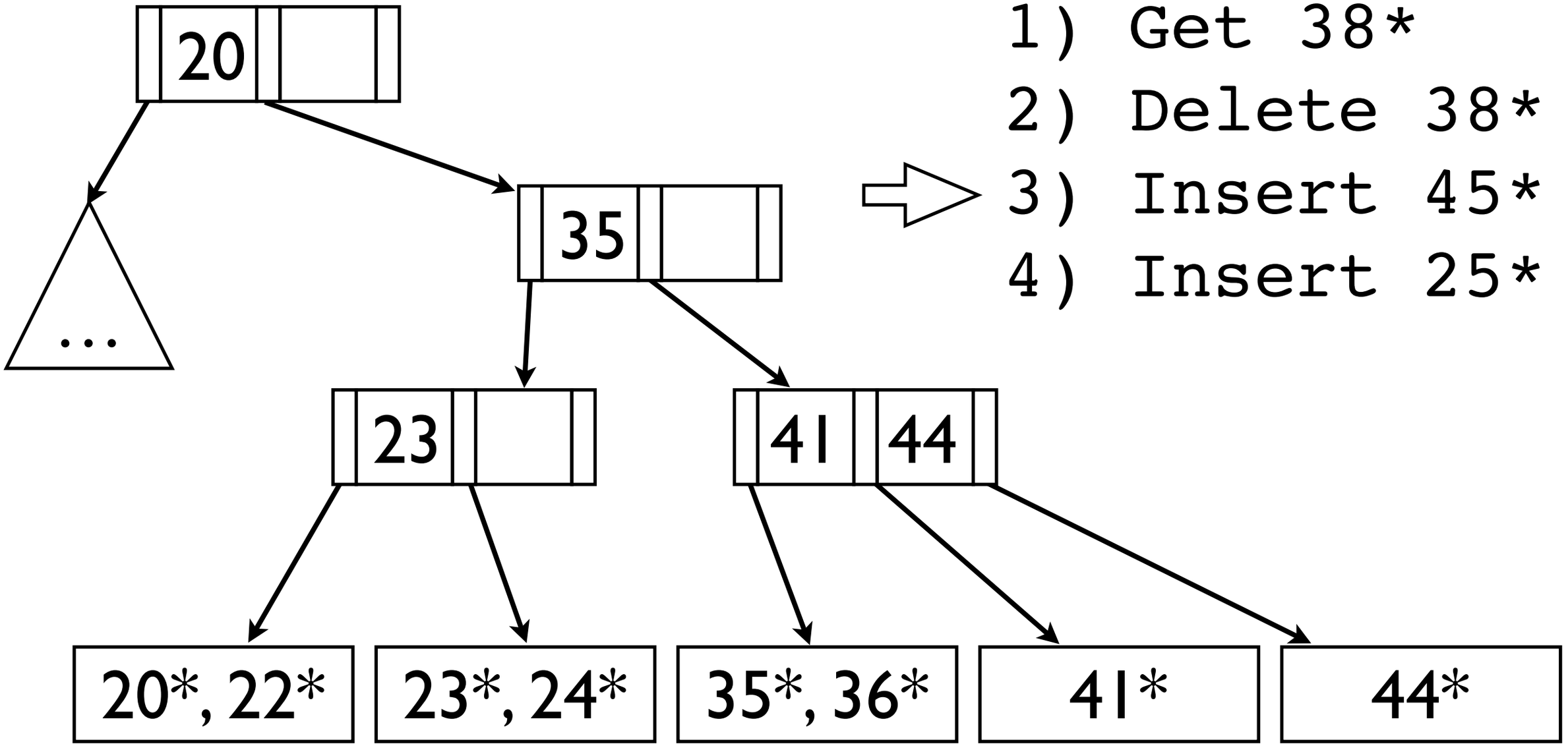
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



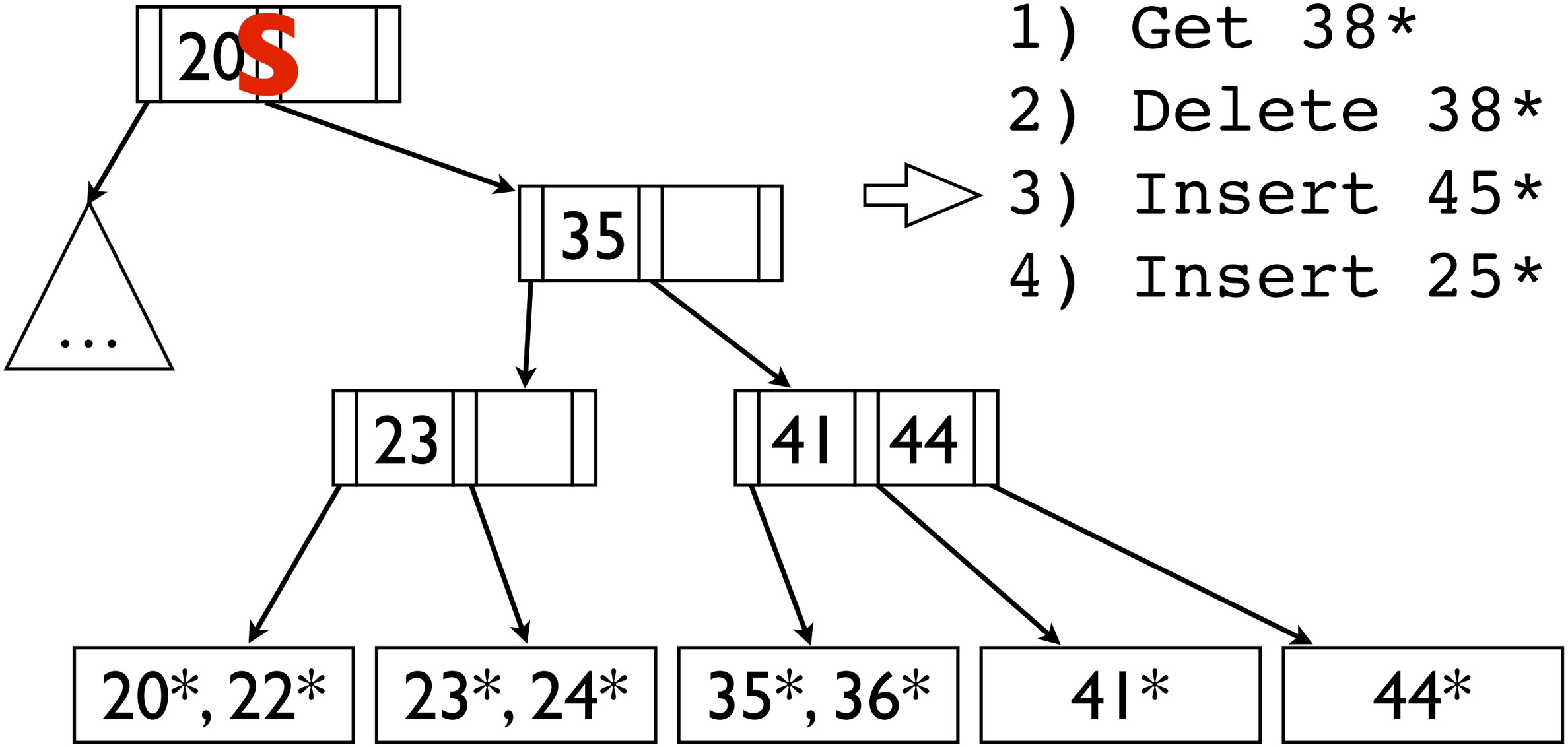
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



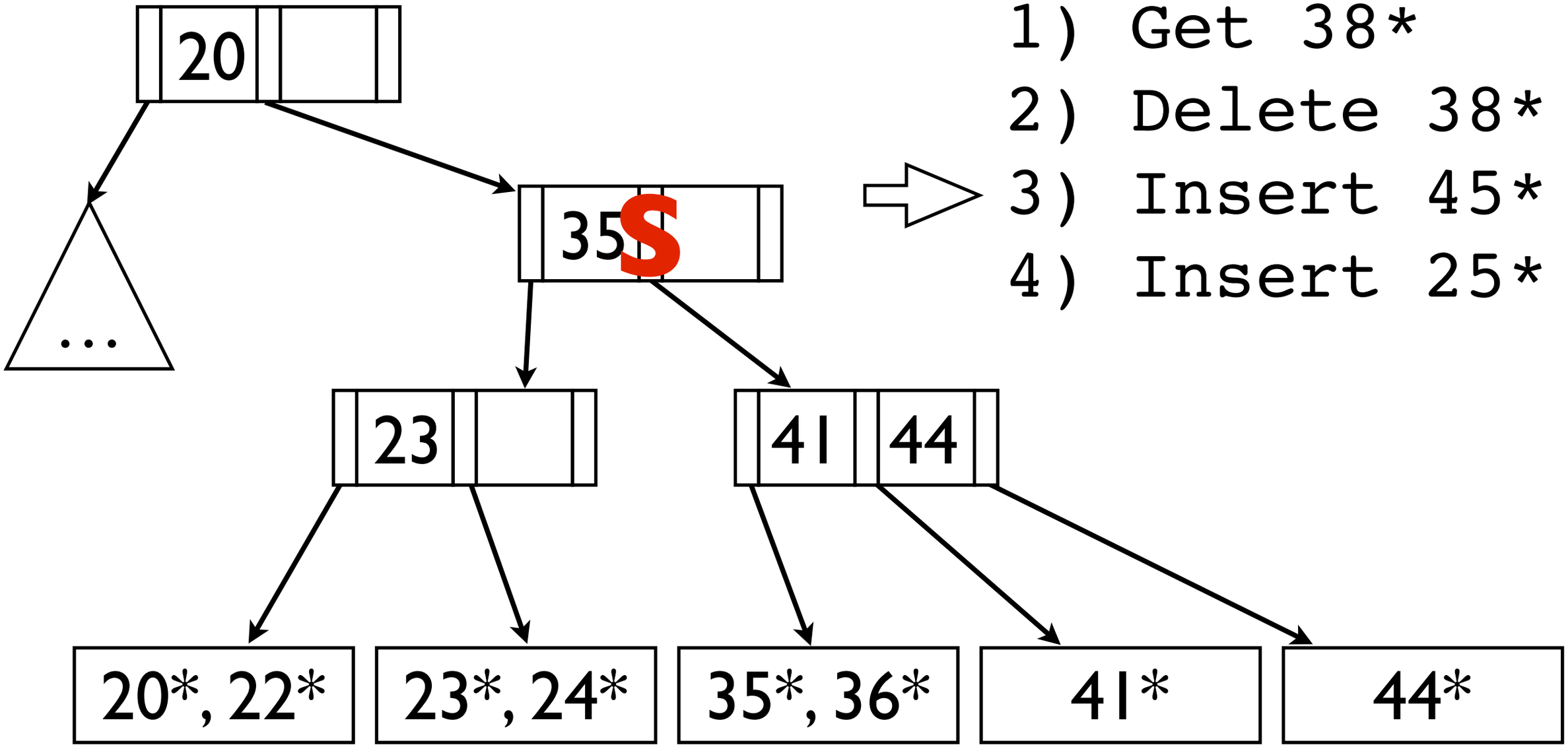
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



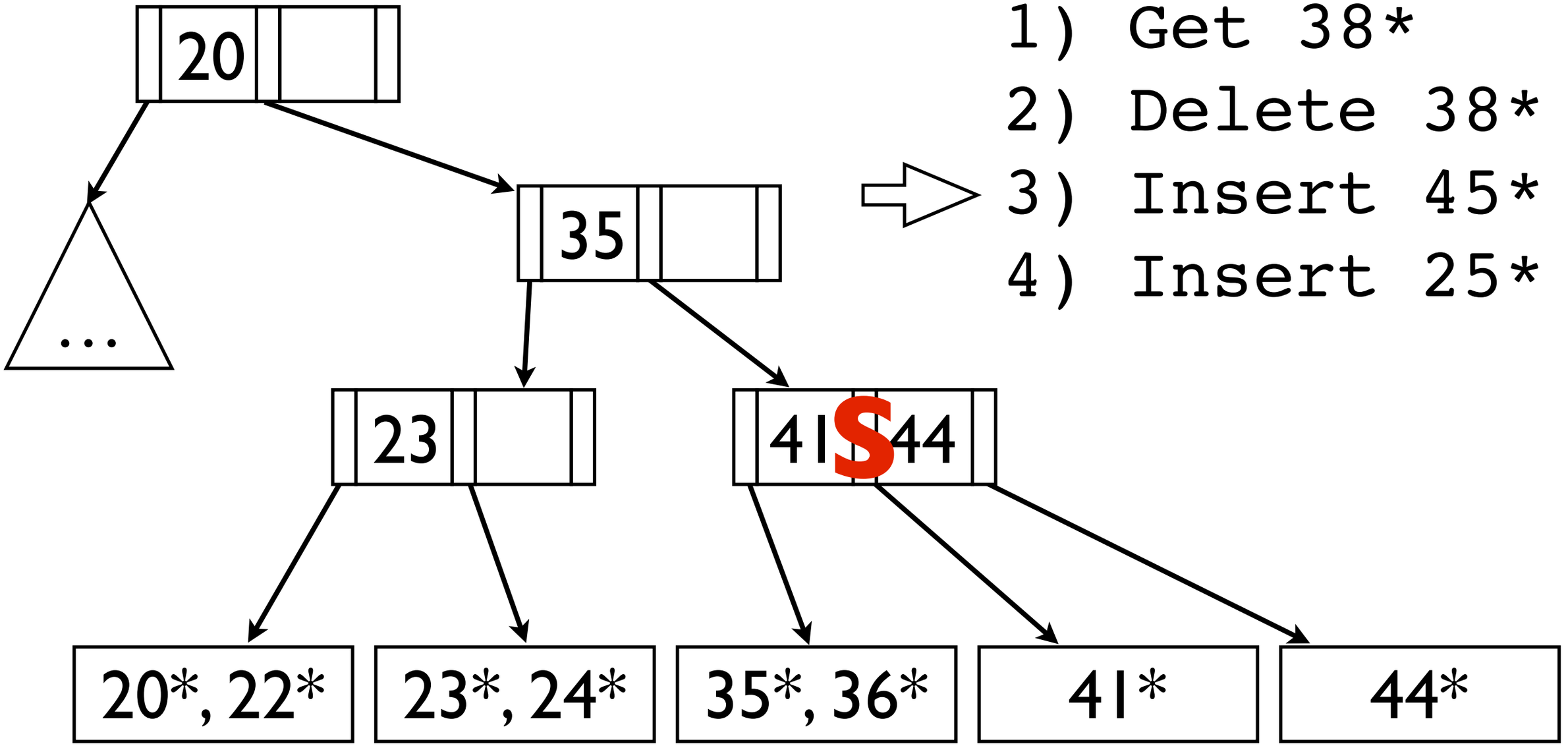
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



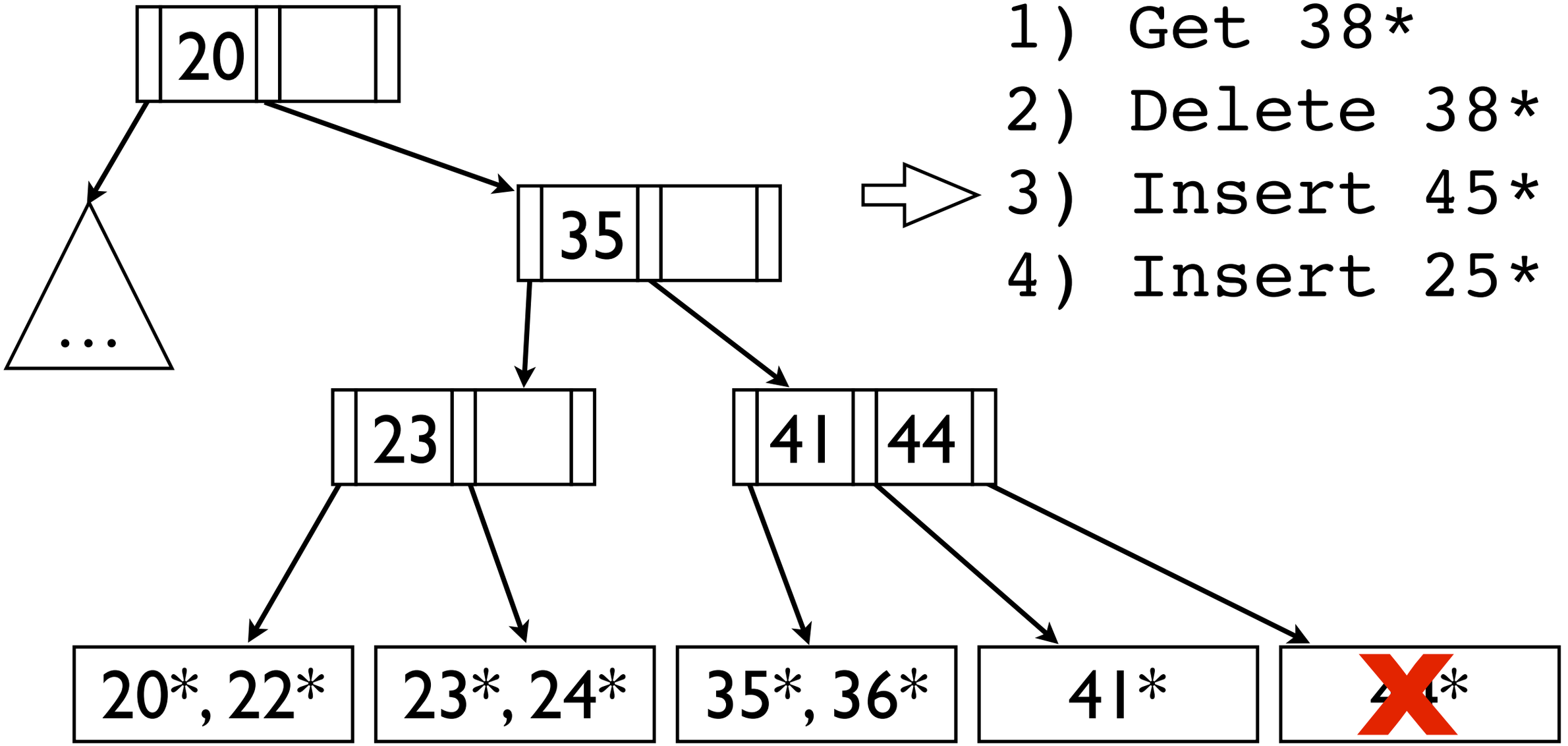
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



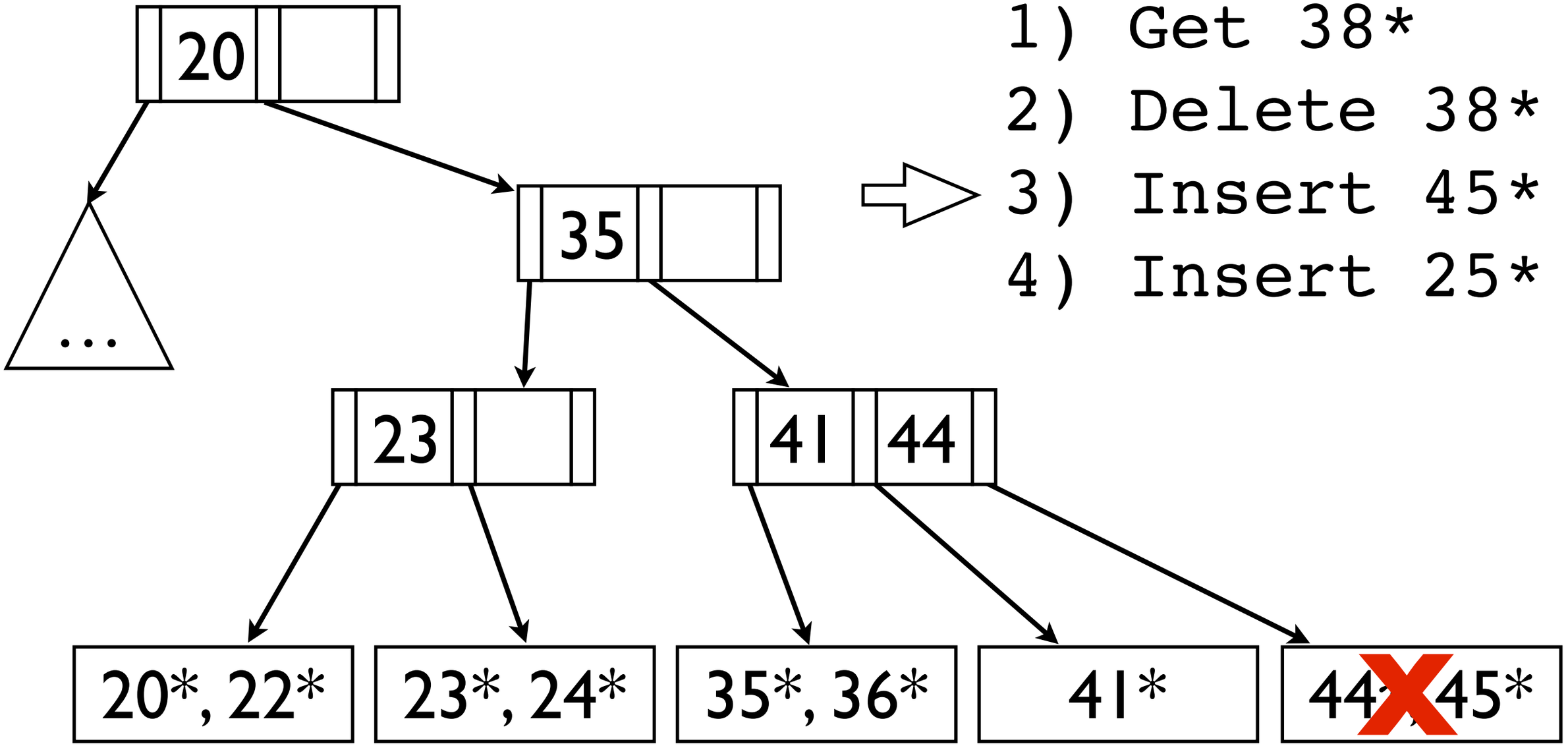
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



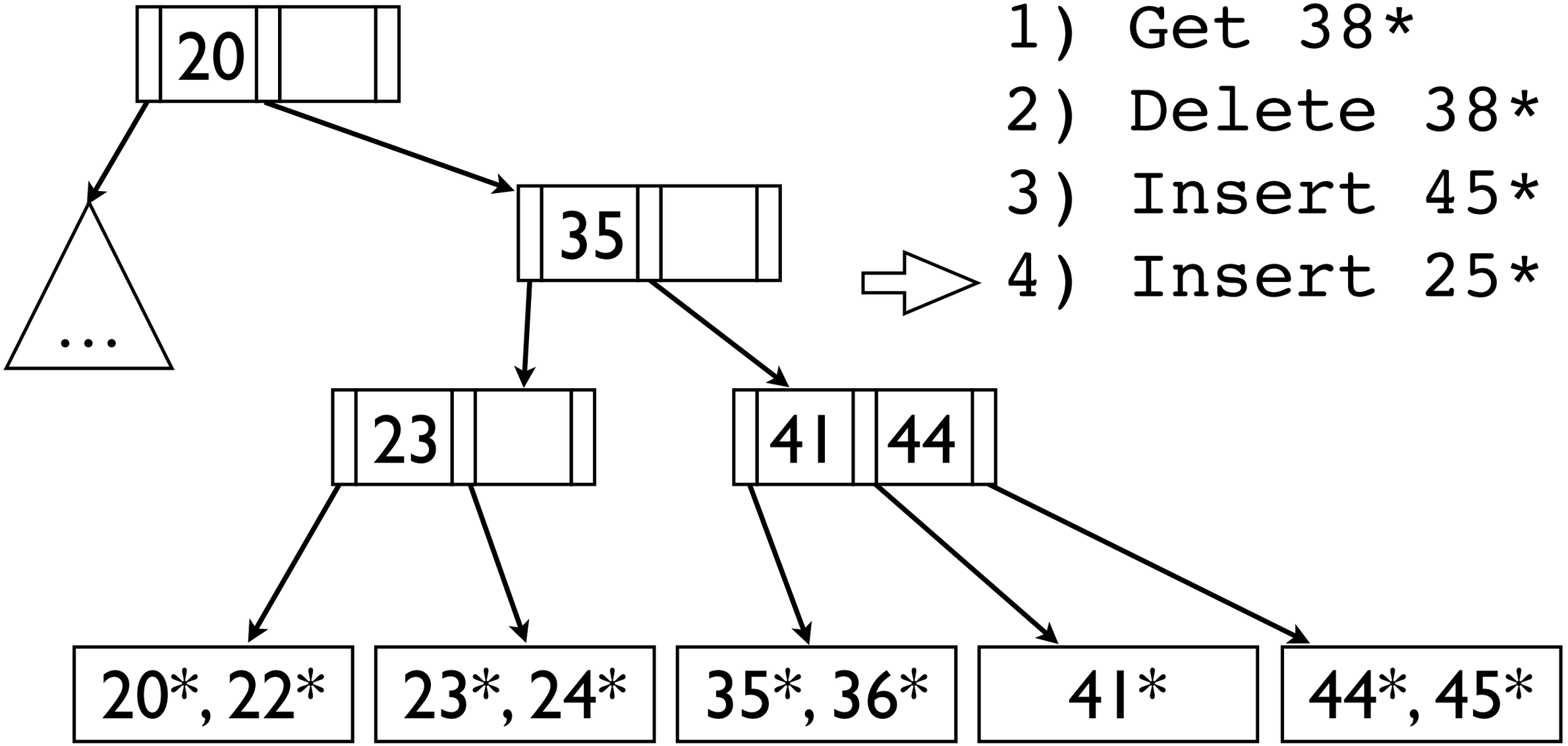
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



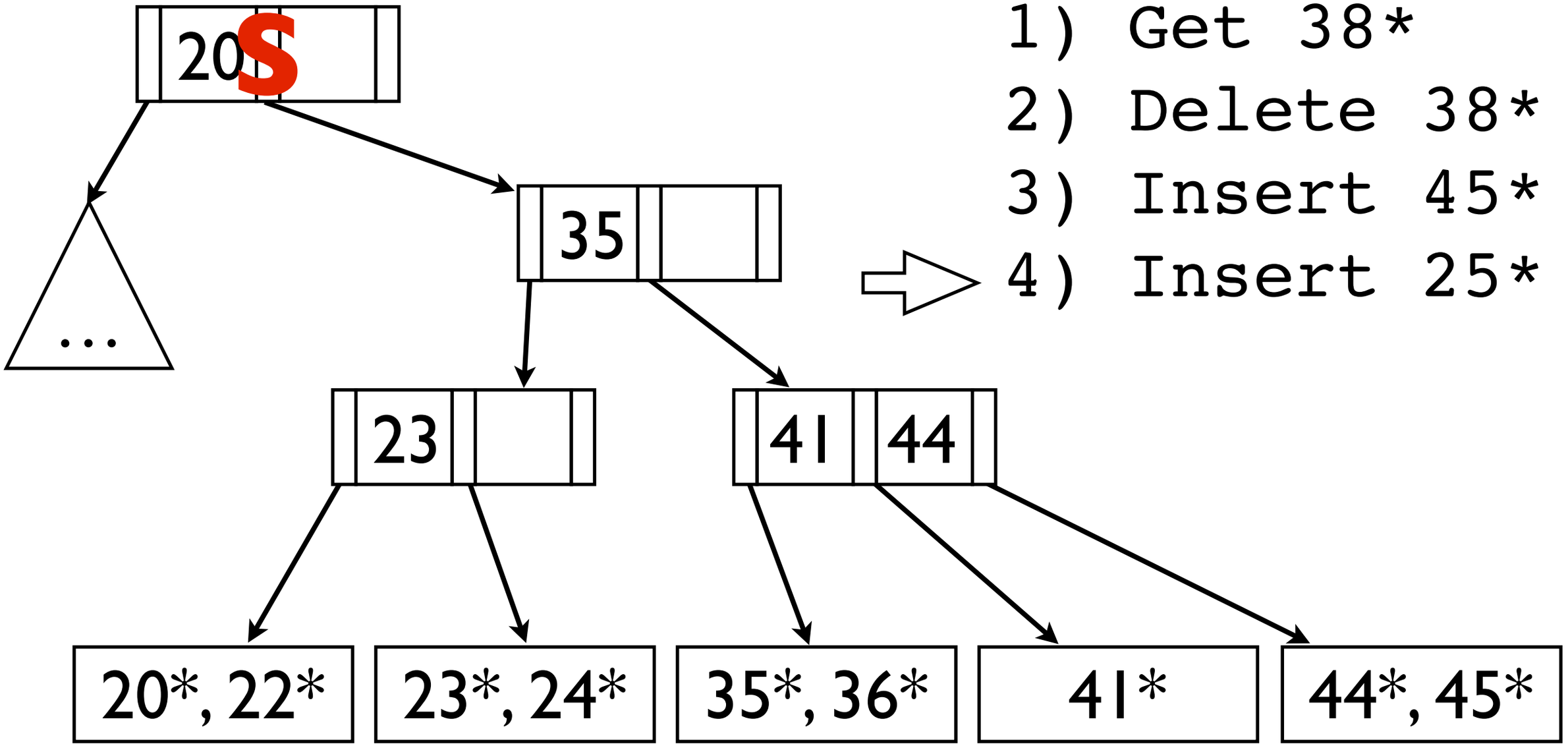
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



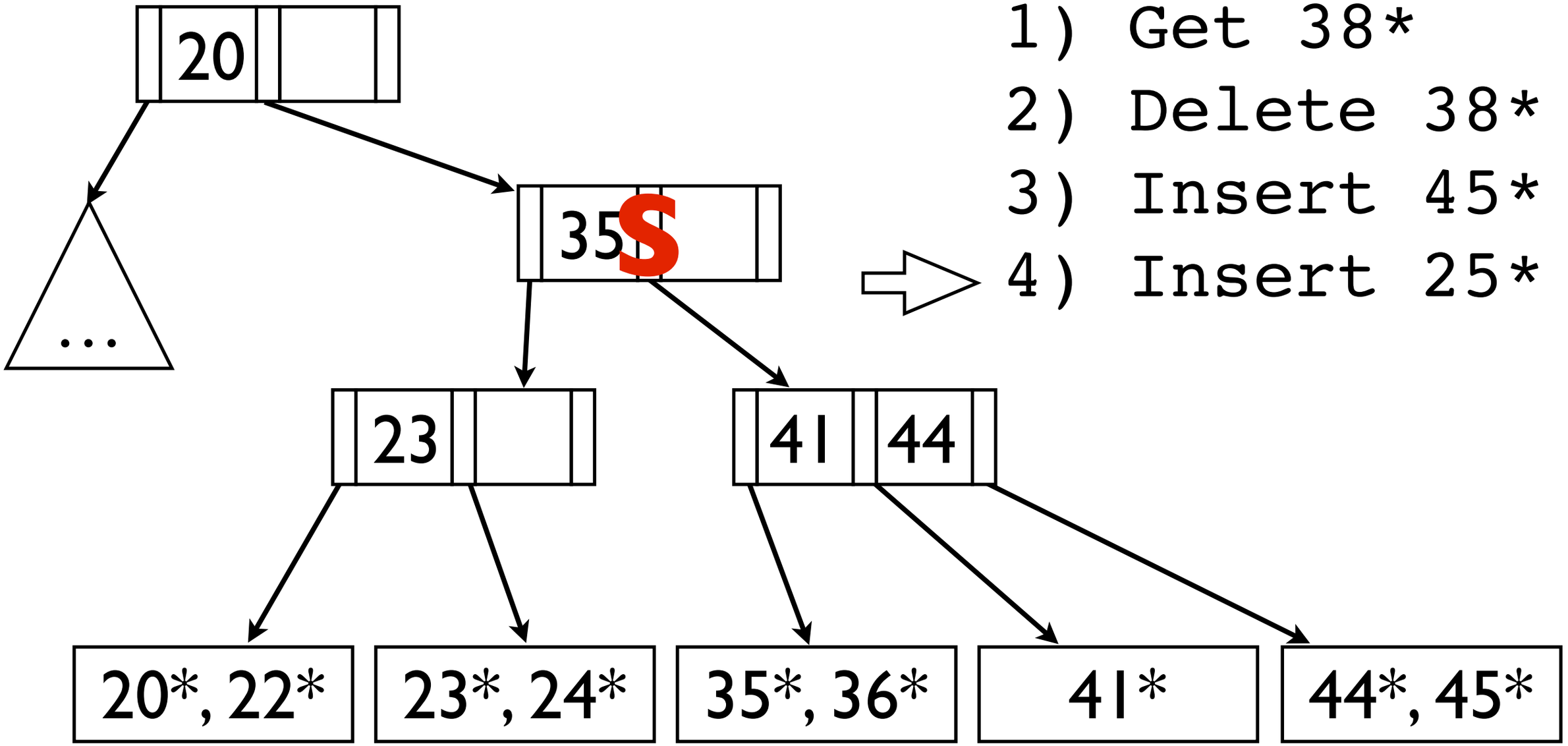
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



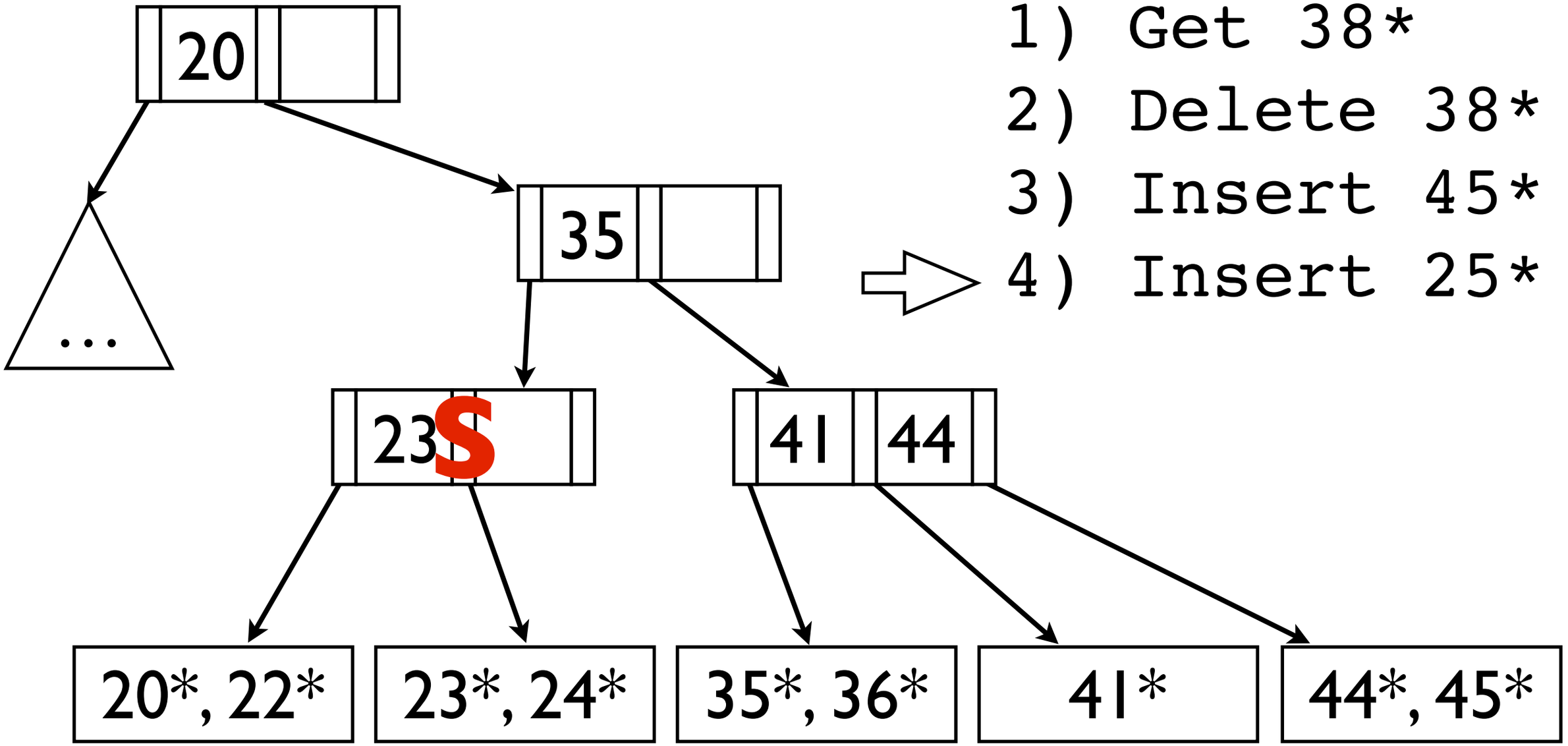
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



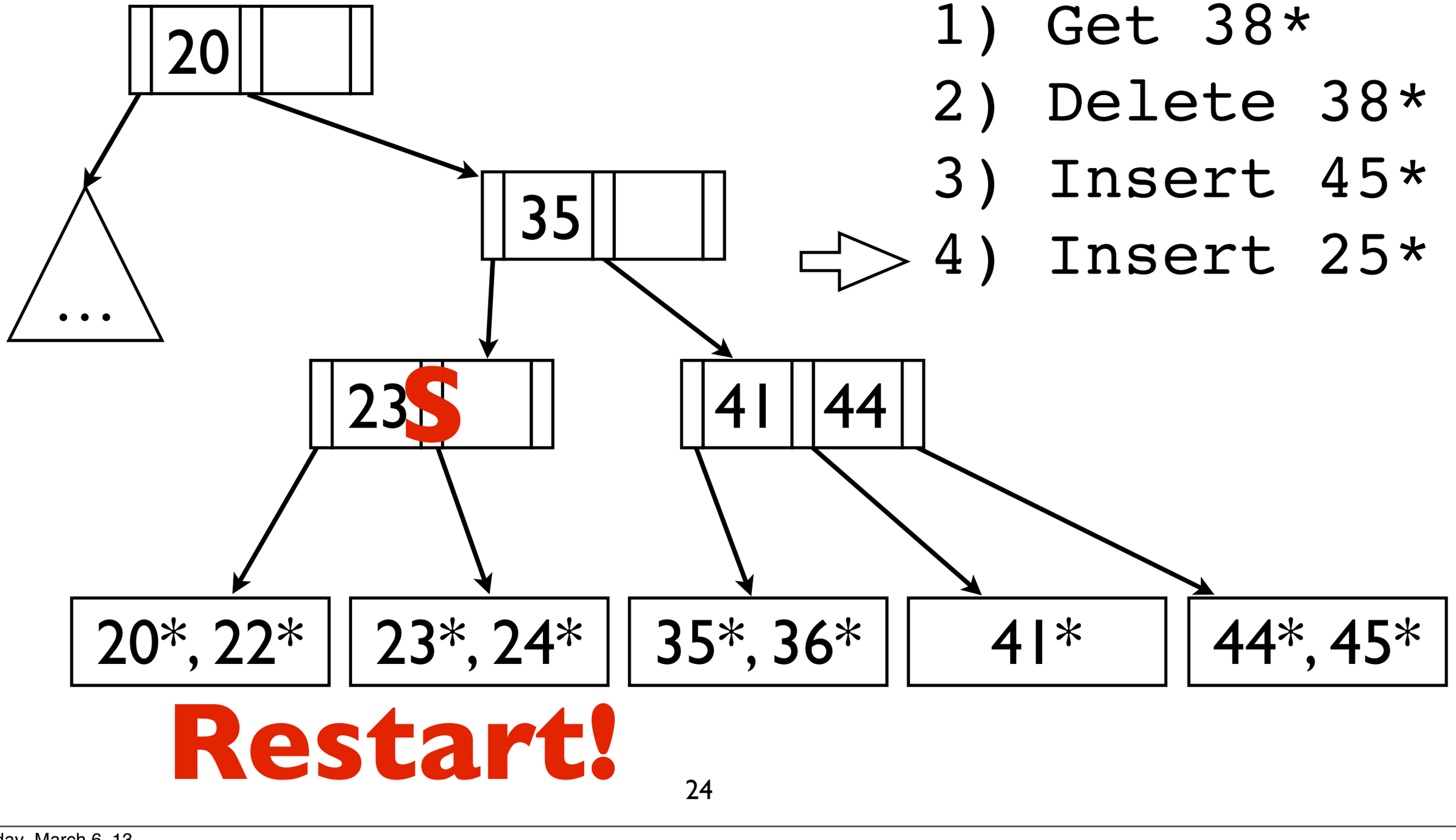
1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Examples



1) Get 38*
2) Delete 38*
3) Insert 45*
4) Insert 25*

Even Better Algorithm

- **Scan:** As before
- **Update**: Acquire IX locks on tree nodes
  - Acquire X lock on the leaf.
  - Convert IX to X locks as needed. Release rest.
    - Do conversion top-down to limit deadlock
- Contrast IX locks here with their use in multiple-granularity locking.

Hybrid Algorithm

- The likelihood that we need an X lock decreases as we move up the tree!