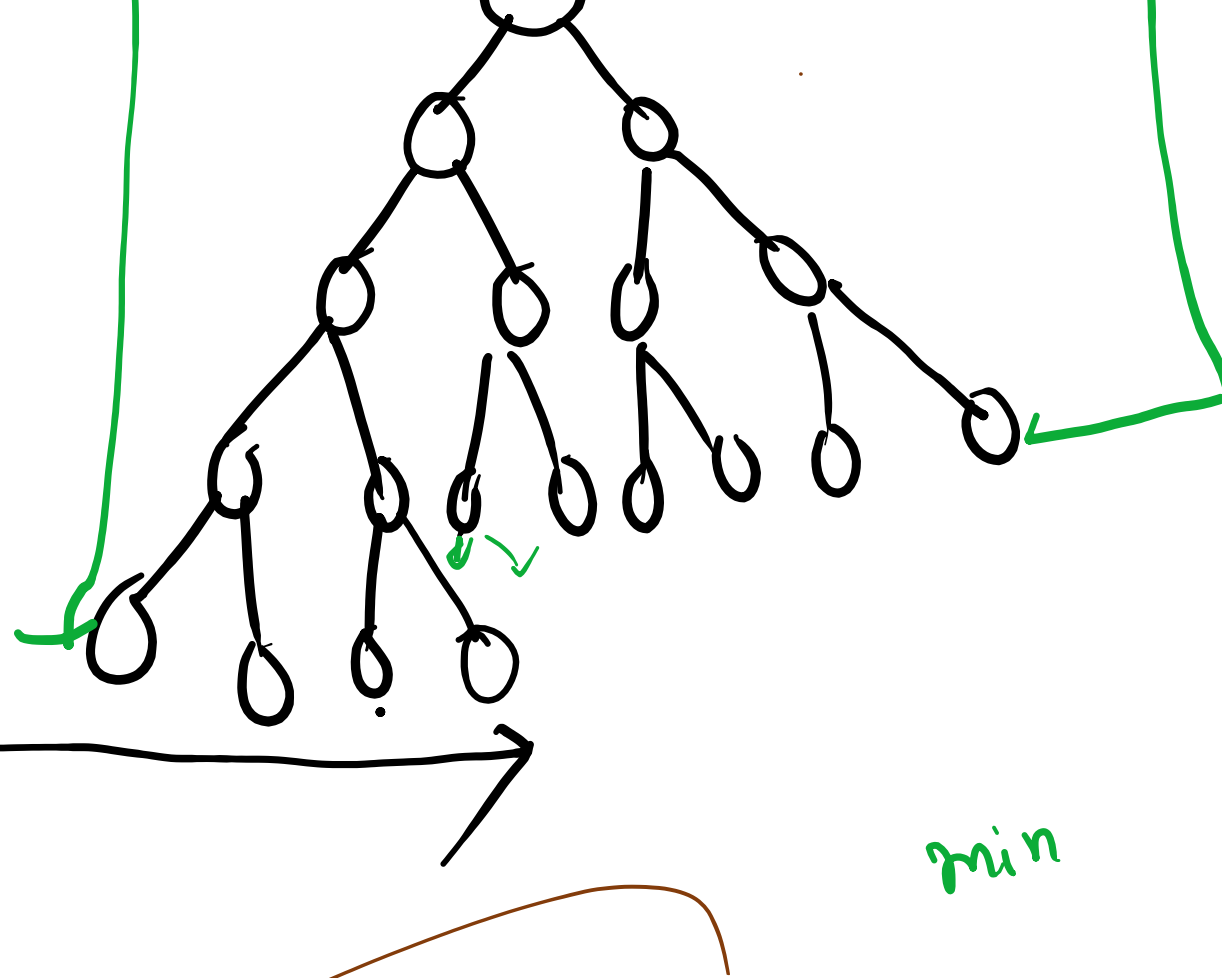


heap priority queue

2, 3, 4, 5

2, 3, 4, 5

① gt ip CBT
50%
same priority



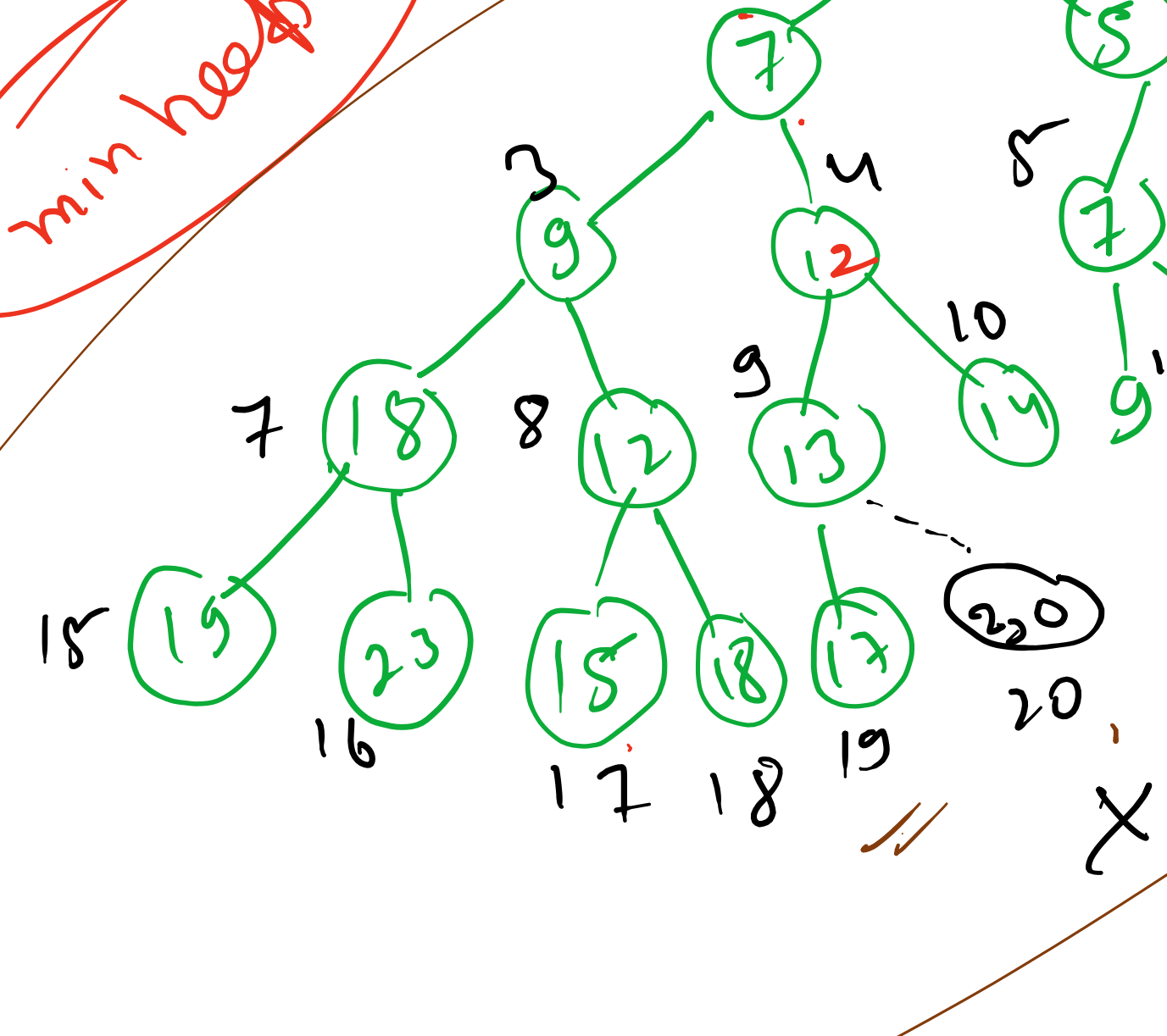
min heap
max heap

min
parent val < both child

$$\frac{ci-2}{2}$$

$$\frac{ci-1}{2}$$

min heap

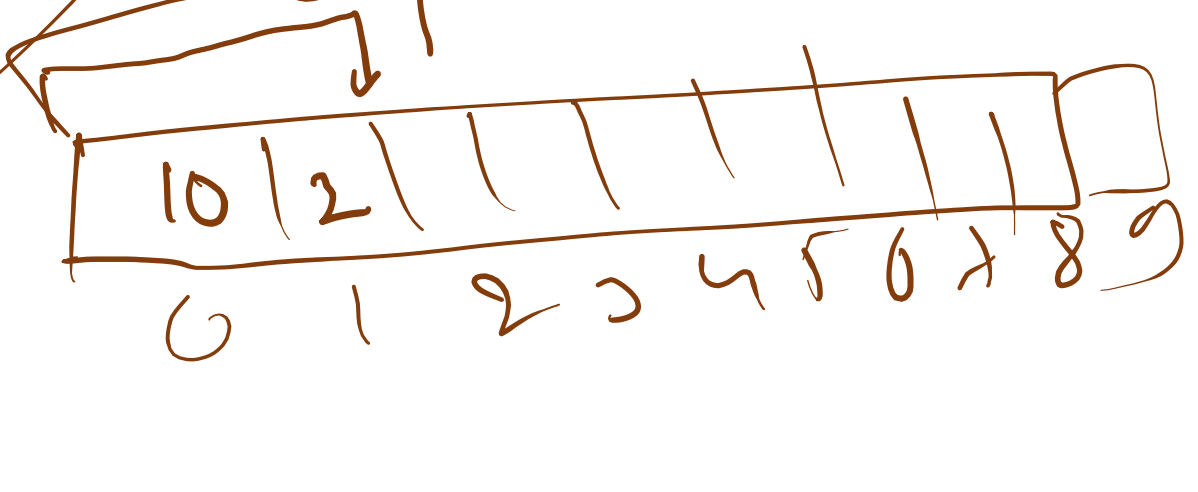


$$pi = 2c$$

$$2ci - 2 \times pi + 1$$

$$RCi = 2 \times pi + 2$$

$$pi = \frac{ci-1}{2}$$



2, 3, 4, 5

Heap

sorted data

unsorted array

Heap

Add

$O(n)$

$O(1)$

$\log(n) = ?$

min

$O(n)$

$O(n)$

$\log(n) = ?$

del min

$O(1)$

$O(n)$

$O(1)$

min heap

$2n = n$
 $n = 1000$

$\{10, 20, 30, 5, 22\}$ CB.T

$$\frac{2-1}{2} = 1$$

$$\frac{1-1}{2} = 0$$

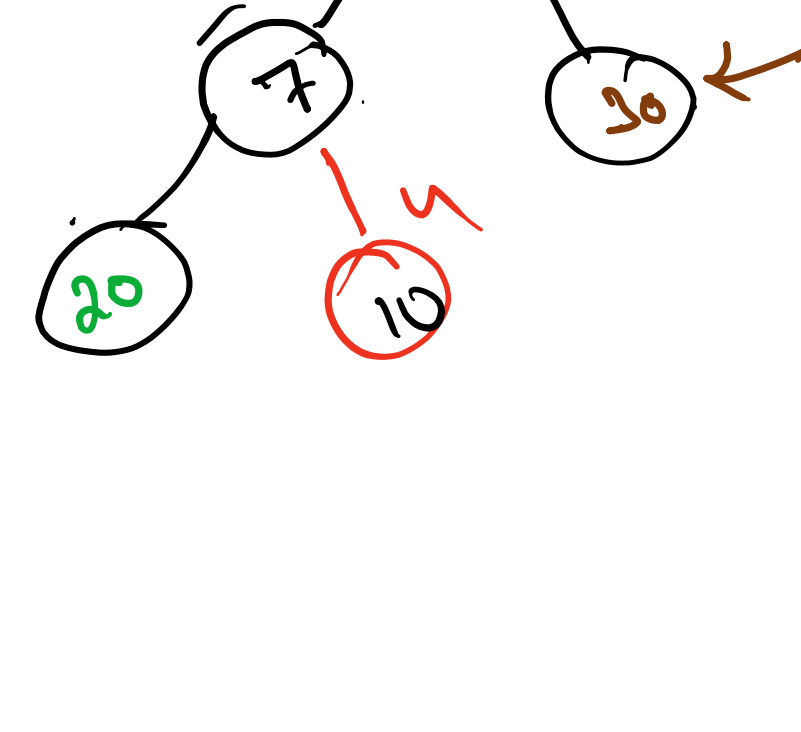
Set - Union

$$\frac{2-1}{2} = 1$$

public void add(int item) {
ll.add(item);
upheapify(ll.size() - 1);
}

private void upheapify(int ci) {
// TODO Auto-generated method stub
}

int pi = (ci-1)/2
if (ll.get(ci) < ll.get(pi))
swap (ci, pi)
upheapify (pi)



$$\frac{0+1-1}{2} = 0$$

$$\frac{2+2-1}{2} = 1$$

$$\frac{2+2-1}{2} = 1$$

$$\frac{2+2-1}{2} = 1$$

int mini = pi
if (ll.get(Lci) < ll.get(mini))
mini = Lci
if (ll.get(Rci) < ll.get(mini))
mini = Rci
if (mini != pi)
swap (mini, pi)
upheapify (mini)

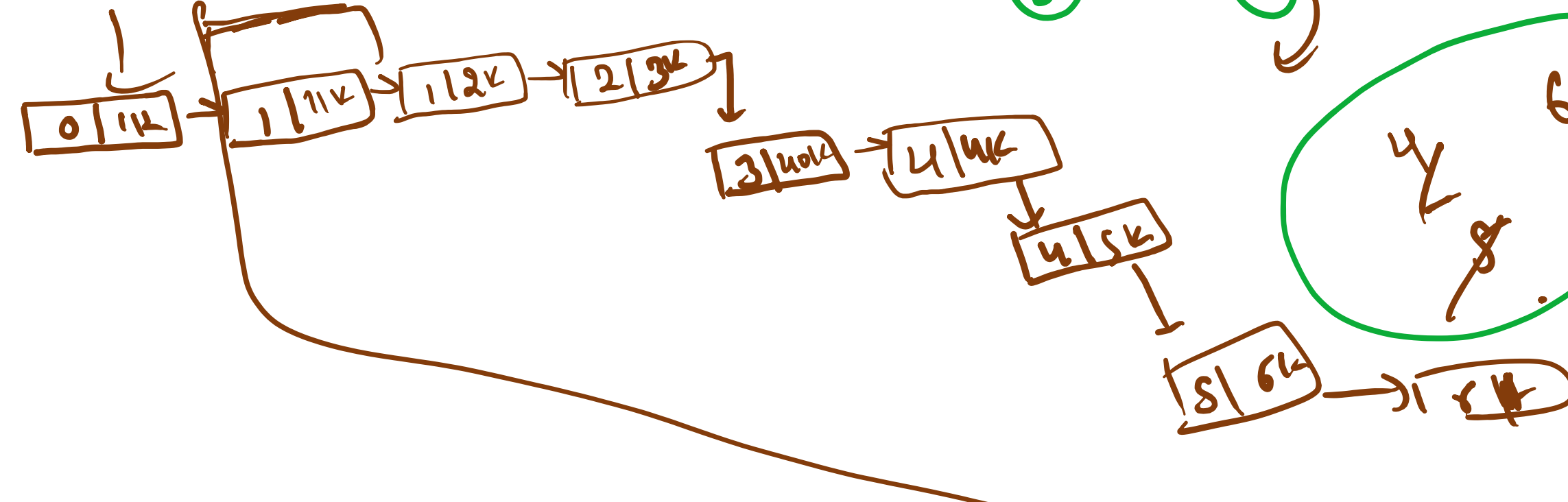
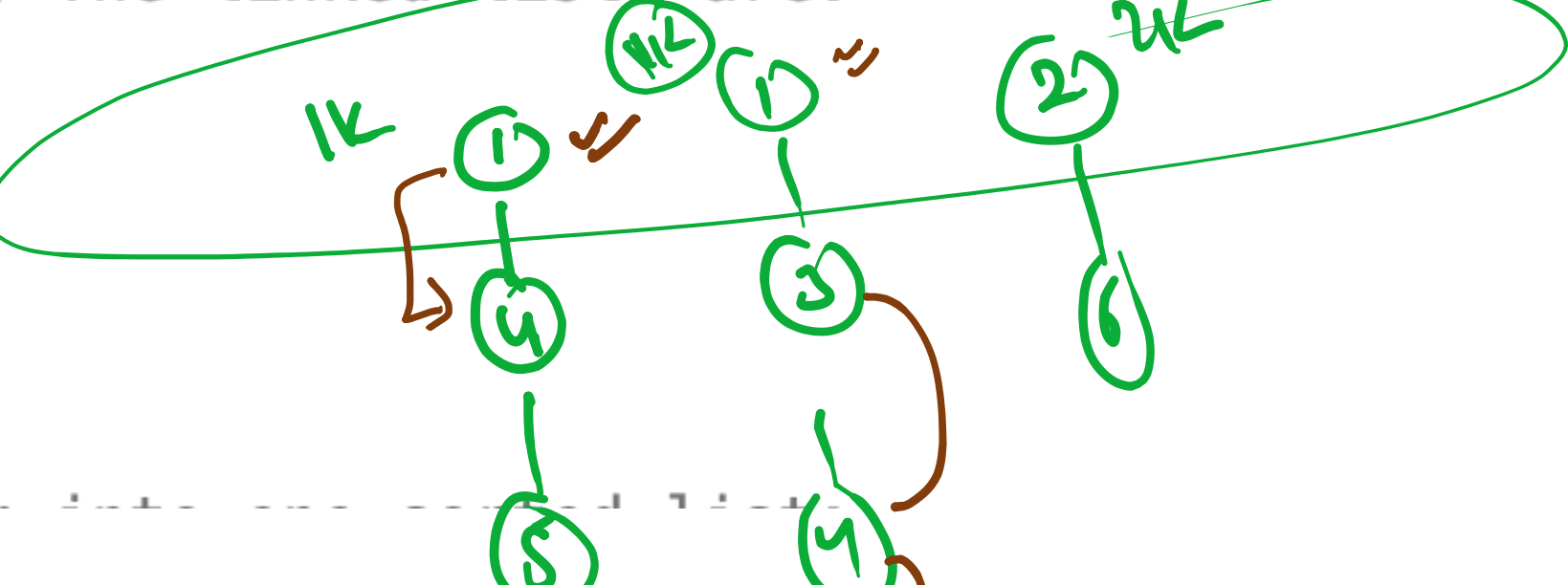
Input: nums = [3, 2, 3, 1, 2, 4, 5, 5, 6, 1, 2], k = 4
Output: 4

arr[ci] > arr[pi] swap (ci, pi)

5 5 6 4

temp array

Input: lists = [[1,4,5], [1,3,4], [2,6]]
Output: [1,1,2,3,4,4,5,6]
Explanation: The linked-lists are:
[
1->4->5,
1->3->4,
2->6
]



$$K \times W$$

$$K \times W$$

$$\frac{10^6}{10} = 10^5$$

$$K \times W \times \log(K \times W)$$

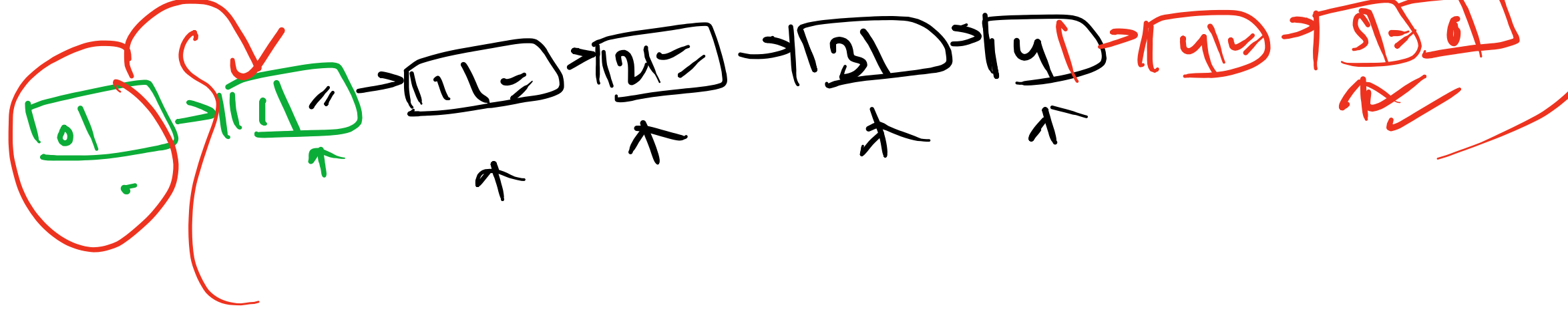
$$K \times W \times \log(K \times W) + K \times W \times \log(K)$$

$$5 \times 10^6$$

$$5 \times 10^6 \times \log 10^6$$

Input: lists = [[1,4,5], [1,3,4], [2,6]]
Output: [1,1,2,3,4,4,5,6]
Explanation: The linked-lists are:
[
1->4->5,
1->3->4,
2->6
]

ListNode dummy = new ListNode();
ListNode temp = dummy;
while (l1 != null || l2 != null || l3 != null)
{
ListNode n1 = l1 != null ? l1.val : 0;
ListNode n2 = l2 != null ? l2.val : 0;
ListNode n3 = l3 != null ? l3.val : 0;
int min = Math.min(n1, Math.min(n2, n3));
temp.next = new ListNode(min);
temp = temp.next;
if (n1 == min) l1 = l1.next;
if (n2 == min) l2 = l2.next;
if (n3 == min) l3 = l3.next;
}



1-2

10, 30

5, 10

15, 20

2, 6

10, 30

12, 18

19, 25

1, 2

3, 5

2-5
5-10
10-20

15-20

19-25

2-6

1-2

2-6

1-2

2-6

1-2

2-6

1-2

2-6

1-2

2-6

1-2

2-6

1-2

2-6

1-2

2-6

1-2

2-6

1-2

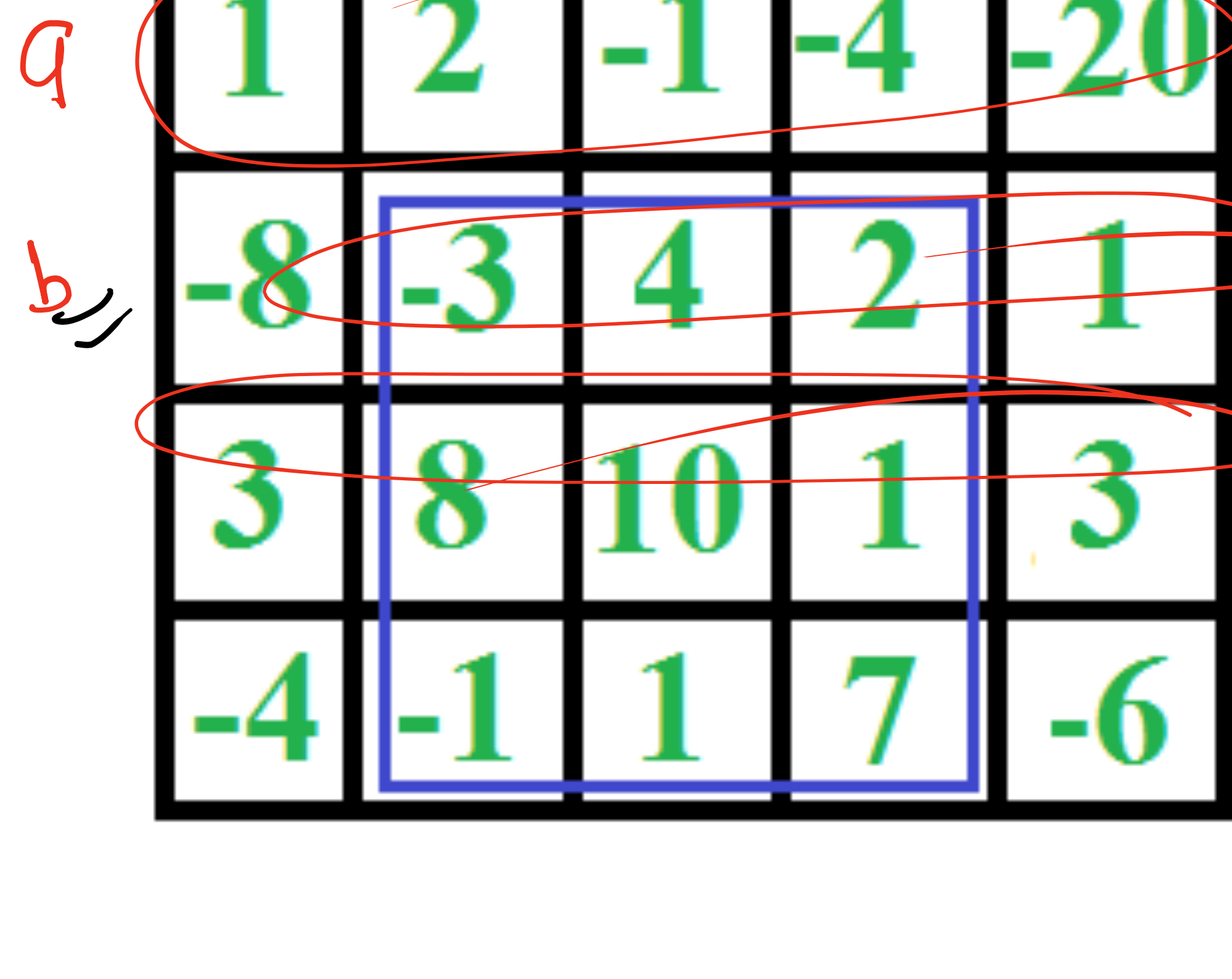
2-6

1-2

2-6

1-2

2-6



-8 -3 4 2 1
-4 -1 1 2 6
1 2 -1 -4 10
3 8 10 1 3

(2, 3, 1, 4)

(2, 3, 1, 4)

(2, 3, 1, 4)

(2, 3, 1, 4)

(2, 3, 1, 4)

20

(2, 3, 1, 4)

(2, 3, 1, 4)

(2, 3, 1, 4)

(2, 3, 1, 4)

(2, 3, 1, 4)

1-2 -> 3

3-3 -> 6

6-4 = 10

10

2, 3

3-4 = 7

2+1 = 3

10

10

2n=3

3n=6

4n=10

(2, 3, 1, 4)

2, 3, 1, 4