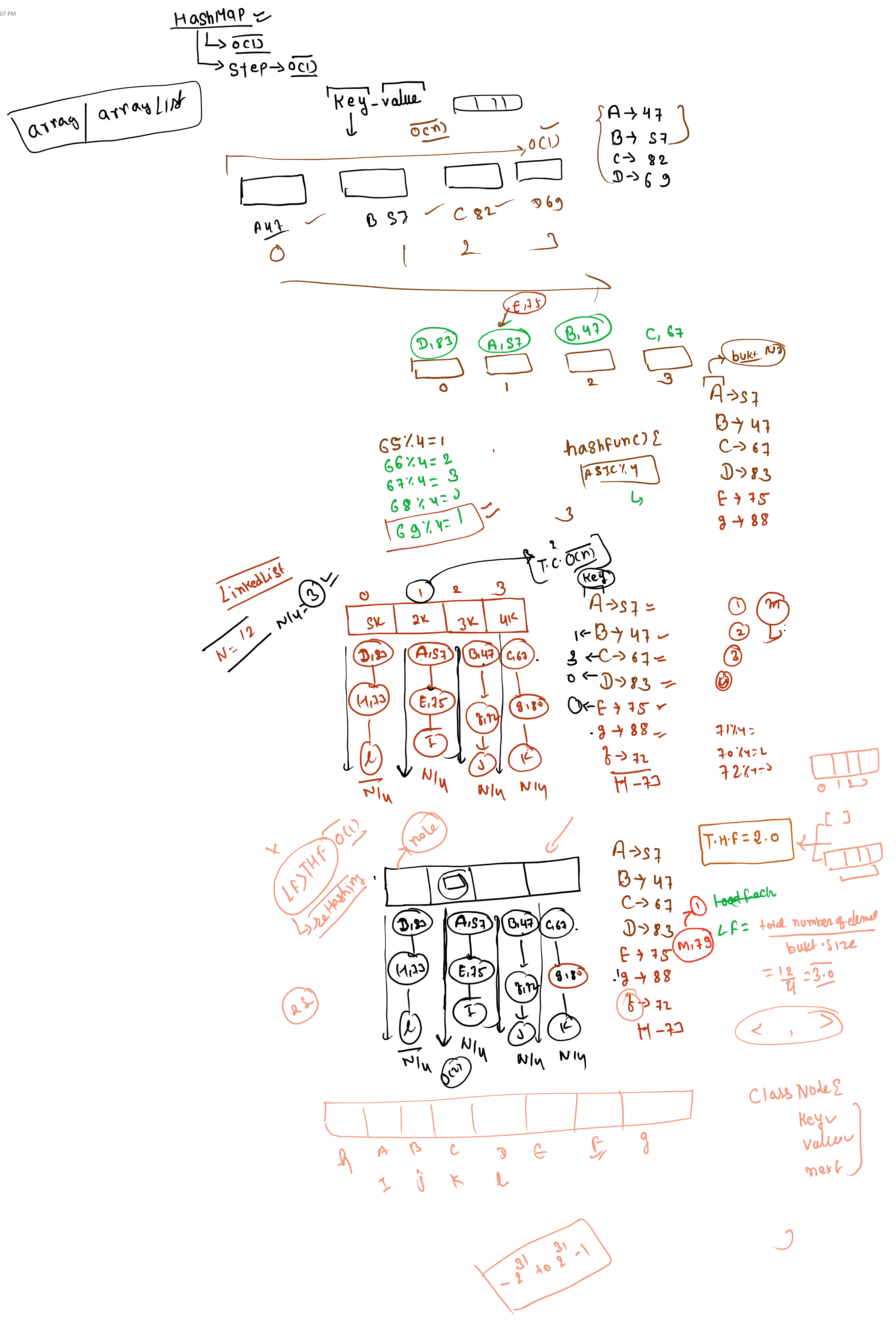
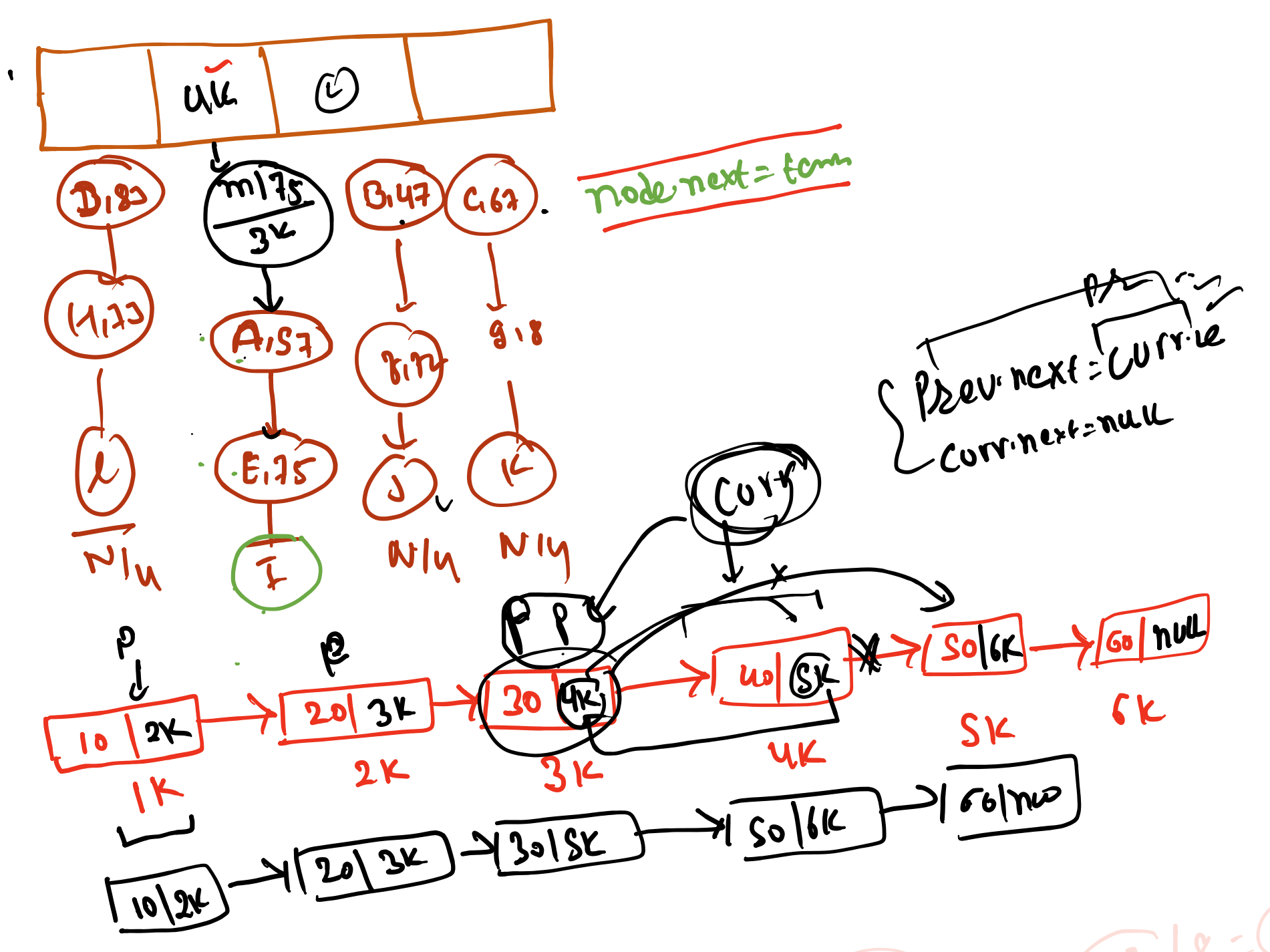
Lec-44

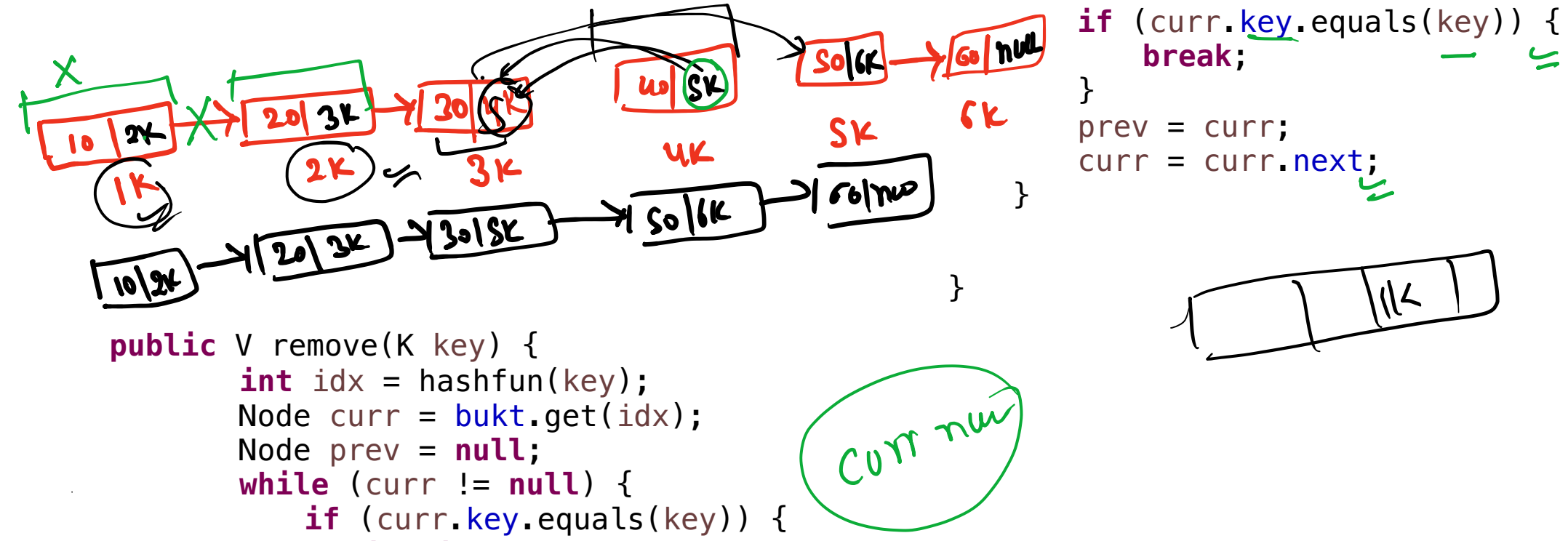
17/08/25 7:07 PM



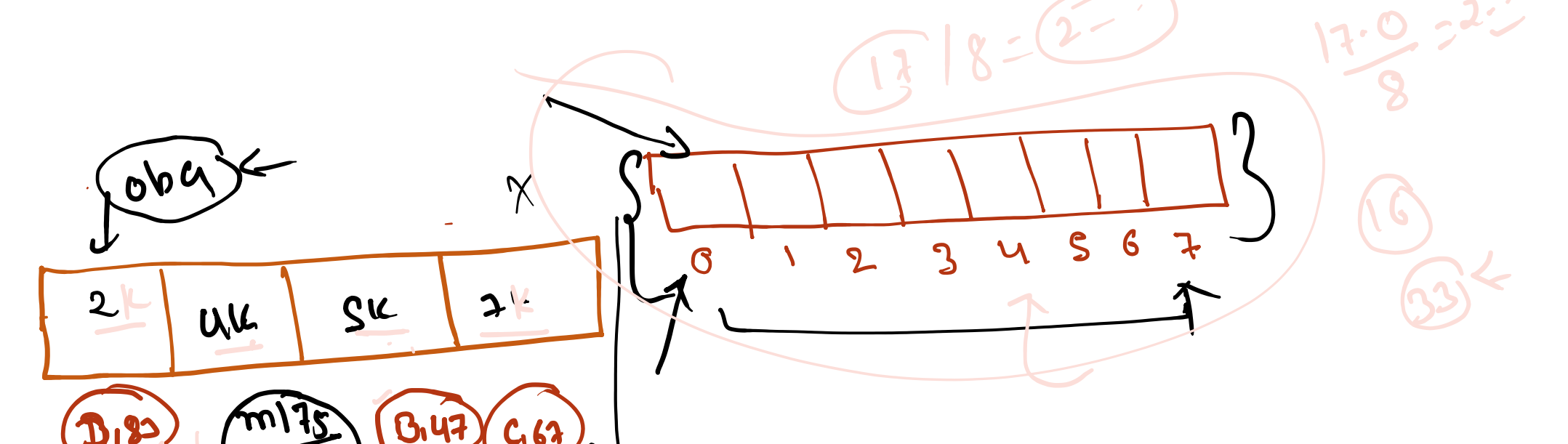
```java
public void put(K key, V value) {
        int idx = hashfun(key);
        Node temp = bukt.get(idx);
        while (temp != null) {
            if (temp.key.equals(key)) {
                temp.value = value;
                return;
            }
            temp = temp.next;
        }
        Node node = new Node();
        node.key = key;
        node.value = value;
        temp = bukt.get(idx);
        node.next = temp;
        bukt.set(idx, node);
        size++;

    }
```

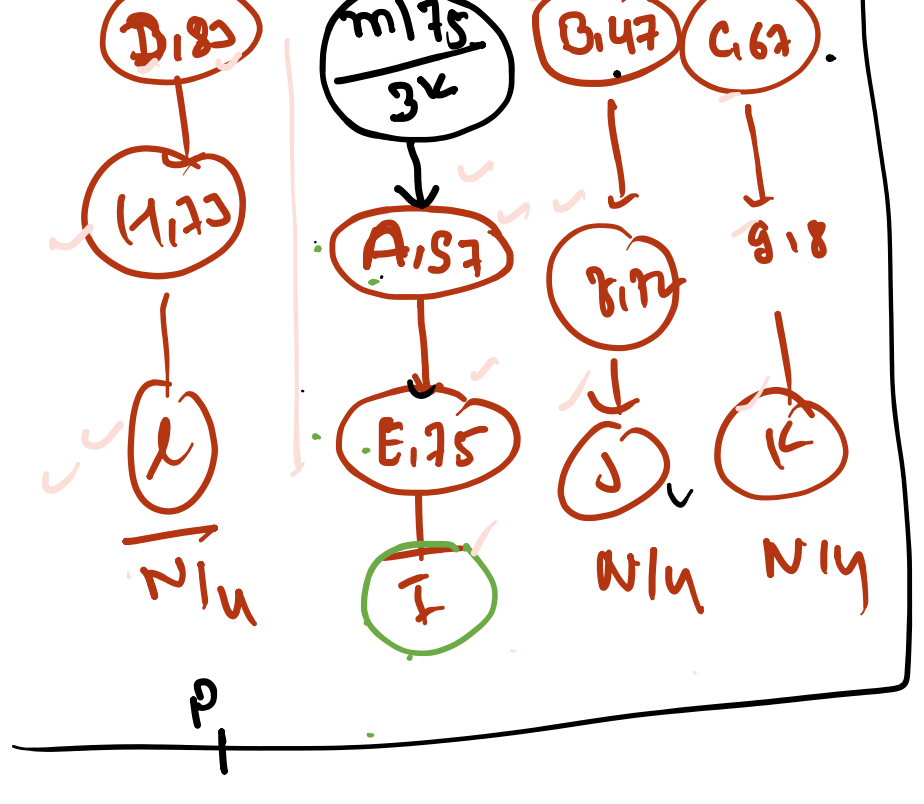


```java
public V remove(K key) {
        int idx = hashfun(key);
        Node curr = bukt.get(idx);
        Node prev = null;
        while (curr != null) {
            if (curr.key.equals(key)) {
                break;
            }
            prev = curr;
            curr = curr.next;
        }

    }
```



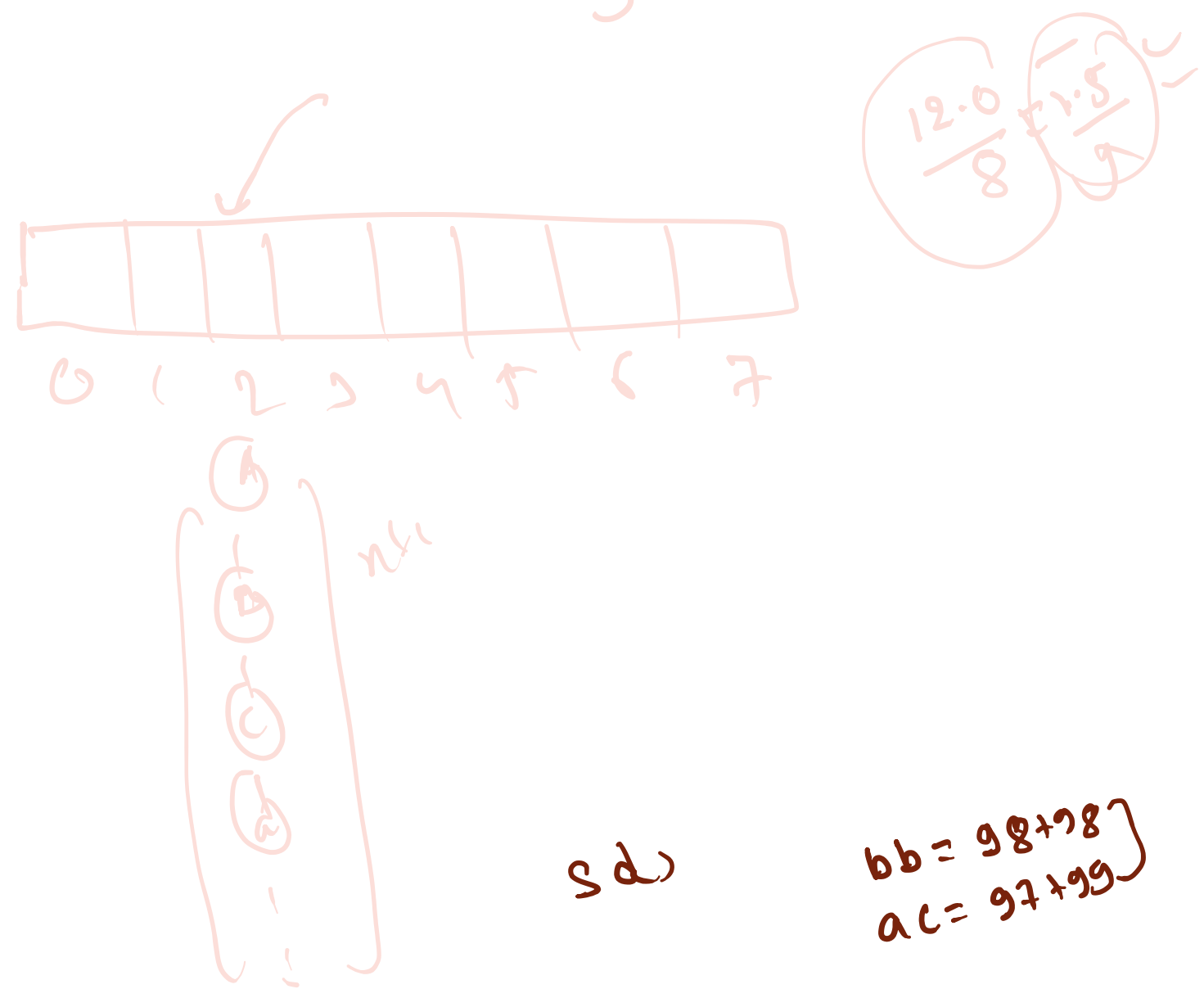
```java
public V remove(K key) {
        int idx = hashfun(key);
        Node curr = bukt.get(idx);
        Node prev = null;
        while (curr != null) {
            if (curr.key.equals(key)) {
                break;
            }
            prev = curr;
            curr = curr.next;
        }
        if (curr == null) {
            return null;
        }
        if (prev == null) {
            bukt.set(idx, curr.next);
        } else {
            prev.next = curr.next;
        }
        curr.next = null;
        size--;
        return curr.value;

    }
```





```java
private void rehashing() {
        // TODO Auto-generated method stub
        ArrayList<Node> new_bukt = new ArrayList<>();
        for (int i = 0; i < 2 * bukt.size(); i++) {
            new_bukt.add(null);
        }
        ArrayList<Node> oba=bukt;
        bukt=new_bukt;
        size = 0;

    }
```

For(Node nn : oba){
while(nn != null){
put(nn.key, nn.value);
nn.next
}
}



sd

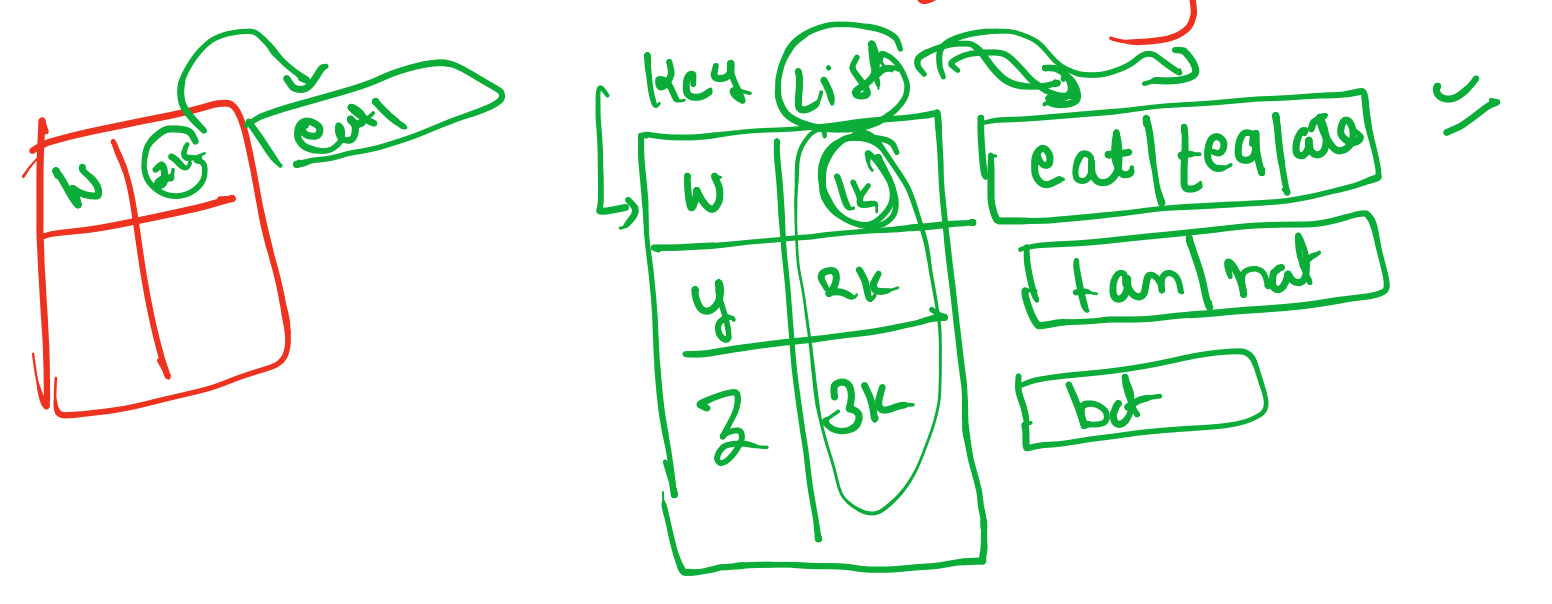
bb = 98+98
ac = 97+99

Given an array of strings `strs`, group the anagrams together. You can return the answer in **any order**.

Example 1:

> **Input:** strs = ["eat","tea","tan","ate","nat","bat"]
>
> **Output:** [["bat"],["nat","tan"],["ate","eat","tea"]]

[[eat, tea, ate]
[nat, tan]
[bat]]



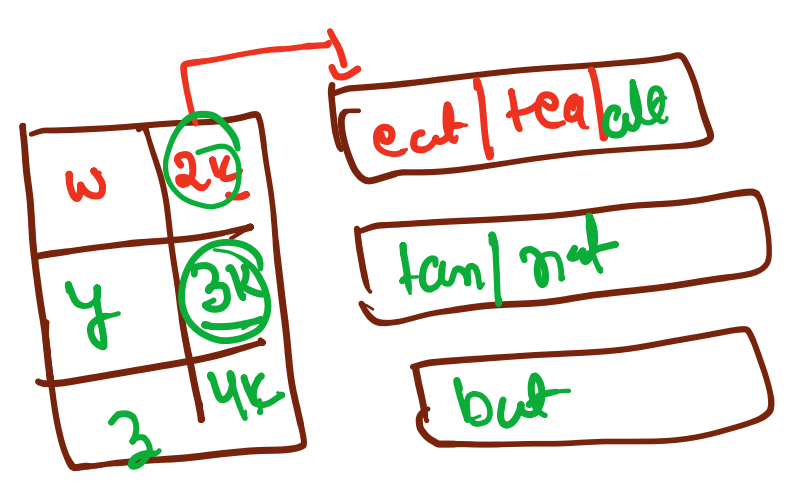
[2K | 3K - 4K]

```java
List<List<String>> ans = new ArrayList<>();
    for(String key:map.keySet()) {
        ans.add(map.get(key));
    }
```

(w, y, z)

```java
public static List<List<String>> group_Anagrams(String[] arr) {
        HashMap<String, List<String>> map = new HashMap<>();
        for (int i = 0; i < arr.length; i++) {

            String key = GetKey(arr[i]);
            if (!map.containsKey(key)) {
                map.put(key, new ArrayList<>());
            }
            map.get(key).add(arr[i]);
        }

    }
```

w y, w 3
eat

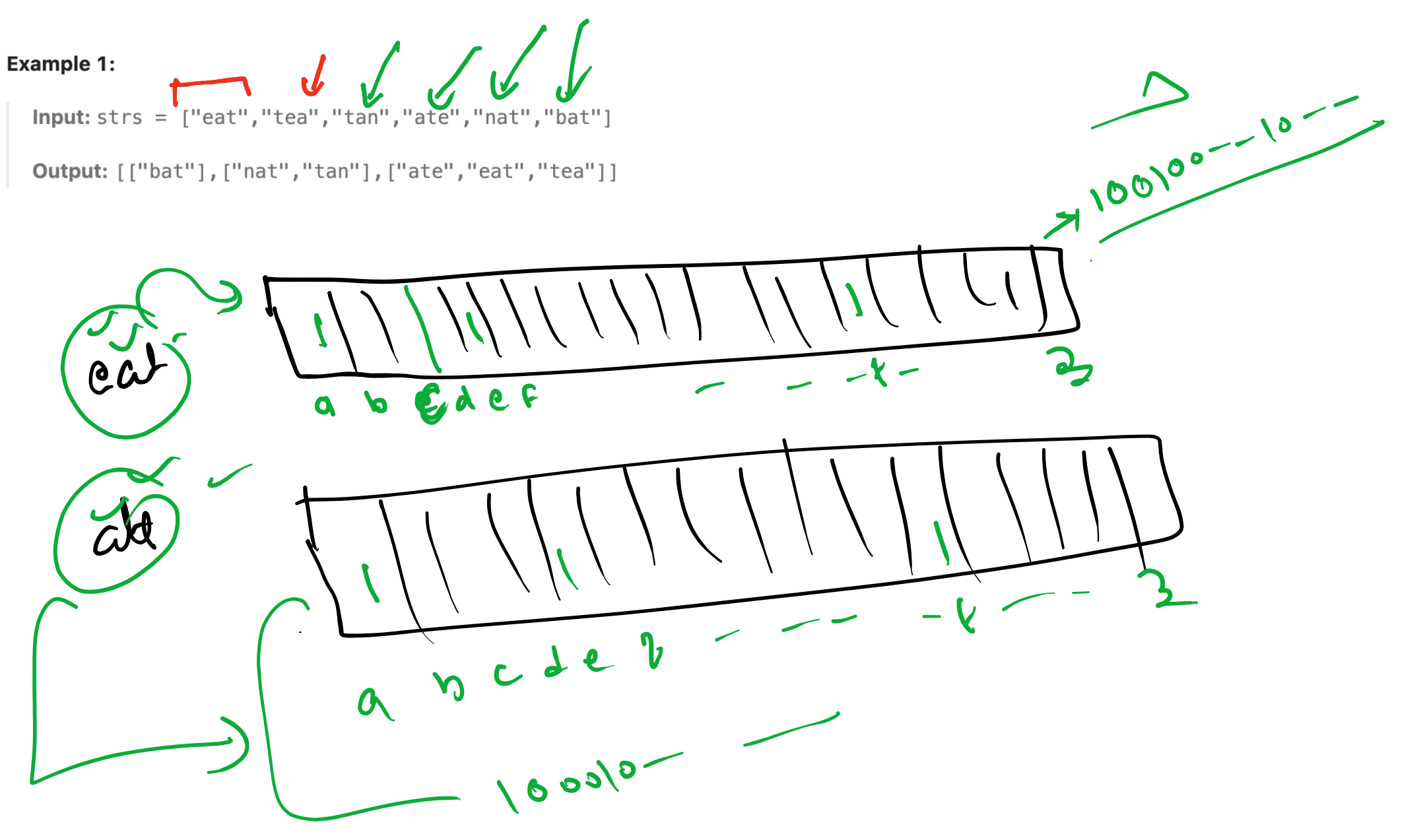


For(String key: map.keySet()){

Given an array of strings `strs`, group the anagrams together. You can return the answer in **any order**.

Example 1:

> **Input:** strs = ["eat","tea","tan","ate","nat","bat"]
>
> **Output:** [["bat"],["nat","tan"],["ate","eat","tea"]]



["bdddddddddd","bbbbbbbbbbc"]