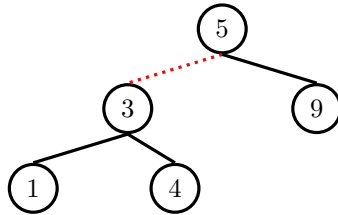


## 1 LLRB Insertions

Given the LLRB below, perform the following insertions and draw the final state of the LLRB. In addition, for each insertion, write the balancing operations needed in the correct order (rotate right, rotate left, or color flip). If no balancing operations are needed, write "Nothing". Assume that the link between 5 and 3 is red and all other links are black at the start.



(a) 1. Insert 7

2. Insert 6

3. Insert 2

4. Insert 8

5. Insert 8.5

6. Final state

(b) Convert the final LLRB to its corresponding 2-3 Tree.

## 2 Hashing Gone Crazy

For this question, use the following TA class for reference.

```
public class TA {
    int semester;
    String name;
    TA(String name, int semester) {
        this.name = name;
        this.semester = semester;
    }
    @Override
    public boolean equals(Object o) {
        TA other = (TA) o;
        return other.name.charAt(0) == this.name.charAt(0);
    }
    @Override
    public int hashCode() { return semester; }
}
```

Assume that the ECHashMap is a HashMap implemented with **external chaining** as depicted in lecture. The ECHashMap instance begins with 4 buckets.

**Resizing Behavior** If an insertion causes the load factor to reach or exceed 1, we resize by doubling the number of buckets. During resizing, we traverse the linked list that correspond to bucket 0 to rehash items one by one, and then traverse bucket 1, bucket 2, and so on. Duplicates are **not** checked when rehashing into new buckets.

Draw the contents of map after the executing the insertions below:

```
1    ECHashMap<TA, Integer> map = new ECHashMap<>();
2    TA jasmine = new TA("Jasmine the GOAT", 10);
3    TA noah = new TA("Noah", 20);
4    map.put(jasmine, 1);
5    map.put(noah, 2);
6
7    noah.semester += 2;
8    map.put(noah, 3);
9
10   jasmine.name = "Nasmine";
11   map.put(noah, 4);
12
13   jasmine.semester += 2;
14   map.put(jasmine, 5);
15
16   jasmine.name = "Jasmine";
17   TA cheeseguy = new TA("Sam", 24);
18   map.put(cheeseguy, 6);
```

### 3 Buggy Hash

The following classes may contain a bug in one of its methods. Identify those errors and briefly explain why they are incorrect and in which situations would the bug cause problems.

(a) The Timezone class below:

```

1      class Timezone {
2          String timeZone; // "PST", "EST" etc.
3          boolean dayLight;
4          String location;
5          ...
6          public int currentTime() {
7              // return the current time in that time zone
8          }
9          public int hashCode() {
10             return currentTime();
11         }
12         public boolean equals(Object o) {
13             Timezone tz = (Timezone) o;
14             return tz.timeZone.equals(timeZone);
15         }
16     }

```

(b) The Course class below:

```

1      class Course {
2          int courseCode;
3          int yearOffered;
4          String[] staff;
5          ...
6          public int hashCode() {
7              return yearOffered + courseCode;
8          }
9          public boolean equals(Object o) {
10             Course c = (Course) o;
11             return c.courseCode == courseCode;
12         }
13     }

```