1 Hashing Gone Crazy

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

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

Assume that the hashCode of a TA object returns charisma, and the equals method returns true if and only if two TA objects have the same first letter in their name.

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

```java
ECHashMap<TA, Integer> map = new ECHashMap<>();
TA sohum = new TA("Sohum", 10);
TA vivant = new TA("Vivant", 20);
map.put(sohum, 1);
map.put(vivant, 2);

vivant.charisma += 2;
map.put(vivant, 3);

sohum.name = "Vohum";
map.put(vivant, 4);

sohum.charisma += 2;
map.put(sohum, 5);

sohum.name = "Sohum";

TA shubha = new TA("Shubha", 24);
map.put(shubha, 6);
```

Assume that the ECHashMap is a HashMap implemented with external chaining as depicted in lecture. The ECHashMap instance begins at size 4 and, for simplicity, does not resize.
2 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) class Timezone {
    String timeZone; // "PST", "EST" etc.
    boolean daylight;
    String location;
    ...
    public int currentTime() {
        // return the current time in that time zone
    }
    public int hashCode() {
        return currentTime();
    }
    public boolean equals(Object o) {
        Timezone tz = (Timezone) o;
        return tz.timeZone.equals(timeZone);
    }
}

(b) class Course {
    int courseCode;
    int yearOffered;
    String[] staff;
    ...
    public int hashCode() {
        return yearOffered + courseCode;
    }
    public boolean equals(Object o) {
        Course c = (Course) o;
        return c.courseCode == courseCode;
    }
}
3  Semi Sorted Heaps

Given a heap represented as an array, determine if it is a valid min-heap and semi-sorted.

For a min-heap to be semi-sorted, all the elements in the left branch must be smaller than the elements in the right branch. The first element of the min-heap is at index 1, and you can assume the array has length of at least 4 for simplicity.

For e.g. if array A = [ ∗, 1, 2, 6, 5, 4, 9, 8] represents a heap, the method should return true, since 2, 5, 4 in left branch are all less than 6, 9, 8 in right branch.

```java
public static boolean isSemiSortedHeap(int[] arr) {
    int rightTop = _________;

    for (__________________________________________) {
        for (__________________________________________) {
            __________________________________________
            __________________________________________
            __________________________________________
            __________________________________________
            __________________________________________
        }
    }
}
```

4  Min Heaps

Fill in the following blanks related to min-heap -

(a) removeMin has a best case runtime of ______________ and a worst case runtime of ______________.

(b) insert has a best case runtime of ______________ and a worst case runtime of ______________.

(c) A ______________ or ______________ traversal on a min-heap can output the elements in sorted order.

(d) The fourth smallest element in a min-heap with 1000 distinct elements can appear in ______________ places in the heap.

(e) Given a min-heap with $2^n - 1$ distinct elements, for an element -

   • to be on the second level it must be less than ______________ element(s) and greater than ______________ element(s).

   • to be on the bottommost level it must be less than ______________ element(s) and greater than ______________ element(s).