1 Counting Stars

Given a String[] args, write a method that counts the number of appearances of "star", case sensitive. If this number is even, simply return args. Otherwise, return a String[] containing all the non-"star" entries (including null entries). Your code shouldn't error for any input, you may not use the modulo (%) operator, and you are not allowed to take more than one pass through the input array.

A potentially helpful method: Arrays.copyOf(String[] array, int newLength) - Copies the specified array, truncating or padding with nulls as necessary so the returned copy has the specified length. For example, if orig is {"a", "2", "3"}, Arrays.copyOf(orig, 2) is {"a", "2"} and Arrays.copyOf(orig, 4) is {"a", "2", "3", null}.

import java.util.Arrays;

public static String[] starCount(String[] args) {

}

2 HugString: Part 1 of 2

You will be helping Josh convert an input String to a singly-linked list of char’s. You’ll first need the building blocks: your linked nodes.

class CNode {
    _________ head;
    _________ next;
    public CNode( _________ head, _________ next) {
}
}
3  HugString: Part 2 of 2

Now implement the method that makes the HugString. You may want to use the `String.charAt(int loc)` method, which returns the character at position `loc`. For example, "hey".charAt(0) returns ‘h’.

```java
/** Converts the input String s into a linked list of CNodes and returns the head of the list. */
public static CNode makeHugString(String s) {
```

4  HugString: Part 3 of 2 (Additional for Aces)

Building off of your code base from above, write a method `swapSpace` to destructively replace every space (" " ) in an input CNode linked-list with "61B". The input is the first CNode node.

```java
public void swapSpace(CNode in) {
```