Strings & Arrays Interview Questions

First two questions with solutions from the class.

```java
//Q1: Write an algorithm to check if a string has all unique characters.
boolean isUnique(String str) {
    //ASCII (128), unicode () //O(1) , constant
    if (str.length() > 128) return false;
    boolean[] chars = new boolean[128];

        // [0,1,2,3...127] , str= "Today is a nice day"
    // [0, 0,0, 1,....1,0,0,1,1 ]
    for (int i = 0; i < str.length(); i++) {
        int val = str.charAt(i);
        if (chars[val]) {
            return false;
        }
        chars[val] = true;
    }
    for (int i = 0; i < 10; i++)
    { System.out.println(chars[i]); }
    return true;
}


//Q2: Write an algorithm to check if a string is palindrome.
boolean isPalin(String str) {
    int i = 0;
    int j = str.length()-1;
    while(i<j) {
        if(str.charAt(i) != str.charAt(j))
            return false;
        i ++;
        j--;
    }
    return true;
}
```

1. We have two strings, write a function to check if one is a permutation of the other.
2. Write a function to check if the given input string is a permutation of a palindrome.
3. Write a function to return the reverse string of the input string.
4. Write a function to count the number of words & spaces in the input string.
   Int countWords&Spaces(String str) { }

5. Write a function to count the number of times a word occurs in an input string.
   int countOccurence(String str, String word) { }
   Hint:

i. *First split the string by spaces in an array. (use, str.split(" "))*
ii. *Next, take a variable count = 0. For every true condition we increment the count by 1*
iii. *Now run a loop from 0 to 'length of string' and check if our string is equal to the word*

iv. If condition true -> increment the value of count by 1 & in the end we return the count.

6. Write a function to left rotate the input string by '*t*' times. (t <= str.length())

```
Ex: Input : str = "SunnyDay"
          t = 2
Output : Left Rotation  : "nnyDaysu"
```

*Hint: Use a temporary string to do rotations. For left rotation, first copy last n-t characters, then copy first **t** characters in order to the temporary string.*

7. Write a function to right rotate the input string by '*t*' times. (t <= str.length())

```
Ex: Input : str = "SunnyDay"
          t = 2
Output : Left Rotation  : "aySunnyD"
```

Hint: Use a temporary string to do rotations. For right rotation, first copy last t characters, then copy n-t characters.

8. Given two input strings s1 and s2, find if s1 can be converted to s2 with exactly one edit.

An edit between two strings is one of the following changes:

Add a character OR Delete a character OR Change a character

Ex:

```
Input:  s1 = "Donny", s2 = "Donn"
Output: yes
Number of edits is 1

Input:  s1 = "Donny", s2 = "Donny"
Output: no
Number of edits is 0
```

Hint:
Let the input strings be **s1** and **s2** and lengths of input strings be **m** and **n** respectively.

```
1) If difference between m an n is more than 1,return false.
2) Initialize count of edits as 0.
3) Start traversing both strings from first character.
    a. If current characters don't match, then
       (i)   Increment count of edits
       (ii)  If count becomes more than 1, return false
       (iii) If length of one string is more, then only
             possible edit is to remove a character.
             Therefore, move ahead in larger string.
       (iv)  If length is same, then only possible edit
             is to  change a character. Therefore, move
             ahead in both strings.
    b. Else, move ahead in both strings.
```

9. Given two strings s1 and s2, find if s1 is a substring of s2. If yes, return the index of the first occurrence, else return -1.
10. Write a function to perform basic string compression. For example, the string “aabbbbccca” would become “a2b4c3a1”. (Can be solved by using Stack, but think about the possible solutions)