Problem Link:

https://leetcode.com/problems/longest-common-prefix/

Problem Description:

Write a function to find the longest common prefix string amongst an array of strings. If there is no common prefix, return an empty string `""`.

Problem Approach:

Two ways to solve the problem O(n logn) solution and O(n * m). Both have equal space complexity as O(1). In one solution, we'll sort the array, and compare. In the other, we'll iterate through the array and find the solution.

Sample Test Case:

Input: strs = ["flower","flow","flight"]

Output: "fl"

Solution:

This is the efficient solution, the O(n * m) one, where n is the number of strings, and m is the length of the shortest string. The approach is quite easy to catch. We start by initializing the prefix variable as the first element in the strings array. Then, we iterate through every element in the array, and check whether the element starts with the prefix or not. We do it by using the method:

.startswith(string) -> It returns a boolean, confirming whether the string starts with the provided prefix in the params.

The same method exists in java too, named as: **.startsWith(string)**.

If it does, then we move to the next iteration, and if it doesn't, we keep on shortening the prefix until the current string in the iteration starts with the prefix. Meanwhile, if the prefix becomes empty, it means that the current string is completely different from the first string of the array. And hence, there is no point of iterating any further, since there will be no "**COMMON PREFIX**". And if that's not the case, then, after iterating through the whole array, we'd have the longest common prefix in the prefix variable, and that's our answer!

Code (Python):

```python
def longestCommonPrefix(self, strs: List[str]) -> str:
        prefix = strs[0]
        for string in strs:
            while not string.startswith(prefix):
                prefix = prefix[:-1]
                if not prefix:
                    return ""
        return prefix
```

In the following solution, we use a simpler approach. Sort the strings. Then, compare the first and last string character by character, in this case, I've maintained a counter that counts the number of common characters between the two strings. And in the end, return the slice of the elements of the first string until the counter's count.

```python
def longestCommonPrefix(self, strs: List[str]) -> str:
        strs.sort()
        str1 = strs[0]
        str2 = strs[len(strs)-1]
        counter = 0
        for i in range(len(str1)):
            if str1[i] == str2[i]:
                counter+=1
            else:
                break
        return str1[:counter]
```