

14. Longest Common Prefix

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 "".

Example 1:

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

Output: "fl"

Example 2:

Input: strs = ["dog", "racecar", "car"]

Output: ""

Explanation: There is no common prefix among the input strings.

Constraints:

- $1 \leq \text{strs.length} \leq 200$
- $0 \leq \text{strs}[i].\text{length} \leq 200$
- $\text{strs}[i]$ consists of only lowercase English letters.

Longest Common Prefix

["VISHAL"] ["VISHU"]

["VISH"]

{ we have to return }
{ this string }

Constraints:

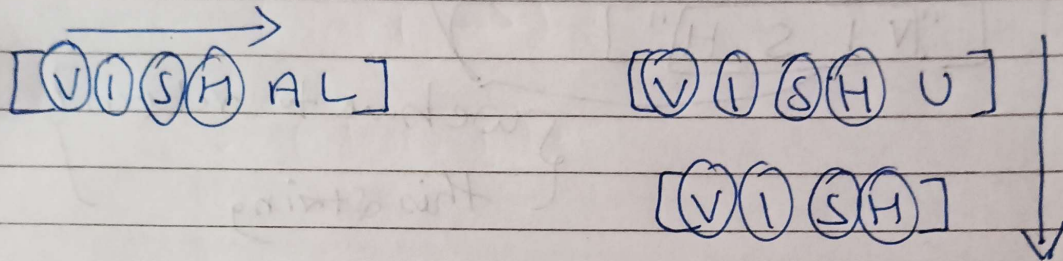
Strings max: 200

every string length max: 200

Strings consists only lower case letters

Approach

∴ On the first string traverse horizontally along with the remaining strings but vertically



Traverse: 1

$$[V] = \begin{bmatrix} V \\ V \end{bmatrix} \quad \text{True}$$

Traverse: 4

True

longest prefix = V

Traverse: 2

$$H = \begin{bmatrix} H \\ H \end{bmatrix}$$

IP = VISH

$$[I] = \begin{bmatrix} I \\ I \end{bmatrix} \quad \text{True}$$

Traverse: 5

longest prefix = VI

Traverse: 3

(One string is over)

$$[S] = \begin{bmatrix} S \\ S \end{bmatrix} \quad \text{True}$$

False

longest prefix = VIS

Code: (Create a string of longest prefix)

- 1) make a variable and store total number of strings.

```
int noOfStrings = str.length;
```

- 2) make a variable who holds 0th string length.

```
int size = str[0].length();
```

- 3) Run first for loop horizontally to str[0]. and also create a variable to holds its char.

```
[ for(int i=0; i<size; i++)  
  char value = str[0].charAt(i); ]
```

- 4) Now, Run a for loop inside this upper loop but vertically and compare this {value} to every string char.

: we start it from 1 because 0 string is taken for [→] traverse.

```
for (int j=1; j<noOfStrings; j++) {
```

```
  if (value != str[j].charAt(i)) {
```

```
    return j;
```

```
  }
```

⑤ Outside the for loop add the char into lp because it presents in all the strings.

```
for (loop 1 →) {  
    value =             
    for (loop 2 ↓) {
```

```
        [lp += str value;]  
    }
```

Now, full code:

```
for (loop for hori → traversing) {
```

```
    value = str[0].charAt( )
```

```
    for ((loop for vertical ↓ traversing) ↓
```

```
        if (value != str[ ].charAt( )) {  
            return lp;
```

```
        }
```

```
    }  
    lp += value;  
}
```

```
return lp;
```


Special case:

⇒ if a string is ends or shorter than the horizontal string

examples:

① [a b] [a]

② ["a b c"] ["a b"] ["a"]

→ for solving this we also check for this condition

if $(\text{str}[i].\text{length}() - 1) < i$ {

return 16;

}

means, if current string length is smaller than the current index on which we are checking on the return the 16; ~~because~~