Lecture 13: Strings

ASCII

'A' - 65	'a' - 97	'0' - 48
'B' - 66	'b' - 98	'9' - 57
⋮		
'Z' - 90	'z' - 122	

'A' $\longrightarrow$ 65 $\longrightarrow$ $(0\,1\,0\,0\,0\,0\,0\,1)_2$

'a' $\longrightarrow$ 97 $\longrightarrow$ $(0\,1\,1\,0\,0\,0\,0\,1)_2$

flip from lower to upper & vice-versa by add/subtract

$2^5 = 32$

ord('a') = 97

chr(97) = 'a'

(42) Given a string flip the case of chars.

```
def flip(A):
        l = list(A)
        for ix in range(len(l)):
                l[ix] = chr(ord(l[ix])^32)
return "".join(l)
```

Follow-up

If any character is in uppercase, convert it to lowercase

```
l = list (A)
ans = ""
for c in l:
        if c > 'A' or c < 'Z':
                c = chr(ord(c)^32)
                ans += c
        else:
                ans += c
return ans
```

(43) Given the string in lowercase, sort it.

Brute Force

```
str → list
list.sort()
"".join(list)
```

[a g d c a b]

⇒ [a a b c d g]

O(NlogN)

Trick

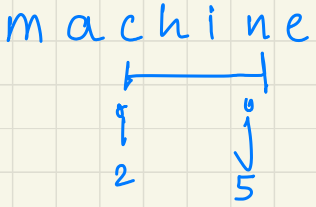
	a	b	c	d	e	f	g	h			
	0	1	2	3	4	5	6	7	8	9	10
freq [	2	1	1	1	0	0	1	0	0	0....	

```
freq = [0]* 26
ans = ""
for x in A :
    freq[ ord(x) - ord('a') ] += 1

for i in range(26):
    for j in range( freq[i]):
        ans += chr(97+i)

return ans
```

TC:
const. time

→ [a a b c d g]

44 Given a substring, reverse it



① convert to list

②
```
while i < j:
    l[i], l[j] = l[j], l[i]
    i += 1
    j -= 1
```

③ return "".join(list)

45 Reverse the words in a sentence

"I am a slave"

1. sentence.split()
2. Reverse the list
3. join()

OR

```
l = s.split()            i=0    j= len(l)
while i<j :
    l[i] , l[j] = l[j], l[i]
    i += 1
    j -= 1
return " ".join(l)
```

46 Given 2 strings, find the length of longest common prefix string

"professor"

"project"

=> 3

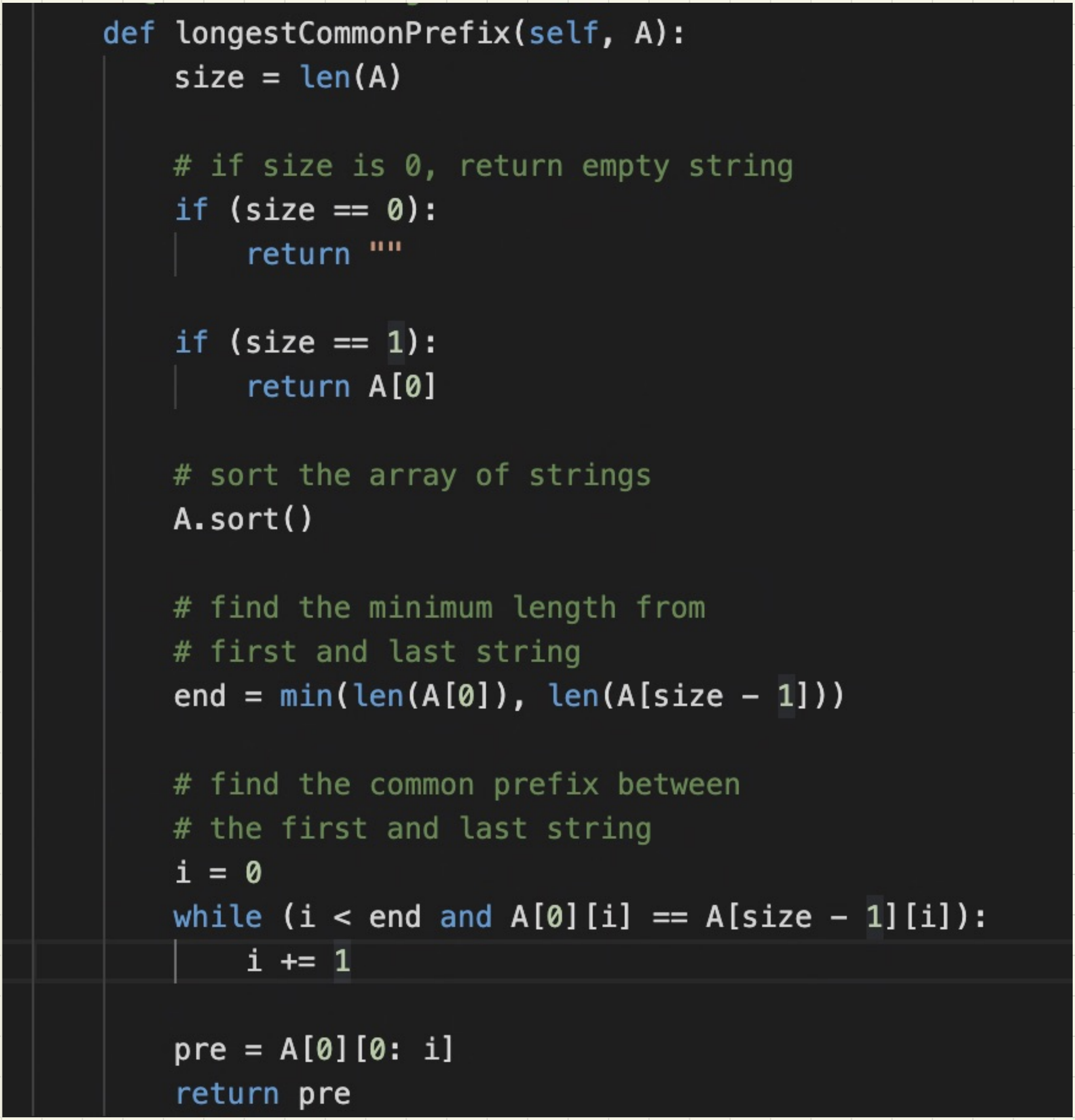
```
ans = 0
for i in range(min(len(a), len(b))):
    if a[i] == b[i]
        ans += 1
    else :
        break
return ans
```

(47) Find longest common prefix string in N strings

[professor
project
pronunciation
pronouns]

```
ans = 0
n = len(l)

min-word-len = len(l[0])

for word in l:
    if len(word) < min-word-len :
        min-word-len = len(word)

for i in range(min-word-len):
    for j in range(n-1):
        if l[j][i] == l[j+1][i]:
            ans += 1

        else:
            break
return ans
```

Alternative solutn :



```
    def longestCommonPrefix(self, A):
        size = len(A)

        # if size is 0, return empty string
        if (size == 0):
            return ""

        if (size == 1):
            return A[0]

        # sort the array of strings
        A.sort()

        # find the minimum length from
        # first and last string
        end = min(len(A[0]), len(A[size - 1]))

        # find the common prefix between
        # the first and last string
        i = 0
        while (i < end and A[0][i] == A[size - 1][i]):
            i += 1

        pre = A[0][0: i]
        return pre
```

(48) Check if a string is palindrome

"nayan" ↔ "nayan"

Brute Force

```
if l == l[: : -1] :
        return 1
return 0
```

O(N)

Optimized

```
i = 0    ,  j = n-1
while i < j :
        if  a[i] != a[j]
                return False
        i + = 1
        j - = 1
return True
```

(49) Find the length of longest palindrome substring

a a b c c b a d

check if the substring is a palindrome

```
def helper (l, r) :
    while (l >=0 and r < len(A)) and
          ( A[l] == A[r]):

        l - = 1
        r + =1
    return ( A [l+1 : r])
res = ""
for i in range(len(A)):
    odd-sub = helper(i,i)
    even-sub = helper(i, i+1)
    if len(odd-sub) > res :
        res = odd-sub
    if len (even-sub) > res:
        res = even-sub
return res
```

abba

i = 0 to 3

i = 0

helper(0, 0) → a

res = "a"

helper(0, 1) → a

res = "a"

i = 1

helper(1, 1) → b

res = "b"

helper (1, 2) → l = 0 r = 3 l = -1 r = 4

×

res = "bb"

[0 : 4]

i = 2

helper(2 2) → "b"

helper

50 Given a string, find no. of vowel substrings

"ABEC" ⇒ "A"
"AB"
"ABE"
"ABEC"
"E"
"EC"
cnt = 6

```
vowels = [        ]
cnt = 0
for i in range(len(A)):
    if A[i] in vowels:
        cnt += len(A) - i
return count % 10003
```