Hi Friends

Parikshit

SDE 2 Amazon

Scaler 1.5 years

Class begins @ 9:05

8510955377

Strings " "

→ Array of chars

'a' '$' '1'

'A' '_' '2'

'3'

⋮

'9'

'10' string

'a' 'a'

Shubhanker Shivam

↓

97

$\boxed{\text{ASCII}}$

Java

```
char ch = 'a' +1
sop (ch) b
```

Python

ord('a') → 97

chr(97) → 'a'

ASCII

ASCII			
'A' = 65	32	'a' = 97	'0' = 48
'B' = 66	32	'b' = 98	'1' = 49
'C' = 67		'c' = 99	⋮
⋮		⋮	
'Z' = 90	32	'z' = 122	'9' = 57

Q1) Given a string, toggle the case of every char.

alphabets

upper case → lower case

lower case → upper case

s = "aBcAEd"

output : AbCaeD

Answer → to me

Question → to everyone

if 'small char' sub 32

if 'big char' add 32

TC: O(N)

SC: O(1)

```
for(i=0; i< N; i++){
        if (s[i] >= 'A' and s[i] <= 'Z')
                s[i] = s[i]+32
        else    s[i] = s[i]-32
}
```

Q Sort an array of 0s & 1s

$$A \; \begin{array}{cccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ [1 & 0 & 1 & 0 & 0 & 1 & 0 & 1] \end{array}$$

A.sort()   Collections.sort(A)

TC: O(N log N)

$$\begin{array}{ccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ [\cancel{1} & \cancel{0} & \cancel{1} & \cancel{0} & \cancel{0} & \cancel{1} & \cancel{0}] \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{array}$$

0 : 4

1 : 3

$$TC: O(\underbrace{N}_{\text{find count}} + \underbrace{N}_{\text{to put 0s and 1s}})$$

TC: O(N)

Q2) Given a string containing only ['a' - 'z'], sort it.

{dictionary order}

s: a b d a c b d e

After sorting

output : a a b b c d d e

Arrays. sort S.sort()

~~a b d a c b d e~~

a a b b c d d e

'a' _ 'z'

a : 2

b : 2

c : 1

d : 2

e : 1

0	1	2	3	4	5	6	7	. . .	25
2	2	1	2	1	0	0	0		0

↓ freq of 'a'

↓ freq of 'b'

	s[i] - 'a'
'a'	0
'b'	1
'd'	3
'c'	2
'c'	4

a b d a c b d e f

↑ ↓

5

```
int count [26] = {0}
for ( i=0; i<N; i++) {
        ind = s[i] - 'a'        # s[i] = 'b'
        count[ind] += 1
}
```

Step 2: # putting in the array

0	1	2	3	4	5	6	7	...	25
2	2	1	2	1	0	0	0		0

count [0] + count [1] + count [2] ... count [25] = len of string

'a'
↓

0	1	2	3	4	5	6	7	...	25
~~2~~ ~~1~~ 0	~~2~~ ~~1~~ 0	~~1~~ 0	~~2~~ ~~1~~ 0	~~1~~ 0	0	0	0		0

0 represents 'a'

2 represent count of 'a'

index to fill = 8
↓
current character that I am filling

0	1	2	3	4	5	6	7
~~a~~	~~b~~	~~d~~	~~a~~	~~c~~	~~b~~	~~d~~	~~e~~
a	a	b	b	c	d	d	e

1) find value to fill
2) fill it in array
3) reduce the count of freq
4) inc index to fill =

i = s[i] - 'a'

```
# given Sting   $\boxed{S}$
indextofill = 0
for (i=0; i<=25; i++){
    chartofill = i + 'a'
    timestofill = count[i]
    while ( timestofill > 0) {
        S[indextofill] = chartofill        # filling in string
        timestofill -=1                    # reduce times to fill
        indextofill +=1
    }
}
```

TC: O(N)

SC: O(1)

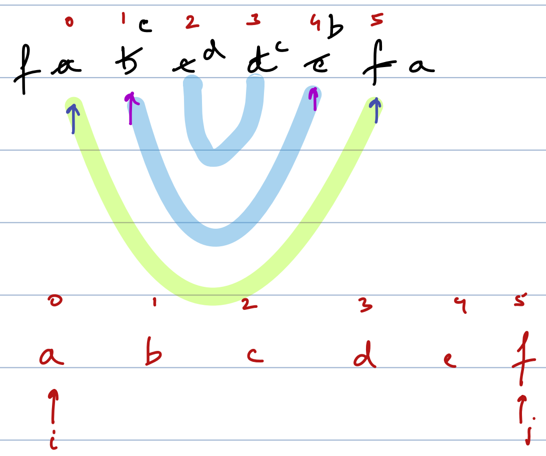
i	times to fill	iteration
0	count[0]	count[0]
		+
1	count[1]	count[1]
		+
:		
:		
25	count[25]	+ count[25]
		N

Break (10:31-10:40)

```
int count[26] = {0}
for (i=0; i<N; i++){
        ind = s[i]-'a'          # s[i]='b'
        count[ind] +=1
}

indextofill = 0
for (i=0; i<=25; i++){
    chartofill = i + 'a'
    timestofill = count[i]
    while (timestofill > 0) {
            S[indextofill] = chartofill
            timestofill -=1
            indextofill +=1
    }
}
```

Reverse a string



```
i = 0 , j = N-1                         TC: O(N)
while ( i < j ) {
        swap (s[i], s[j])
            i++
            j--
}
```

Q3) Check if a string is palindrome or not?

S = N I T I N

rev(s) = N I T I N

s = rev(s)

a b c d c b a

↑

↑

a b c d a

↑ ↑

```
i=0    j=N-1
while( i<j ){
    if (s[i] != s[j]){
        return false
    }
    i++
    j--
}
return true
```

Monther in Law Hindi

→ Saas

Chain ki saas

Vicks

$[s \quad e]$

$e-s+1$

Q4) Given a string find length of longest palindromic substring ?

0 1 2 3 4 5

a b a c a b

Ans: 5

a → 1

aba → 3

bacab → 5

0 1 2 3 4 5 6 7 8 9 10 11

f c f a b d k d b a l l

Ans : 7

ll → 2

fcf → 3

abdkdba → 7

1) Find every substring

2) Check if palindrom or not

s <= e

↑

0 1 2 3 4 5 6 7 8 9 10 11

f c f a b d k d b a l l

i

TC: $O(N^3)$

```
maxlen = 0
for (e=0; e<N; ++){
    for (s=0; s<=e ;j++){
        # substring [s e]
            # check palindrom
            if (palindrome){
                len = e-s+1

                maxlen = max(maxlen, len)
```

3

3 3

a b c | b a

Center

a   0 1 2 3 4
     a b c b a
   ↑ p1   ↑ p2
                        p2 - p1 - 1

len = 1

b   0 1 2 3 4
     a b c b a
     ↑ p   ↑ p2
                 len = 1

c   0 1 2 3 4
     a b c b a
         ↑
  ↑ p1   ↑ p2
                        len = 5

b   0 1 2 3 4
     a b c b a
                        len = 1

a   0 1 2 3 4
     a b c b a
             ↑
                        len = 1

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
x	b	d	y	z	z	y	d	b	d	y	z	y	d	x
								↑ p1						↑ p2

$[p1 \quad p2] : p2-p1+1 \xrightarrow{-2}$

$(p1 \quad p2) : p2-p1-1$

0	1	2	3	4	5	6	7	8	9	10	11
f	c	f	a	b	d	k	d	b	a	l	l
		↑ p1								↑ p2	

TC: O(N)

```
int expand(s, p1, p2){
        while(p1>=0 and p2<N and s[p1] == s[p2]){
                p1--
                p2++
        }
        return p2-p1-1
}
```

given string

```
int  expand(s, p1, p2){
    while(p1>=0 and p2<N and s[p1] == s[p2]){
            p1--
            p2++
    }

    return p2-p1-1
}
```

```
maxlen = 0
for (i=0; i<N; i++){
        p1=i , p2=i
        len = expand(s, p1, p2)
        maxlen = max(maxlen, len)
}
```

TC: O(N^2)

single center

```
for (i=0; i<N; i++){
        p1 = i     p2 = i+1
        len = expand(s, p1, p2)
        maxlen = max(maxlen, len)
}

return  maxlen
```

double center

f a b b a   len = 1

↑

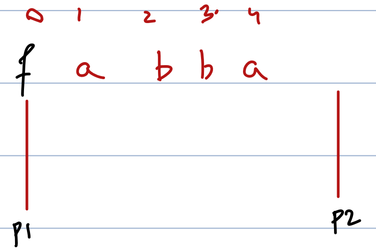
f a b b a   len= 1

  ↑

f a b b a   len=1

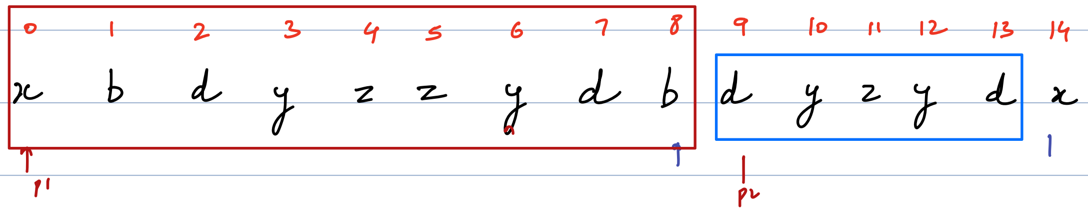
    ↑

actual ln = 4

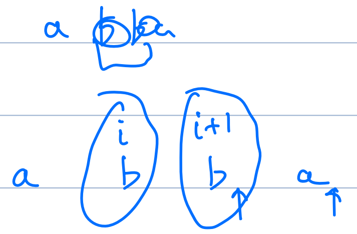


len = 4

DONE!



0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
x	b	d	y	z	z	y	d	b	d	y	z	y	d	x


a
b
b
a
a
i
b
i+1
b
a