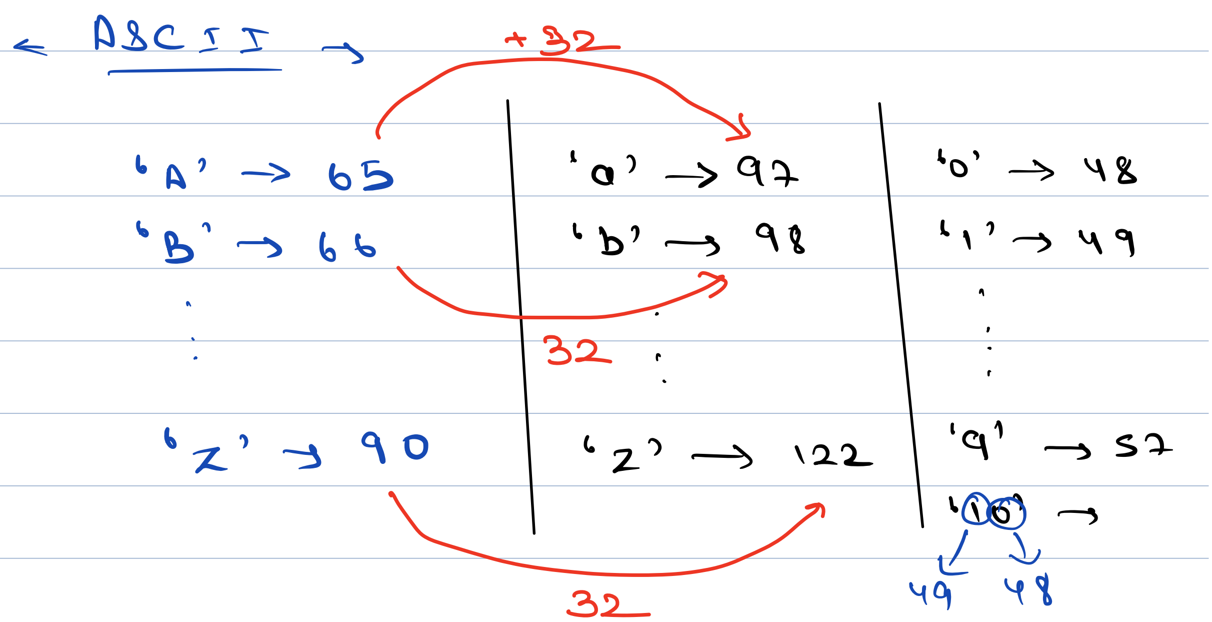
Test → 4 Questions.
↓
Contest.

Strings :- Set of char. / Group of char ✗
Sequence of char
Array / List of char

abcd
bcda



← ASCII →

'A' → 65
'B' → 66
⋮
'Z' → 90

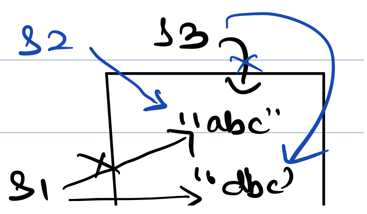
'a' → 97
'b' → 98
⋮
'z' → 122

'0' → 48
'1' → 49
⋮
'9' → 57
'10' →

Immutable Strings

(Java / python & other languages)

String s1 = "abc"
s1[0] = 'd'

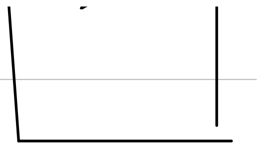


//s1 → "dbc"

String s2 = "abc"

String s3 = "abc"

s3 = "dbc"



String pool

Append char to immutable string :-

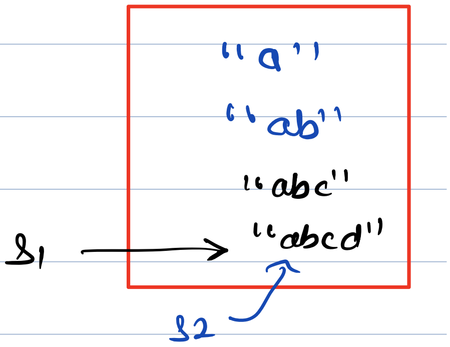
String s1 = "a"

s1 = s1+'b' ~~O(1)~~ O(n)

s1 = s1+'c' ~~O(1)~~, O(n)

s1 = s1+'d' ~~O(1)~~ O(n)

s2 = 'abcd'



If we append n times :-

T.C → O(n^2) | Garbage collector cleans out useless strings from memory.

S.C → O(n^2)

→ strings are not Immutable.

StringBuilder :-

Array of characters.

```
StringBuilder sb = new StringBuilder();
    sb.add('c')          → O(1)
    sb.append("abc");    → O(1).
    sb.toString();       → Java string.
```

Ques) Given a String s. Toggle the case of every char.

upper case → lower case
lower case → upper case.

s: → aBcDEd → AbCaeD

```
toggle (s) {
    for ( i=0; i< s.size(); i++) {
        //if s[i] is lower case;
        if(s[i] >= 'a' && s[i]<='z'){
            s[i] -= 32;
        } else {
            s[i] ^= 32;
        }
    }
}
```

T.C → O(n)
S.C → O(n)/O(1)
Immutable strings

'a' → 97
'b' → 98
⋮
'z' → 122

```
0 1 1 _ _ _ _ _
7 6 5 4 3 2 1 0
```

'A' → 65

'B' → 66

⋮

'Z' → 90

0	1	0	_	_	_	_	_
7	6	5	4	3	2	1	0

'z' → 122

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

'Z' → 90

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

s[i] = s[i] ^ (1<<5)

↓

One step for every characters.

10:14 → 10:24 pm :-

Ques) Given a string of lowercase char. Sort it in dict order.

S: d a b a e d b

→ a a b b d d e

Use library method of sorting

↳ O(nlogn)

HashMap / Dict :-

d : 2

a : 2

b : 2

e : 1

0 → 2

1 → 2

2 → 0

3 → 2

4 → 1

int count[26] = {0}

'a' - 97

'b' - 97

'c' - 97

0	1	2					25
2	2	5	1	1	1	1	1

aabb

```
int count[26] = {0}
for (i=0; i<n; i++){
```

→ O(n).

```
        char ch = s[i];
         int idx = ch-'a'
          count[idx] = count[idx]+1
    }
```

```
k=0

for(i=0; i<26; i++){
      // ith index char is coming how
                          many times.

          int freq = count[i];
           for(j=0; j<freq; j++){
              s[k] = char(i+'a')
              k=k+1
           }                          → O(n).
```

d a a b e

a	b	c	d	e
0	1	2	3	4
↓	↓	↓	↓	↓
2	1	0	1	1

SC → O(n) / O(1)

O(n) ↓ String builder

O(1) ↓ If your language allows mutability.

lc char → 26

lc + uc → 52

All char → 256

(Counting sort) ⇒

11:03 we'll start :-

Ques) :- Given a string s & two indices l & r, reverse the substring from l to r.

0	1	2	3	4	5	6
a	b	d	e	a	g	f

```
reverse(s, l, r)
    while (l<r) {
        swap(s[l], s[r])
        l++
        r--
    }
}
```

T.C → O(N)

S.C → O(1) → mutable string

O(N) → immutable string

Amazon :-

Ques) Given a character array storing a sentence.
Reverse it word by word.

→ No Extra space is allowed.

→ Every word is seperated by a single white space.

→ No inbuilt method can be used.

h	e	r	e		i	s		a		b	o	y

→ boy a is here

* "Are you as clever as I am"

→ "am I as clever as you Are"

"mailmen bring letters"

reverse the whole string

"srettel gnirb nemliam"

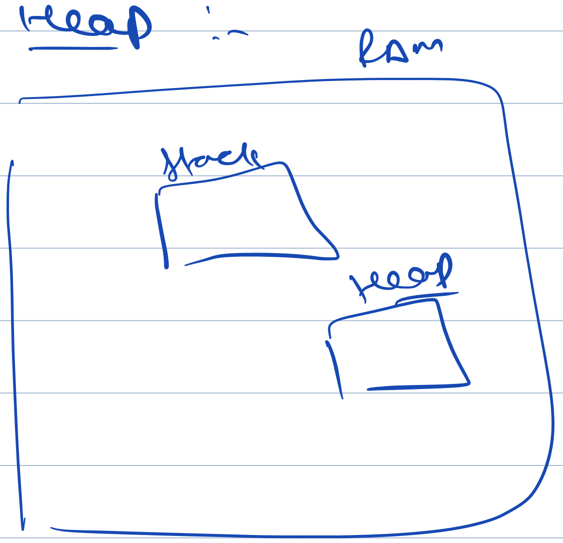
reverse each word

"letters bring mailmen"

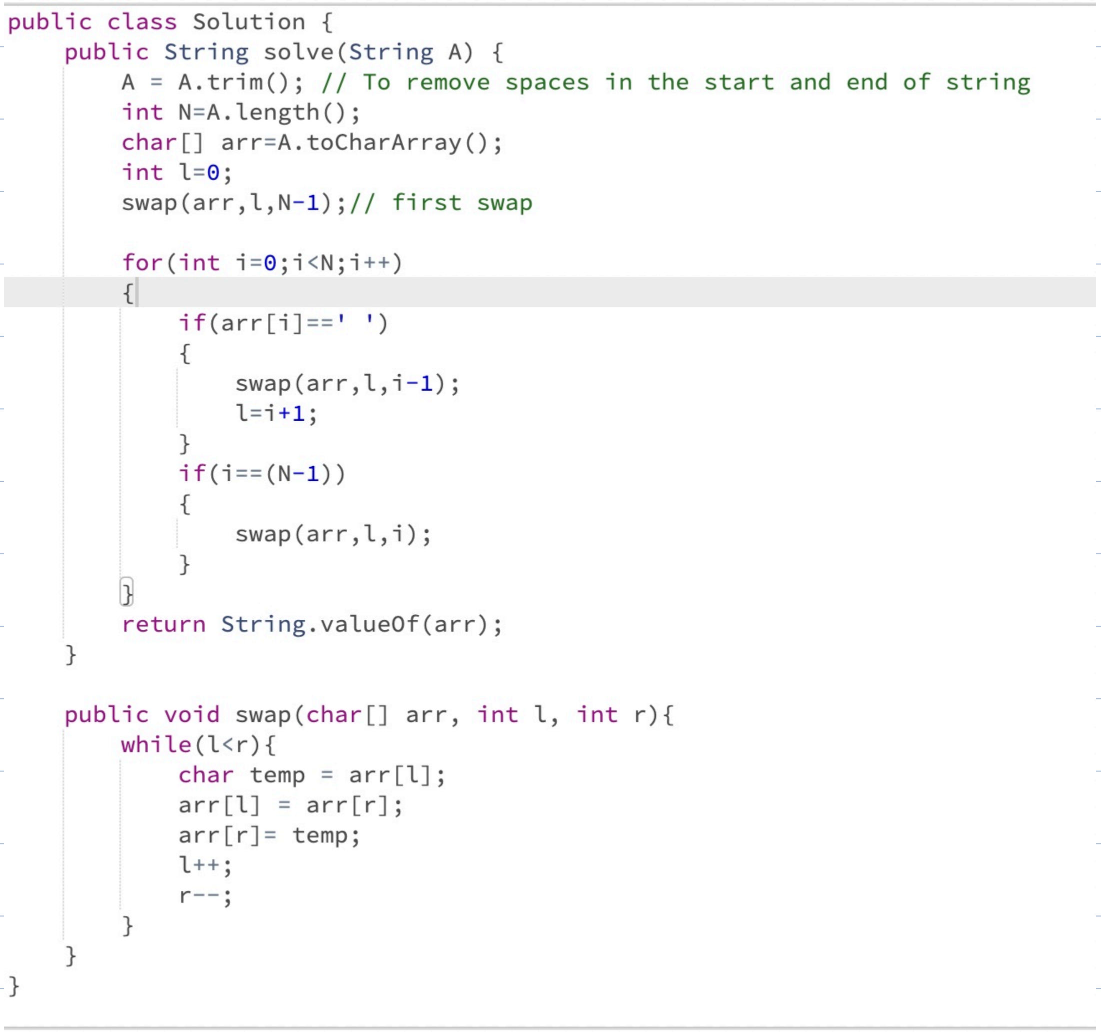
* : Stack Heap : in

int

new



int [] arr = new int [10];



```java
public class Solution {
    public String solve(String A) {
        A = A.trim(); // To remove spaces in the start and end of string
        int N=A.length();
        char[] arr=A.toCharArray();
        int l=0;
        swap(arr,l,N-1);// first swap

        for(int i=0;i<N;i++)
        {
            if(arr[i]==' ')
            {
                swap(arr,l,i-1);
                l=i+1;
            }
            if(i==(N-1))
            {
                swap(arr,l,i);
            }
        }
        return String.valueOf(arr);
    }

    public void swap(char[] arr, int l, int r){
        while(l<r){
            char temp = arr[l];
            arr[l] = arr[r];
            arr[r]= temp;
            l++;
            r--;
        }
    }
}
```