Strings (class 7 PPT)

Sequence of characters
(array)

$\boxed{\text{"data"}}$, $\boxed{\text{"structure"}}$

Operations

1) concatenation of strings

"abc" + "def" + "ghi" ⇒ "abcdefghi" ✓

abdefcgih ✗

2) Trim in string

"__a_bc_d__" trim() a_bc_d

3) Reverse ~~of~~ a string

"abcd" → "dcba"

4) Subsequence of a string

↓

sequence derived by removing zero/more characters of a string without changing the order of remaining elements

substring of a string

↓

contiguous part of a string

abcdefg

ad f, ag, abcdefg

afe ✗ acf ✓

ab, a, abcd, def, g, fg, abcdefg

5) Palindrome strings

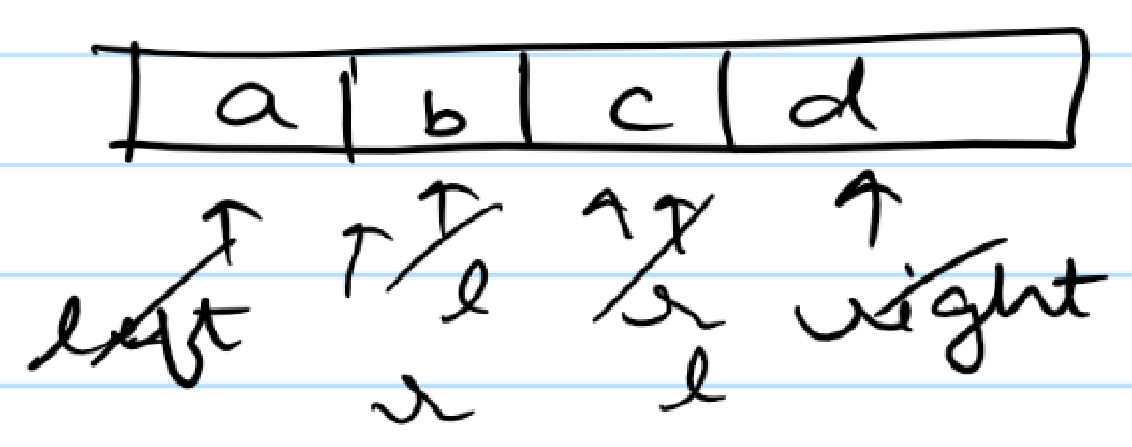
abba → abba

string = reverse(string)

Q1 Reverse a string

1st approach: Inbuilt operations   reverse()

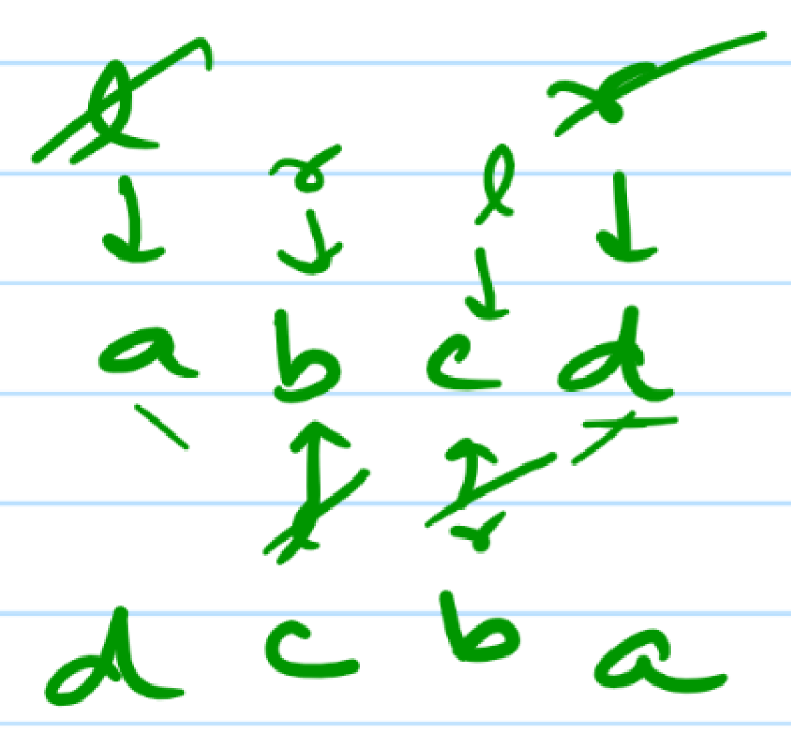
2nd approach: 2 pointer approach



d c b a

string s

```
public void reverse (char[] s)
{
    int left = 0;
    int right = s.length - 1;
    while ( left < right )
    {
        char tmp = s[left];
        s [ left] = s[right];
        s [ right ] = tmp;
          left ++;
          right --;
    }
}
```



$l = 0$
$r = 3$ $(n-1)$

" "

$O(n/2) = O(n)$

TC: $O(n)$
SC: $O(1)$

Q2 First unique character

leetcode   0

HashMap   Key   No. of occurences

Algorithm: 1) Go through the string, create a hashMap & save the no. of times each character occurs in the string.

2) Iterate over the string, use hashmap to see if current character at position i occured 1 time or not.
If it occured 1 time, return i ~~& break~~.

leetcode

l	1
e	~~2~~ 3
t	1
c	1
o	1
d	1

= 0

26 characters

TC: O(N)

SC: O(1)

leleleg 0

hm.

int x = get or Default (c, 0)) + 1

x = 1 x = 0

1) cefc 1+1

HM

~~e~~	~~2~~

c 1

c	~~1~~ 2
e	1
f	1

Q3: Length of last word

Trim ()

abc def ghi

Algorithm: 1) Remove the trailing spaces
2) Start from the last character & keep on incrementing the length till we find a space.
3) Return the length

TC: O(N)

SC: O(1)

Q4: Longest common prefix

"flow", "flower", "flight" , ... n

(0) (1)

$\boxed{fl}$

Brute force : f , fl , flo , ...

Best : flow

LCP

string[i] . indexOf (LCP) == 0

flower . indexOf (flo) = 0

flower. index(flow) = 0

flight . indexOf (flow) -1

flo

flight . indexOf (flo) = -1

fl

flight . indexOf (fl) = 0

$\boxed{\text{Ans} = fl}$

4 leets , 8 leetcode , 4 leet , 5 leeds

$\underbrace{4+8+4+5}$

~~leets~~

lee

leeds . indexOf (leet) -1

leeds indexOf (lee) = 0

$\boxed{\text{Ans} = lee}$

TC : O(n) n : sum of all characters of all strings

SC : O(1)

index of

leetcode . indexOf (~~abc~~ a)

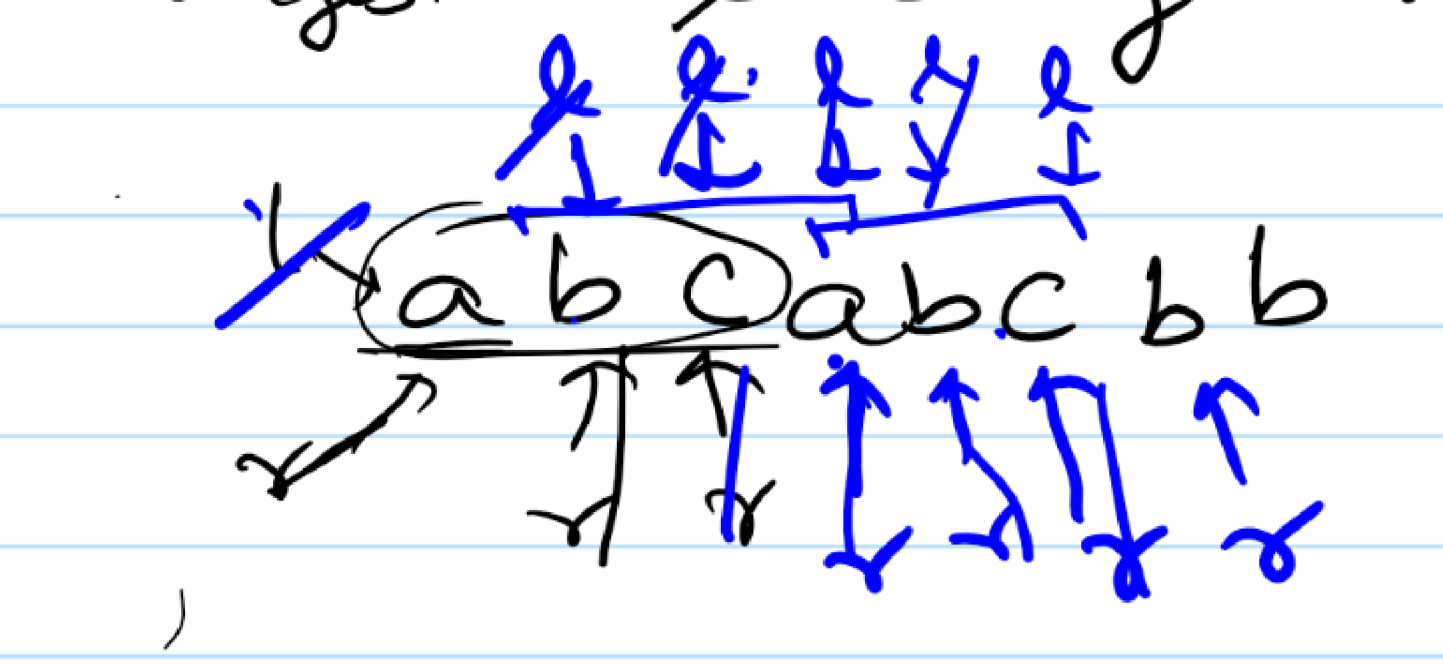
= O(length (leetcode))

string 1 , string 2 , string 3 , string 4

O(length(string2)) + O(length(string3)) + O(length(string4)

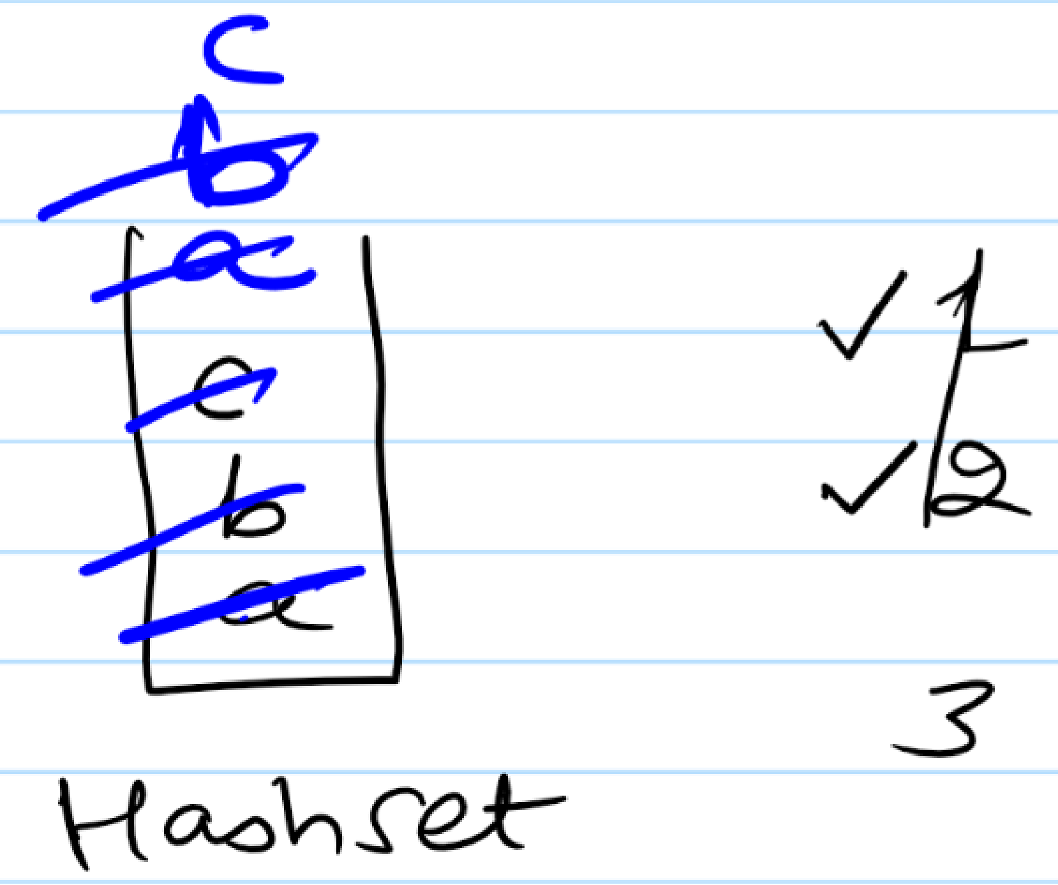
TC: $\boxed{n}$

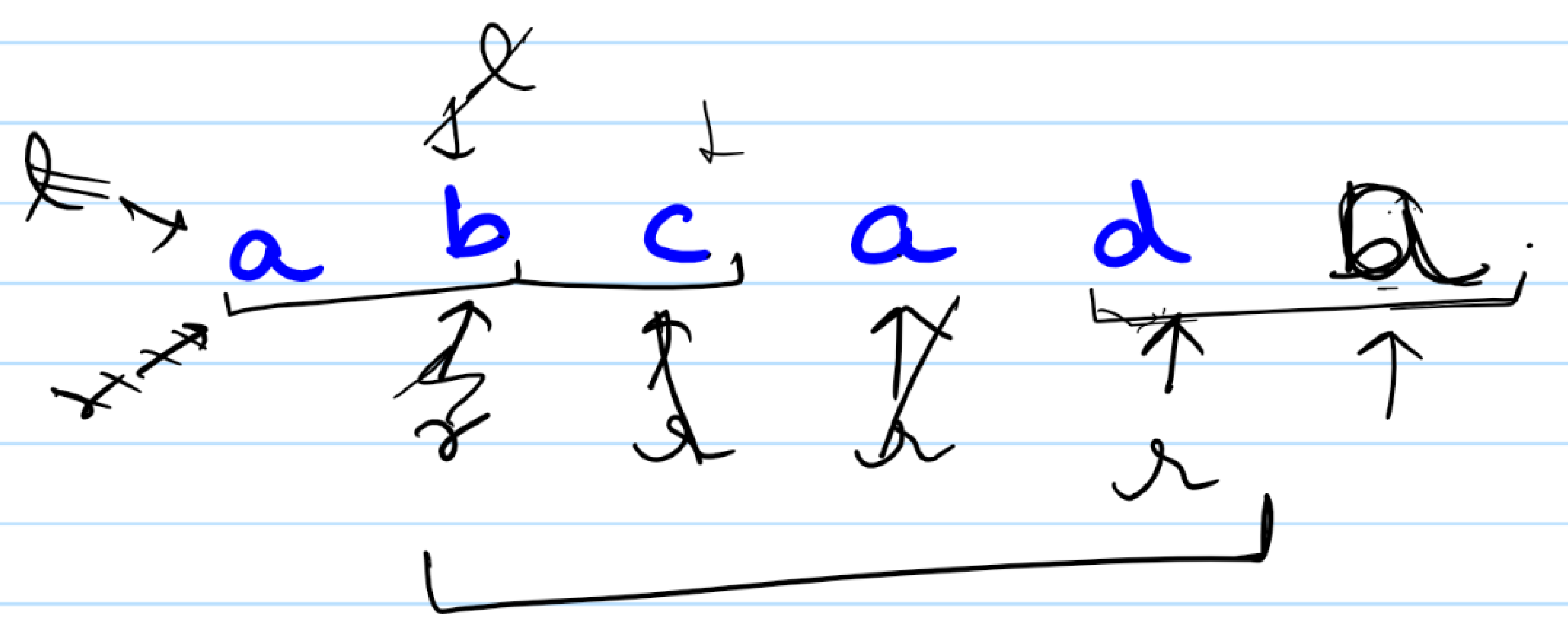
Q5: Longest substring without repeating characters

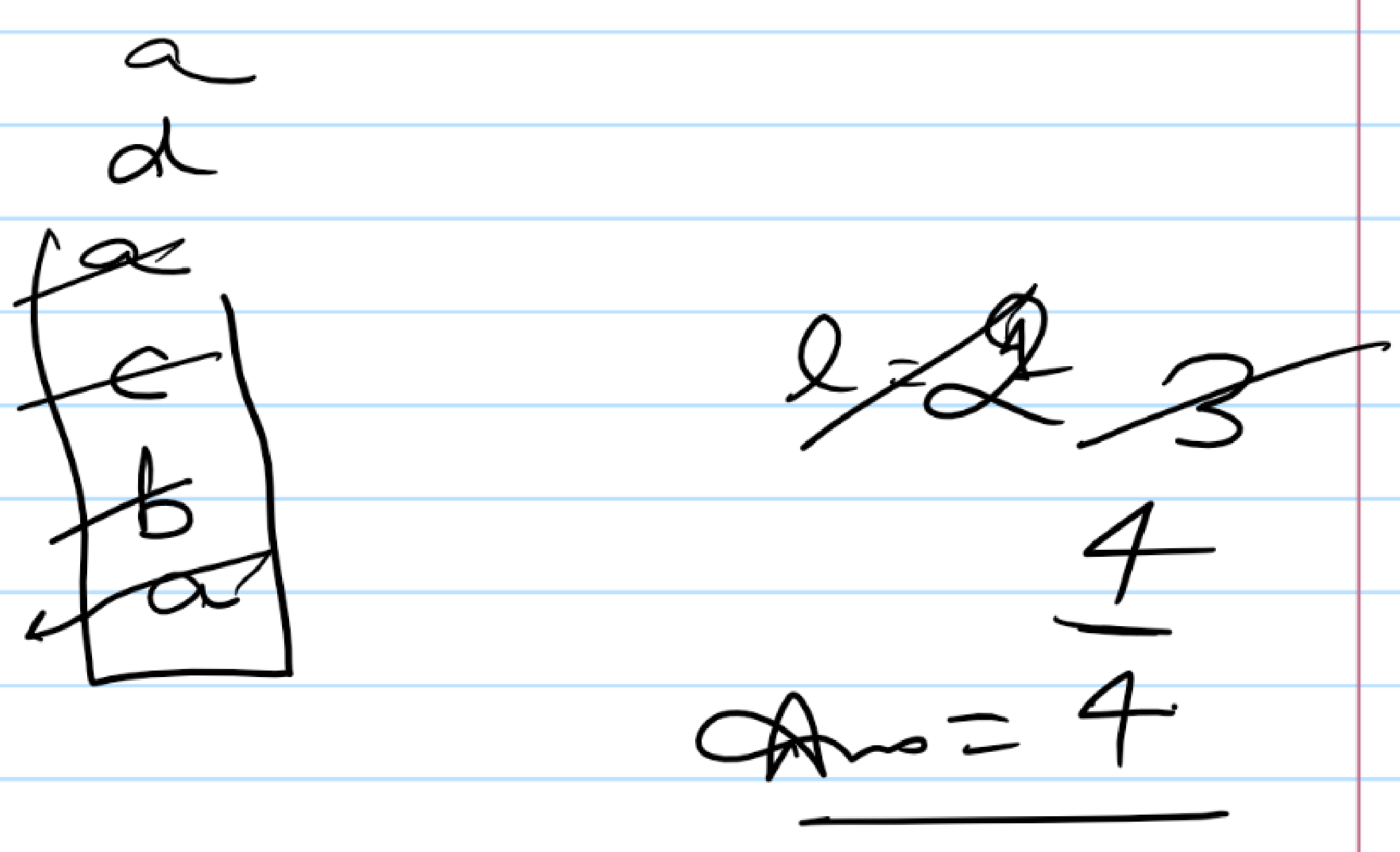


abc

l = 3







SLIDING WINDOW

TC: O(N)

SC: set of smaller case characters → O(1)
26 letters.

Q6: Reverse words in a string

Approach 1:
1) Trim the string
2) split the string with 1 or more spaces
3) Reverse the above list & join by single space.

```
reverse Words ( string s)                "\\s+"
{
  ✓ s = s.trim ();
  ✓ List<String> words = Arrays.asList(s.split("\\s+"));
Collections.reverse(words);
return String.join(" ", words);
}
```

_ _ coding _ _ _ is _ _ _ _ fun _ _ _ _

Trim :→ coding _ ~~_~~ _ is _ ~~_~~ _ fun

list	coding	is	fun
Reverse	fun	is	coding

"fun_is_coding"

TC : $O(n)$

SC : $O(n)$