PEPCODING

PURSUIT OF EXCELLENCE AND PEACE

MODULE

HASHMAP & HEAP

+91 11 4019 4461

PepCoding, 3rd Floor, 15 Vaishali,
Pitampura Opposite Metro Pillar 347,
Above Karur Vysya Bank, New Delhi,
Delhi 110034

www.pepcoding.com

/pepcoding

Valid-Invalid Approach

Q Smallest Window in a string containing all the characters of another string.

What?
Find the length of the smallest window that contains all the characters of other string.

How?
→ We use Valid-Invalid approach
- Make Frequency map of second string.

eg:
S = a b c a a d e d d b c d a a b a c d a c
P = a b b c a d

PMap for P = a→2, b→2, c→1, d→1
= a2b2c1d1

st ↓

a0 b1 c2 a3 d4 e5 d6 d7 b8 c9 c10 d11 a12 a13 b14 a15 c16 d17 a18 c19

↑ end ↑ ↑ ↑ ↑ ↑ ↑ ↑ (not in P, both so don't add)

SMap ⇒ a→X2
b→X2
c→1
d→X1X3

match count = 1 X 2 X 3 X 4 X 5 6.

When match count = characters of P.
Then start moving st.

When character gained by end add that character to hashmap. If current frequency of character (before updation) is less than frequency of that char in PMap then increment match count.

Match count ~~and~~ keeps no. of character matching b/w S & P.

Position at which start is the character that is recently removed from window

st

(II) $a_0\ b_1\ c_2\ a_3\ d_4\ e_5\ e_6\ d_7\ b_8\ c_9\ c_{10}\ d_{11}\ a_{12}\ a_{13}\ b_{14}\ a_{15}\ c_{16}\ d_{17}\ a_{18}\ c_{19}$

end.

map a-~~2~~ 1 b-2 c-1 d3   mc= ~~6~~ 5

at end = (8)

Since we loosed a character of use smpfreq (char) <= pmap(char)

* Remove from start till matchcount remain same.
Since it has changed we will aquire in window.

ws → window st = we = (st+1) = (-1+1) = 0
we = window end = $\boxed{we = end = 8.}$

[when matchcount is same check these & update)

st st

(II) a b c a d e d d $b_8\ c_9\ c_{10}\ d_{11}\ a_{12}\ a_{13}\ b_{14}\ a_{15}\ c_{16}\ d_{17}\ a_{18}\ c_{19}$.

mc=5 mc=5

map= a ~~1~~ 2 b2 c ~~1~~ ~~2~~ 3 d ~~3~~ 4   mc= ~~5~~ 6 at(12)

After start a2 b ~~2~~ 1 c3 d4   mc = 5

$\boxed{ws = 0 , we = 8}$

St = 1 end = 12 (12-1+1) > (8-0+1) so no update.

st

(III) $a_0\ b_1\ c_2\ a_3\ d_4\ e_5\ d_6\ d_7\ b_8\ c_9\ c_{10}\ d_{11}\ a_{12}\ a_{13}\ b_{14}\ a_{15}\ c_{16}\ d_{17}\ a_{18}\ c_{19}$.

map = a ~~2~~ 3 b ~~1~~ 2 c3 d4   mc = ~~5~~ 6.

Start re = a ~~3~~ 2 b ~~2~~ 1 c ~~3~~ 2 d ~~4~~ ~~3~~ ~~2~~ 1   mc = ~~6~~ 5

when start was at 8 end at 14 gap = (14-8+1) = 7

gap < (8-0+1)

8 ws = 8 , we = 14 .

(IV)

st

a b c a d e d d b c c d a a b a c d a c
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19

end

smap = a 1 b 1 c 1 d 1 mc = 5
3 3 4 2

match count remain 5 so no updation

Ans ws = 8 we = 14

String : bccdaab.

(a*) Acquire till matchcount is not equal to characters in p. i.e acquire till valid.

(s) settle → calculate the size of window on the basis of current start and end. if better update.

(r*) Release → Release character till match count remain same. as soon as matchcount decreases settle.

Q Smallest window containing all characters of Yourself (smallest distinct window)

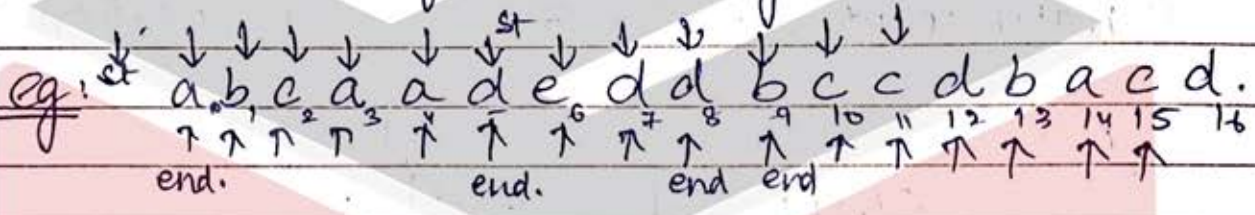
What?

Find the length of smallest window that contains distinct characters of athis string.

How?

Same approach as before.

Make hashset of the string.



(I) map = a1 b1 c1 d1

(map size = set size) So found a window.

hashset <a, b, c, d>.

st = a0 b1 c1 d1

	st	end	end-st
gap =	1	5	(6)
	5	5	(5)
	11	15	(4)

(II) map = a2 c1 d1 b1

st = a2 c1 d3 b1

(III) map = a2 d3 b1 c1

st = a1 d3 b1 c1

(IV) map = d1 b1 c1 a1

st = d0 b1 c2 a1

(V) map = d1 b1 a1 c1

st = d1 b1 a1 c1

Ans 4 size.

dbac

Q Longest Substring with unique characters.

What?
Find the length of the longest substring which has unique character ie no character is repeating.

How?
Valid Invalid approach.
a* → acquire till no duplicacy.
s → compare the length of current string with maxlen.
r → release a character.

Make Hashset for characters.

eg: a b c a b c b
1st 1st
↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑
and end end.

(I) set = <a, b, c> curlen=3 maxlen = 3
After release <b c>

(II) set <a b c> maxlen=3
release <a c b>

(IV) acquire <a c b e> maxlen=4
release <b c e>
ans=4

III acquire <a c b>
release <a b>

Date
DELTA Pg No.

Q Count Substring with unique characters

What?
Count all the substrings with unique characters.

How?
Same as before Just instead of updating maxlength update count by .(end - st -1)

st
eg: a b c a b c e b
0 1 2 3 4 5 6 7
end.

(I) set < ab c > st = -1 settle count += (end-st-1)
end = 3 = (3-(-1)-1)
(Stopped aquired with a) count = 3

Release a0. set = < b.c>

(II) After aquire set < b c a > end = 3 (can't acquire B)
st = 0

settle count += (4-0) = 3+3 = 6

After release < c a >

(III) After aquire < ca b > settle
a count = 6+(5-1-1) = 9
end = 5
st = 1 After release < ab>

Similarly count = 22 Ans = 22.

Q Longest Substring with K Unique Characters

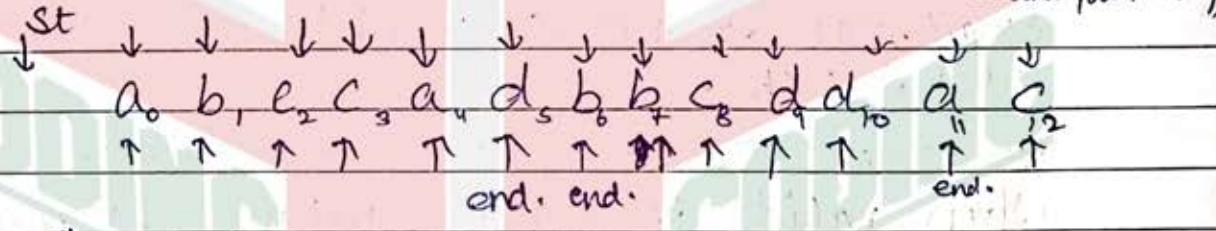
What?
Find the longest substring that has K unique characters.

How?
Same approach as all unique characters here valid till unique character count remains K.
Use hashmap to store character of current subarray

acquire * → acquire till count of unique characters is less than or equal to K. (add to hmap)

settle → Calculate current length and compare with max length. and update if greater.

release * → release till count of unique character is not less than K. (remove/decrease count from hmap)



(I) acquire * till end = 5
settle: count = end - st + 1 = (5 - 1) - 1) = 5 max = 5
release * till st = 2 length remains

(II) acquire * till end = 6
settle = end - st - 1 = (6 - 1 - 1) = 4 max = 5
release * till st = 3

(III) acquire * till end = 8
settle = end - st - 1 = (8 - 3 - 1) = 4 max = 5
release * till st = 4
st = 4 equal value.

(IV) acquire* till end = 11
settle count = (11-4-1) = 6 max = 6
release* till st = 7

(V) acquire* till end = 13
settle count = (13-1-7) = 5 max = 6.
release* till st = 13

Ans = 6

Q Count Substring with K unique characters

What?
Count all the substrings that have K unique characters.

How?
We will use 2 hashmap to store characters of current subarray.
Small map → This map will acquire characters till size is <K
Big map → This map will acquire character when valid ie no. of unique characters is <= K.
Maintain 2 ends end1 & end2 one for each hmap.
acquire* → If no. of unique characters in small map is less than K add current character to both small map & big map. and increment both j1 & j2.
If (j1 < j2) then only acquire in small as characters already in big map.

Count*

When no of unique characters in small is equal to K i.e size of small == K > big.size() == K acquire in big till it has K unique characters. and increment ~~f~~ end2

Settle : Increment count by (end2 - end1 + 1)

remove/release : Release the character from both maps. and Increment count. till not valid to acquire again

eg: st ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓

a0 b1 b2 c3 b4 c5 c6 a7 a8 d9 b10 c11

e1 e2 e1 e2 e1 e2 e1 e2 e1 e2 e1 e2 e1 e2 e1 e2 e1 e2 e1 e2 e1 e2 e1 e2

(I) Acquire* — acquire in small till 3. end1 = ~~4~~

acquire in big till 9 end2 = 9 (this no)

Count = 9 - ~~5~~ + 1 = ~~6~~.

release st = 0 count += (en2 - end1 + 1) = ~~12~~. small = ~~<b2 c~~ <b2>

~~st = 1~~ ~~count += (.~~ big <b3 c3 a2>

(II) Acquire* → acquire in small as end1 < end2.

small till end1 = ~~8~~ 7 <b3 c3 a1>

big till 9 end2 = 9 <b3 c3 a2>

count = 6 + (9 - ~~8~~ 7 + 1) = 8.

release ⇒ till st = 1 count += 8 + (9 - 8 + 1) = 10

st2 = 2 count += 10 + (2) = 12

st = 3 count += 10 + (2) = 14

(count is incremented as can't acquire)

st = 4 again valid.

Date
DELTA Page No. that shows don't include.

(III) Acquire * Acquire in small till end1 = 10
end2 = 10

count = 14 + (10 - 10 + 1) = 15

release and count = st = 5 count = 15 + 1 = 16
st = 6 (valid again)

(IV) Acquire* → end1 = 11
end2 = 11

count = 16 + (11 - 11 + 1) = 17.

release and count st = 7 ⇒ 17 + 1 = 18

(V) Acquire * end1 = 12
end2 = 12

count = 18 + (12 - 12 + 1) = 19 Ans = 19

Q Equivalent Subarrays

What?
Count all subarrays containing all distinct character of given array. (Make Hashset of array)

How?
acquire* → acquire in map till our map.size is less than set.size. (

count → count += (array length - end) when map.size == set.size()

release → Release cond check if setsize == map.size
count increment similar way as above.

st

Eg: arr 1 2 3 4 1 2 4 3
0 1 2 3 4 5 6 7

Set = <1, 2, 3, 4>

(I) acquire → till end = 3
count = (8 - 3) = 5
release st = 0

(II) acquire till end = 4
count += (8 - 4) = 9
release st = 1

(III) acquire till end = 5
count += (8 - 5) = 12
release st = 2

(IV) acquire till end = 7
count += (8 - 7) = 13
release st = 3 count += (8 - 7) = 14
(as map size = set size)

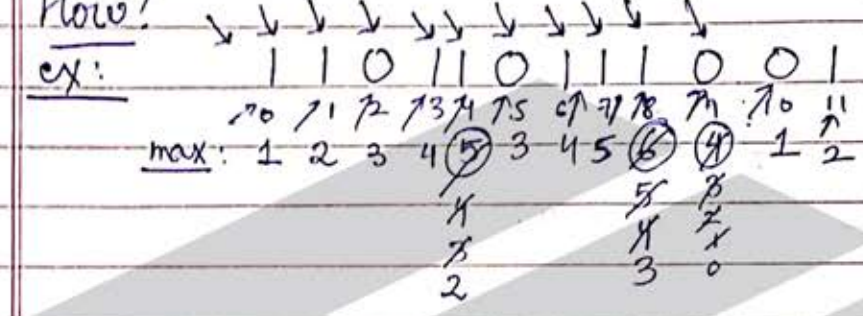
Ans = 14

Q Maximum Consecutive Ones (Ans)

a) find maximum length substring which has maximum consecutive ones where substring can have at most one 0.

How?

ex:



At each index i,

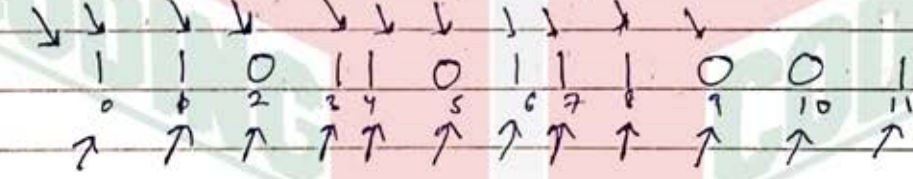
a) acquire till you are valid, i.e; till you have only 1 zero in your substring (a^*)

b) settle, i.e; update max. (s)

c) release till you loose a zero. (r^*)

→ Initially start = -1, end = 0 which means end will now acquire 0th index, and start will now leave first 0th index.

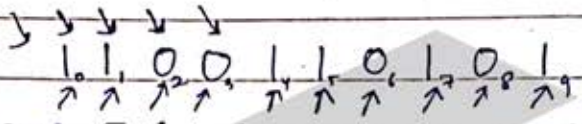
→



0 - 4 : end++ and it acquires, length = 5.
Now end at 5; update max = 5.
Since at 5 value is 0, start ++ till you release 0.
Now start at 2, length = 3.
Now end acquires till 8, and is at 9.
Update max = 6 and start leaves till it gets a 0 at 5.
Now end acquires 9th index, start leaves till 9th and this works till you reach end of array.

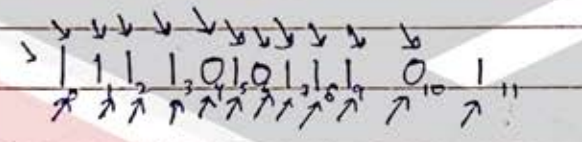
b) This question has two more variations, one where k zeroes are allowed and k is given, other where k=0, i.e., only 1's substring.

ex: k = 2.



ans = 6 (0–5 / 4–9)

ex: k = 0



ans = 4 (0–3)

Q. Binary strings with substrings representing 1 to N (AnS).

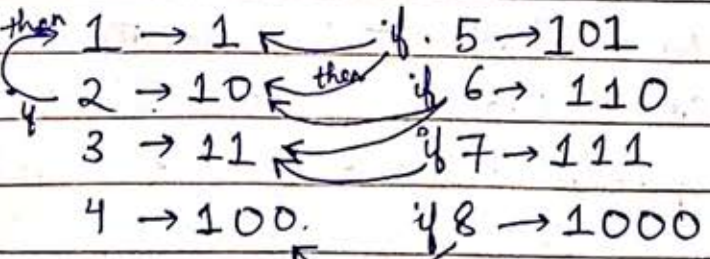
What?

You are given a binary string, return true if the string can represent all numbers from 1 to N, both inclusive. N will be given.

How?

For a binary string to represent 1 to N, if it represents $\frac{N+1}{2}$ to N, then it represents all 1 to N.

This is because:



If 8 is present, 4 is also present. (if N is present, N/2 is also present)

Use start-end approach, keep start = -1 and end = 0 initially.
Create a hashset of values $\frac{N}{2}+1$ to N.
→ acquire till you are invalid, i.e., binary substring represents a number <= N (a*), if greater, stop.
→ release till the substring is valid again.

ex: N = 8 | 5 | 6 | 7 | 8 | hashset

I) 0 1 0 1 0 1 1 0 1 1 1 0 0 0 1 0

number 0 1 2 5 10 → 10 > 8, release till valid.

Cut 5 in hashset.

II) 0 1 0 1 0 1 1 0 1 1 1 0 0 0 1 0

0 1 2 5 11 > 8, release.

III) 0 1 0 1 0 1 1 0 1 1 1 0 0 0 1 0

0 1 3 6 13 > 8, release

Cut 6 in hashset

IV) 0 1 0 1 0 1 1 0 1 1 1 0 0 0 1 0

1 2 5 11 > 8, release.

V) 0 1 0 1 0 1 1 0 1 1 1 0 0 0 1 0

0 1 3 7 13 > 8, release.

Cut 7 in hashset

VI) 0 1 0 1 0 1 1 0 1 1 1 0 0 0 1 0

1 3 6 12 > 8, release.

(2 release shown here)

VII) 0 1 0 1 0 1 1 0 1 1 1 0 0 0 1 0

1 2 4 8 → Cut 8 in hashset, return true.

Q. Find subarray with given sum. (Ans)

What?
Find a subarray with sum to a target.

How?
Use start - end approach,
→ acquire till your sum is <= target.
→ if your sum == N, settle, i.e. return answer.
→ release till your sum is <= target.

ex: [2₀, 8₁, 3₂, 4₃, 9₄, 5₅]

start = -1 & end = 0 initially.

target = 16

I) start = -1 end = ~~0~~ ~~1~~ ~~2~~ 3
array = [2, 8, 3, 4]
sum = ~~2~~ ~~10~~ ~~15~~ (17) → invalid, so release now.

II) start = ~~-1~~ 0 end = 3.
array = [~~2~~, 8, 3, 4]
sum = ~~17~~ 15 → valid, so acquire again.

III) start = 0 end = ~~3~~ 4
array = [~~2~~, 8, 3, 4, 9]
sum = ~~15~~ 24 → invalid, release now.

IV) start = ~~0~~ 1 end = 4.
array = [~~2~~, ~~8~~, 3, 4, 9]
sum = ~~24~~ [16] → sum == N.

return [3, 4, 9] as answer.

// Similiar question:
Q. Find subarray with product less than K.
→ Just calculate product instead of sum.

Anagrams Based

Q. find all anagrams

What?

You are given a string and a pattern, return count of all anagrams of the pattern in the string.

How?

Use Match Count strategy to do this (MC) where match count tells how many characters are being matched in current string and pattern string.

→ a* : with each acquire, if frequency of element in current map is less than that in pattern map, then MC++, else if it is equal or greater, MC does not change.

→ a : If MC == pattern.length(), count of anagram is increased by 1.

→ a* : Release a character after checking that if its frequency in current map was less or equal to that in pattern map, then MC--, else MC does not change.

ex: string : b a d c c d a b b a d a c c a b c d b

pattern : a b b c d

pattern map : | a→1 | b→2 | c→1 | d→1 |

fmap

-1 b a

1) b a d c c d a b b a d a c c a b c d b
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18

Current map:

| ~~b→1~~ | a→1 | d→1 | c→2 |

b_0 a d_2 c_3 ● → when b_0 was released.

MC = 1 2 3 4 3 When c_4 is acquired, current map size == fmap size, so release as MC not = 5.

2) b a d c c d a b b a d a c c a b c d b

MC 1 2 3 4 ~~3~~

current map | a→1 | c→2 | d→2 |

MC = 2

3) b a d c c d a b b a d a c c a b c d b

| a→1 | c→2 | d→~~2~~ 1 |

MC = ~~2~~ 3

release d. ⇒ MC = 3.

4) b a d c c d a b b a d a c c a b c d b

| a→1 | b→1 | c→~~2~~ 1 | d→1 |

MC = ~~3~~ 4

Release c ⇒ MC = 4.

~~5) acquire~~

5) b a d c c d a b b a d a c c a b c d b

| a→1 | b→2 | c→1 | d→1 |

⇒ MC = ~~4~~ (5) ✓ count = 1.

6) b a d c c d a b b a d a c c a b c d b

(0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18)

a→~~1~~	b→~~2~~	c→~~1~~	d→~~1~~
a→~~2~~	b→~~1~~	c→~~1~~	d→~~1~~
~~1~~	~~0~~	~~1~~	~~0~~
~~2~~	~~1~~	~~3~~	1
~~1~~	2	~~2~~	
~~2~~		1	
1			

MC = ~~5~~ ~~4~~ (release c_4)

~~3~~ (release d_5) ~~4~~ (acquired$_{10}$)

~~3~~ (r b_7), ~~4~~ (a c_{12}), ~~3~~ (r b_8)

~~2~~ (a d_{10}), ~~3~~ (a b_{15})

~~4~~ (a d_{17}), (5) (a b_{18})

→ count = 2.

∴ We have 2 anagrams of abbcd,
which are cdabb and abcdb
4 5 6 7 8 14 15 16 17 18

Q. K-anagrams.

What?

Tell if 2 strings are K-anagrams of each other.

How?

ex: s1: d b a a b c d a K=3.
s2: a b c d a b c d.

1) Make fmap of s1.

| d → 2 | b → 2 | a → 3 | c → 1 |

2) Update fmap using s2 such that if element exists in fmap, frequency --.

After update: | d → 0 | b → 0 | a → 1 | c → -1 |

frequencies > 0 denote extra characters in s1, and < 0 denote extra characters in s2.

3) Sum either all frequencies > 0 or < 0 and if sum <= K, return true.

Here, sum = 1 where 1 < 3, ∴ these strings are K-anagrams.

Q. Are two arrays equal?

What?

Return if 2 strings are same / 2 arrays are same.

How? K anagrams where K = 0.

Q. Anagram Mappings

What?

You are given 2 strings which are anagrams of each other, return an array which tells the indexes of a character in string2 present in string1. Randomizing is done as mapping where P[i] = j where character at i in string1 is at j in string2.

How? # Return i indexes corresponding to each j array.

1. Without duplicacy: Make hashmap <Char, Int> where each character stores index of it in string1.
→ Now create an array by iterating in string2, for each character, find it in hashmap and add its index in array.

ex: s1: f c d h a e b g
0 1 2 3 4 5 6 7

s2: c a f d h b g e
0 1 2 3 4 5 6 7.

HashMap: [f→0 d→2 a→4 b→6 / c→1 h→3 e→5 g→7]

Iterate: s2: c a f d h b g e
ans: [1 4 0 2 3 6 7 5]
0 1 2 3 4 5 6 7

2. With duplicacy: Make hashmap of character vs. either arrayList to store multiple indexes and always add in last in List to avoid remove first operation, or use a priority queue / a linked list.

ex: s1: f d b a c e d g f a c g b h
0 1 2 3 4 5 6 7 8 9 10 11 12 13

s2: a b b g e f d f g h c c a d
0 1 2 3 4 5 6 7 8 9 10 11 12 13

HMAP: ~~f → 8, 0~~

char vs Priority Queue (ascending order).

f → 0, 8
d → 1, 6
b → 2, 12
a → 3, 9
c → 4, 10
e → 5
g → 7, 11
h → 13

Now iterate:

s2: a b b g e f d f g h c c a d
0 1 2 3 4 5 6 7 8 9 10 11 12 13

ans: [3 2 12 7 5 0 1 8 11 13 4 10 9 6]
0 1 2 3 4 5 6 7 8 9 10 11 12 13
→

if you use arraylist, and do add last while adding indexes & remove last while removing indexes, then just make the ans array from right to left (←)

Q. Valid Anagram

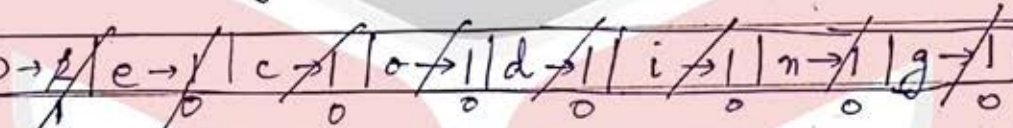
What?

You are given 2 strings S and T, return 'yes' if T is an anagram of S or 'no' if it is not.

How?

Simply create a fmap of S, and if any character of T is not in fmap, or frequency of any character in S is >0 after cancelling with character frequencies in T, return false.

ex: S: pepcoding
T: dongpcpie



returns true as all 0 and all characters present.

Q. Group Anagrams

What?

Given an array of strings, return a 2D list which has each list as a group of anagrams. Case-sensitive anagrams are not grouped together, like Abc and abc.

How?

We keep a HashMap of HashMap vs ArrayList.

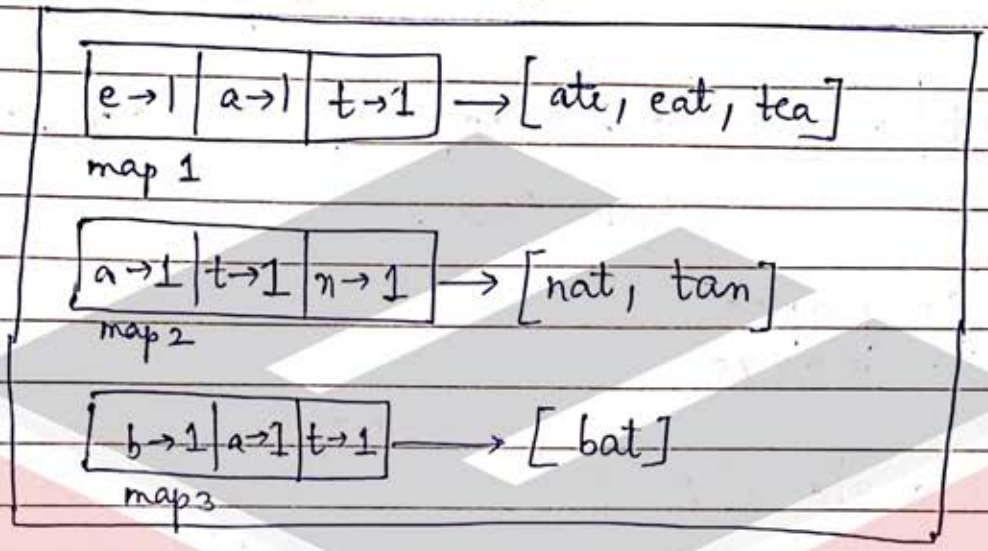
HashMap<Char, Int> which will be frequency map for each string.

For each fmap, it will have all words having the same fmap.

ex: [cat, tea, tan, ate, nat, bat].

HM<HM<Char, Int>, ArrayList> map.

map



ex // Make fmap for ate, check if fmap already exists in map, if yes, add to its list ate, else create a new entry in map for fmap vs list having ate.

Q. Isomorphic strings

What?

Given two strings S and T, tell if they are isomorphic. isomorphic strings are those in which we can map the character of one string to characters in other string.

How? Create two maps, and at any given index i, map1[S.charAt(i)] = T.charAt[i] and map2[T.charAt(i)] = S.charAt[i].
If it already exists and the character to be mapped with is different than what is stored, return false.

ex1 S: paper
T: title.

map1

p→t	a→i	e→l	r→e

map 2

t→p	i→a	l→e	e→r

returns true.

ex2 S: paper
T: xxxx.

map1

p→x	a→x	e→x	r→x

map 2

x→p

a ? fails.

// this example explains why checking & creating both maps is important.

Q. Word Pattern

What?

You will be given pattern string and words string, check if words follow exact order as pattern string.

How? # same as isomorphic strings above, just characters are mapped to strings in map1, and strings to characters in map2.

ex: S: abab W: pep code pep code

map1

a→pep	b→code

map2

pep→a	code→b

returns true.

Prefix Sum Based Questions

Q Subarray Range with given target (count subarray with sum K)

What?

Count all the subarrays of a array that has sum equal to given target.

How?

→ Make a Hashmap that stores sum (prefix sum) as key and count of occurence of that sum as value.

FMAP → PS V Count.

target = 4

eg:

0	2	3	-1	4	-5	1	6	-4	3	-1	4
0	2	5	4	8	3	4	10	6	9	8	12

Initially added: 0 → 1

2 → 1

5 → 1

4 → X2

8 → X2

3 → 1

10-1

6-1

9-1

12-1.

4-4=0 present

So count += 1

Total count = ~~0~~ 1 ~~2~~ ~~3~~ 4 ~~5~~ ~~7~~ 9

at(4) at(8) at(4) at(6) at(9)

Ans = 9

At each sum check if (sum - target) is present or not in hashmap. If present then increase Total count with frequency of (sum - target).

* Increase count ~~then~~ before frequency update.

Page No.
Date

Q Subarray With Zero Sum

What?
count all the subarrays whose sum is equal to zero.

How?
* same strategy as before here target = 0

eg: 2 3 -2 -1 -2 4 -2 -2
0 2 5 3 2 4 2 0
(+1) (+2) (+2)

0-2 5-1 4-1 count = 1 2 3 5 7
2-2 3-1

check if same prefixsum is in map or not if present update count.

Q Subarray With Sum Divisible by K

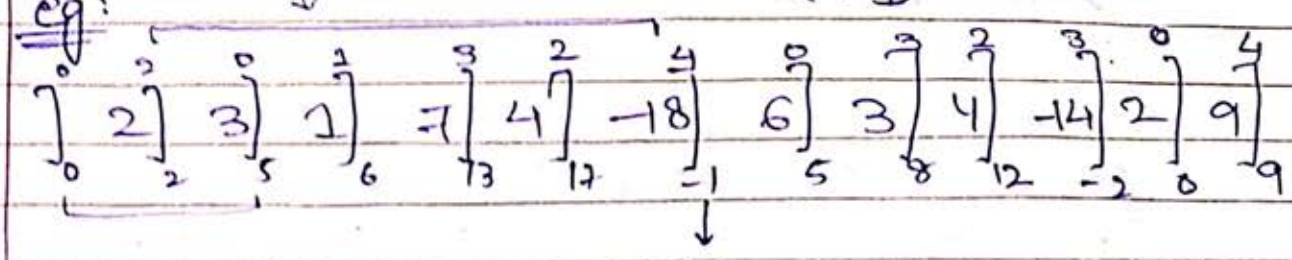
What?
count all subarrays whose sum is divisible by K.

How?
same strategy as before with map.

Here map will store (prefixsum % k.) as key and value will be count of this remainder.

Page No.
Date

eg: K = 5



0 → X X 4
1 → 1
2 → X X 3
3 → X X 3
4 → X 2

Count = ϕ X X 4 5 X X 17 13
= 13

Here when PS = -1
Key = (-1 % 5) + 5
= -1 + 5
= 4

* In case of negative sum
Key = (prefixSum % K) + K

* At each PS (remainder) check if already present or not. If present then increment count by frequency.

* when no. (remainder) are repeating it means all the numbers in between has sum divisible by K

Q Count subarrays with ~~sum~~ Equal 0's & 1's

What?
count all such subarrays in a array of 0's 2's where zero & 1's are equal.

How?
since zero & 1's should be equal then we can do this by replacing all 0's with -1's and apply 0 sum subarray strategy

When 0 & 1 are equal then only sum will be zero.

eg: [1 0 1 0 1 1 0]
idx: 0 1 2 3 4 5 6
prefix sum: 1 0 1 0 1 2 1

considered = -1

0 → X 2
1 → X X 3
2 → 1

Ans = 9

count = 1 + 2 + 4 + 6
(at idx 1) (at idx 2) (idx 3) (idx 4)
9
(idx 6)

Q Binary Subarray With Sum

Same Question as subarray with sum equal to target here just array is binary. Use same strategy.

target = 2

eg: [0 1 1 0 1 1 0 1 0 1 1 0]
idx: 0 1 2 3 4 5 6 7 8

0 → X 2
1 → X
2 → X 2

3-1
4 → X 2

6-1
7 → X 2

5 → X 2

count = 1 + 4 + 5 + 7 + 9 + 10 + 12 + 14 + 16

Ans = 16

Q Count Substrings with equal no of 0's, 1's & 2's

What?
Count the no. of substrings with equal no of zeroes & one's & twos.

How?

- Make Hashmap where key is a string made like (count1 - count0) + '*' + (count2 - count1).
  - '*' → any character

eg:

	1	0	2	1	0	0	2	1	1	
count0	0	0	1	1	1	2	3	3	3	3 → when zero occur increment zero count & similarly with 1 & 2
count1	0	1	1	1	2	2	2	2	3	4
count2	0	0	0	1	1	1	1	2	2	2
	0*0	1*1	0*-1	0*0	1*-1	0*1	-1*-1	-1*0	0*1	-1*2

① ② ③

(2-1) * (1-2)

1*-1

HMAP

0*0 = ~~1~~ 2

1*-1 = ~~1~~ 2

0*-1 = ~~1~~ ~~2~~ 3

-1*-1 = 1

-1*0 = 1

-1*2 = 1

count = ~~1~~ 2 3 5

at(2) at(1) at(0) at(1)

Ans = 5

When count was 2 of 0*-1 it means 2 of 0*-1 can make combination with third 0*-1

Why?

When delta of counts (string) is repeated again it means that no. of increments in 0's is same to 1's and no. of increments in 1's is same to 2's. So this results in substring with equal no. of zero's & 1's.

Page No.
Date

Q Longest substring with equal no. of 0's & 1's.

What?
Find the longest substring which has equal no. of 0's & 1's

How?
using same strategy as count substring / subarray with equal no. of zeroes & 1's.
Consider 0 as -1.
Here we will check the length b/w first occurrence of sum and current occurrence

eg: 1 0 1 0 1 1 1 0

0 1 2 3 4 5 6 7

sum: 1 0 1 0 1 2 3 2

Map
0 → -1
1 → 0
2 → 5
3 → 6

sum vs index

gap = 1-(-1) = 2 | maxlen = 0 2 4 (at (1) at 3)

gap = 3-(-1) = 4

at index 4 gap = 4-0 = 4 so no updation

Ans = 4

Q Longest Subarray with sum equal to K.

What?
Find the longest subarray whose sum is equal to K.

How?
Same strategy as count subarray with sum equal to target. Just save the first occurence of prefix sum and calculate gap instead of count.

target = 4

eg: 2 3 -1 4 -5 1 6 -4 3 -1 4

0	2	5	4	8	3	9	10	6	9	8	12
				gap=1		gap=6		gap=7	gap=6	gap=7	gap=7

0→1 8-3
2→0 3→4
5→1 10-6
4-2 6→7
9-8 12-10

4-4 (ps-K) present
gap = i - map.get(ps-K)
= 2-(-1) = 3

maxlen = ~~3~~ ~~6~~ 7

Ans = 7

MAP PS vs index.

Q longest subarray with sum divisible by K

Same strategy as above Just store sum as prefix sum modulus K. And for negative sum (prefix sum % K) + K. Calculate the Gap.

K=5

eg: 2 3 1 7 4 -18 6 3 4 -14 2 9

0	2	5	6	13	17	-1	5	8	12	7	10	9
		gap=2			gap=4		gap=5	gap=4	gap=8		gap=11	

PS vs index
0→-1
2→0
1→2
3→3
4→5

maxlen = ~~1~~ ~~4~~ ~~5~~ ~~8~~ 11 Ans = 11

gap = [i - map.get(ps)]

Date
DELTA Pg No.

Q **Largest Subarray with zero Sum**

Same as largest subarray with sum equal to target here sum is zero (target=0). Solve similarly.

Q **Smallest Subarrey with all occurrences of most Frequent Element**

what?

Find the smallest subarray of the array which also has x as the maximum frequency element.

How?

Make a pair class that stores 3 properties for each element.

Pair → frequency (frequency of element)
→ start (start of subarray)
→ end. (end of subarray)

~~Initially~~ calculate the frequency and make a pair for each element and store in hashmap.

eg: 4^0 2^1 3^2 1^3 2^4 1^5 4^6 2^7 4^8 3^9 1^{10}

element vs Pair ob (freq, st, end)

4 → (~~0~~ 1, 0, ~~6~~ 8)
2 → (~~1~~ 1, 1, ~~7~~ 7)
3 → (1, 2, ~~9~~)
1 → (~~3~~, 3, ~~5~~) 10 → no change as gap
gap small so → change

bfv	bf	bs	bend
4	1	0	0
2	2	1	4
1	2	3	5
2	3	1	7

Ans = 2 3 1 2 1 4 2

On first occurrence frequency is set to 1 and st & end is index of element. Add this pair to hashmap.

When frequency increases check for bf (best frequency) if gap current frequency greater than bf. update best frequency value (bfv), best start (bs), best end (bend)

When frequency same as bf check if current subarray is smaller than best subarray if yes than update (gap = end - st + 1). the bf, bfv, bs, bend.

subarray b/w bs & bend is smallest subarray.
length = (bend - bs + 1) = (7 - 1 + 1) = 7

Q Count of substrings containing k ones.

What?
count all the substrings in a binary array that contains k one's

How?
Solve in same strategy as count subarray. with sum equal to target. Here target will be k
target = k (Just array is binary else same)

HashMap Questions

Q. Rat in a Jungle

What?

In a forest, each rat has some colour. Some subset of rats tell you how many other rats have same colour as them.
Return minimum number of rabbits that could be in a forest.

How?

ex: [2, 2, 2, 2, 4, 4, 4, 6, 6, 6, 3, 3, 3, 3, 3, 3, 7, 7].

Create a fmap where map[a[i]] += 1.

i.e; fmap: 2 → 4. ✓ ✓ ✓ ✓ _ _ = 6
4 → 3 ✓ ✓ ✓ _ _ _ = 5
6 → 3 ✓ ✓ ✓ _ _ _ = 7
3 → 6 ✓ ✓ ✓ ✓ ✓ ✓ _ _ = 8
7 → 2 ✓ ✓ _ _ _ _ _ _ = 8
= 34 rabbits

Why?

At a given point, fmap <key, value> pair says that: value number of ~~people~~ rabbit said that these were key number of ~~people~~ rabbits more like me.

ex: 2 → 4 : 4 rabbits said that there are 2 more rabbits like them.

→ If key >= value, ans += key + 1.
else ans += (key + 1) / nor where nor is numbers of rabbits calculated as ceil((value * 1.0) / (key + 1)).

ex: 2 < 4, ans += [ceil(4/3) * (key + 1)] = 2 * 3 = 6.

Q. Double Pair Array.

What?

Given an array of integers with even length, return true iff it is possible to reorder it such that $A[2*i+1] = 2*A[2*i]$ for every $0 <= i < len(A)/2$.

How? // Arrange such that every odd index is its previous even index elements twice.

ex: A = [-96, -96, -72, -48, -48, -48, -36, -24, -24, -12, 0, 0, 2, 2, 3, 3, 4, 4, 4, 6, 6, 6, 8, 12]

to be considered

1) For each of these, first we create a fmap:

fmap

[-96 → 2 -72 → 1 -48 → 3 -36 → 1 -24 → 2
  -12 → 1 0 → 2 2 → 2 3 → 2 4 → 3 6 → 3
  8 → 1 12 → 1]

2) Sort the array A.

3) Now, for each element, these things are considered:

a) If fmap[A[i]] == 0, continue.

b) If A[i] == 0 and fmap[A[i]] % 2 != 0, return false.

→ Because, 0 can occur only even times for creating a double pair array.

c) If fmap.containsKey(A[i] * 2) && fmap[A[i] * 2] < fmap[A[i]], return false.

→ Because, if we have 2, 2, 2, 4, 4, ... then 2, 4, 2, 4, ... is possible. Third 2 has no more (2*2) i.e. 4 element left.

d) If above all are checked, then simply make fmap[A[i]] -= minv and fmap[A[i]*2] -= minv where minv = fmap[A[i], fmap[A[i]*2);

Date
DELTA Pg No.

exsolved : for sorted array :

a) -96 : A[i] = -96 , fmap[A[i]] = 2
A[i]*2 = -48 , fmap[A[i]*2] = 3
⇒ [-96, -48, -96, -48]
0 1 2 3

fmap[A[i]] = 0
fmap[A[i]*2] = 1

b) -72 : [-72, -36]
4 5

c) -48 : [-48, -24]
6 7

d) -24 : [-24, -12]
8 9

e) 0 : [0, 0]
10 11

f) 2 : [2, 4, 2, 4]
12 13 14 15

g) 3 : [3, 6, 3, 6]
16 17 18 19

h) 4 : [4, 8]
20 21

i) 6 : [6, 12]
22 23

∴ The array is : [-96, -48, -96, -48, -72, -36, -48, -24, -24, -12, 0, 0, 2, 4, 2, 4, 3, 6, 3, 6, 4, 8, 6, 12]

* Also check that if their is a negative odd number, then return false [as no double pair array can be formed using -55.]

Date
DELTA Pg No.

Q. Is Frequency Equal

What?

Given a string with lowercase alphabets, check if its possible to remove at most one character from this string in such a way that frequency of each distinct character is same in string.

How?

Consider 3 cases:

a) a2 b2 c2 d2 : when all have same frequency.
b) a2 b3 c2 d2 : when 1 has extra character.
c) a3 b3 c3 d1 : when 1 character is extra.

ex: x x x y y z z.

fmap: x : 3
y : 2
z : 2.

↙

* Calculate max and min frequency (maxf, minf) and elements with maxf & minf as maxc & minc.

Here, maxf = 3, minf = 2, minc = 2, maxc = 1.

→ If (minf == maxf)
return true

→ If (maxc == 1 && minc == fmap.size() - 1
&& maxf - minf == 1)
return true

// Answer.

→ If (maxc == fmap.size() - 1 && minc == 1 &&
maxf == 1)
return true.

→ else
return false.

Q. Check AP Sequence

What?

Given an array of N integers, check whether AP can be formed using all these elements?

How?

ex: 0, 12, 4, 8.

1) Make a hashset. [0|12|4|8]
2) Calculate 'd' of AP as. $d = (max - min)/(array\ size - 1)$
   $\Rightarrow d = (12-0)/3$
   $= 4$
3) Now check if $min + d*i$ exists in set or not.

min = 0, d = 4.

1) $min + d*0 = 0+0 = 0$ ✓
2) $0 + 4*1 = 4$ ✓
3) $0 + 4*2 = 8$ ✓
4) $0 + 4*3 = 12$ ✓.

All elements of hashset used, return true, else false.

Q. Pair Sum Divisibility.

What?

Given an array of N elements and number K, check whether array can be divided into pairs such that sum of every pair is divisible by K.

How ?

1) for each element in array, store element % K in hashmap. If element % K is negative, store K + element % K.
2) Check following cases: (Loop in keyset)
   a) If map has a key = 0, then if map[0] % 2 != 0, return false.
   b) e If map.containsKey(K - val) == false, return false.
      [val is each element in keyset, or a key of hashmap.]
   c) e If K - val == val, check if map[val] is even, else return false.
   d) e If map[K - ele] != map[ele], return false.

ex: N = 4, A = {9, 7, 5, 3}, K = 6.

Hashmap: 3 → 2.
1 → 1
5 → 1.

val : 3 , 6 - 3 == 3, and map[3] = 2 which is even. ✓
val : 1 , 6 - 1 = 5, and map[1] == map[5] so return true.

Date
DELTA Pg No.

Q. Max points on a line

What?

Given n points on a 2D plane, find maximum number of points that lie on same line.

How?

* Slope formula : $\frac{y_2 - y_1}{x_2 - x_1}$
* Create a gcd function which reduces all points as dy = dy/gcd and dx = dx/gcd.
* Store for this (dy, dx) pair, in a hashmap as frequency map, count of all points with this reduced (dy, dx) pair.
  i.e: (dy, dx) → count of points.
* Return the max count possible for any such pair.

N = 6

ex: [(1,1), (3,2), (5,3), (4,1), (2,3), (1,4)]

1) (1,1) , (3,2).
   dy = 1 dx = 2 gcd = 1 ⇒ (1,2)
2) (1,1) , (5,3)
   dy = 2 dx = 4 gcd = 2 ⇒ (1,2)
3) (1,1) , (4,1)
   dy = 0 dx = 3 gcd = 3 ⇒ (0,1)
4) (1,1) , (2,3)
   dy = 2 dx = 1 gcd = 1 ⇒ (2,1)
5) (1,1) , (1,4)
   dy = 3 dx = 0 gcd = 3 ⇒ (1,0)
6) (3,2) , (5,3)
   dy = 1 dx = 2 gcd = 1 ⇒ (1,2)

map for (1,1)

(1,2)	1
(0,1)	1
(2,1)	1
(1,0)	1

max here = 2 + 1 = 3 (I)

(+1 because (1,1) is also on this dy, dx)

Date
DELTA Pg No.

7) (3,2) , (4,1)
dy = -1 dx = 1 gcd = -1 ⇒ (1,-1)
8) (3,2) , (2,3)
dy = 1 dx = -1 gcd = 1 ⇒ (1,-1)
9) (3,2) , (1,4)
dy = 2 dx = -2 gcd = 2 ⇒ (1,-1)
10) (5,3) , (4,1)
dy = -2 dx = -1 gcd = -1 ⇒ (2,1)
11) (5,3) , (2,3)
dy = 0 dx = -3 gcd = -3 ⇒ (0,1)
12) (5,3) , (1,4)
dy = 1 dx = -4 gcd = 1 ⇒ (1,-4)
13) (4,1) , (2,3)
dy = 2 dx = -2 gcd = 2 ⇒ (1,-1)
14) (4,1) , (1,4)
dy = 3 dx = -3 gcd = 3 ⇒ (1,-1)
15) (2,3) , (1,4)
dy = 1 dx = -1 gcd = 1 ⇒ (1,-1)

map for (3,2)
(1,2) → 1
(1,-1) → ~~1~~ 2 3
max = 4.

map for (II)
(5,3)
(2,1) → 1
(0,1) → 1
(1,-4) → 1
max = 2 (III)

map for (4,1)
(1,-1) → ~~1~~ 2
max = 3 (IV)

map for (2,3)
(1,-1) → 1
max = 2 (V)

Using all maxs for each point,
(I) = 3 , (II) = 4 , (III) = 2 , (IV) = 3 , (V) = 2
Overall max = 4
→ answer.

We use gcd to manage quadrants.
If class does not implement hashCode & equals, implement it. (Line class : int dx, int dy)

Java

→ public int hashCode()
{ return this.dx + this.dy;
}

→ public boolean equals(Object o)
{ Line l = (Line o);
if (this.dx == l.dx && this.dy == l.dy)
return true
else
return false.
}

Q Morning Assembly.

What?
Given an array of N elements which contains heights of students in order which they are standing. Return minimum transfers needed to arrange the students in in order of heights.

How?
eg: | 7 | 4 | 1 | 3 | 5 | 2 | 6 | N = 7

res: | 1 | 1 | 1 | 1 | 2 | 2 | 3 | (0 1 2 3 4 5 6)

(4-1) index not present in hashmap so]

(5-1)=4 present in hashmap

res[5] = res[map.get(4)] + 1

map
7-0
4→1
1→2
3→3
5→4
2→5
6→6

max = ~~X~~ ~~X~~ 3.

→ For each value ~~of index~~ check if value -1 is present in map or not. If present means this element has map.get(value-1) persons before it who are in order.

→ max count contains maximum no. of elements that are orderd.
Mininimum transfers = array length (N) - max
= 7 - 3 = 4

Q Identify Strings

What?
Given a string arrange the characters of string such that adjacent characters are distinct.

How?
→ Make frequency Map fmap <key, value> where key is character of string & value is # frequency of that character is value.

→ If max frequency is greater than (string.length())/2 then string can't be arranged.

Make max heap of pair class.

Pair class → char c
& int freq.

In Java.

```
public static class pair implements Comparable<pair>{
    int freq;
    char ch;
    public pair(int freq, char ch)
    { this.freq = freq;
      this.ch = ch;
    }
    public int compareTo(pair o)
    {
        if (this.freq == o.freq)
            return o.ch - this.ch;
        else
```

```
        return this.freq - o.freq;
    }
}
```

Priority queue will be max heap

```
PriorityQueue<pair> pq = new PriorityQueue<>
                        (Collections.reverseOrder());
```

In C++

```
struct frequencyCompare {
    bool operator()(const pair& a, const pair& b)
    {
        if (a.freq == b.freq)
            return b.ch < a.ch;
        else
            return a.freq < b.freq;
    }
};

class pair {
public:
    int freq;
    char ch;

    pair() {}
    pair(int freq, char ch)
    { this->freq = freq;
      this->ch = ch;
    }
};
```

priority_queue < Pair , vector < pair > , frequencyCompare > pq;

eg: aaabcb

Ans a b a b a c

- pop 2 height frequency
- print chars
- push with decremented frequency.

Q Substring with similar first and last characters

What?
Count the no. of substrings of a given string which has same first and last occurrence.

How?
Make frequency map for each character

eg: a b c b d a e d b c a
0 1 2 3 4 5 6 7 8 9 10

a → ~~1 2~~ 3
b → ~~1 2~~ 3
c → ~~1 2~~ 3
d → ~~1~~ 2

count = ~~1 2~~ 3 ~~4~~ 5 ~~6~~ ~~7~~ ~~8~~ 10 ~~11~~ ~~12~~ ~~13~~ ~~14~~ ~~15~~ ~~16~~ 21

When frequency is 1 that is also valid substring as per given condition that is why added in count.

Ans = 21

- update frequency in map and then and in count resultant frequency

Q Group Shifted Strings

What?
Given an array of strings. Group all those strings together which are shifted in same way. Shifting is when we change each letter of string to its successive letter.

How?
eg: arr = { abc, bcd, acef, xyz, az, ba, a, z }

1) For each string calculate the gap b/w characters (adjacent) and make a string.
for eg: s = abc
key = *1*1* ⇐ (*(b-a)*(c-a)*)

s = za gap = (a-z) = (1-26) = -25
= -25+26 = 1
key = *1

*when gap is negative add 26.

2) If string is already present then add to the arraylist currentString.

Map	Ans
*1*1* → [abc, bcd, xyz]	abc bcd xyz
* → [a, z]	a z
*2*2*1* = [acef]	az ba
25 = [az, ba]	acef.

All the string that belong to same shifted group will form same key string.

Q Max No. of characters between 2 same Characters.

What?

Given a string find maximum no. of characters that are present b/w 2 same characters.

How?

→ Make a Hashmap to store first occurence for each character.

At second ; Third - .. occurences check the gap b/w first occurence & current occurance if greater then max update max.

eg: a b c d a b c b d a c c d f d

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14

max ~~0 5 7~~ 10 12

		Map
at 4	(4-0+1) > 0e.5	a → 0
at 5	(5-1+1) = 5	b → 1
at 6	(6-2+1) = 5	c → 2
at 7	(7-1+1) = 7	d → 3
at 11	(11-1+1) = 10	f → 13.
at 14	(14-3+1) = 12	ans = 12

If no char repeats then maxLen will remain 0.

Q Logger Rate limiter

What?

Given a stream of message along with timestamps each message should be printed if only if its not printed in last 10 seconds. return true if msg should be printed

How?

Make a hashmap that contains messages as key. and value will be the timestamp at which it was last printed.

eg: a b c a b a c b c.

✓ ✓ ✓ x x ✓ x ✓ ✓

1 8 10 11 14 16 18 20 22

11-10=1 (1 > 1 so false)

14-8 < 10

20-8 > 10 print True.

Result = [T T T F F T F T T]

Map

a → ~~1~~ 16

b → ~~8~~ 20

c → ~~10~~ 22

For each time stamp check (timestep - map.get(msg))

>=10 then return True and update time stamp

else false.

Q Longest Consecutive Subsequence

What?

Given an array of integers find the longest consecutive subsequence in an array.

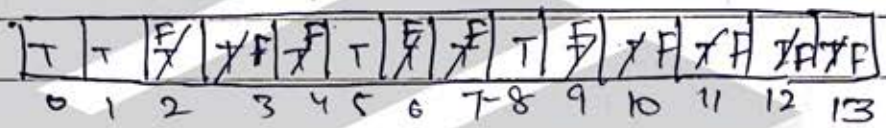
How?

1) Find maximum & minimum of the array elements and. Fill array elements in a set.
2) Make boolean array of size max+1 and contains True in all indexes.
3) For each num b/w min to max check if num-1 is present in Hashset or not if present then mark it false in boolean array as it can't be start of sequence.

4) For each index (num) in array (given number) check if boolean array is true. If true means this is start of subsequence.
Increment length count till you find cur+1 element in set.

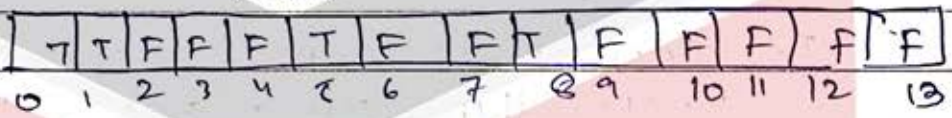
Eg: | 1 | 9 | 12 | 11 | 6 | 2 | 8 | 3 | 5 | 10 |
0 1 2 3 4 5 6 7 8 9

boolean starts array.



2 in set

⇒



1 → Start[1] = true ⇒ count = ~~1~~ ~~2~~ 3 (2 previous in set) (3 present) max = 3

9 → Start[9] = false. max = 3

12 → Start[12] = false. max = 3

8 → start[8] = true ⇒ count = ~~1~~ ~~2~~ ~~3~~ 4 5 max = 5

5 → start[5] = true count = 2 3 4 5 (9 10 11 12) max = 5

Ans = 5

* we are finding the starts of subsequences and checking the length from that.

Heap Based Questions.

Q. IPO

What?

Help pepcoding design the best way to maximize its total capital after finishing at most k distinct projects. Each projects' price as profit and capital needed to start it is given. Initial Capital is W. Return maximised final capital.

How?

1. Sort on the basis of Capital in ascending order.
2. Sort on the basis of Profit in descending order if capital is same (used in priority queue).

ex:

	a	b	c	d	e	f
C :	80	150	90	200	250	350
P :	40	60	50	70	50	300

, W = 100
, k = 4.

After sorting:

	a	b	c	d	e	f
C:	80	150	90	200	250	350
P:	40	60	50	70	50	300

3. Now priority queue uses capital to perform profit comparison, initially ~~priority queue~~ a list has all elements. sorted array
4. Now one by one elements are added in priority queue after checking if its capital <= the Capital we currently have.

I) k=1, W=100

a	c
80	90
40	50

added in 1st pass.

⇒ c gives more profit in this, capital so c is removed & its profit ~~capital~~ is added in answer.

II) W = 150 (100 + 50)

k=2

a	b
80 / 40	150 / 60

⇒ b gives more profit.

III) W = 210 (150 + 60)

k=3

a	d
80 / 40	200 / 70

⇒ d gives more profit

IV) W = 280 (210 + 70)

k=4

a	c
80 / 40	250 / 50

⇒ c gives more profit.

∴ Total profit earned = 330 ~~using~~ by doing c, b, d, c projects.

Implementing comparators in C++:

```
class Project {
    public:
    int c;
    int p;
    Project (int p, int c) {
        this→p = p;
        this→c = c;
    }
    Project () {          // default constructor is needed
    }                     // to create objects
};

bool CapitalComparator(Project &a, Project &b) {
    return a.c < b.c;
}
```

```
struct ProfitComparator{
    bool operator()(const Project& a, const Project& b)
    {   return a.p < b.p;
    }
};
```

In main, create vector<Project> v and after inserting all Project pairs in it, sort it as:

→ sort(v.begin(), v.end(), CapitalComparator);

Now create a priority queue as:

→ priority_queue<Project, vector<Project>, ProfitComparator> pq;

Implementing comparators in Java:-

```
public static class Project{
    int p;
    int c;
    Project(int p, int c){
        this.p = p;
        this.c = c;
    }
}
public static class profitComparator implements
  Comparator<Project>{
    public int compare(Project p1, Project p2){
        return p1.p - p2.p;
    }
}
public static class capitalComparator implements
  Comparator<Project>{
    public int compare(Project p1, Project p2)
```

```
    { return p1.c - p2.c;
    }
}
```

In main, sort array in which you insert all elements as Project pairs as:

→ Arrays.sort(projects, new capacityComparator());
projects is the array.

After this, create priority queue as:

→ PriorityQueue<Project> pq = new PriorityQueue<>(Collections.reverseOrder(new profitComparator()));

Q. Print k smallest elements in an array in given order.

What?

Print first k smallest elements in an array in the order in which they appear in the array.

How? ex: {4, 2, 7, 11, 9, 15, 7, 6, 3, 8, 20, 19}, k=4.

→ Create a priority queue of size k of max type.

I) 4 | 2 | 7 | 11 | 11 — 4, 2, 7, 11

II) 4 | 2 | 7 | 9 | 9 — 9

III) 4 | 2 | 7 | 9 | acq 9 — 15

IV) 4 | 2 | 7 | 7 | 7 — 7

V) 4 | 2 | 7 | 6 | 7 — 6

VI) 4 | 2 | 6 | 3 | 6 — 3

VII) 4 | 2 | 6 | 3 | 6 — 8, 20, 19

Now, for printing, if array element < peek, print, and if it is == peek, print & remove (so that peek can change).

→ 4, 2, 6, 3 is ans.

Q. Print Kth smallest fraction.

What?

You will be given an array of prime numbers and a number K, find Kth smallest fraction n/d such that both belong to array of prime numbers.

How? ex: [3 5 7 2 11]

All cases: $\frac{2}{11}$ $\frac{3}{11}$ $\frac{5}{11}$ $\frac{7}{11}$

$\frac{2}{7}$ $\frac{3}{7}$ $\frac{5}{7}$

$\frac{2}{5}$ $\frac{3}{5}$

$\frac{2}{3}$.

If the array is not sorted, sort the array to only make pairs in the way shown above.

→ Now simply apply a comparator which sorts fraction pairs :—

C++

```
→ bool FractionComparator (fpair &a, fpair &b) {
      return (a.n*(1.0)/a.d) < (b.n*(1.0)/b.d);
  }
```

* Now return K^{th} smallest fpair with numerator = n & denominator = d.

Java Implement compareTo in class.

```
→ public int compareTo (fpair o)
  { double val1 = this.n*(1.0) / this.d;
    double val2 = this.o.n*(1.0)/ o.d;
```

```
if (val1 < val2) {
    return -1;
} else if (val1 == val2) {
    return 0;
} else {
    return 1;
}
}
```

Q. Top K Frequent Elements

What?

Given a non-empty array of integers, find top K most frequent elements.

How? ex: N = 6

{1, 1, 1, 2, 2, 3}

K = 2

1. Make a frequency map.
2. Insert frequency map pairs in priority queue sorted on the basis of frequency.
   (Pair with frequency greater are on top).

PQ / Vector | (1-3) | (2-2) | (3-1) |

3. Now pop and print values of first k pops.

Ans: [1, 2]

Similiar question: Top K Frequent Words.
[numbers replaced by words, rest everything is exactly same.]

Page No.
Date

Q. Height of heap

What?

You will be given a binary heap of size N, return height of heap.

How?

Use log function to calculate height of heap.

↓

Log floor method used.

for 8: ans = 3 $(2^3 = 8)$.

// Code

```
int solve (int N)
{   int x = 0;
    while ((1 << x) <= N) {
        x++;
    }
    return x-1;
}
```

↓

Returns floor as for all 4, 5, 6, 7 it should give ~~floor~~ height as 2; ||ly for other levels..

Q. Kth largest element in a stream.

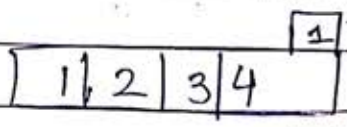
What?

You are given input stream of n integers, find kth largest element for each element in system stream.

How? ex: {1, 2, 3, 4, 5, 6}, k = 4.

Simply create a ~~min~~ priority queue of k size, if ~~elements pt.~~ you have not inserted k elements, answer is −1, else, return the peek if kth

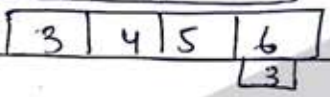
is being inserted, else, if element > peek, remove ~~ele~~ peek element (pop), add current element and now return peek.

I) 

For 1, 2, 3 → ans = -1, -1, -1

For 4 → ans = 1

II) For 5 → ans = 2.



III) For 6 → ans = 3.

∴ ans ⇒ [-1, -1, -1, 1, 2, 3].

HuffMann Encoding and Decoding

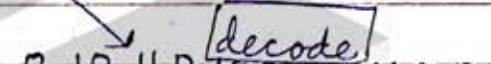
a) You will be given a string, and you will have to decode it into binary : decoding.
b) You will be given a binary string, and you will have to encode it into a character string : encoding.

ex: a b c b a d c a

encode

0 10 110 10 0 111 1100

a → 0
b → 10
c → 110
d → 111



decode

0 10 110 10 0 111 1100
a b c b a d c a

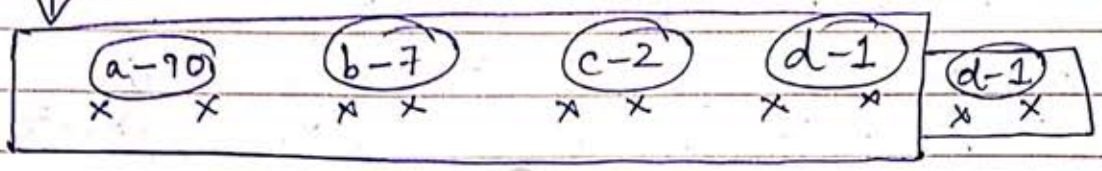
0 → a
10 → b
110 → c
111 → d.

Design goals to be implemented:
1) Each character has different encoding (no prefixes common).
2) Character which occurs more should have less number assigned to it, i.e; it should be given a shorter binary code.
ex: a - 90, b - 7, c - 2, d - 1

Algorithm
1) Feeder : A feeder function tells you frequency of each character; Parses string to frequency map. [a → 90, c → 2; b → 7, d → 1]

2) Frequency Map → Priority Queue of Nodes where each Node has value as char-freq, and left and right as NULL initially.

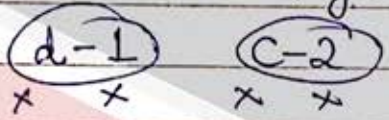
pq



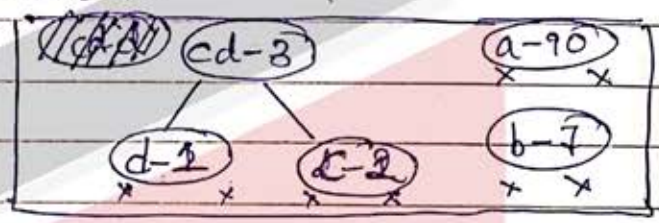
3) While (pq.size() > 1),

[remove 2 nodes.
merge into 1, root is both data + both values.
add the merged node back.

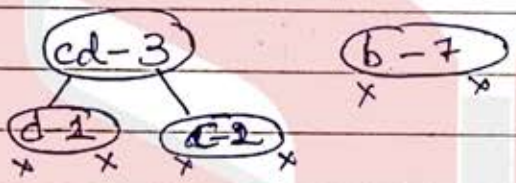
I)



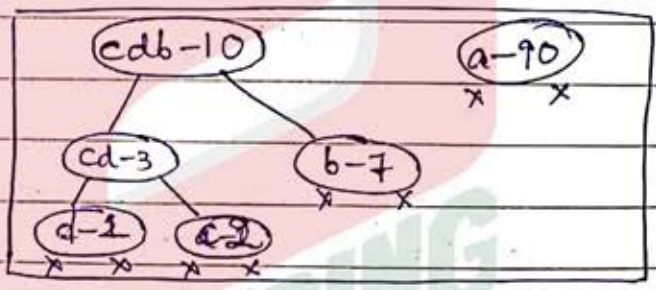
data = cd, val = 3



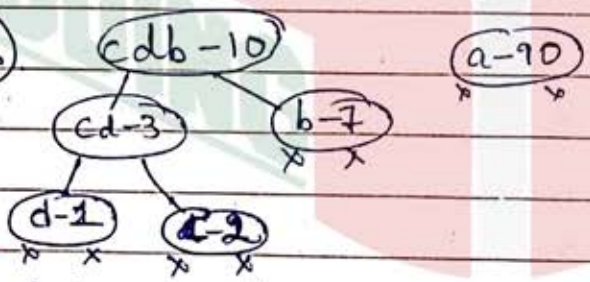
II)



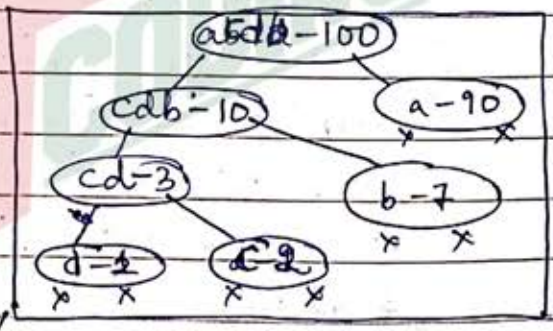
data = cdb, val = 10



III)

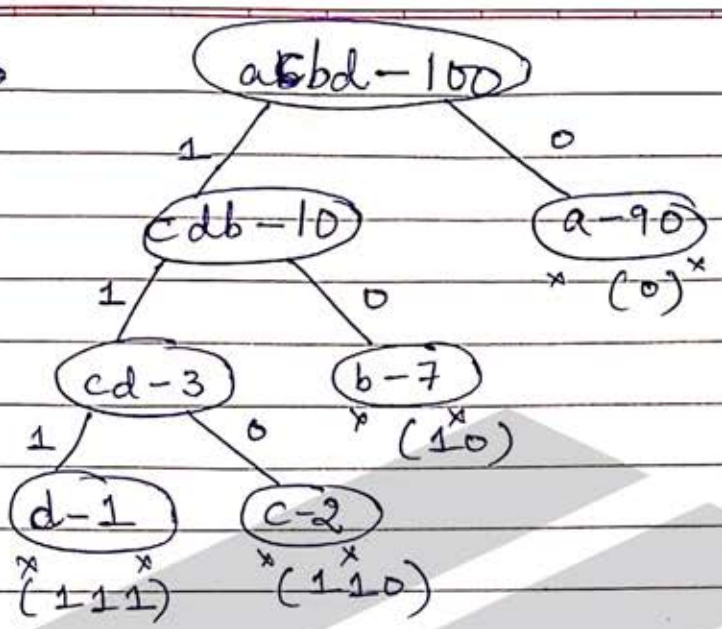


data = abcdb, val = 100



Now use this tree to create codes for each character at leaf node. Since left has all smaller, left calls add 1 in string, right calls add 0 in string.

⇒



∴ mapping done as:

a = 0
b = 10
c = 110
d = 111

⇒ The mappings as we see meet all design goals, because since we use min priority queue, the characters used least have more depth and longer codes.

Also since mapping is done at leaf nodes, which contain only characters, we have no common prefixes.

BIT Masking Based Questions

Q Find the Difference

What?
Given 2 strings find ~~all~~ the character which is different in both strings

How?
XOR all the characters of string 1 and string 2 last remaining character is different character

eg: s1 = a b c d e
s2 = e c b d a g.

XOR = a ∧ b ∧ c ∧ d ∧ e ∧ e ∧ c ∧ b ∧ d ∧ a ∧ g
= (a ∧ a) ∧ (b ∧ b) ∧ (c ∧ c) ∧ (d ∧ d) ∧ (e ∧ e) ∧ g
= 0 ∧ g
= g.

a ∧ a = 0
a ∧ 1 = a' (complement)
a ∧ 0 = a

Q Find the Odd occurrence

What?
Given an integer array find the element with odd occurence. (only element like this)

How?
XOR all the elements of array last remaining is the result.

Page No.
Date

Q Uncommon Characters

What?
Find and print uncommon characters of two given strings. Here uncommon characters means that either the character is present in one string or it is present in other but not in both. (string has only lowercase chars).

How?
→ For each string make a bitmask (for 26 alphabets) If the string contains that alphabet set the bit otherwise it is set to 0.

eg: s1: a b c d f g p q r
s2: g h i a n d f p.

mask1 00000000001100000011011111 (c, b, a)

mask2: 00000000101000011110100 1

XOR 00000000100100001100000110
q r ih c b

Ans bching

Why
If both have characters then XOR(1^1) is 0 result won't contain that character.

* Set Bit x = x | (1<<k) (To set Kth bit)
* Unset Bit x = x & (~(1<<k))
* Toggle Bit x = x ^ (1<<k)

Page No.
Date

Q Is-Sudoku Valid

What?

Given a 9x9 board of Sudoku. Validate the filled cells. Each row/col must contains digits 1 to 9 without repetition & each of the 9 3x3 subboxes must contain the digits 1-9 without repeat.

How?

	c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8
r_0	5	3	.	.	7	.	.	.	.
r_1	6	.	.	1	9	5	.	.	.
r_2	.	9	8	.	.	.	.	6	.
r_3	8				6				3
r_4	4			8		3			1
r_5	7				2				6
r_6		6					2	8	.
r_7				4	1	9	.		5
r_8					8		.	7	9

We will use bitmask to store which no.(digit) is used in this row/column/box.

integer row[9] → contains mask for each row
integer col[9] → contains mask for each column
integer sn[3][3] → contains mask for each box

eg: For row 1 mask is 0010 1 0 1 000
(bit positions 9 8 7 6 5 4 3 2 1 0)
this means digit 3, 5, & 7 are used

Page No. _____
Date

For every box 3 conditions are checked following
i & j are row & col no.

→ row[i] & (1 << (digit)) != 0 (already present so false)
→ col[j] & (1<<(digit)) != 0 → already present in colm
→ sm[i/3][j/3] & (1 << digit) != 0 (already present in box)

If all checks fail ie box is valid.
set bit true in all more masks.

Q. Meeting Rooms

What?

You are given start time and end time of various meetings where $s_i < e_i$ for each one. Return minimum number of rooms required to hold these meetings.

How?

Approach 1: left - right.

Sort the two ~~sort~~ start times and end times array. Keep left on start[0] and right on end[0]. While comparing, if start[i] <= end[i], count++, else count-- (if end[i] < start[i]). Answer is maximum count at any particular i.

ex:

starts: 2, 3, 4, 5, 8, 10, 12, 13, 14, 15, 16, 17, 18

ends: 6, 6, 7, 9, 11, 13, 14, 15, 18, 19, 20, 20, 20

count = ~~1~~ ~~2~~ ~~3~~ ~~4~~ ~~3~~ ~~2~~ ~~1~~ ~~2~~ ~~1~~ ~~2~~ ~~1~~ ~~2~~ ~~3~~ ~~2~~ ~~3~~ ~~2~~ ~~3~~ ~~2~~
~~3~~ ~~4~~ (5) ~~4~~ ~~3~~ ~~2~~ ~~1~~ 0

→ maximum count at any moment was 5.

\# If you first decrement in case of ==, ans is (4)

Approach 2: Create an array of size = max value of all starts and ends array.

Put +1 at all indices of starts[i] in array, and -1 and all ends[i]. Answer is maximum prefix sum at any point.

index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
array			1	1	1	1	-2	-1	1	-1	1	-1	1	~~1~~0	~~1~~0	~~1~~0	1	1	~~1~~0	-1
psum:			1	2	3	(4)	2	1	2	1	2	1	2	2	2	2	3	4	4	3

20: -3 → 0

(4) → ans