Largest Subarray with zero sum

int[] arr = { 2, 3, 5, -7, -1, -2, 8, -8 }

len = ~~2~~ 4

0 length = 2

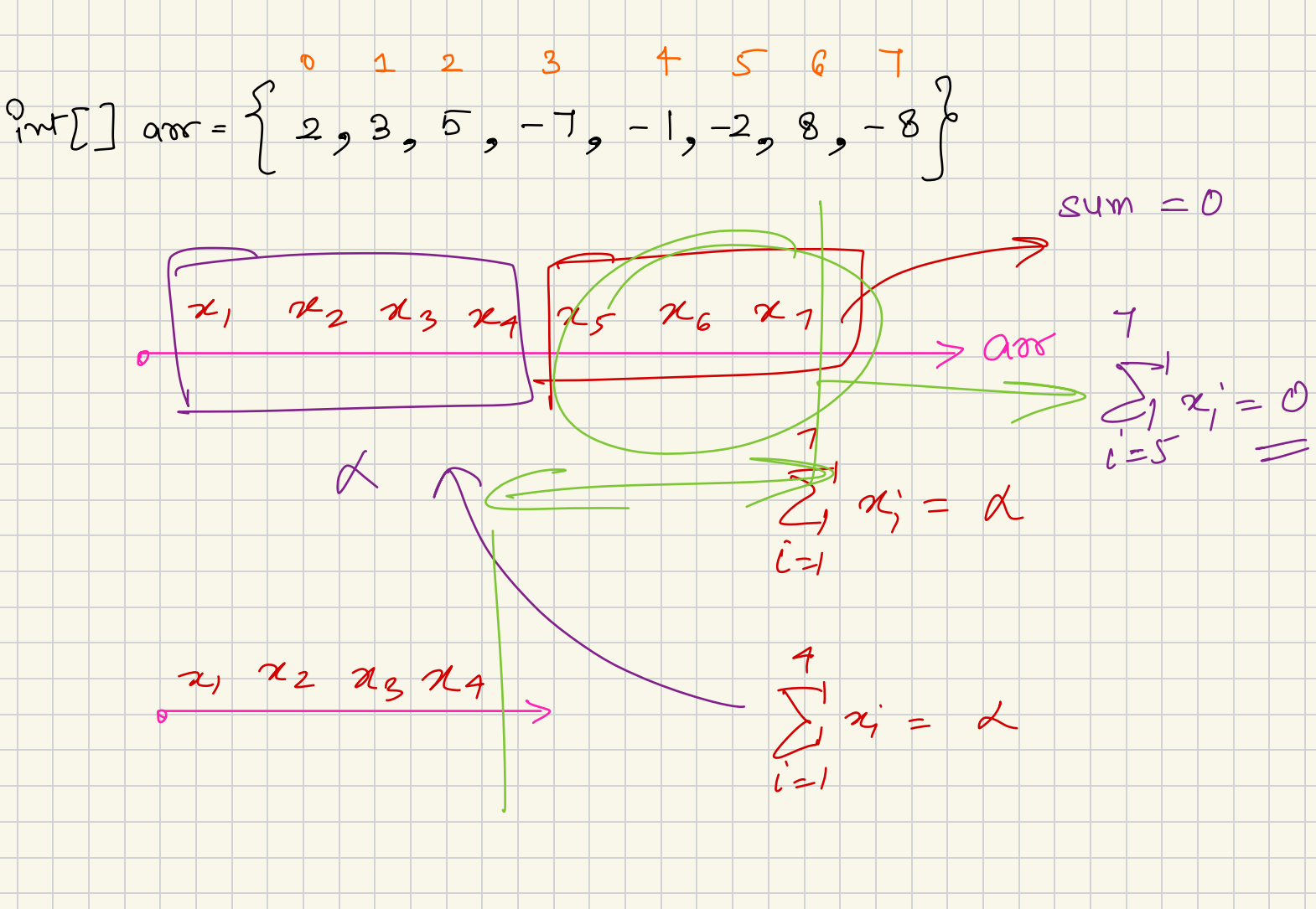
Brute Force

→ Calc. all the subarrays sum

→ Store largest len of subarray with zero sum

TC: $O(N^2)$

SC: $O(1)$


0 1 2 3 4 5 6 7
int[] arr = { 2, 3, 5, -7, -1, -2, 8, -8 }
sum = 0
x1 x2 x3 x4 x5 x6 x7
arr
$\sum_{i=5}^{7} x_i = 0$
$\sum_{i=1}^{7} x_i = \alpha$
α
x1 x2 x3 x4
$\sum_{i=1}^{4} x_i = \alpha$

int[] arr = { 2, 3, 5, -7, -1, -2, 8, -8 }

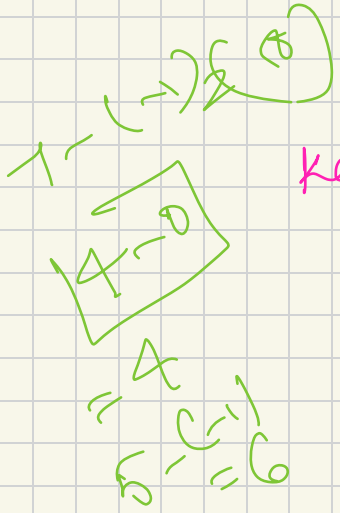
	0	1	2	3	4	5	6	7
	2	3	5	-7	-1	-2	8	-8
Sum	0	2	5	10	3	2	0	8

HashMap

key { Sum }

value { index }

Seen for the first time



key (Sum)	value (index)
0	6
2	0
5	1
10	2
3	4
	-1

Group Anagrams

String[] words = { "act", "tac", "dog", "cat", "god" }

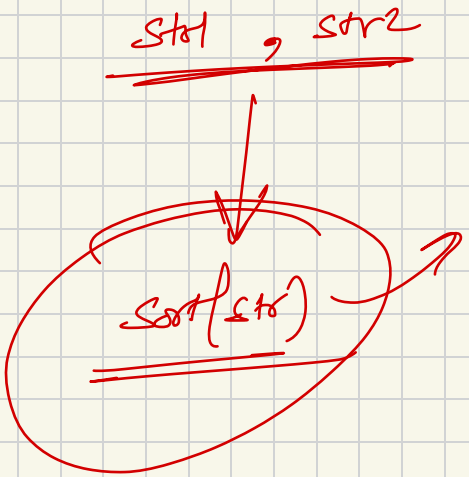
O/P : { { "act", "tac", "cat" }, { "dog", "god" } }

Brute force :

TC : $O(N^2 \times M)$ — N: Number of words, M: avg length of word

SC : O(1) — O(26)

String[] words = { "act", "tac", "dog", "cat", "god" }



unique

HashMap

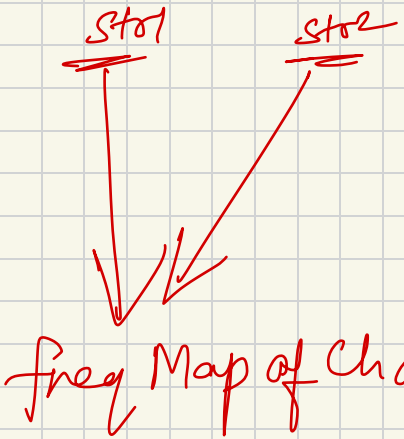
key (String) — value (AL<words>)

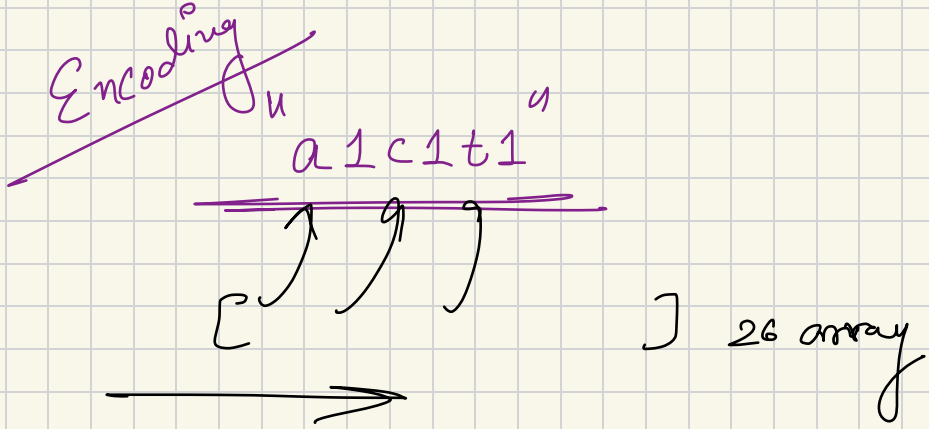
"act" — {"act", "tac", "cat"}

"dgo" — {"dog", "god"}

$$\begin{cases} TC : O(N \times M \log M) \\ SC : O(N) \end{cases}$$

String[] words = { "act", "tac", "dog", "cat", "god" }





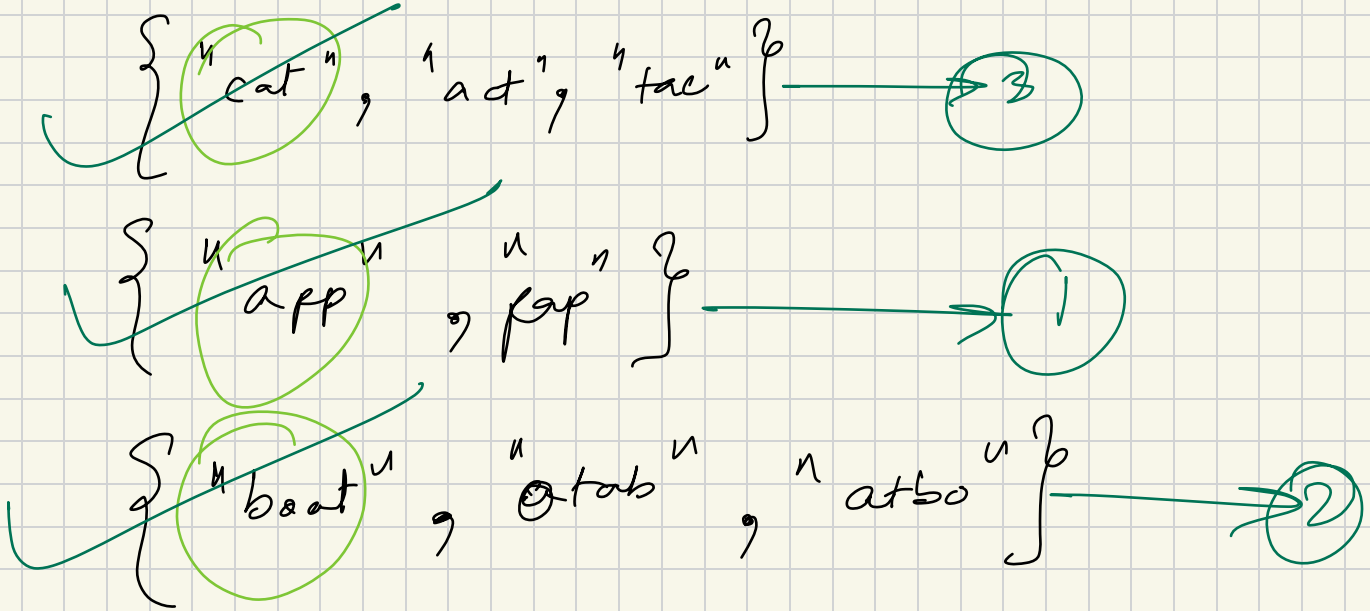
String[] words = { "act", "tac", "dog", "cat", "god" }

key

a1c1t1 {act, tac, cat}

d1g1o1 {dog, god}

TC: O(N × M) SC: O(N)


{ "cat", "act", "tac" } → 3
{ "app", "pap" } → 1
{ "boat", "otab", "atbo" } → 2