→ prefix sum

	3	2	4	5	8
	0	1	2	3	4
ps	3	5	9	14	22

ps[i] → 0 to i

i = 2

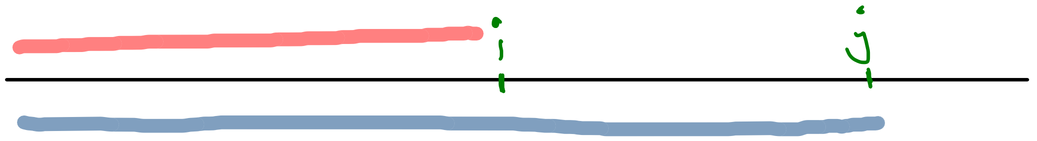
j = 3

sum(i to j) ⟶ ps[j] - ps[i-1]

Largest subarray with 0 sum

```
A[] = {15,-2,2,-8,1,7,10,23}
```

	15	-2	2	-8	1	7	10	23
	0	1	2	3	4	5	6	7
PS	15	13	15	7	8	15	25	48



$$\underbrace{ps[i]}_{\Sigma\, 0 \text{ to } i} == \underset{\Sigma\, 0 \text{ to } j}{ps[j]} \longrightarrow \Sigma\, (i+1) \text{ to } j = 0$$

	15	-2	2	-8	1	7	10	23
	0	1	2	3	4	5	6	7
ps	15	13	15	7	8	15	25	48

hashmap

prefix sum vs first idx

map :

0	$\rightarrow$ (-1)
15	$\rightarrow$ 0
13	$\rightarrow$ 1
7	$\rightarrow$ 3
8	$\rightarrow$ 4
25	$\rightarrow$ 6
48	$\rightarrow$ 7

0 ans = ~~0~~ ~~2~~ 5

	-2_0	1_1	1_2	4_3	-9_4
ps	-2	-1	0	4	5

$0 \rightarrow -1$

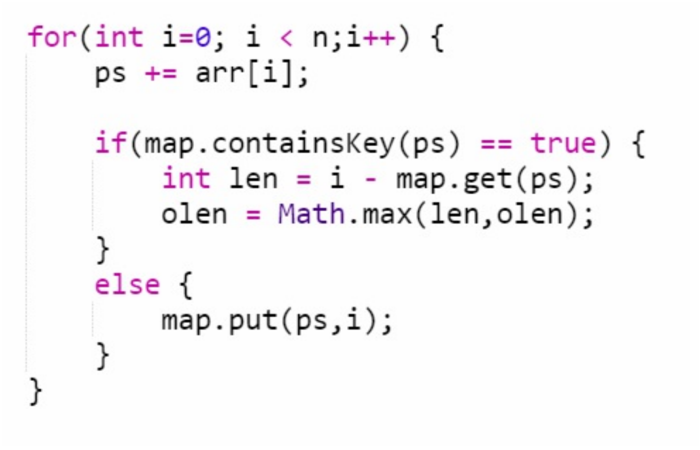
$-2 \rightarrow 0$

$-1 \rightarrow 1$

$4 \rightarrow 3$

$5 \rightarrow 4$

olen = ~~0~~ 3



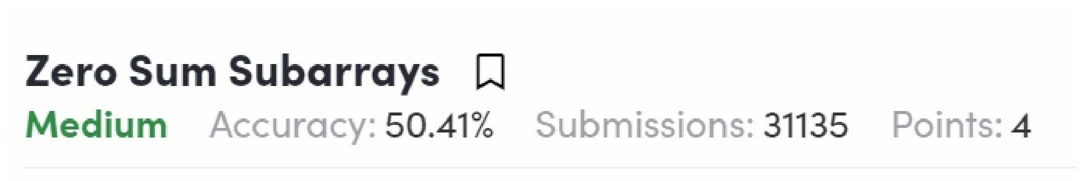
```
for(int i=0; i < n;i++) {
    ps += arr[i];

    if(map.containsKey(ps) == true) {
        int len = i - map.get(ps);
        olen = Math.max(len,olen);
    }
    else {
        map.put(ps,i);
    }
}
```

.	0	-1	1	15	-2	2	-8	1	7	10
-1	0	1	2	3	4	5	6	7	8	9
ps	0	-1	0	15	13	15	7	8	15	25

map

0 →	(-1)
-1 →	1
15 →	3
13 →	4
7 →	6
8 →	7
25 →	9

olen = ~~0~~ ~~1~~ ~~3~~ 5



Zero Sum Subarrays

Medium Accuracy: 50.41% Submissions: 31135 Points: 4

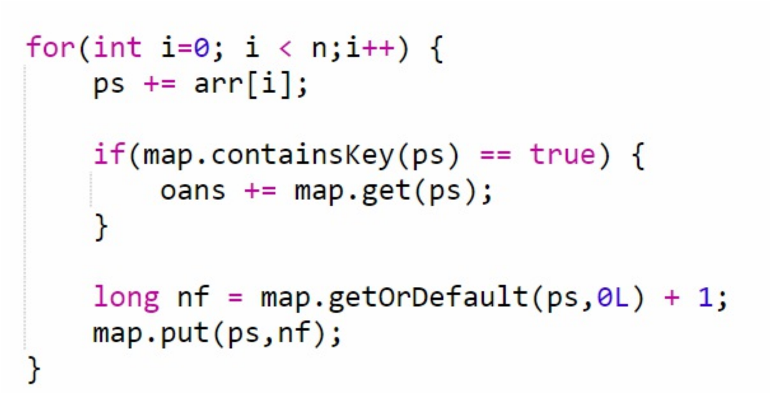
$0_0 \quad -1_1 \quad 1_2 \quad 15_3 \quad -2_4 \quad 2_5 \quad -8_6 \quad 1_7 \quad 7_8 \quad 10_9$

ps $\quad 0 \quad -1 \quad 0 \quad 15 \quad 13 \quad 15 \quad 7 \quad 8 \quad 15 \quad \overset{}{\underset{i}{25}}$

map: ps vs count

0 → 3	7 → 1
-1 → 1	8 → 1
15 → 3	25 → 1
13 → 1	

0 ans = 1 + 2 + 1 + 2



```java
for(int i=0; i < n;i++) {
    ps += arr[i];

    if(map.containsKey(ps) == true) {
        oans += map.get(ps);
    }

    long nf = map.getOrDefault(ps,0L) + 1;
    map.put(ps,nf);
}
```

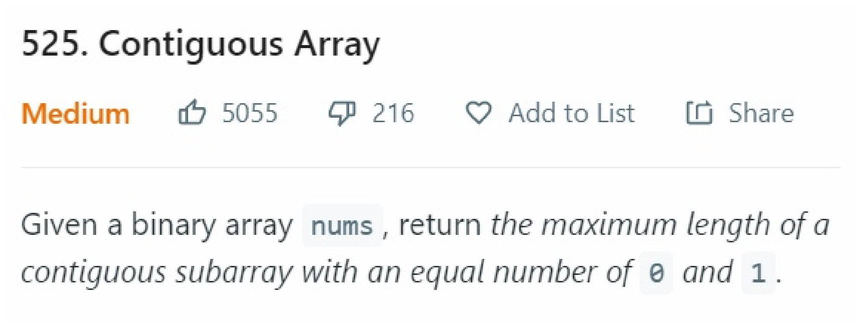
	0_0	-1_1	1_2	15_3	-2_4	2_5	-8_6	1_7	7_8	10_9
ps	0	-1	0	15	13	15	7	8	15	25

map

ps, count

$0 \to 3$	$7 \to 1$
$-1 \to 1$	$8 \to 1$
$15 \to 3$	$25 \to 1$
$13 \to 1$	

oans = 1 + 2 + 1 + 2


525. Contiguous Array

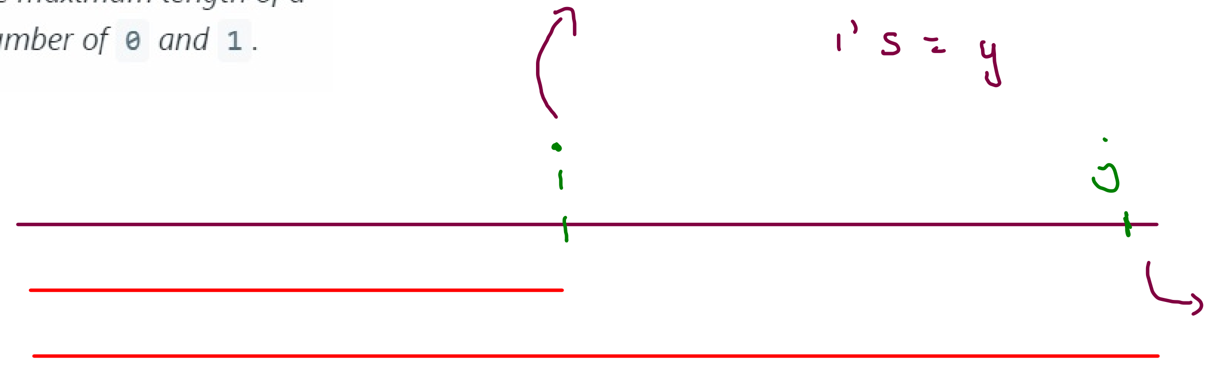
Medium 5055 216 Add to List Share

Given a binary array `nums`, return *the maximum length of a contiguous subarray with an equal number of* `0` *and* `1`.

① - using gap

0 to i 0's = x

1's = y



0 to j

0's = a

1's = b

i+1 to j → $a = x + k$

$b = y + k$

$\boxed{a - b = x - y}$

[equal no. of 0's & 1's]

```
6
011011
```

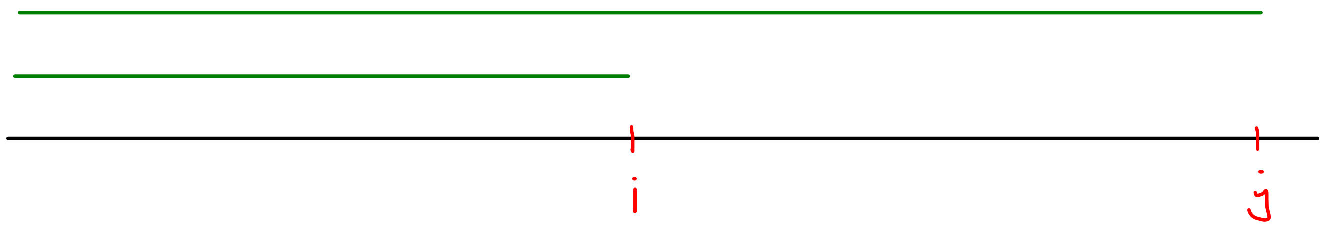
	0_0	1_1	1_2	0_3	1_4	1_5
c0	1	1	1	2	2	2
c1	0	1	2	2	3	4
(c0 - c1)	1	0	-1	0	-1	-2

$0 \rightarrow (-1)$
$1 \rightarrow 0$
$-1 \rightarrow 2$
$-2 \rightarrow 5$

olen = ~~2~~ 4

map

$(c_0 - c_1) \longrightarrow$ first idx



0's = x

1's = y

gap = $x - y$

0's = a ,

1's = b

gap = $a - b$

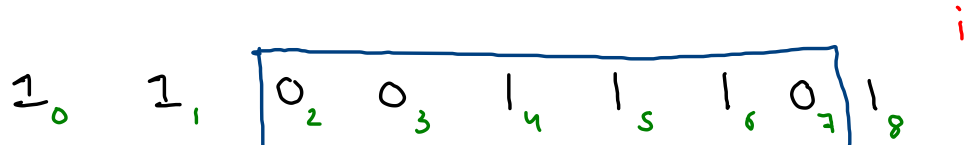
$a = x + k$ — ①

$b = y + k$ — ②

① − ②

$$\boxed{a - b = x - y}$$

```java
map.put(c0-c1,-1);

for(int i=0; i < nums.length;i++) {
    if(nums[i] == 0) {
        c0++;
    }
    else {
        c1++;
    }

    int gap = c0-c1;

    if(map.containsKey(gap) == true) {
        int len = i - map.get(gap);
        olen = Math.max(len,olen);
    }
    else {
        map.put(gap,i);
    }
}
```



c_0	0	0	1	2	2	2	2	3	3
c_1	1	2	2	2	3	4	5	5	6
$c_0 - c_1$	-1	-2	-1	0	-1	-2	-3	-2	-3

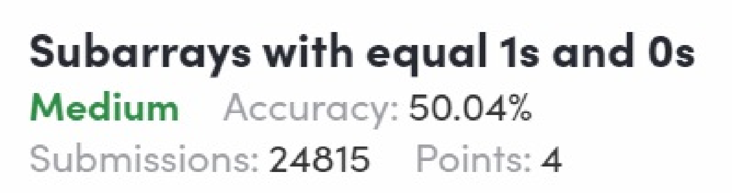
$0 \rightarrow (-1)$

$-1 \rightarrow 0$

$-2 \rightarrow 1$

$-3 \rightarrow 6$

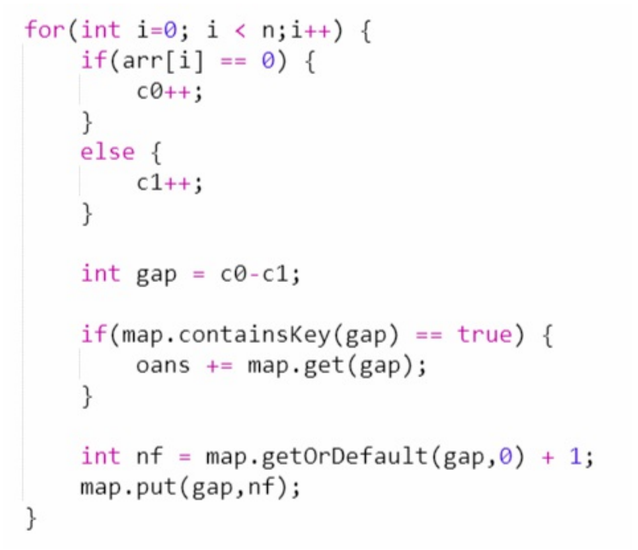
olen = ~~2~~ ~~4~~ 6



Subarrays with equal 1s and 0s

Medium Accuracy: 50.04%

Submissions: 24815 Points: 4



```
for(int i=0; i < n;i++) {
    if(arr[i] == 0) {
        c0++;
    }
    else {
        c1++;
    }

    int gap = c0-c1;

    if(map.containsKey(gap) == true) {
        oans += map.get(gap);
    }

    int nf = map.getOrDefault(gap,0) + 1;
    map.put(gap,nf);
}
```

	0_0	1_1	1_2	0_3	1_4	1_5
c0	1	1	1	2	2	2
c1	0	1	2	2	3	4
gap	1	0	-1	0	-1	-2

$0 \to 3$

$1 \to 1$

$-1 \to 2$

$-2 \to 1$

gap vs count

oans = 0 + 1 + 2 + 1

② . using largest subarray having sum = 0

$\overset{-1}{\cancel{0}}_0 \quad 1_1 \quad 1_2 \quad \overset{-1}{\cancel{0}}_3 \quad 1_4 \quad 1_5$

(i) replace 0's with -1

(ii) apply subarray having sum=0