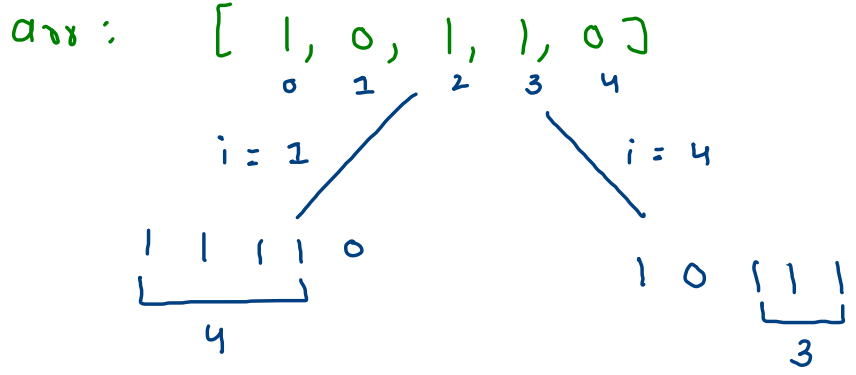
883 · Max Consecutive Ones II

you can flip a 'zero' to a 'one'

```
Input:  nums = [1,0,1,1,0]
Output:  4
```

cz

1 0 1 1 0

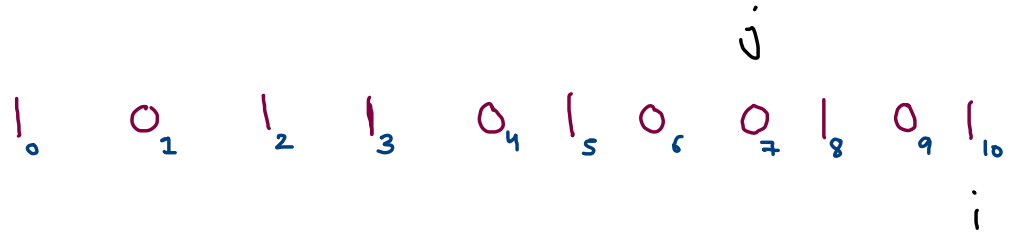


1. aquire : while(cz ≤ 1), stop when cz == 2

2. release : we need to release a zero.

1. aquire : while(cz ≤ 1), stop when cz == 2

2. release : we need to release a zero.



cz = 1

olen = ~~0 1 2 3~~ 4

1004. Max Consecutive Ones III

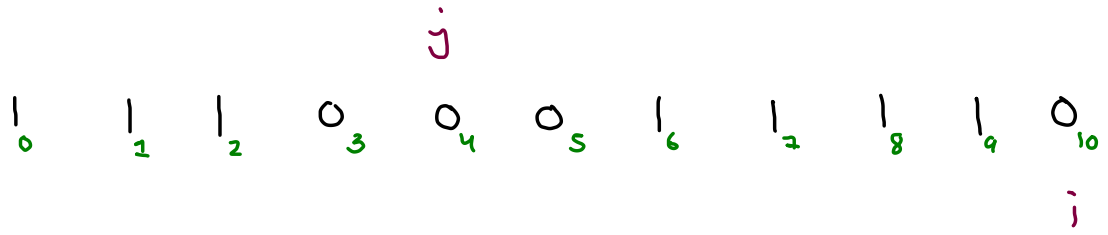
Medium 3998 55 Add to List Share

Given a binary array `nums` and an integer `k`, return *the maximum number of consecutive* `1`*'s in the array if you can flip at most* `k` `0`*'s.*

```
while(i < n-1) {
    //aquire
    while(i < n-1) {
        i++;

        if(nums[i] == 0) {
            cz++;
        }

        if(cz <= k) {
            //update ans
            int len = i - j;
            olen = Math.max(olen,len);
        }
        else {
            break;
        }
    }

    //release
    while(j < i) {
        j++;

        if(nums[j] == 0) {
            cz--;
            break;
        }
    }
}

return olen;
```

olen = ~~0~~ ~~1~~ ~~2~~ ~~3~~ ~~4~~ ~~5~~ 6

k = 2



cz = 2

386 · Longest Substring with At Most K Distinct Characters

Algorithms Medium Accepted Rate 31%

Description Solution Notes Discuss Leaderboard

Description

Given a string *S*, find the length of the longest substring *T* that contains at most k distinct characters.

k = 2

a a b c b c d b c a

aabcbcdbca
2

1. aquire: while (map.size() <= k)

2. release: while (map.size() > k)

Conclusions:

tuc (total unique chars)

→ tuc ≥ k

ans(atmost) == ans(exactly)

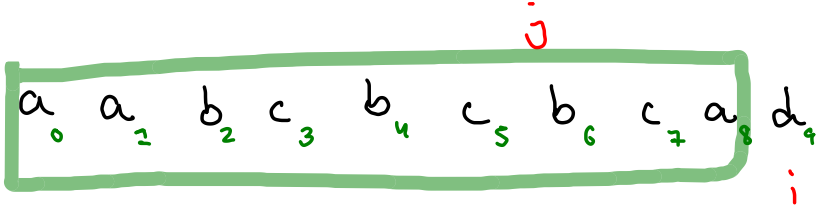
→ tuc < k

ans(atmost) == complete string

```
while(i <  s.length()-1) {
    //aquire
    while(i < s.length()-1) {
        i++;

        char ch = s.charAt(i);
        int nf = map.getOrDefault(ch,0) + 1;
        map.put(ch,nf);

        if(map.size() <= k) {
            //ans updation
            int len = i - j;
            olen = Math.max(olen,len);
        }
        else {
            break;
        }
    }
    //release
    while(j < i) {
        j++;

        char ch = s.charAt(j);

        if(map.get(ch) == 1) {
            map.remove(ch);
            break;
        }
        else {
            int nf = map.get(ch) - 1;
            map.put(ch,nf);
        }
    }
}
```

olen = ~~0~~ ~~1~~ ~~2~~ ~~3~~ ~~4~~ ~~5~~ ~~6~~ ~~7~~ ~~8~~ 9

k = 3



map a - 1

c - 1

d - 1

Count Of Substrings Having At Most K Unique Characters

```
while(i < str.length()-1) {
    //aquire
    while(i < str.length()-1) {
        i++;

        char ch = str.charAt(i);
        int nf = map.getOrDefault(ch,0) + 1;
        map.put(ch,nf);

        if(map.size() <= k) {
            //ans updatation
            count += (i-j);
        }
        else {
            break;
        }
    }

    //release
    while(j < i && map.size() > k) {
        j++;

        char ch = str.charAt(j);

        if(map.get(ch) == 1) {
            map.remove(ch);
            //valid again
            count += (i-j);
        }
        else {
            int nf = map.get(ch) - 1;
            map.put(ch,nf);
        }
    }
}
```

k = 3

a_0 b_1 a_2 d_3 c_4

a	ab	aba	abad	adc
	b	ba	bad	dc
		a	ad	c
			d	

map a-1

d-1

c-1

count = 1 + 2 + 3 + 4 + 3