String

- Array of characters
- Sequence of characters

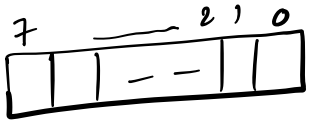
chars ?

```
char x = 'a';
```

→ 1B / 2B

16 bits → 2^{16} → 65536

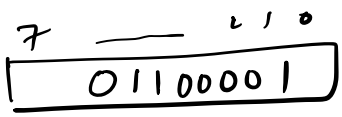
1B



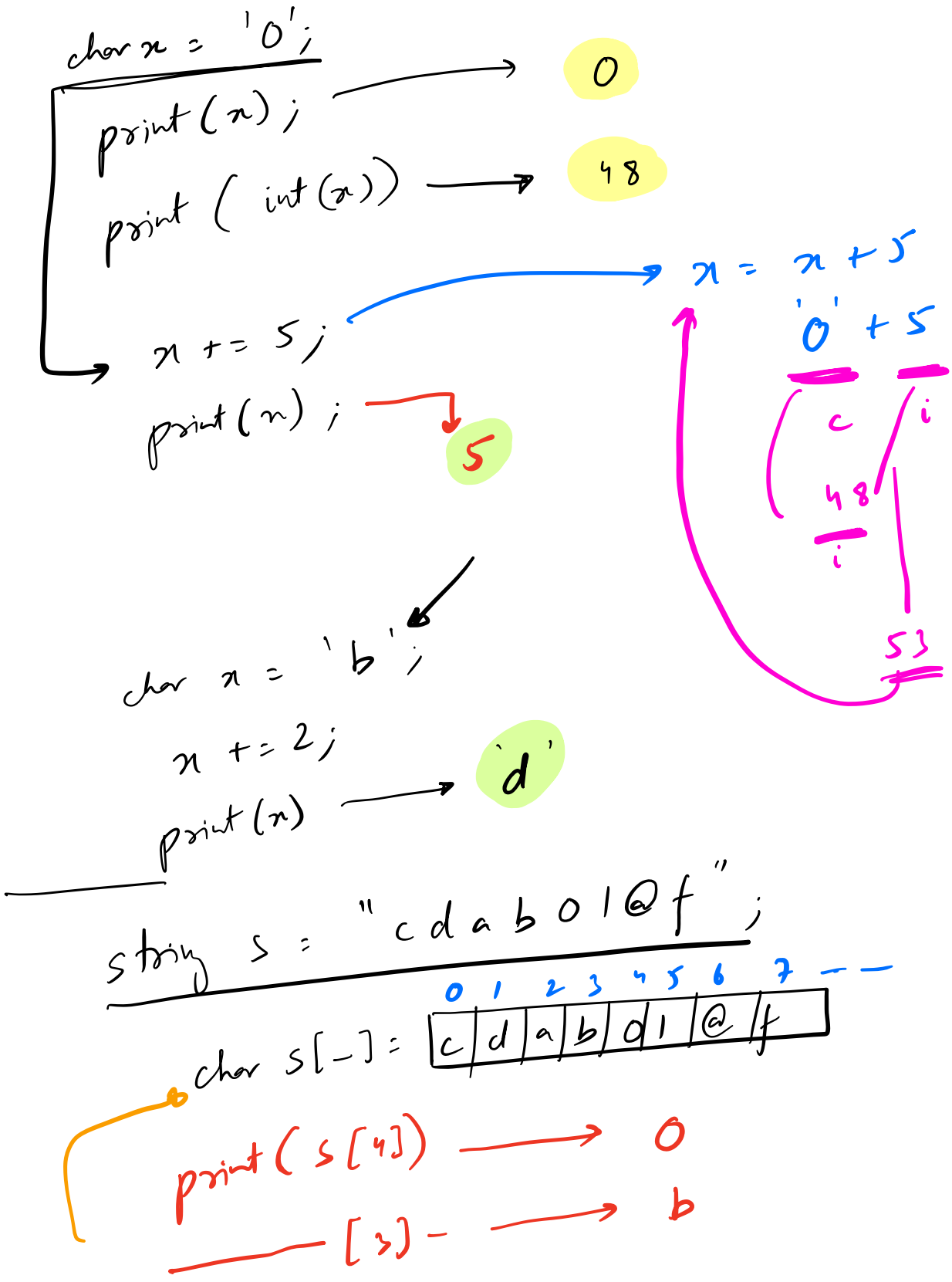
ASCII System

'A'	65	'a'	97	'0'	48
'B'	66	'b'	98	'1'	49
⋮	⋮	⋮	⋮	⋮	⋮
'Z'	90	'z'	122	'9'	57

```
char x = 'a';
```



97 → bin


char x = '0';
print(x);
0
print (int(x))
48
x += 5;
print(x);
5
x = x + 5
'0' + 5
c
i
48
i
53
char x = 'b';
x += 2;
print(x)
'd'
string s = "cdab01@f";
0 1 2 3 4 5 6 7
char s[_] = c d a b 0 1 @ f
print(s[4])
0
[3]
b

```
string s = "rahul ";
string y = "yadav";
string z = s + y;
print(z)
```

→ rahul yadav

Q Given a string. Toggle every char

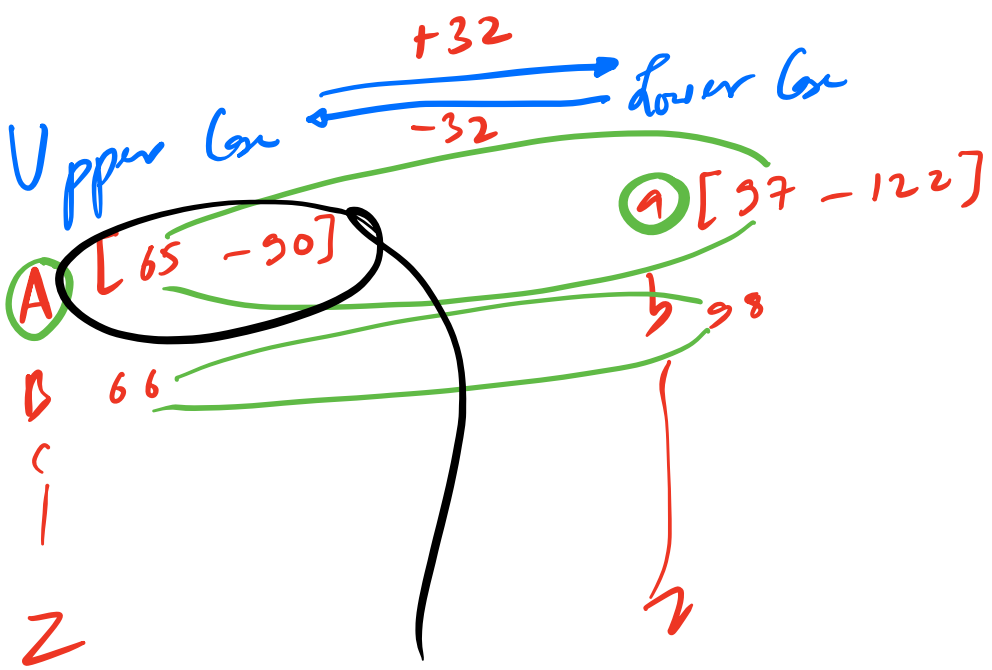
alphabets

Cap ⇄ Lower Case

aNa ConDA

↓

AnAcONda



```
for( i=0; i<s.size(); i++){
    if( 'A' <= s[i] && s[i] <= 'Z'){
        s[i] += 32;
    }else{
        s[i] -= 32;
    }
}
return s;
```

32 → 'a' - 'A'

TC = O(N)

SC = O(1)

		7 6 5 4 3 2 1 0
'A'	: 65	0 1 0 0 0 0 0 1
'B'	: 66	0 1 0 0 0 0 1 0
		0 1 0 0 0 0 1 1
'Z'	: 90	0 1 0 1 1 0 1 0

		7 6 5 4 3 2 1 0
'a'	: 97	0 1 1 0 0 0 0 1
'b'	: 98	0 1 1 0 0 0 1 0
		0 1 1 0 0 0 1 1
'z'	: 122	0 1 1 1 1 0 1 0

+32 = 2^5

```
for(i=0; i<S.size(); i++){
    S[i] = (S[i] ^ (1<<5));
}
ret S;
```

Q Given a string. Sort it alphabetically!

lower case

S = "axbaacd";

↓ SORT

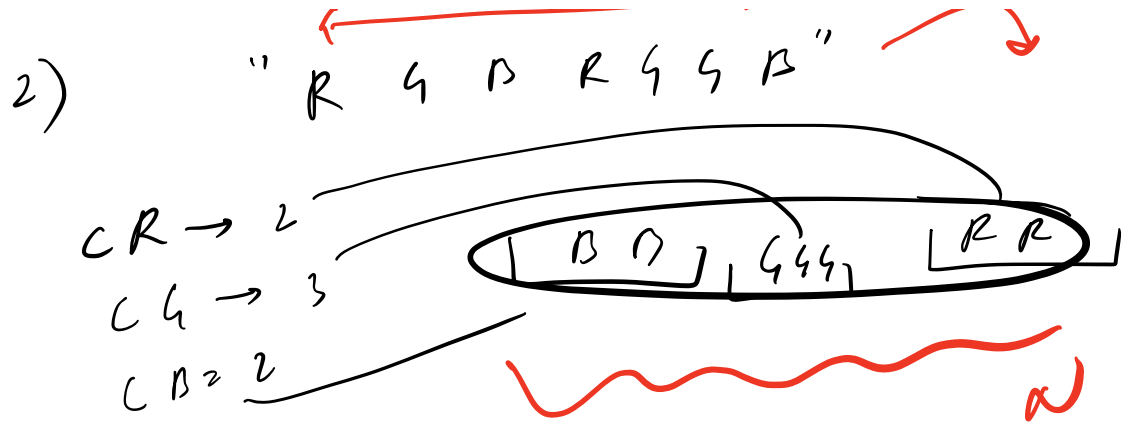
"aaabcdx"

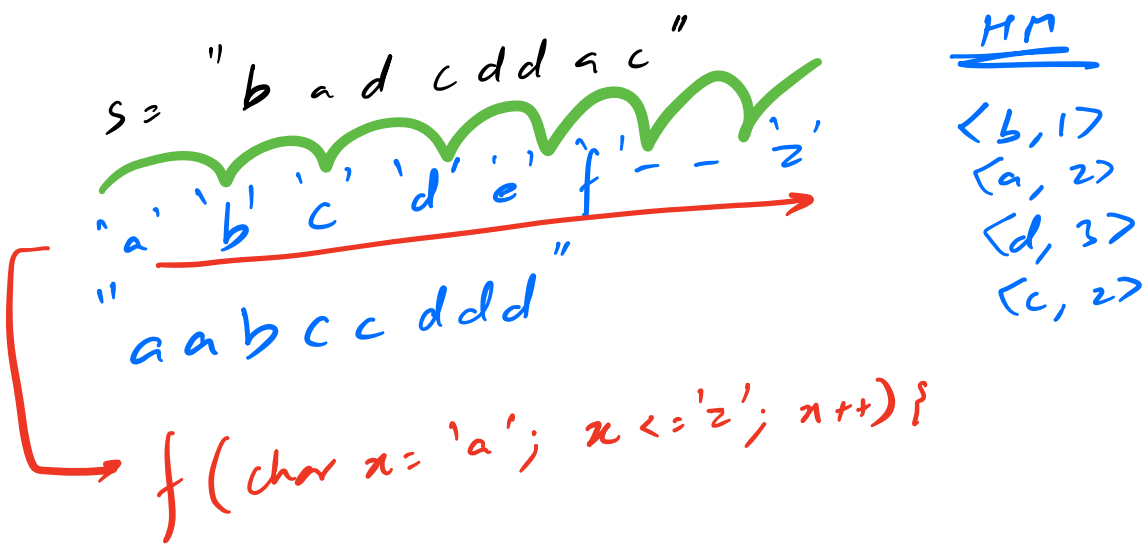
1) Using inbuilt sort + Comparator

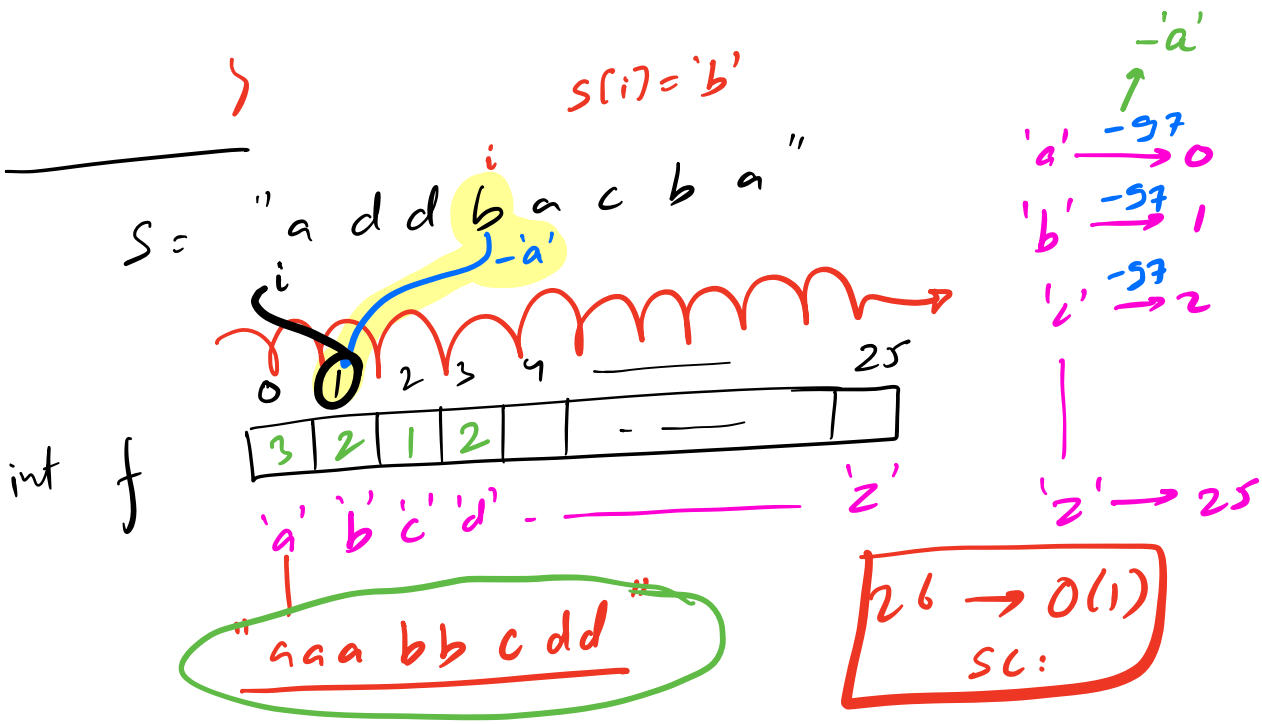
TC = O(N log N)

SC = O(1)

N


2)
"R G B R G G B"
cR → 2
cG → 3
cB = 2
B B
GGG
R R
W


s = "b a d c d d a c"
'a' 'b' 'c' 'd' 'e' 'f' - - 'z'
"a a b c c d d d"
HM
<b, 1>
<a, 2>
<d, 3>
<c, 2>
f(char x = 'a'; x <= 'z'; x++) {


}
s[i] = 'b'
s = "a d d b a c b a"
i
i
-'a'
0 1 2 3 4 25
3 2 1 2
int f
'a' 'b' 'c' 'd' - 'z'
"a a a b b c d d"
-'a'
'a' -97 → 0
'b' -97 → 1
'c' -97 → 2
'z' → 25
26 → O(1)
SC:

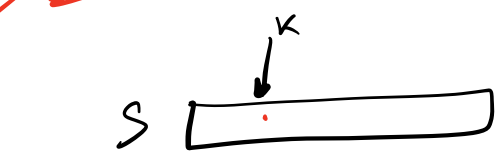
```
int f[26] = {0};
for (i = 0; i < s.size(); i++) {
    idx = s[i] - 'a';
    f[idx]++;
}
k = 0;
for (i = 0; i < 26; i++) {
    int numTimes = f[i];
    char ch = i + 'a';
    for (j = 0; j < numTimes; j++) {
        s[k] = ch;
        k++;
    }
}
```

N

f[s[i] - 'a']++;

26

2 + 'a' → 'c'



N

TC = N + N + 26

TC = O(N)

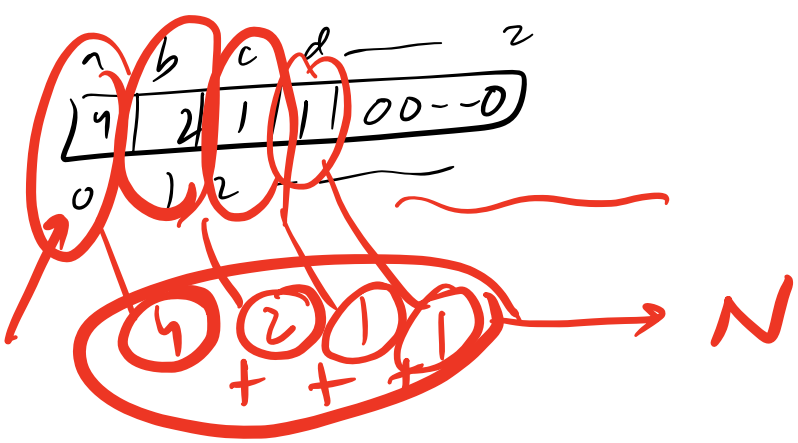
O(N + K)

K : Alphabet set size

SC = O(1)

O(K)

s = a b a a b c d a



④ Substring

↳ A continuous part of a string

[SAME AS SUBARRAY]

S = "a b x c d f"

"bxc" ✓

Q Given a string. check if any of it's substring is a palindrome!

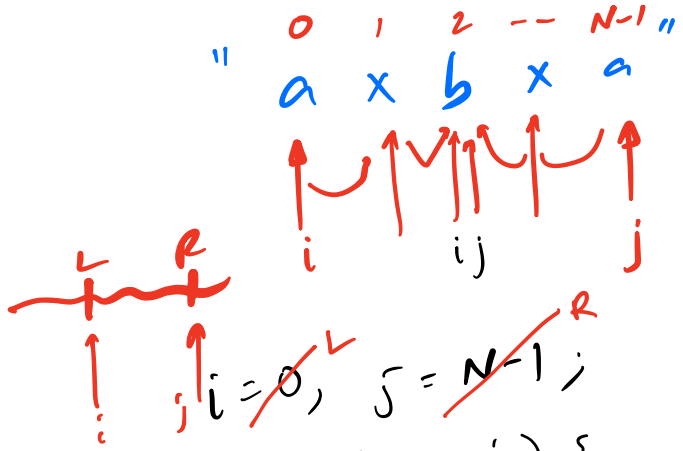
S = "xabc"

SINGLE chars are palindromes

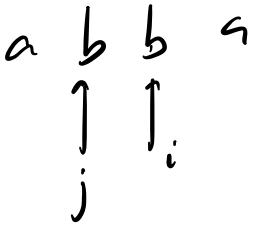
ALWAYS TRUE →

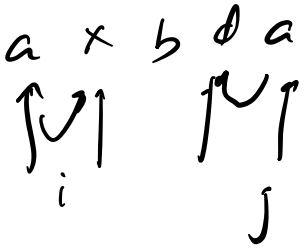
Q Given a string. check if it is a palindrome!

"a b a b a"




```
i = 0; j = N-1;
while( i < j ) {
    if( s[i] != s[j] ) {
        ret false;
    }
    i++; j--;
}
ret true;
```





TC = O(N)

SC = O(1)

Q Given a string. Find the largest palindromic len of/ substring!

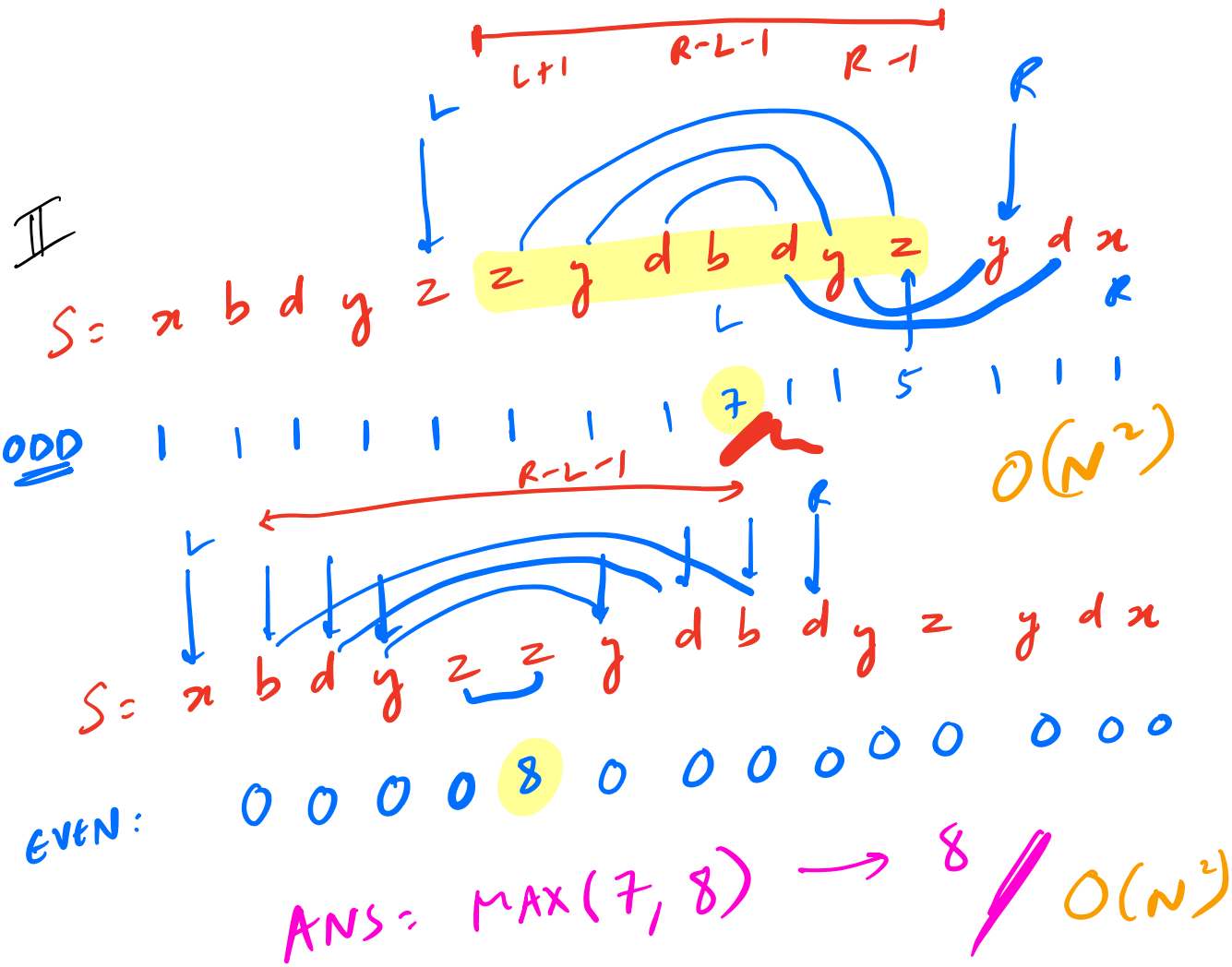
S: a b a c a b → 5

I) BF

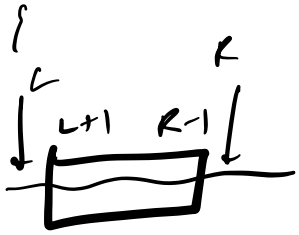
```
ANS = 0
for(L:0 ⟶ N-1) {                    N²
    for(R:L ⟶ N-1) {
        if(isPalindrom(S, L, R)){    N
            ANS = max(ANS, R-L+1)
        }
    }
}
ret ANS;
```

TC: O(N³)

SC: O(1)




```
int expand(S, L, R){
  while(L>=0 && R < S.size()){
    if(S[L] != S[R]){
      break;
    }
    L--; R++;
  }
  ret R-L-1;
}
```



3 4
L R

```
ANS = 0
for ( i = 0; i < N; i++) {
    ANS = max(ANS, expand(S, i-1, i+1));
    if ( i < N-1){
        ANS = max(ANS, expand(S, i, i+1));
    }
}
print ANS;
```

TC = $O(N^2)$

SC = $O(1)$

DP

N^2

Rabin Karp + BS

$N \log N$

Manacher's Algo

N