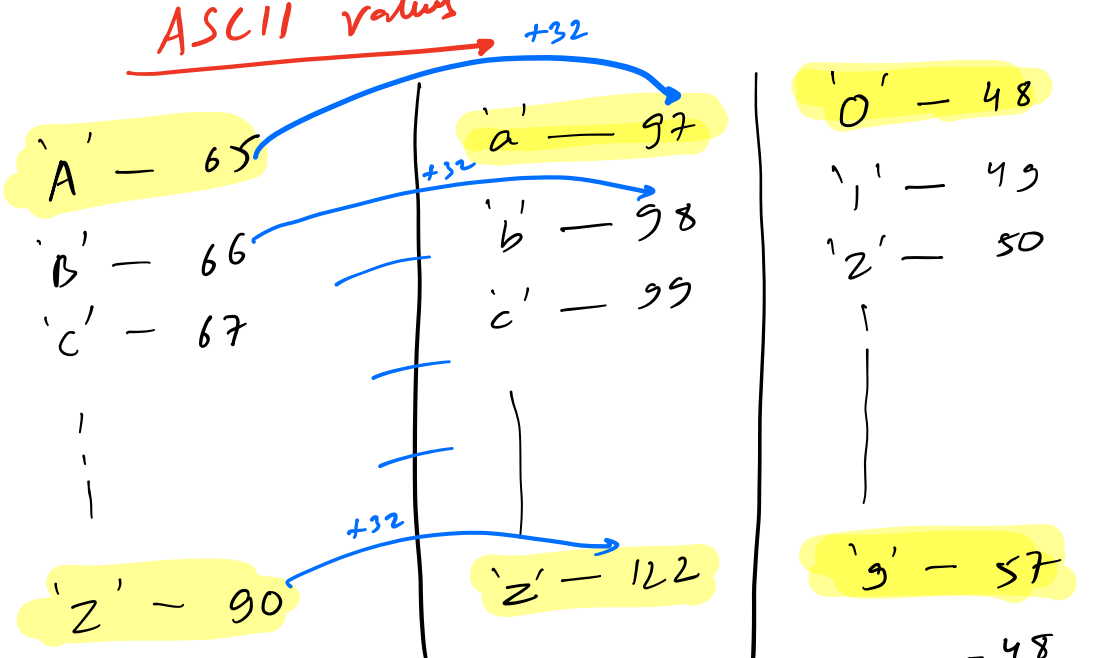
String

- array of characters
- sequence of chars.

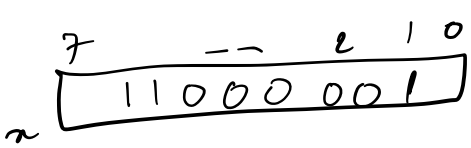
Character : 1 / 2 B

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

ASCII values




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




```
char x = '0';
```

48

53

```
x = x + 5;
print(x);  → 5
print(int(x)) → 53
```

print(x): 0

string s = "abc"; ⇒ char s[3] = "abc";

print(s[1]) ⟶ 'b'

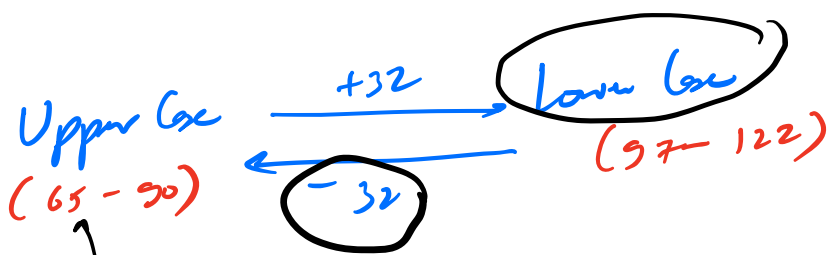
Q: Given a string. Toggle every char!

Caps ⇄ Lower

alphabets

Q: "aNa CondA"

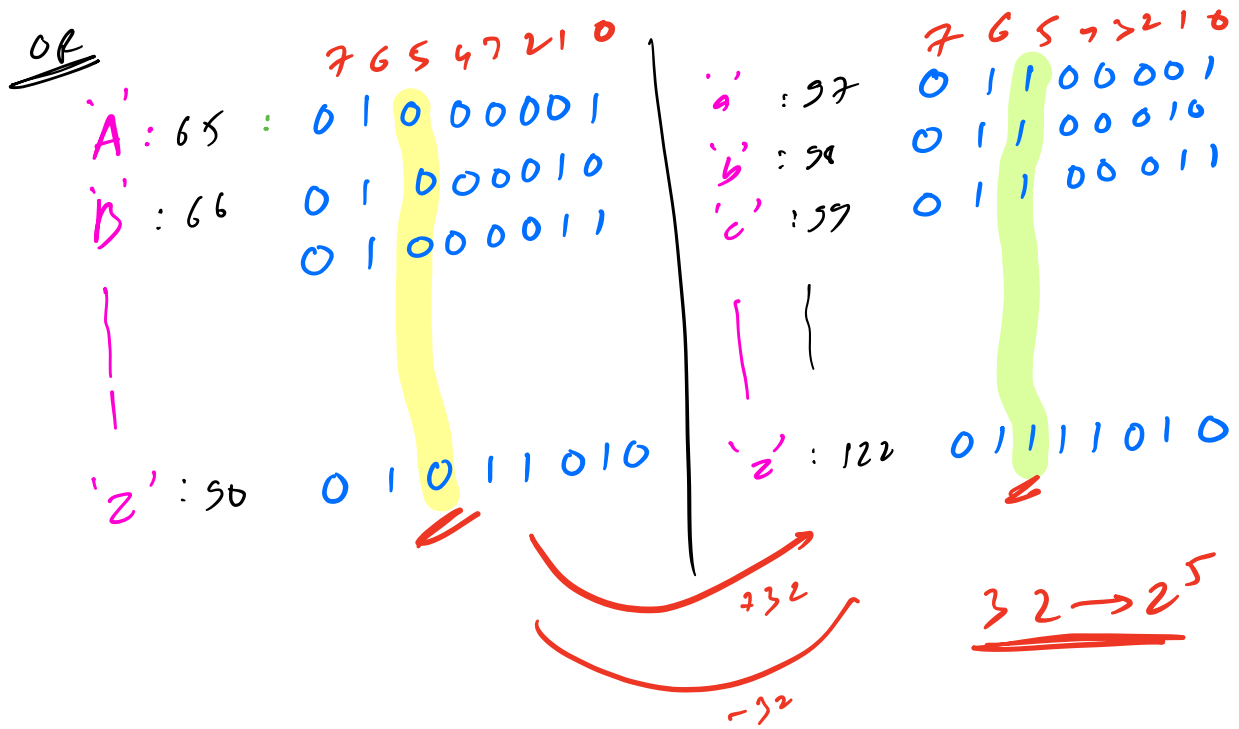
↳ "AnAcONDa"




```
N = s.length();
for(i=0; i<N; i++){
    if( s[i] >= 65 && s[i] <= 90){
        s[i] = s[i] + 32;
    }
    else{
        s[i] = s[i] - 32;
    }
}
ret s;
```

TC: O(N)

SC: O(1)

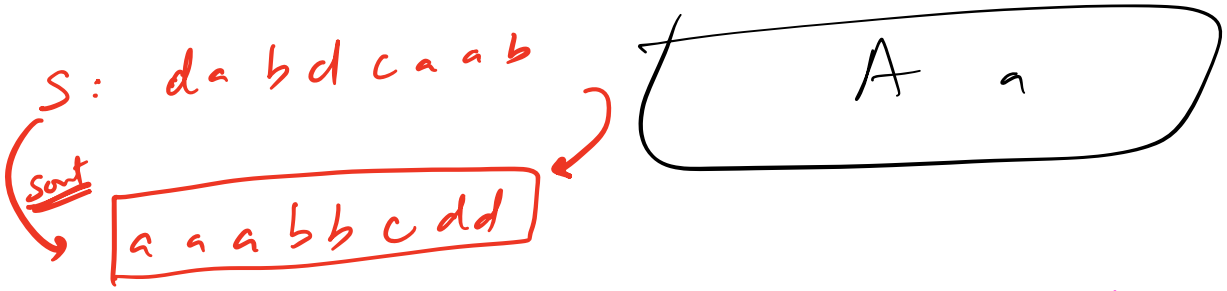



```
for (i=0; i<N; i++){
    s[i] = (s[i] ^ (1<<5));
}
```

Q Given a string. Sort it in alphabetical order.

lower case

a b c — z

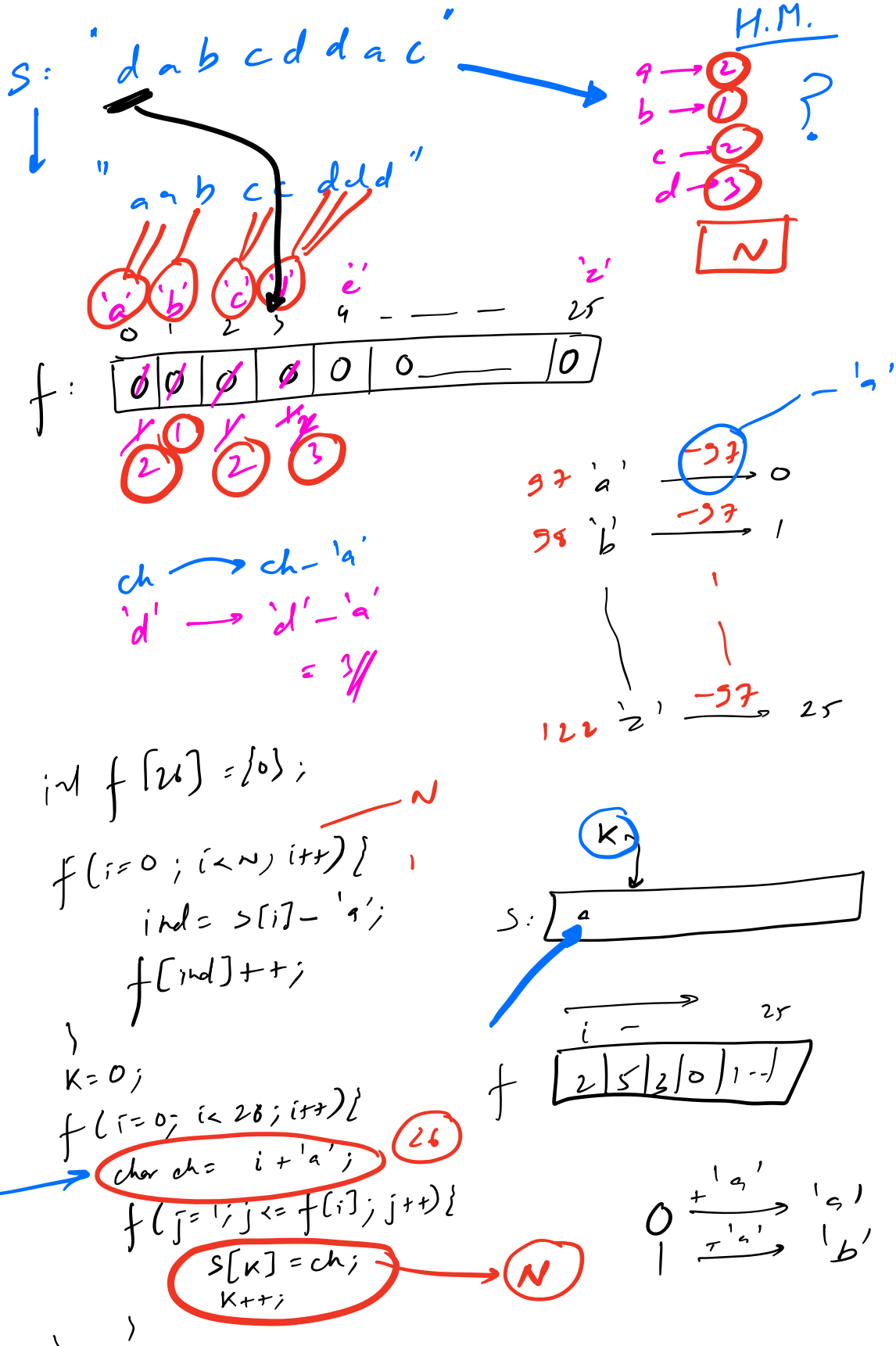


1. Sort

$O(N^2)$

2. Comparator + In built sort

TC: $O(N \log N)$


S: "dabcddac"
"aabcddd"
H.M.
a → 2
b → 1
c → 2
d → 3
N
f:
0 1 2 3 4 25
'a' 'b' 'c' 'd' 'e' 'z'
ch → ch - 'a'
'd' → 'd' - 'a'
= 3
-97
'a'
'b'
'z'
122
25
int f[26] = {0};
for(i=0; i<N; i++){
ind = s[i] - 'a';
f[ind]++;
}
K=0;
for(i=0; i<26; i++){
char ch = i + 'a';
for(j=1; j<=f[i]; j++){
s[K] = ch;
K++;
}
}
N
26
K
S:
a
f
i
25
2 5 3 0 1 ...
+'a'
-'a'
'a'
'b'

$TC = O(N + 26 + N)$

$TC = O(N)$ → $O(N+K)$

$K =$ # of alphabets

$SC = O(26)$

$SC = O(1)$ → $O(K)$

① Substring →

Similar to a sub Array

→ A continuous part of a string.

S: a b c x a b c y

② Given a string. Check if it is a palindrome!

S: a b c b a

isP(L,R)

i = 0 (L), j = N-1 (R)

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

TC: O(N)

Q Given a string. Find the longest palindromic substring!

S = a b a c a b

L <= R

L

1) ans = 0

TC: O(N^3)

```
for( L = 0; L < N; L++) {
    for(R = L; R < N; R++) {
        // [L, R]
        if( isP(L,R) == true) {
            ans = max(ans, R-L+1);
        }
    }
}
```

i i+1

S: x b d g z z g d b d g z g d x

ODD: 1 1 1 1 1 1 1 1 7 1 1 5 1 1 1

EVEN: 0 0 0 0 8 0 0 0 0 0 0 0 0 0

len → R-L-1

ODD: i → center

L → i-1

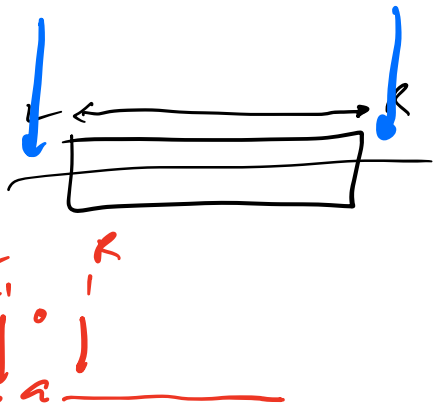
R → i+1 → i

EVEN: i, i+1 → center

L → i

R → i+1

```
expand ( S, L, R) {
    while( L>=0 && R<N) {
        if( S[L] != S[R]) {
            break;
        }
        L--, R++;
    }
    ret (R-L-1);
}
ANS = 0;
for( i=0; i<N; i++) {
    ANS=max(ANS, expand( S, i-1, i+1));          // ODD
    if (i<N-1 && S[i]==S[i+1]) {                 // EVEN
        ANS=max(ANS, expand( S, i, i+1));
    }
}
ret ANS;
```



TC = O(N²)

SC = O(1)

DP
O(N²)

Rabin Karp + BS
O(N log N)

Manacher's Algo
O(N)