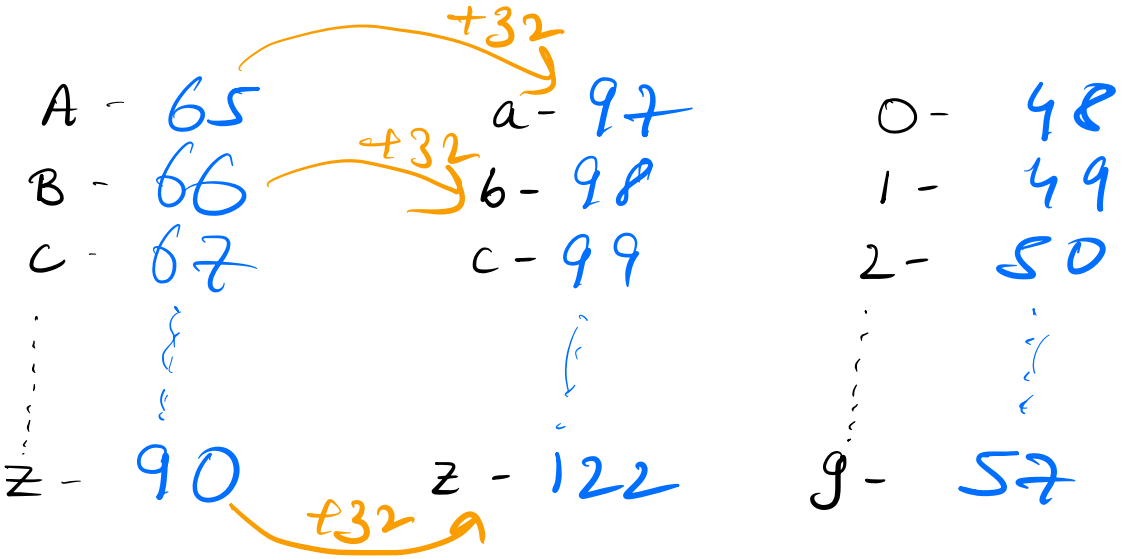
Strings

- Array of characters
- Sequence of characters
- Bunch of characters → {abd} {adb} ✗

0 1 2 3 4 5
o m a n s h ✓

How are characters stored ? ASCII



Obs: Between capital & small letters, ASCII value difference is 32

Java Specific

String: Cannot change {Immutable}

char s[] : Can change any index.

String Builder : Can change any index

class in Strings

Q1) Given a char s[], Toggle every character.
uppercase & small case
Capital → small
small → capital.

Eg: AnaConDa → aNAcONdA

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

TC: O(N) SC: O(1)

	6	5	4	3	2	1	0
65	1	0	0	0	0	0	1
66	1	0	0	0	0	1	0
67	1	0	0	0	0	1	1
⋮							
90	1	0	1	1	0	1	0

	6	5	4	3	2	1	0
97	1	1	0	0	0	0	1
98	1	1	0	0	0	1	0
99	1	1	0	0	0	1	1
⋮							
122	1	1	1	1	0	1	0

1 ^ 1 = 0
0 ^ 1 = 1

for(i=0; i<N; i++) {

s[i] = s[i] ^ (1<<5)

}

74 ⟹ J

74 ^ (1<<5)

74^32
= 106
= j

Q2 Given char s[], sort the array.
lowercase alphabets.

Sol^n 1 → Arrays.sort TC: nlogn

Eg - d a b a c d b
a a b b c d d

a → 2
b → 2
c → 1
d → 2
e → 0
f → 0
g → 0
⋮
z → 0

-97

a → 97 → 0
b → 98 → 1
c → 99 → 2
⋮
z → 122 → 25

index = ascii - 97

0	1	2	3	-------	25
2	2	1	2		

```
int c[26] = {0};

for(i=0; i<n; i++) {
    idx = s[i] - 97
    c[idx]++
}

int cur_idx = 0
for(i=0; i<26; i++) {
    for(j=0; j<c[i]; j++) {
        s[cur_idx] = 'a' + i
        cur_idx++
    }
}
```

b

N

	0	1	2	3
c →	2	2	1	2

d a b a c d b

a a b b c d d

TC: $2N \Rightarrow O(N)$

adbadba

0	1	2	3	----
~~0~~	~~0~~	0	~~0~~	
~~1~~	~~1~~		1	
~~2~~	2		2	
3				
3	2	0	2	

aaabbdd

i	j	$C_0 + C_1 + C_2$
0	C_0	$---+C_{25}$
1	C_1	
2	C_2	N
25	C_{25}	

Q3 Given a string. Find longest palindrome substring.

```
   0 1 2 3 4 5 6 7
S: a b c d e f g h
```

s[4:6] : efg len = 6-4+1

Palindrome: String that reads the same forward & backward.

LEVEL
LEVEL

NAMAN

NITIN
NITIN

```
isPalin(s, start, end) {
    i = start
    j = end

    while(i<j) {
        if(s[i] != s[j])
            return false.
        i++
        j--
    }
    return true
}
```

NITIN

TC: O(N)

```
ans = 0
for (i=0; i<n; i++) {
    for (j=i; j<n; j++) {
        if (ispalin(s, i, j))
            ans = max(ans, j-i+1)
    }
}
```

TC: $O(N^3)$

$(i, j) \Rightarrow [i+1, j-1]$

$\Rightarrow j-i-1$

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
x	b	d	y	3	3	y	d	b	d	y	3	y	d	x

a x x a

b c y y c b

a a b b a a

a x a

b q r q b

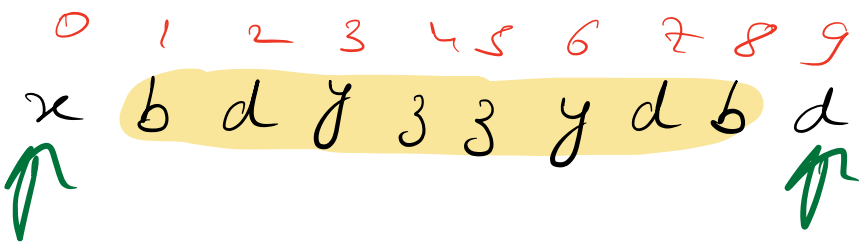
abede

Take all possible centers & expand.

2 types of centers

2 char

1 char.



left = 0
right = 9

What is actual length?

9 - 0 - 1 = 8

j - i - 1

```
int extend (char s[], int i, int j){
    while ( s[i] == s[j] && i >= 0
                         && j < n )
        i--;        j++;
    return j-i-1;
}
```

```
int largest_pal_substring(char s[]){
    int ans = 0
    for(i=0; i<n; i++){
        // take idx i as center.
        len = expand(s, i, i)
        ans = max(ans, len)
    }

    for(i=0; i<n-1; i++){
        // take s[i], s[i+1] as centes.
        len = expand(s, i, i+1)
        ans = max(ans, len)
    }

    return ans.
```

TC: $O(N^2)$ SC: $O(1)$

a b a c d c a b

1 3 7

d dou 4

0 0 0 0 0 0 1

7 6 5 4 3 2 1

1

0 0 0 0 0 1 0

2

0 0 0 0 0 1 1

3