__/__/__

283. Move Zeroes

[0, 1, 0, 3, 12]
 1 0 ↳ In place

[1, 3, 12, 0, 0]

[0, 1, 0, 3, 12]
 1 0

[0, 1, 0, 3, 12]

index = 0

[1, 3, 12, 0, 0]

whenever you see a non-zero element, nums[index++] = nums[i];

290. Word Pattern

"abba"

hash[a]++

hash[a] → 1

hash[b] → 2

a → 1

b → 1

0, 1, 1, 0

"abba a" → "dog cat cat dog"

vec(a) → 1 2 2 1

int cnt = 1;

for(auto x : str){

if(hash[x])

vec.pb(hash[x]);

else {

hash[x] = cnt++;

vec.pb(hash[x]);

}

}

cnt = 0;

for(auto x : a)

if(hash[x])

vec.pb(hash[x])

else {

hash[x] = cnt+1

vec.pb(hash[x]);

}

dog cat cat dog

dog → 0 0 1 1 0

vec(b) → 1 2 2 1

Compare a & b

↓ ↓ ↓
"dog cat cat dog"

__/__/____

Split

vector<string> splitString (string s, ~~delim~~ string delim) {

```
vector<string> ans;
int start = 0;
int end = s.find(delim);
while(end != string::npos) {
    ans.pb(s.substr(start, end - start));
                            len
    start = end + delim.length();
    end = s.find(delim, start);
                        after
}
ans.pb(s.substr(start));
```

str.find("hello") , str.find("hello", 5); → start from idx 5

↳ returns the position of the 1st occurence if found

↳ else returns string::npos

str.substr(3, 5);

start len

str.substr(str.find("pos")); // if 2nd param not specified ⇒ to the end

↳ from pos ⇒ to the end

OR we can map each word ⟷ letter

jab a b y a

an dog cat cat dog

a → dog

May solution is better

__/__/__

299. Bulls & Cows

secret = "1807"
guess = "7810" 1A 3B

1807 xA
7810

1→1 7→1
0→1 1→1
7→1 0→1

secret = "1123"
guess = "0111" 1A 1B

1123 1→2 2→1 3→1

0111
0→1
1→3

1→(2) − 1 = (1)

1807 7888
7810 8888

<8> 8→(8→2)
8→1

303. Range Sum Query Immutable

0 1 2 3 4 5
[-2, 0, 3, -5, 2, -1]
-2 0 1 -4 -2 -3

20/8 → 9/2 → 3 → 3/1 → 1

326. Power of Three

```
bool isP3(int n) {
    if (n<=1)
        return 0;          // Imp
    while (n%3==0) {
        n/=3;
    }
    if (n==1)
        return true;
    else
        return false;
}
```

Imp

342. Power of Four (32-bit integer)

2^6 2^4 2^3 2^2 2^0

$(2^n) = (4^{n/2})$

$= n/2 \geq 1$

2^6

$\boxed{n \geq 2}$

$n \geq 2$ and $n \Rightarrow$ even

344. Reverse String

In-place ≠ constant space complexity.

Recursion is ~~not in-place~~

in-place bcoz no other D.S is beingused but it is not constant space as it uses recursion stack.

Base → { n=0, [ ], "", 'A','a', 0 }

__/__/__

345. Reverse Vowels

→ "hello"

→ "holle"

Leetcode

Leetcode

```
string reverseVowel(string s){
int n=s.length();
if(n==0||n==1) return s;
int i=0, j=n-1;
while(i<=j){
    char l = s[i];
    if(!vowel(l)){
        i++;
        continue;
    }
char r = s[j];
if(!vowel(r)){
    j--; continue;
}
char tmp = s[i];
s[i] = s[j];
s[j] = tmp;
i++;
j--;
}
```

Imp

Most Imp

Imp

when both are vowels

```
bool vowel(char c){
c = tolower(c); → A, a.
if(c=='a'-----)
{
}
```

349. '0' of two Arrays.

same approach → hash map + intersection

350. '1' of two arrays-II

367. Valid Perfect Square

Figuring out the sqrt of a number by Binary Search.

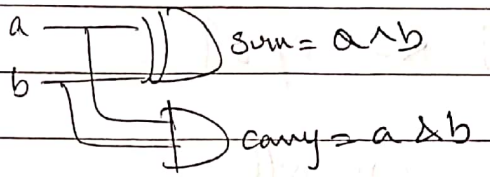
```
bool isPerfSq (int num) {

    if (num == 0) return true;

    // l = 1, r = num;
    while (l <= r) {
    // mid = l + (r-l)/2; // sqrt assumed
    // tmp = mid * mid;

    if (tmp == num)
        return true; // Perf Sq
    else if (tmp > num)
        r = mid - 1;
    else
        l = mid + 1;

    }
    return false;
}
```

__/__/__

371. Sum of Two Integers

- add 2 integers without using (+) sign

Half Adder



x y
→ add(15,32)d

```
while(y!=0) {
    carry = x & y;
    x = x ^ y;
    y = carry<<1;
}
```

```
int add (int x, int y) {
    if (y == 0)
        return x;
    unsigned int carry = (x & y);
    return add(x ^ y, (carry) << 1);
}
```

Note
left shift op cannot be applied on negative numbers.

$$(a+b) = (2^a \times 2^b)$$

$$= \frac{\log(2^a \times 2^b)}{\log_2} = \log_2 2^{(a+b)} = a+b$$

__/__/____

PayPal

383. Ransom Note

("aa", "aab") → true

mp[] <= mp[]

389. Find the Difference.

s = "abcd" a x x x mp

s = "abcde" a x x x x mp

392. Is Subsequence

s → t

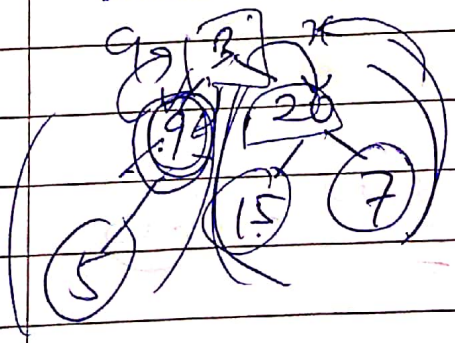
check if 's' is a subsequence of 't'

s = "abc"

t = "ahbgdc"

i=0

404. Sum of Left Leaves



__/__/__

```
int sum(node* root) {
    if (root == NULL)
        return 0;

// check if it's a parent of left leaf
    if (root->left)
        if (root->left->left == NULL and root->left->right == NULL)
            return root->left->val + sum(root->right);

return sum(root->left) + sum(root->right);

}
```

Scanned by CamScanner