LeetCode 53. Maximum Subarray

Kadane

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

→ 6 → [4, -1, 2, 1]

[-2, 1, (-3), 4, -1]

↳ either a part of subarray / new subarray from this pt.

```
S=0
for(i) {
  S += a[i]    ans = max(ans, sum);
  if (S<0) {
      S=0;
  }
}
```

1) Case 1: all -ve [(-2), -5, -3] → ans = max-element (max_element(), max.end())

2) Case 2: all +ve [2, 5, 3]

3) Case 3: +/- [-2, 5, -1]

→ 53(Index)

→ 53(Sum)

Divide & Conquer

{1, 6, -5, 7, -9, -11}

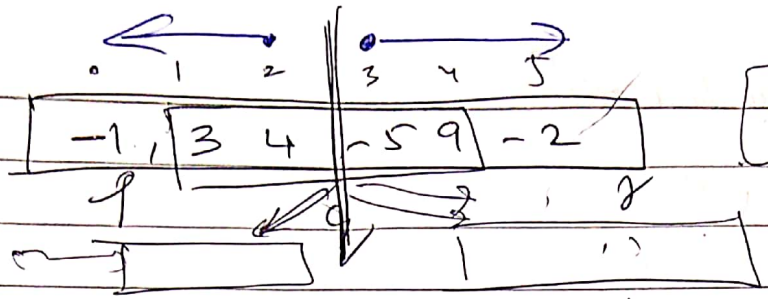
{1, 6, -5} {7, -9, 11}

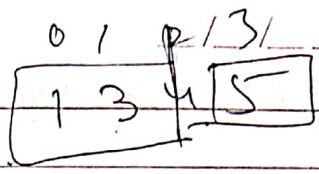
7

{1, 6} {5} {7, -9} {11}

6

{1} {6} {7} {-9}





- any subarray will lie 1) either completely to the left
2) either completely to the right
3) cross the border

↓

So we know 2 indices already,
go left ← and go right →
keeping track of max sum.

D&C

DEC1 DEC2

```
int MaxSubArray (A, p, r) { T(n)
    if (n == 1) return A[0];
    q = (p+r)/2;
    L = MaxSubArray (A, p, q); T(n/2)
    R = MaxSubArray (A, q+1, r); T(n/2)
    C = MaxCrossing (A, p, q, r); Θ(n)
    return max (L, R, C);
}
```

$T(n) = 2T(n/2) + \Theta(n)$

same as merge sort → $O(n \log n)$

__/__/____

58. Length of Last Word leetcode (*)

Python3 →

```
arr = s.split()
if (len(arr) == 0:
    return 0
else:
    return len(arr[-1])
```

Note (WA) → the string also contains empty space characters.

" Hello world "
4 3 2 1 0

(1) — Since we want the len(last word), overlook the spaces & then start counting.

(2) In C++, string elements are 'char' not "string"

```
if(s[i] == ' ')
```

(3)

```
for (j=0; j<n; j++){

};
```

j → (n)

```
for(j=0; j<n; j++){
    if(s[j] == ' ')
        break;
}
```

j

__/__/__

66. Plus One

I/P: [1, 2, 3]

O/P: [1, 2, 4]

Python approach — Easiest

num = "123"

- just collapse the array into a list → list str +1 from the array

- for Leetcode vector inputs, have a condition where you check for empty vector.

This is done so as to handle overflow cases of BigInts.

[1, 2, 3, 9] → [1, 2, 4, 0]

→ {2, 1, 3, 0}

→ {0, 2, 1}

= {2, 1, 5, 1}

2130
21
2151

ans=0;

[1 | 5 | 2 | 5]

500 + 20 + 5

1, 2, 5, 9
9, 1, 5, 3
1, 0, 4, 1, 2

1, 2, 3, 5
0, 2, 3, 5

67. Cpp Binary Sum

String a = "1010"
String b = "11"

Same logic as prev.

Just modify the sum

0/1 0/1 0/1 → max → 1+1+1 → (3)

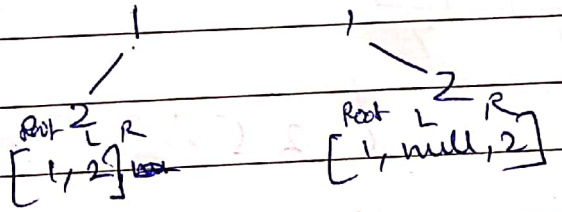
int sum = v1[i] + v2[j] + carry;

sum = add(sum) → int add(int a){
int arr[] = {0, 1, 10, 11};
return arr[a];
}

100. Same Tree

→ to check if 2 Binary Trees are equal

→



1, 2, -1

1, 2

If tree is given in pre-order style, you also store the 2 pre-orders in 2 vectors.

for → void preOrder(node* root, vector<int> &v){
if (root == NULL){
v.pb(-1); // → replacement for null
return;
}
v.pb(root->val);
preOrder(root->left, v);
preOrder(root->right, v);
}

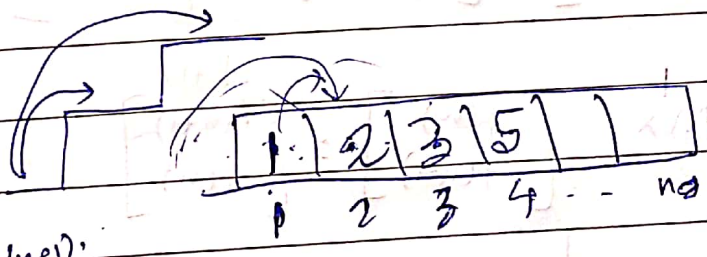
Note

(Inorder + Pre-Order) or (Inorder + Post-Order) gives an unique soln.

100. Same Tree (Expert Notes)

```
int check (node* p1, node* p2){
    if (p1==NULL && p2==NULL)
        return 1;
    if (p1!=NULL and p2!=NULL){
        if (p1->data==p2->data and check(p1->left, p2->left) and
            check(p1->right, p2->right))
            return 1;
    }
    return 0;
}
```

70. Climbing Stairs



dp[n+1];

Don't say dp[1]=1; dp[2]=2; ✗ input might be n=1 and it will give an error.

~~for(i=1 -> n+1 if i==1 -> dp[1]=1;~~ if i==2, dp[i]=2;

dp[i]=dp[i-1]+dp[i-2];

}

Matrix Exponentiation

→ $f(n) = f(n-1) + f(n-2)$

$f(0), f(1) \rightarrow$ known

so (2×2)

$$M \times \begin{bmatrix} f(n) \\ f(n+1) \end{bmatrix}_{k^{th}} = \begin{bmatrix} f(n+1) \\ f(n) \end{bmatrix}_{(k+1)^{th}}$$

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \times \begin{bmatrix} f(n) \\ f(n+1) \end{bmatrix} = \begin{bmatrix} f(n+1) \\ f(n) \end{bmatrix}$$

$\Rightarrow a \cdot f(n) + b \cdot f(n+1) = f(n+1) \rightarrow a=1, b=1$

$c \cdot f(n) + d \cdot f(n+1) = f(n) \rightarrow c=1, d=0$

$$\rightarrow \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \times \begin{bmatrix} f(n) \\ f(n+1) \end{bmatrix} = \begin{bmatrix} f(n+1) \\ f(n) \end{bmatrix}$$

$$\rightarrow M \times \begin{bmatrix} f(n) \\ f(n+1) \end{bmatrix} = \begin{bmatrix} f(n+1) \\ f(n) \end{bmatrix}_{(K+1)}$$

Now, $M \times \begin{bmatrix} \quad \\ \quad \end{bmatrix}$

$$M \times \begin{bmatrix} f(n+1) \\ f(n) \end{bmatrix} = \begin{bmatrix} f(n+2) \\ f(n+1) \end{bmatrix}_{(n+2)^{th}}$$

$$M^k \times \begin{bmatrix} f(n) \\ f(n+1) \end{bmatrix} = \begin{bmatrix} f(n+k) \\ f(n+k-1) \end{bmatrix}$$

$$\times \begin{bmatrix} f(0) \\ f(1) \end{bmatrix}_{n=0} = \begin{bmatrix} f(\\ \end{bmatrix}$$

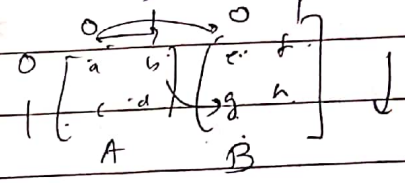
0 1 1 2 3 5 8 13 21
0 1 2 3 4 5 6 7 8 $n=0$ (8)

__/__/__

$n=1$

$$M^k \times \begin{bmatrix} f(1) \\ f(0) \end{bmatrix} = \begin{bmatrix} f(k+1) \\ f(k) \end{bmatrix}$$

$$M^{k-1} \times \begin{bmatrix} f(1) \\ f(0) \end{bmatrix} = \begin{bmatrix} f(k) \\ f(k-1) \end{bmatrix} \rightarrow \text{Nth fibonacci No}$$



$a^b \rightarrow \log(b)$ time

∴ Nth fibonacci $\rightarrow O(\log n)$.

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} f(1) \\ f(0) \end{bmatrix} = \begin{bmatrix} f(n) \\ f(n-1) \end{bmatrix}$$

$a \cdot f(1) + b \cdot f(0) = f(n)$

$a = f(n)$

$f(0) = 0$
$f(1) = 1$

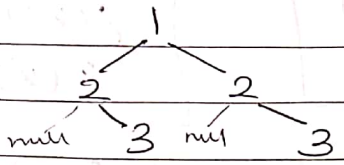
$A^n \rightarrow A^{n/2} \times A^{n/2}$ (even)
$\rightarrow (A^{n/2} \times A^{n/2} \times A)$ node

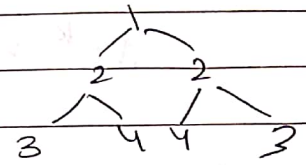
Optimized Power Function

pow(x, n) $\rightarrow x \times$ pow(x, n-1)

```
int pow (int x, int n) {
    if (n==0) return 1;
    if (n==1) return x;
    int val = pow (x, n/2);
    val = val * val;
    if (n&1)  val = x * val;
    return val;
}
```

101. Symmetric Tree

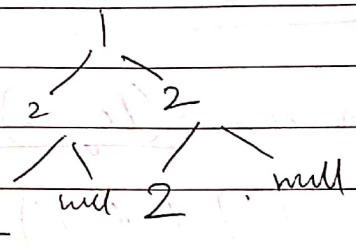




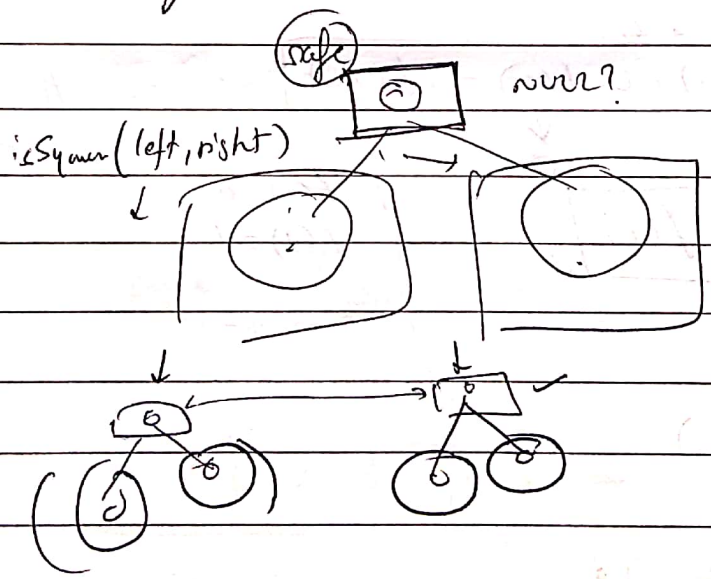
null 2 3 | null 2 3

×

3 2 4 | 4 2 3



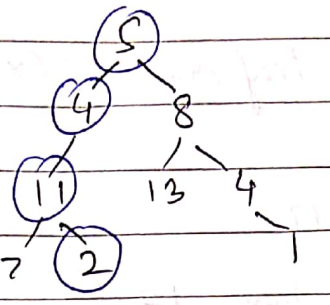
- comparing palindromically the inOrder traversal O.N seem to work.



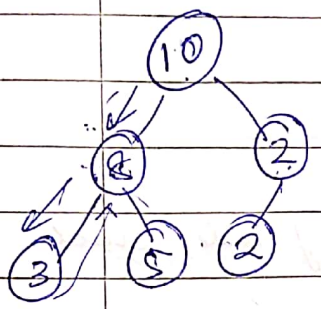
__/__/__

112. Path Sum / 257. Binary Tree Paths.

Root-to-leaf path



sum = 22



len = 1, 2, 3

10-8-3 → 10, 8, 3

10-8-5

10-2-2

10, 8, 3

preOrder 10-8-3

10-8-5-4

10-2-2

so overwriting on the vector happens.

(*). the preOrder method has a T.C of $O(n^2)$

→ string tmp = "";

char x = 2 + '0';

tmp = tmp + (x) + "->" ✓

→ Pass vectors around causes TLE.

→ Never forget to add endl after every test case.

"10->8->3" + 10->8

10->8->

str(20) // Py

↪ to_string(root->val); (*).

__/__/____

121.

Best Time to Buy & Sell Stock

[7, (1), 5, 3, (6), 4]

Buy (min) ↓ ↑ Sell (max)

- Maintain a dp array maxR that stores max till now from Right

	7	1	5	3	6	4
dp[] →	7	6	6	6	6	4

←

ans[] →	0	6-1	6-5	6-3	6-6	4-4

ANS = max(0, 5, 1, 3, 0, 0) = (5)

122. Best time to buy & Sell Stock II

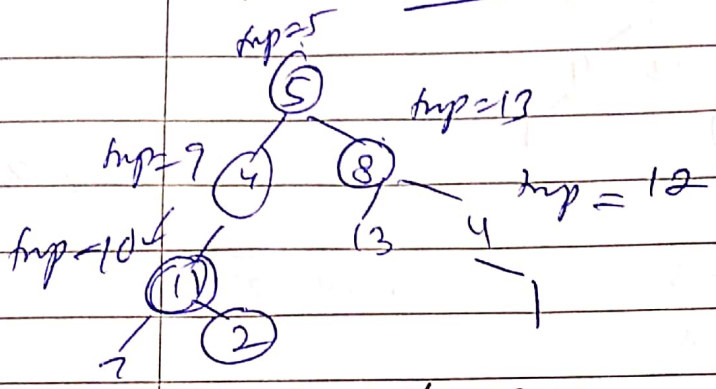
[7, (1) (5) (3) 6 (4)]

Small → big

__/__/____

112. Path Sum

- find if root → leaf sum == 22 (X).



- Same DFS traversal and when you reach any leaf node, check if (tmp + root → val == sum), then pb (1) ~~xx~~

else

pb(-1).

- In the end, check for 1's in the vector.

118. Pascal's Triangle

n=0 ⟶ if (n==0) return {5~~13~~ return { } ✓ [ ]

n=1 [[1]]

n=2 [[1], [1, 1]]

111. Min Depth of Binary Tree

dfs + check in ~~xxx~~ end-vectors for min depth.