Page 1

PSA Class 11
Ruchika Shashidhara
NU 002245068

Techniques to Solve Problems:

- Brute Force
- Greedy Algo
- Divide & Conquer
- Dynamic Programming
- Back Tracking

1) Find the ~~max~~ indexes that can be taken from 1D array, that cannot be adjacent to get max sum

	0	1	2	3	4	5
Ex:	6	1	2	7	3	5

{6}, {1}, {2}, {7}, {3}, {5}

{6, 2}, {6, 7}, {6, 3}, {6, 5}
8, 13, 9, 11

{6, 2, 5} 13

{6, 7, 5} (indexes 0, 3, 5) 18 ⇒ best indexes {0, 3, 5} for max sum

Brute Force: Find all combinations, then find max sum of each combination

0	1	2	3
1	2	3	4

{1} 1 {1, 3} 4
{2} 2 {1, 4} 5
{3} 3 {2, 4} 6
{4} 4

sum non-adj
∴ Max index (soln) = {1, 3}

works only for small case

Time complexity = O(N!)

2) Given amount, find the min no of coins that can be given for a purchase of that item (cost)

Ex: Amount = \$1
Coins Available: 1¢, 5¢, 10¢, 25¢

Cost of item = 13¢
Return amount = 87¢

$$\begin{array}{r} 87 \\ -75 \\ \hline 12 \end{array}$$

87 > 25?
87 // 25 = 3
87 % 25 = 12
12 > 10?
12 // 10 = 1
12 % 10 = 2
2 > 5? 2 > 1?
2 // 1 = 2
2 % 1 = 0

∴ We need
3, 25¢
1, 10¢
2, 1¢ coins
Total coins = 6

classmate

Date

Page (3)

Coins Available :

12 ¢, 10 ¢, 5 ¢, 1 ¢

change, Return Amount : 16 ¢

Greedy Approach doesn't work every time,

16 > 12 ?

16 // 12 = 1

16 % 12 = 4

4 > 10 ?   4 > 5 ?   4 > 1 ?

4 // 1 = 4

4 % 1 = 0

Greedy Algo gives

4, 12 ¢

1, 1 ¢

Total coins = 5

But best case (least no. of coins) :

1, 10 ¢

1, 5 ¢   Total coins = 3

1, 1 ¢

3) Divide & conquer

Ex: Binary Search

4) Stamp Denominations (DP)

90, 30, 24, 10, 6, 2, 1

Ex: 34 cent : 24 cent, 10 cent

DP:
Base case
Memoization - Remember prev answers
Optimal Sub Structure

1¢, 3¢, 4¢

Change = 6   Ans: 3¢, 3¢ (2 coins)

0	1	2	3	4	5	6
∞	∞	∞	∞	∞	∞	∞
-1	-1	-1	-1	-1	-1	-1

	0	1	2	3	4	5	6
1-x	0	∞	∞	∞	∞	∞	∞
3-x	0	-1	-1	-1	-1	-1	-1
4-x							

	0	1	2	3	4	5	6
1	0	1	∞	∞	∞	∞	∞
3-x	0	1	-1	-1	-1	-1	-1
4-x	0	1	1				

	0	1	2	3	4	5	6
1-1,1	0	1	2	∞	∞	∞	∞
3-x	0	1	2	-1	-1	-1	-1
4-x	0	1	1				

3

	0	1	2	3	4	5	6
1-1,1,1	0	1	2	3	∞	∞	∞
3-3,1	0	1	2	1		-1	-1
4-x	0	1	1	3	-1		

	0	1	2	3	4	5	6
1	0	1	2	1	1	∞	∞
4	0	1	1	3	4	-1	-1

	0	1	2	3	4	5	6
3	0	1	2	1	1	2	∞
4	0	1	1	3	4	4	-1

	0	1	2	3	4	5	6
3	0	1	2	1	1	2	2
4	0	1	1	3	4	4	3

Space complexity = $\Theta(N)$

Time complexity = $O(N * \text{No. of Denominations})$

If No. of denom is less → constant

$\approx O(N)$

1, 2, 6, 10, 24, 30, 90

0	1	2	3	4	5	6	7	8	9
0	1	1	2	2	3	1	2	2	3
0	1	2	2	2	1	6	1	2	1

10	11	12	13	14	15	16	17	18	19
1	2	2	3	3	4	2	3	4	2
10	1	2	1	2	1	6	2	1	10

20	21	22	23	24	25	26	27	28	29
2	3	3	4	1	2	2	3	3	4
10	1	2	1	24	1	2	1	2	1

30	31	32	33	34
1	2	2	2	2
30	2	2	1	10

3
- 1^{1} 2^{1} = 2
- 2^{1} 1^{1} = 2

5
- 1^{1} 4^{2} = 3
- 2^{1} 3^{2} = 3

4
- 1^{1} 3^{2} = 3
- 2^{1} 2^{1} = 2

7
- 1^{1} 6^{1} = 2
- 2^{1} 5^{2} = 3
- 6^{1} 1^{1} = 2

8
- 1^{1} 7^{2} = 3
- 2^{1} 6^{1} = 2
- 6^{1} 2^{1} = 2

9
- 1^{1} 8^{2} = 3
- 2^{1} 7^{2} = 3
- 6^{1} 3^{2} = 3

11
- 1^{1} 10^{1} = 2
- 2^{1} 9^{3} = 4
- 6^{1} 5^{2} = 3
- 10^{1} 1^{1} = 2

12
- 1^{1} 11^{2}
- 2^{1} 10^{1}
- 6^{1} 6^{1}
- 10^{1} 2^{1}

and so on

33
- 1^{1} 32^{2} = 3
- 2^{1} 31^{2} = 3
- 6^{1} 27^{3} = 4
- 10^{1} 23^{4} = 5
- 24^{1} 9^{3} = 4
- 30^{1} 3^{2} = 3

34
- 1^{1} 33^{2} = 3
- 2^{1} 32^{2} = 3
- 6^{1} 28^{3} = 4
- 10^{1} 24^{1} = 2
- 24^{1} 10^{1} = 2
- 30^{1} 4^{2} = 3

0/1 Knapsack

0/1 → means either take into soln or not

Knapsack → get the sum of max value 'v' with least sum of cost/weight 'w' within a sack of max weight = k

		TV	Laptop	Cellphone
Ex:	V	5	3	4
	W	3	2	1
#		Item 1	Item 2	Item 3

max weight = 5

	V	W	
TV, Laptop =	$8	5kg	
TV, Cellphone =	$9	4kg	✓ choose this

	0	1	2	3	4	5
No item	0	0	0	0	0	0
1	0	0	0	5	5	5
1, 2	0	0	3	5	5	8
1, 2, 3	0	4	4	7	9	9

	0	1	2	3	4	5
No item	0	0	0	0	0	0
1	0	0	0	1	$\boxed{1}$	1
2	0	0	1	0	$\boxed{0}$	1
3	0	1	1	1	1	$\boxed{1}$

Max Profit = 9

Item 1
v {3} Remaining Profit = 9 − 4 = 5
w 1 Remaining Bag Weight = 5 − 1 = 4

Item 2
Nothing is selected

Item 3
v {3, 1} Remaining Profit = 5 − 5 = 0 → Alg Terminate
w {1, 3} Remaining Bag Weight = 4 − 3 = 1

We get the same answer for any order of selection of items

1	or 2	or 3
1, 2	2, 3	3, 2
1, 2, 3	2, 3, 1	3, 2, 1

and so on

if n is no. of items

if k is max bag weight

space complexity = O(n * k)   2(n * k) space

1) House Robber - Leetcode

	0	1	2	3	4	5
	6	1	2	7	3	5
	6	6	8	13	13	18
V	6	6	8	3	3	5
K	0	0	2	3	13	3

i1) Steal 1, N Steal 6

i2) steal 2+6=8, NS 6

i3) S 7+6=13, NS 8

i4) S 3+8=11, NS 13

i5) S 13+5=18, NS 13

∴ output = {5, 3, 0} index   {5, 7, 6} nums

How to generate enumerations or (permutations) for Brute Force:

Ex:

a	b	c	d	
0	0	0	0	{ }
0	0	0	1	{d}
0	0	1	0	{c}
0	0	1	1	{c, d} X
0	1	0	0	{b}
0	1	0	1	{b, d}
0	1	1	0	{b, c} X
0	1	1	1	{b, c, d} X
1	0	0	0	{a}
1	0	0	1	{a, d}
1	0	1	0	{a, c}
1	0	1	1	{a, c, d} X
1	1	0	0	{a, b} X
1	1	0	1	{a, b, d} X
1	1	1	0	{a, b, c} X
1	1	1	1	{a, b, c, d} X

Computing 2^n : $1 \ll n$

Bit shift operator

Ex: 1<< 0001 = 1 = 2^0

1<< = 0010 = 2 = 2^1

1<< 0100 = 4 = 2^2

1<< = 1000 2^3

Ex: Set = 0101
How do we know if a/b/c/d is in the set?

	{a}	{b}	{c}	{d}
	0101	0101	0101	0101
AND	1000	0100	0010	0001
	0000	0100 {b}	0000	0001 {d}

XNOR Truth Table

			XNOR
0	0	⇒	1
1	0	⇒	0
0	1	⇒	0
1	1	⇒	1

AND

0	0	0
1	0	0
0	1	0
1	1	1

If result == {b} {d}
it is in the set

	1	2	3
	1,	2,	4
V	1	2	5
K	0	1	3

[0, 2]

{4, 1} 5

0 1 2 3
0 1 2 4
0 1 2 $\boxed{5}$ {4, 1}
0 1 2 4
0 1 2 3 4
0 1 2 3 4
0 1 2 4 $\boxed{6}$
0 1 2 3 4
{4, 2}

	0	1	2	3
	1	2	3	4
V	1	2	4	6
K	0	1	2	3

{4, 2} 6

[1, 3]

	0	1	2	3	4	5
	6	1	2	7	3	5
V	6	6	8	13	13	18
K	0	0	2	3	3	5

Page 11

$n = 6$

	0	1	2	3	4	5	6
	0	6	1	2	7	3	5
v	0	6	6	8	13	13	$\boxed{18}$
k	0	6	6	2	7	7	5

{5, 7, 6}
[0, 3, 5]

	0	1	2	3	4	5	6	7
	0	3	100	4	1	5	100	6
v	0	3	100	100	101	105	201	$\boxed{201}$
k	0	3	100	100	1	5	100	100

{100, 1, 100}
[2, 3, 5]

	0	1	2	3	4	5
	0	5	99	199	99	7
v	0	5	99	204	204	$\boxed{211}$
k	0	5	99	199	199	7

{7 199, 5}
[0, 2, 4]

	0	1	2	3	4	5
	0	1	3	1	3	100
v	0	1	3	3	6	$\boxed{103}$
k	0	1	3	3	3	100

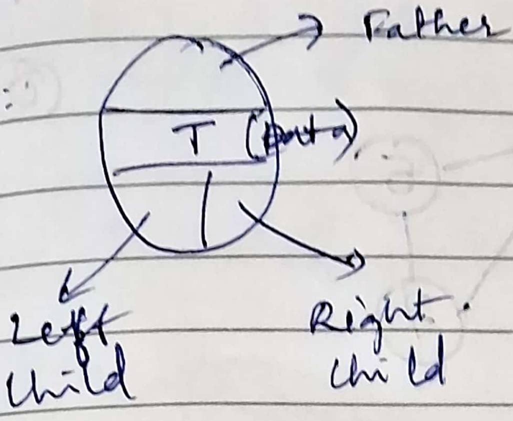
{100, 3} [1, 4]

	0	1	2	3	4	5	6	7	8	9	10	1
	0	6	3	10	8	2	10	3	5	10	5	
v	0	6	6	16	16	18	26	26	31	36	36	3
k	0	6	6	10	10	2	10	10	5	10	10	3

{3, 10, 10, 10, 6} [0, 2, 5, 8, 10

Trees :

Node :

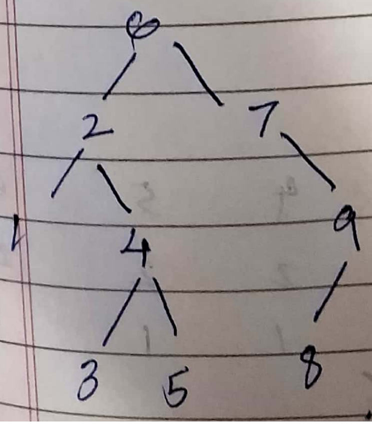


∴ For a tree with n nodes, I require 3n pointers

- We store trees as nodes rather than a python list, since ~~it~~ trees are not a complete trees always. So python lists are not efficient since many trees spots will be empty.
- Root of a tree → node that has no father
- Leaf of a tree → node that has no right child, no left child

	visit	left	right
Preorder	V	L	R
Postorder	L	R	V
Inorder	L	V	R

Postorder without recursion requires 2 stacks



Preorder
6 2 1 4 3 5 7 9 8

```
f (node n) {
    if (n) {
        f (n.val); f (n.left);
                   f (n.right);
    }
```