1, 3, 7, 2, 1, 3, 2

res = 0 → 0000

1 0001

0001

3 0011

0010

7 0111

0101

2 0010

0111

0111

1 0001

0110

3 0011

0101

2 0010

0111 = 7

1, 2, 3, 4, 2, 1

xorall → 3^4 = 0011
0100
0111 = 7

Rightmost set bit :→

1 1 2 2 3 3 7

rsb 7 & −7 = 0111
& 1001
0001 → rsb

−7 = 1000
+1 = 0001

2's com 1001

Two groups

G1 → 1, 3, 1

n1 = 0 $\frac{1}{1}$ $\frac{1}{1}$ $\frac{1}{1}$

G2 → 2, 4, 2 (0 / 1 above each)

n2 = 0

if (arr[i] & rsb) → true
1 3 1

num1 ^ = arr[i]
1 ^ 3 ^ 1 = 3

if (arr[i] & rsb) → false
2 4 2

num2 ^ = arr[i]
2 ^ 4 ^ 2 = 4

Toggle Compiler Design DMS

5 → 0101

Not Truth Table

I	O
0	1
1	0

I	O
F	T
T	F

~5 → 1010
= 10

−499 = −(−499) − 1 = 498
500 = −500 − 1 = −501

−6 = 10

−6 → abs (−6) = 6 → 0110
1's 1001
+1 1
TV ← 1010

↓
PV

2's com

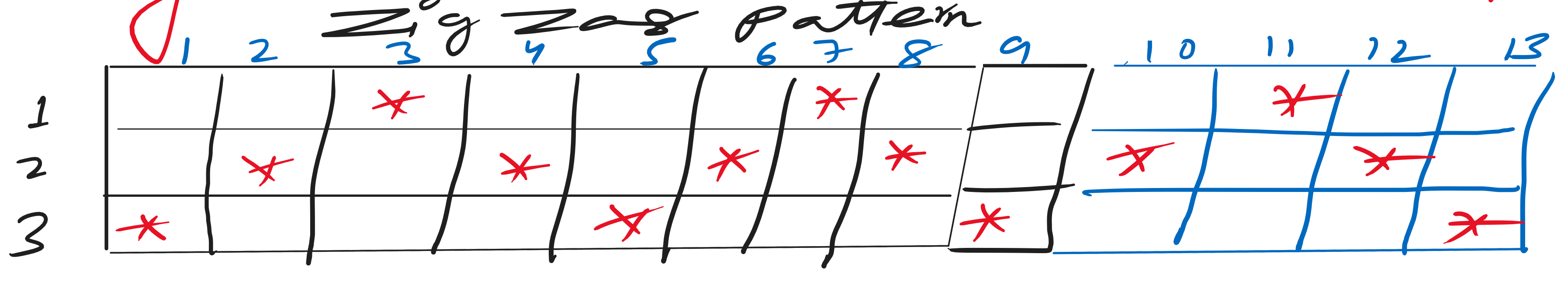
~5 → 2, 3, −6, 10

Formula :→ $\boxed{\sim n = -n - 1}$

Dynamic Patterns :→

2 mins TCS NOT

Zig Zag pattern



row = 3

col = 9, 13, 17, 21, 25 and so on

r1 = 3, 7, 11 % 4 = = 3 *, " "

r2 = Even Number

r3 = 1, 5, 9, 13 % 4 = = 1

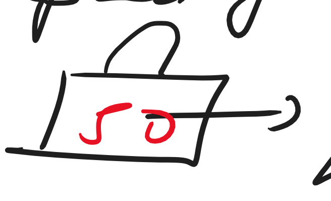
Fractional Knapsack :→

N → items

wt → [ ]

val → [ ]

Knapsack capacity



50 − 10
= 40 − 20
= 20 kg − 20
= 0

N = 3, W = 50

val → [60, 100, 120]
wt → [10 kg, 20 kg, 30 kg]

Maximize the value

kg
10 × 6 + 20 × 5 + 20 × 4
60 + 100 + 80
= 240 kg

20 × 5 = 100
30 × 4 = 120
50 kg → 220 ✗

Per unit value
1. $\frac{60}{10} = 6$
2. $\frac{100}{20} = 5$
3. $\frac{120}{30} = 4$

6, 5, 4 →

How to calculate the per unit value :

vector< pair <double, Item>> V;

50 → 10 = 40

↓ puv {60, 10}

$\frac{V}{wt}$

puv, Item

pair (6, 60, 10), (5, 100, 20), (4, 120, 30)

puv Item puv Item puv Item

struct Item {
int value ;
int weight;
}

pair. first puv

pair. second Item (v, wt)