Coin Change Problem

if arr = [1|2|3] (0 1 2) target = 5

you have to find minimum no. of element → to reach the target SUM.

{1,1,1,1,1} → 5
{1,2,1,1} → 4
{2,2,1} → 3
{3,1,1} → 3
{3,2} → 2 → minimum no. of element to reach target SUM.

Note Target Reach krne ke we have 2 way

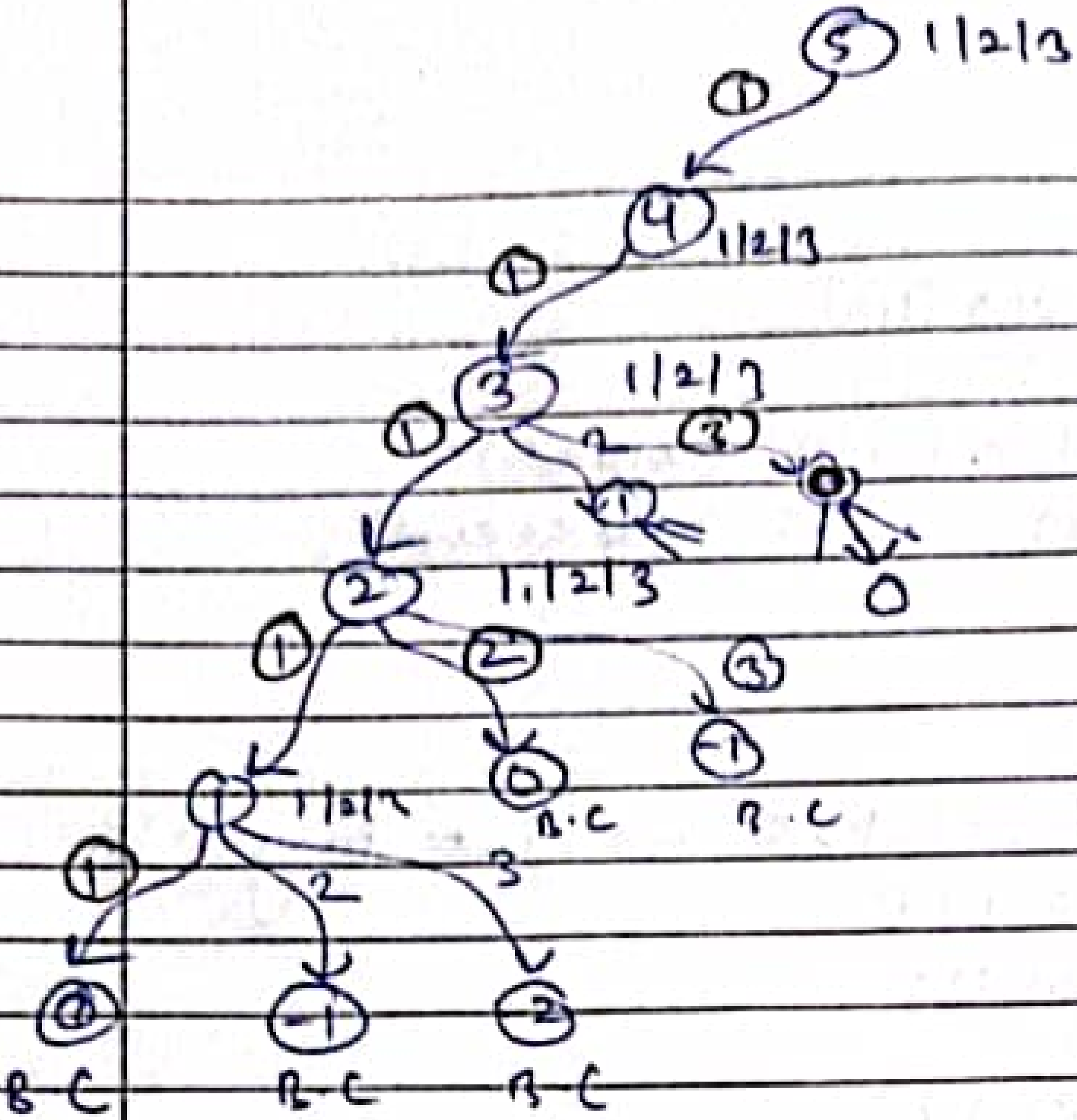
1) target → 0 (if target == 0) means reach target

original value ko decrease krega 0 tk jane ke lie

2) 0 → 5(target) → target Reach hojaega, but here use Additional variable, jo ki 5 ko store krega, jisse hum 0 me update krke 5 ko reach krw.

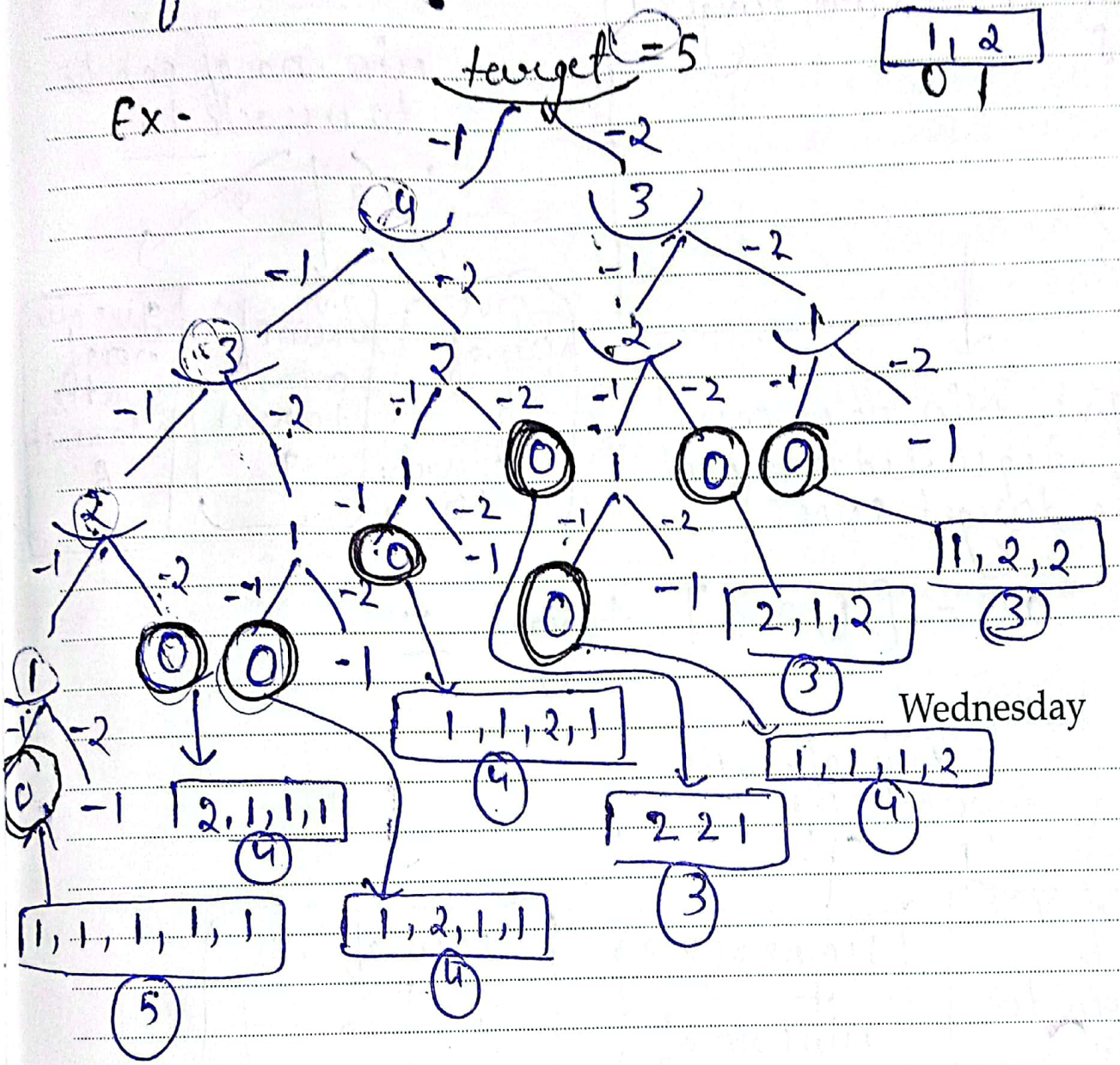
original ko store for comparison

PAGE No
DATE



If both equal means, we reach target

Ex-



Wednesday 20

21 Thursday

Sun	Mon	Tue	Wed	Thu	Fri	Sat
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		

```
int solve (arr, target)
{

}
```

give min. no. of coins required to reach the target SUM

min. no. of coins to reach 11

1 | 2 | 5

give min. of coins to reach 10

give min coins to reach 9

give min no. of coins to reach 6

arr = [1, 2, 5] target = 11

ye sb hume function dekha. hai

22 Friday

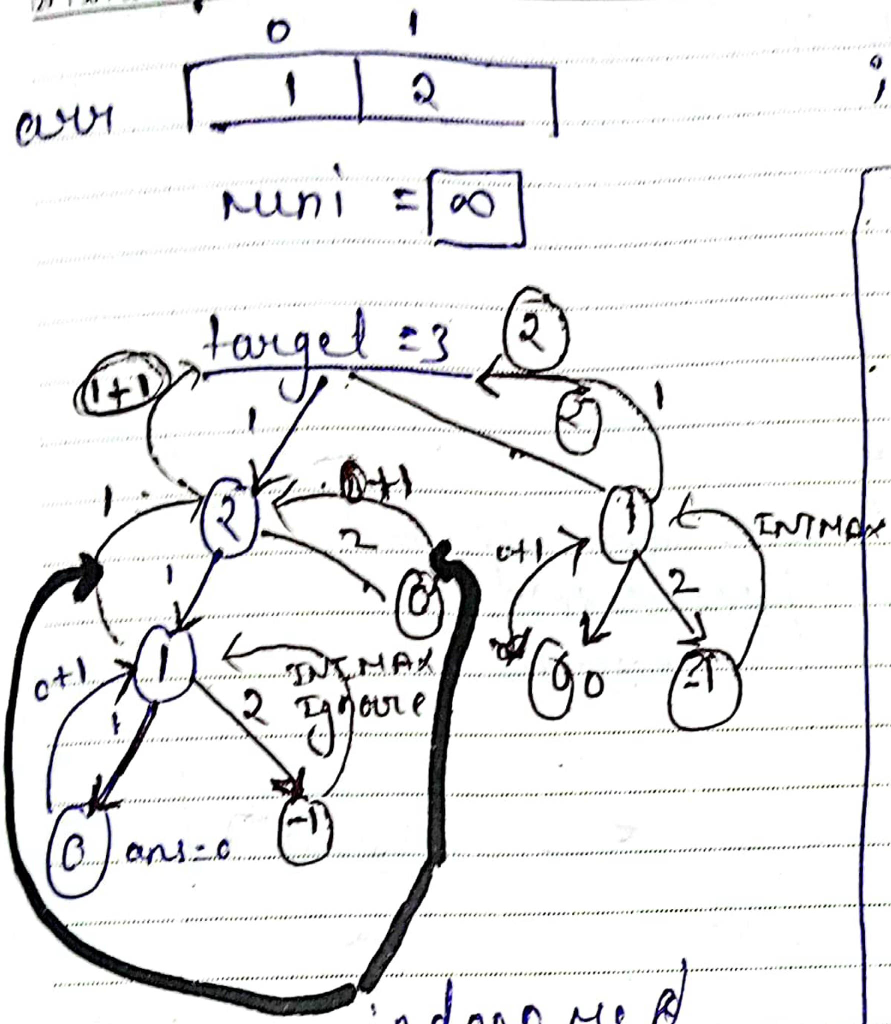
Min. of coins to reach 10

1 | 2 | 5

1 coin of 1₹ + min coin to 10₹

1 coins of 2₹ + min coin of 9₹

1 coin of 5₹ + min coin of 6₹



in dono me se jo minimum hoga wo upar jayega

```
int solve(arr, target)
{
    if(target==0)
        return 0;

    if(target<0)
        return INT_MAX

    int mini = INT_MAX;
    for(i=0; i<n; i++)
    {
        int ans =
        solve(arr, target-arr[i])

        if(ans!=INT_MAX)
        {
            mini = min(mini, ans+1)
        }

    return mini;
}
```

Cut into segments

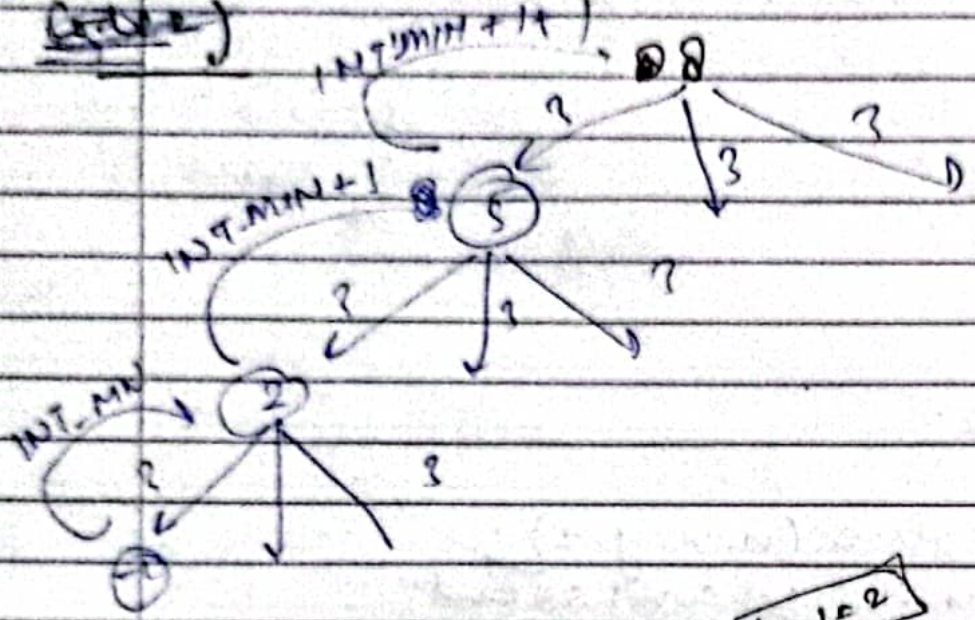
ip → N = rod length.
Maximum no of segments but only with x, y, z.

eg. N = 7
x = 5
y = 2
z = 2

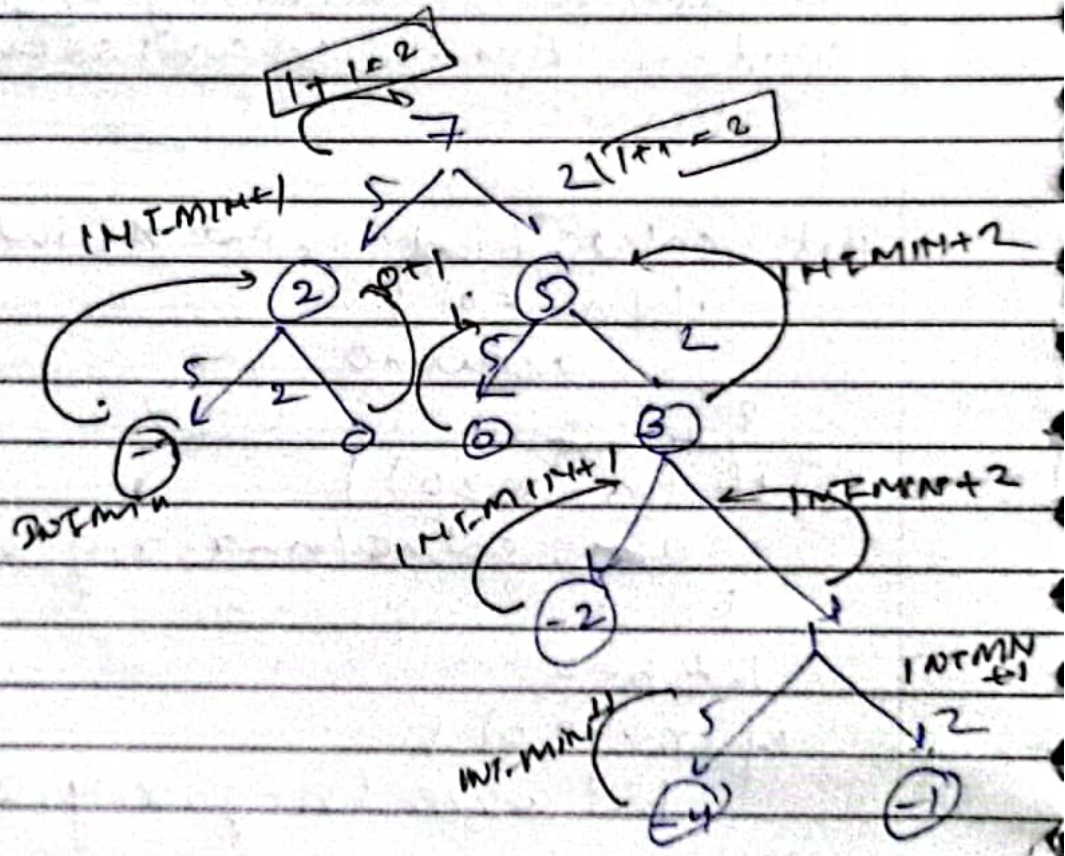
```
int main() {
    int n=7;
    int x=5;
    int y=2;
    int z=2;

int ans = solve(n,x,y,z);
cout << "Ans is " << ans << endl;
  return 0;
}
int solve(int n, int x, int y, int z) {
    if(n==0) {
        return 0;
    }
    int ans=0;
    if(n-x >=0) {
      ans = solve(n-x,x,y,z)+1;
    }
    int b=0;
    if(n-y>=0) {
        b= solve(n-y,x,y,z)+1;
    }
```

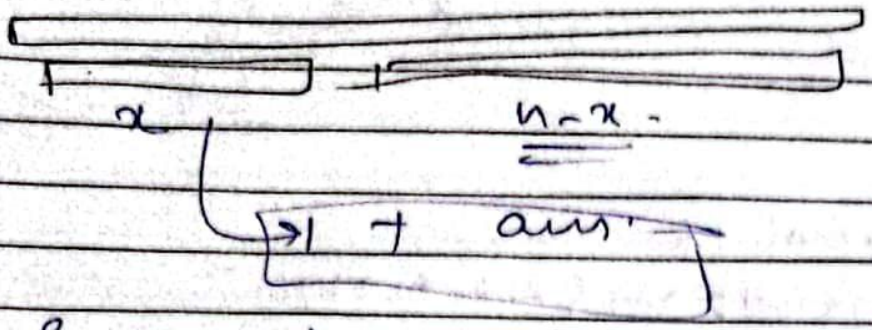
```
if(ans<0)
    ans=0;
```

```
int c=0;
if(n-2>=0){
  c= solve(n-2,x,y,z)+1;
}
int ans = max(a, max(b,c));
return ans;
}
```



eg. n=7, x=5, y=2, z=2





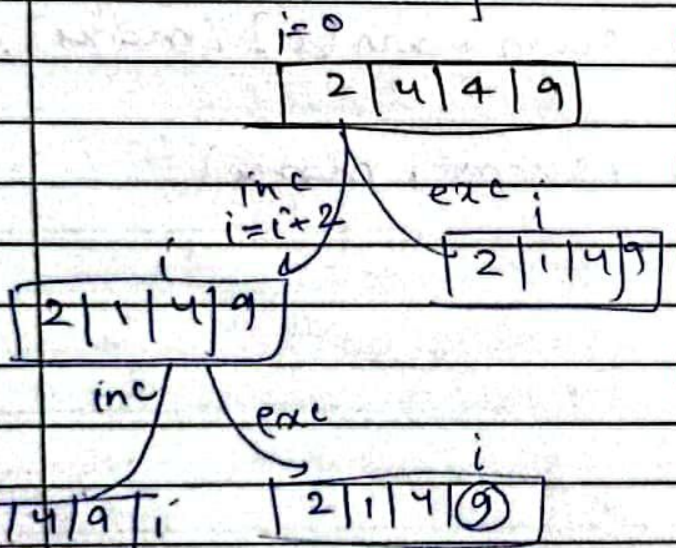
Reason of 1 + ans

3) Maximum sum of non adjacent elements

i/p. | 2 | 1 | 4 | 9 |

Returns the max sum of seleequence in which no two elements are adjacent

2+4 = 6
1+9 = 10
2+9 = 11



B. C.

Question me Sub sequence ki bt kia hai mtlb sub sequence me element ka preset or absent hota hai

means Use Include / Exclude Technique