⇒ Pre Computations :-

→ Prefix Sum :-

	0	1	2	3	4	5
arr :	3	6	2	8	9	2
sumArr :	3	9	11	19	28	30

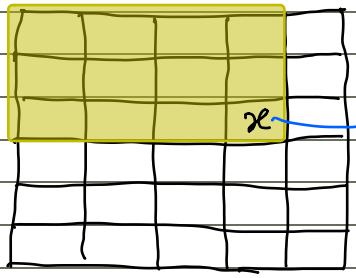
Query : Return Sum of ith to jth Index in Arr

O/P : sumArr[j] − sumArr[i−1]

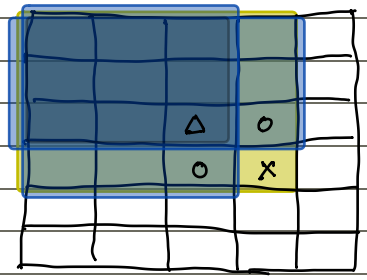
E.g. → 1 to 4 index Sum = 28 − 3 = 25

expected : 6 + 2 + 8 + 9 = 25

→ Prefix Sum 2D :-



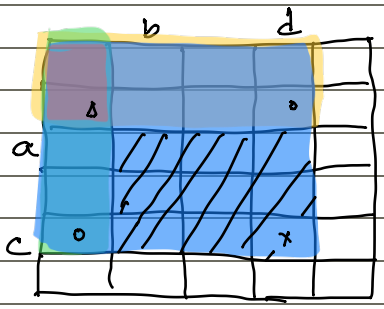
Stores Sum of this box



sum[x] = sum[o] + sum[o] + arr[x] − sum[Δ]

sumArr[i][j] = Arr[i][j]
+ sumArr[i−1][j] + sumArr[i][j−1]
− sumArr[i−1][j−1]

✓ Query :- return sum of (a, b) to (c, d)



▨ = x − o − o + Δ

O/P = sumArr[c][d]
− sumArr[c][b−1] − sumArr[a−1][d]
+ sumArr[a−1][b−1] .

These 3 term may become 0 when a == 0 or b == 0 accordingly.

Q// array = [1, 2, 3 ...] return GCD of elements from index = 2 to 5.

↳ Store forward GCD & backward GCD.

Q//

	0	1	2	3	4	5	6	7	8
	0	0	0	0	0	0	0	0	0
i = 2 to 5 add 25	0	0	25	0	0	0	−25	0	0
i = 3 to 4 sub 11	0	0	25	−11	0	11	−25	0	0

#Trick

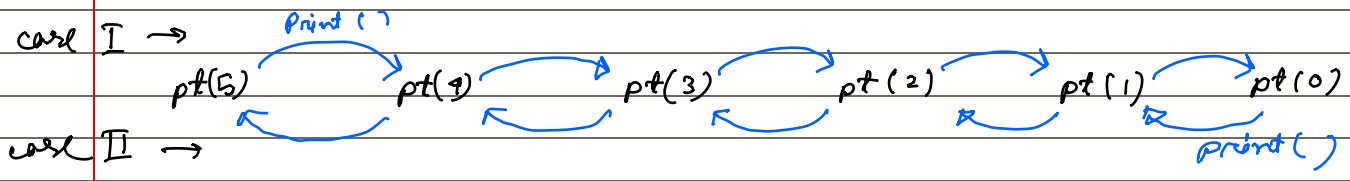
Recursion & Backtracking

(I)

```
def pt(n):                    op
    if n==0: return           5
    print(n)                  4
    pt(n-1)                   3
                              2
                              1
```

(II)

```
def pt(n):                    op
    if n==0: return           1
    pt(n-1)                   2
    print(n)                  3
                              4
                              5
```



Q. Parantheis generator :- ip → count of open == close.

```
void paranGen ( & temp, int open, int close){
    if (open == 0 && close == 0) {
            ans.pushback(temp);
            return
    }
    if (open > 0) {
            temp.pushback("(");
            paranGen(temp, open-1, close);
            temp.popback();
    }
    if (close > 0 && close > open) {
            temp.pushback(")");
            paranGen(temp, open, close-1);
            temp.popback();
    }
}
```