4/09/2025

Ⓠ arr : 10 -2 8 15 6 -8 17 4 -1

0 1 2 3 4 5 6 7 8

Q : i	j	$\sum_{k=i}^{j} arr(k)$
2	4	29
5	8	12
0	2	16

Ⓠ given array of size N and Q queries. For every Find subarray sum from i to j

1)

```
int sum(int arr[], int i, int j){        Q*(j-i+1)
    int sum=0;
    for (int k=i; k<=j; k++){
        sum +=arr[k];

    }
    return sum;
}
```

precompute ↑ (N + Q(1)) → query

2) Find cumulative array of arr

                                   29 46 50 49

c_arr = 10 8 16 31 37

```
int ~~sum(int arr[], int i, int j, int q) {~~
    int ~~cum_sum[n] = ;~~
    w~~hile (q-->0) {~~
        int ~~i = sc.nextInt();~~
        int ~~j = sc.nextInt();~~
        i
  int c_sum[] = new int(n]; c_sum[0]=arr[0];
  for (i=1; i<n; i++){
      c_sum[i]= arr[i] +c_sum[i-1];

  }
  while (q-->0) {
  ans= solve (i,j,arr[],c_sum);
     print(ans);
  }
}
int solve (        ){
  else return c_sum[j]-c_sum[i-1];
  → if (i==0) return c_sum[j];
}.
```

Todo

↓

nlogn + Σ*1

n + q

n+q*1

```
//computing suffix:
    suf[n] = {0};
    suf[n-1] = arr[n-1];
    for (int i = n-2 ; i >= 0; i--){
        suf[i] = suf[i+1] + arr[i];
    }
```

Q

A = a b b x y a y b b a

B = y a b b a x y a

a[i to j] == b[k to l] → True.

!= → False.

Q=	i	j	k	l	o/p
	0	2	1	3	T
	4	6	0	2	F
	7	9	2	4	T
	2	3	5	7	F

1) O(Q × N) → $\forall_{i=0}$ if(A[i] != B[i]) → T / → F

2) prefix array.

PHA: a a+b a+b+b ....

PHB: y y+a y+a+b ...

```
if ( PHA[j] - ((i!=0)? PHA[i-1] : 0) ==
        PHB[l] - ( k!=0)? PHB[l-1] : 0))
        return True;
else
    return False;
```

} Hash string collision

bba ↓↓↓ 8 9 10

bba ↓↓↓ 3 4 5

To handle

→ compute power array O(N)

→ PHA
PHB } compute O(N)

→ HA compare with HB O(1)

→ calculate d → |k-i| → ~~(k>p)~~ ? O(1)
(k>i) ? $HA * p^d$: $HB * p^d$

{ # modulo arithmetic where necessary

→ return true.

→ $\boxed{\max(N,M) + N+M+Q(1)}$

$\boxed{\text{LCS}}$

HA [1:3]: $[ap^2 + bp^3 + bp^4]$: PHA[3] - PHA[0]

HB [2:4]: $[ap^3 + bp^4 + bp^5]$: PHB[4] - PHB[1]

code:

```
p[0] = p   (p is variable) (if doesn't work increase p)

$\forall_{i=1}^{max(N,M)}$ p[i] = (p[i-1] * p) % mod

int PHA[N] = {0};
PHA[0] = A[0] * P[0]
$\forall_{i=1}^{N}$ PHA[i] = (PHA[i-1] + A[i] * P[i] % mod ) % mod

int PHB[M];
PHB[0] = B[0] * P[0]
$\forall_{i=1}^{M}$ PHB[i] = (PHB[i-1] + B[i] * P[i] % mod ) % mod

loop(Q)
    Read(i, j, k, l)
    HA = PHA[j] - ((i != 0)? PHA[i-1] : 0)
    HB = PHB[l] - ((k != 0)? PHA[k-1] : 0)

    d = abs(k-i)
    if (k > i)
        HA = (HA * P[d-1]) % mod        }  equal powers
    else                                }
        HB = (HB * P[d-1]) % mod        }

    if(HA == HB)
        return True;
    else
        return False;
```