QN Link : <https://www.desiqna.in/15920/microsoft-oa-september-2023-freshers-hiring-sde1-set-115>

Question Summary :

* You re given an array of size “N” and you need to find the minimum rehabilitation time for X days.
* Each session (1 to X – 1 ) has Y days between them .
* Return the minimum time taken for rehabilitation.

Brute Force :

* Find the X indexes from i to till we get X indexes
* If we do not get x indexes then don’t taken it into account.
* **for**(int i=1;i<=n;i++){
* j = i
* sum = 0
* c = 0
* **while**(j>=1 && c<x){
* sum = sum + b[j]
* c++
* j = j - y
* }
* **if**(c==x){
* answer = min(answer,sum)
* }

* }

Efficient Approach :

* Use an prefix array to store the sum of values from j to j – (x \* y)
* We can calulate the prefix sum for index I by prefix[i] = nums[i] + prefix[I – y]
* We need to calaulate the minimum of it .

Example TC :

[4 , 2 , 5 , 4 , 3 , 5 ,1 , 4, 2 , 7] X = 3 , Y = 2

We need to select 3 session each with an gap of 2 days

For index 0 , we can’t calculate previous index value so let it be [4 , 0 , 0, 0 , 0 , 0 , 0 , 0, 0, 0] [0] = 4

For index 1 , we can’t calculate previous index value so let it be [4 , 2 , 0 , 0 , 0 , 0 ,0 , 0 , 0, 0] [1] = 2

For index 2 , 2 – 2 🡪 0 , then the prefix array should be [4 , 2 , 9 , 0 , 0 , 0 ,0 , 0 , 0, 0] [2] = 5

For index 3 , 3 – 2 🡪 1 , then the prefix array should be [4 , 2 , 9 , 6 , 0 , 0 , 0 , 0, 0 , 0] [3] = 2

For index 4 , 4 – 2 🡪 2 , [4 , 2 , 9 , 6 , 12 , 0 ,0 , 0 , 0, 0] [4] = 3

For index 5 , 5 – 2 🡪 3 , [4 , 2 , 9 , 6 , 12 , 11 ,0 , 0 , 0, 0] [5] = 5

For index 6 , 6 – 2 🡪 4 , [4 , 2 , 9 , 6 , 12 , 11 ,13 , 0 , 0, 0] [6] = 1

* Here we can observe that the for index [i] , the prefix value should be prefix[I – y] + nums[i]

But how will you calulate for index I , with (x \*y ) previous values ?

* Simple , ans = prefix[i] – prefix[I – (x \* y)]
* Only if I – (x \*y) >= 0

class Solution {

    public int solution(int [] nums , int X , int Y) {

        int n =  nums.length;

        int min = n + 1;

        int [] prefix = new int[n];

        for(int i = 0 ; i < n ; i++){

            if ( i - y >= 0){

                prefix[i] = prefix[i - y] + nums[i];

            }else{

                prefix[i] = nums[i];

            }

        }

        for(int i = 0 ; i < n ; i++){

            if (i - (X \* Y) >= 0){

                min = Math.min(min , prefix[i] - prefix [ i - X \* Y]);

            }

        }

        return min;

    }

}