HOMEWORK DAY: 13.3 HW 6

01 October 2023 17:51

HW 06: Painters Partition Problem (GFG & Code studio)

Problem Statement:
Dilpreet wants to paint his dog's home that has **n boards** with different lengths.
The length of **ith board** is given by **arr[i]** where **arr[] is an array** of n integers.
He hired **k painters** for this work

Note: each painter takes 1 unit time to paint 1 unit of the board.
Like --> Agar kisi ek board ki length 10 unit hai to ek painter ko iss board ko paint karne me bhi 10 unit time legega

---

Observation:
Board = ith
Board ki length = arr[ith]
Dog's Home = arr[]
Home ki length/Total numbers of board = n
Painters = k

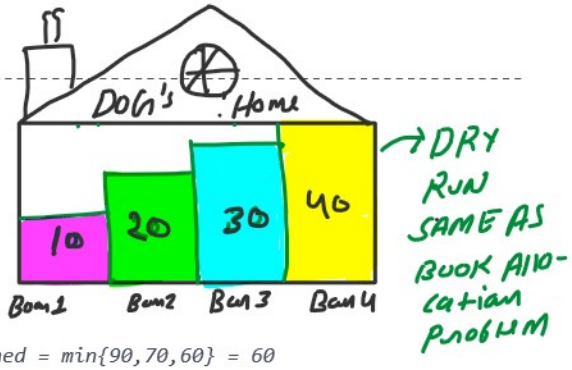
The problem is to find the **'minimum time'** to get this job done
if all painters start together with the constraint that **any painter will only paint 'continuous boards'**,

continuous boards:
say boards numbered {2,3,4} or only board {1} = increasing order me board ki length hai 📈

non continuous boards:
or nothing but not boards {2,4,5} = decreasing order me board ki length hai 📉

Allocate boards in a way such that:
- Each painter gets at least one board.
- Each board should be allocated to a painter.
- Board allocation should be in a contiguous manner.

Example 01:
Input: n = 4, k = 2, arr[] = {10,20,30,40}
Output: 60



Observation:
Home ki length/total numbers of board = n = 4
Painters = k = 2
Board ki length = arr[ith] ,Here arr[0] = 10 length of 1st board
Dog's Home = arr[]
Board = ith, Here i--> 0, 1, 2, 3
The minimum time of all painters of the maximum time of all painters assigned = min{90,70,60} = 60

Explanation: The most optimal way to paint:
Painter 1 allocation : {10,20,30}
Painter 2 allocation : {40}
Job will be complete at time = 60

OPTIMAL APPROACH: Define search space and predicate function
Step 01: find total sum of array to create search space's end point (**Highest Maximum Time=sum**)
Step 02: apply binary search on search space BinarySearch()
Step 03: create predicate function isPossibbleSolution()

Always Yaad Rahe: Jab hum **mid** nikalte hai to uska mtlb **Maximum time** hota hai Yani ki **Highest Maximum time** se less

Corner case 01: If number of painter(k) are greater then number of board(n) then return -1
Corner case 02: If number number of case greater then number of painetr(k) then return false

---

Time Complexity: O(N*log(Sum)),here **Sum** of all boards' length, and **N** is length of array
Space Complexity: O(1), no extra space used

Resource: https://www.geeksforgeeks.org/painters-partition-problem/

```cpp
// HW 06: Painters Partition Problem (GFG & Code studio)
class Solution
{
  public:

    // Predicate Function
    bool isPossibleSolution(int arr[],int n,int k,long long mid){
        int cases = 1;
        long long timeSum = 0;

        for(int i=0; i<n; i++){
            if(timeSum  + arr[i]<=mid){
                timeSum  += arr[i];
            }
            else{
                cases++;
                if(cases>k || arr[i]>mid){
                    return false;
                }
                // reset time sum
                timeSum  = arr[i];
            }
        }
        return true;
    }

    // Binary Search
    long long BinarySearch(int arr[],int n,int k,long long end){
        long long start = 0;
        // Mid is Mamimum Time only
        long long mid = start + (end - start)/2;
        // Ans is Minimum Time of Maximum Time
        long long ans = -1;

        while(start<=end){
            if(isPossibleSolution(arr,n,k,mid)){
                ans = mid;
                end = mid - 1;
            }
            else{
                start = mid + 1;
            }
            mid = start + (end - start)/2;
        }
        return ans;
    }

    long long minTime(int arr[], int n, int k)
    {
        long long sum = 0;
        // Total sum (Highest Maximum Time)
        for(int i=0;i<n;i++){
            sum+=arr[i];
        }
        // Binary Search
        long long ans = BinarySearch(arr,n,k,sum);
    }
};
```