https://course.acciojob.com/idle?question=d7165d6a-4fcd-42af-aed5-13f8ab4881a2

MEDIUM

Max Score: 40 Points

Min Cut Tree

Given an integer `K` and an array height[] of size `N`, where height[i] denotes the height of the `ith` tree in a forest. The task is to make a cut of height `x` from the ground such that at max `K` unit wood is collected. Find the minimum value of `x`.

If you make a cut of height `x` from the ground then every tree with a height greater than `x` will be reduced to `x` and remaining part of wood can be collected as wood.

Your task is to complete the function `minHeight` which receives the input array, `n` and `k` as its parameters and returns the minimum value of `x`.

Input Format

The first line contains two integers `N` and `K`. Next line contains `N` integers denoting the elements of the array height[]

Output Format

Print a single integer the value of `x`.

Example 1:

Input:

```
4 2
1 2 1 2
```

Output

```
1
```

Explaination

Make a cut at height 1, the updated array will be {1, 1, 1, 1} and the collected wood will be {0, 1, 0, 1} i. e. 0 + 1 + 0 + 1 = 2.

Example 2:

Input:

```
6 3
7 4 9 2 1 8
```

Output

```
7
```

Explaination

Make a cut at height 7, the updated array will be {7, 4, 7, 2, 1, 7} and the collected wood will be {0, 0, 2, 0, 0, 1} i. e. 0 + 0 + 2 + 0 + 0 + 1 = 3.

Constraints:

1 <= N <= 10^5
1 <= arr[i] <= 10^5
1 <= K <= 10^7

Topic Tags

Binary Search

My code

```
import java.util.*;
import java.io.*;

class Solution{
    int minHeight(int[] arr, int n,int k){
        // write code here
                int l=1, r=Integer.MIN_VALUE;
                int ans=0;
                        for(int i=0; i<n; i++){
                          r=Math.max(arr[i],r);
                        }
                while(l<=r){
                        int mid=(l+r)/2;
                        if(isPossible(arr,l,r,mid,k)){
                                ans=mid;
                                r=mid-1;
                        }else l=mid+1;
                }
                return ans;
    }
        boolean isPossible(int[]arr, int l, int r, int mid, int k){
                int sum=0;

                for(int i=0; i<arr.length; i++){
                        if(arr[i]-mid>0)
                        sum+=arr[i]-mid;
                }

                return sum<=k?true:false;
        }
}
```

```
public class Main {
   public static void main(String args[]) {
        Scanner input = new Scanner(System.in);
        int n = input.nextInt();
        int k = input.nextInt();
        int a[] = new int[n];
        for(int i = 0; i < n; i++){
            a[i] = input.nextInt();
        }
       Solution Obj = new Solution();
       int ans = Obj.minHeight(a,n,k);
        System.out.println(ans);
    }
}
```