ASSIGNMENT – 6

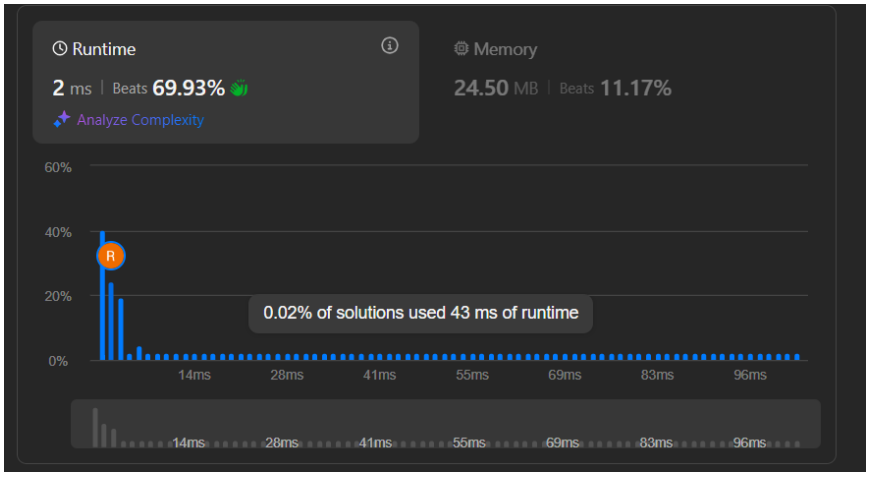
Name – Raj Singh

UID – 22BCS10624

CLASS – FL_IOT_604-A

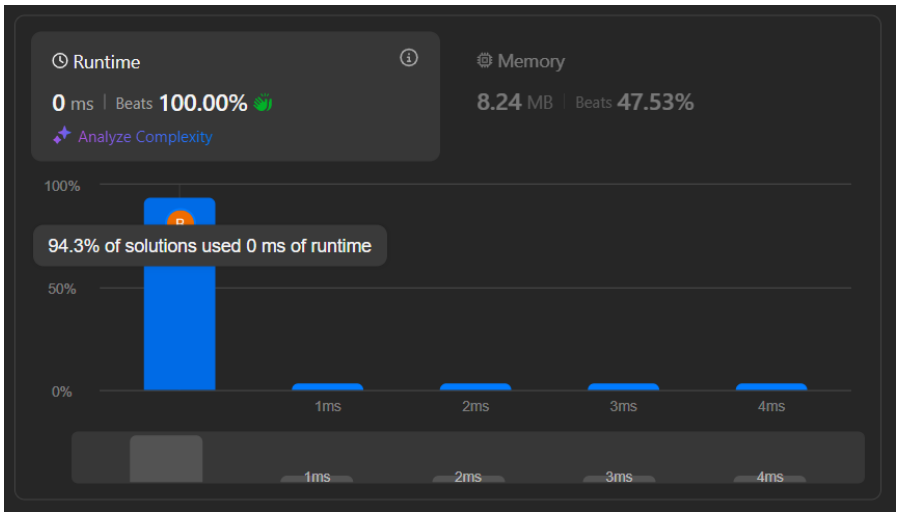
108.Convert Sorted Array to Binary Search Tree

```
/**
 * Definition for a binary tree node.
 * struct TreeNode {
 *     int val;
 *     TreeNode *left;
 *     TreeNode *right;
 *     TreeNode() : val(0), left(nullptr), right(nullptr) {}
 *     TreeNode(int x) : val(x), left(nullptr), right(nullptr) {}
 *     TreeNode(int x, TreeNode *left, TreeNode *right) : val(x), left(left), right(right) {}
 * };
 */
class Solution {
public:
    TreeNode* sortedArrayToBST(vector<int>& nums) {
        nums.push_back(0);
        TreeNode * res = sorting(nums,0,nums.size()-1);
        return res;
    }
    TreeNode* sorting(vector<int> &nums,int low,int high){
        TreeNode* res = new TreeNode();
        if(low>=high){
            return NULL;
        }
        int mid = low + (high-low)/2;
        res->val = nums[mid];
        res->left = sorting(nums,low,mid);
        res->right= sorting(nums,mid+1,high);
        return res;
    }
};
```



191.[Number of 1 Bits]

```
class Solution {
public:
    int hammingWeight(int n) {
        int res=0;
        while(n>0){
            if(n & 1){
                res++;
            }
            n = n>>1;
        }
        return res;
    }
};
```

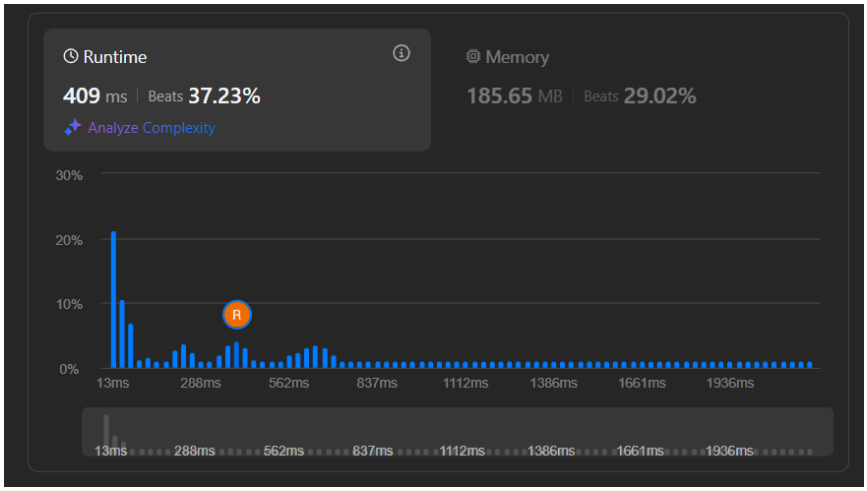


912.[Sort an Array]

```
class Solution {
public:
```

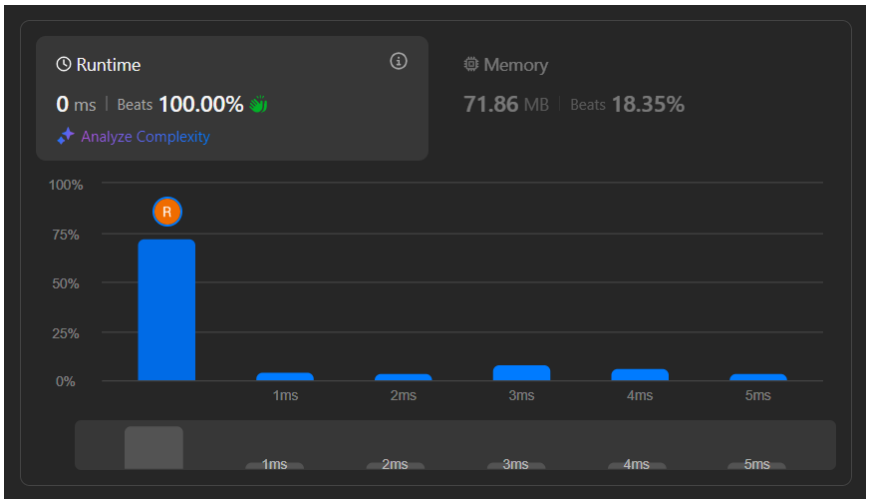
```
    vector<int> sortArray(vector<int>& nums) {
        mergesort(nums,0,nums.size()-1);
        return nums;
    }
    void mergesort(vector<int> &vec,int low,int high){
        if(low>=high){
            return;
        }
        int mid = low + (high-low)/2;
        mergesort(vec,low,mid);
        mergesort(vec,mid+1,high);
        merge(vec,low,high,mid);
    }
    void merge(vector<int> &arr,int low,int high,int mid){
        int n1 = mid-low+1,n2 = high-mid;
        vector<int> left(n1);
        vector<int> right(n2);
        for(int i =0;i<n1;i++){
            left[i]=(arr[low+i]);
        }
        for(int i =0;i<n2;i++){
            right[i]=(arr[mid+1+i]);
        }
        int i =0,j=0,k=low;
        while(i<n1 && j<n2){
            if(left[i]<=right[j]){
                arr[k] = left[i];
                i++;
            }
            else{
                arr[k] = right[j];
                j++;
            }
            k++;
        }
        while(i<n1){
            arr[k++] = left[i++];
        }
        while(j<n2){
            arr[k++] = right[j++];
        }

    }
};
```



53.Maximum Subarray

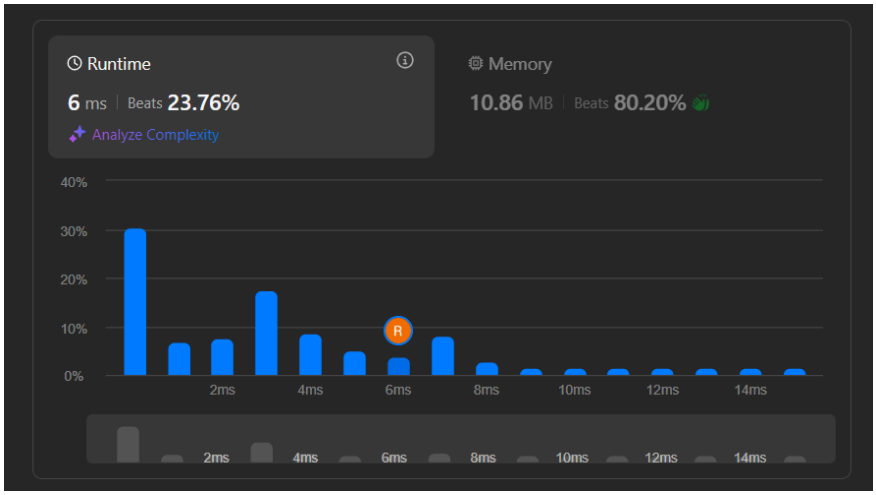
```
class Solution {
public:
    int maxSubArray(vector<int>& nums) {
        int res=nums[0],maxi = nums[0];
        for(int i=1;i<nums.size();i++){
            maxi = max(nums[i]+maxi,nums[i]);
            res = max(res,maxi);
        }
        return res;
    }
};
```



932.Beautiful Array

```
    vector<int> beautifulArray(int N) {
        vector<int> res = {1};
```

```
        while (res.size() < N) {
            vector<int> tmp;
            for (int i : res) if (i * 2 - 1 <= N) tmp.push_back(i * 2 - 1);
            for (int i : res) if (i * 2 <= N) tmp.push_back(i * 2);
            res = tmp;
        }
        return res;
    }
```



372.[Super Pow]

```
class Solution {
public:
    int superPow(int a, vector<int>& b) {
        long long pow = b[0];
        int n=b.size();
        if(n<1 || a==1){
            return 1;
        }
        for(int i=1;i<n;i++){
            pow *= 10;
            pow += b[i];
            pow %= 1140;
        }
        if(pow ==0){
            pow = 1140;
        }
        return power(a,pow)%1337;
    }
    long long power(int &a,long long &pow){
        if(pow==0){
            return 1;
        }
        long long temp = pow/2;
        long long hpow = power(a,temp);
        if(pow%2==0){
```

```
            return hpow*hpow%1337;
        }
        else{
            return hpow*hpow*a %1337;
        }
    }
};
```