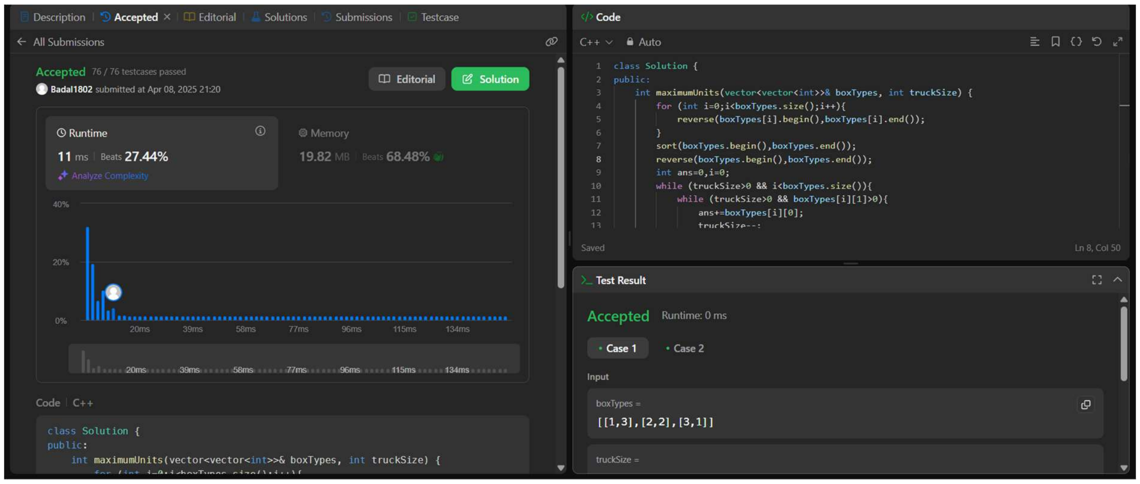
ASSIGNMENT 8

Max Units on a Truck

CODE:

```
class Solution {
public:
    int maximumUnits(vector<vector<int>>& boxTypes, int truckSize) {
        for (int i=0;i<boxTypes.size();i++){
            reverse(boxTypes[i].begin(),boxTypes[i].end());
        }
        sort(boxTypes.begin(),boxTypes.end());

        reverse(boxTypes.begin(),boxTypes.end());

        int ans=0,i=0;
        while (truckSize>0 && i<boxTypes.size()){
            while (truckSize>0 && boxTypes[i][1]>0){
                ans+=boxTypes[i][0];
                truckSize--;
                boxTypes[i][1]--;
            }
            i++;
        }
        return ans;
    }
};
```

OUTPUT:



Accepted 76 / 76 testcases passed
Badal1802 submitted at Apr 08, 2025 21:20

Runtime 11 ms | Beats 27.44%

Memory 19.82 MB | Beats 68.48%

Code | C++

```
class Solution {
public:
    int maximumUnits(vector<vector<int>>& boxTypes, int truckSize) {
        for (int i=0;i<boxTypes.size();i++){
            reverse(boxTypes[i].begin(),boxTypes[i].end());
        }
        sort(boxTypes.begin(),boxTypes.end());
        reverse(boxTypes.begin(),boxTypes.end());
        int ans=0,i=0;
        while (truckSize>0 && i<boxTypes.size()){
            while (truckSize>0 && boxTypes[i][1]>0){
                ans+=boxTypes[i][0];
```

Saved   Ln 8, Col 50

Test Result

Accepted Runtime: 0 ms

Case 1   Case 2

Input

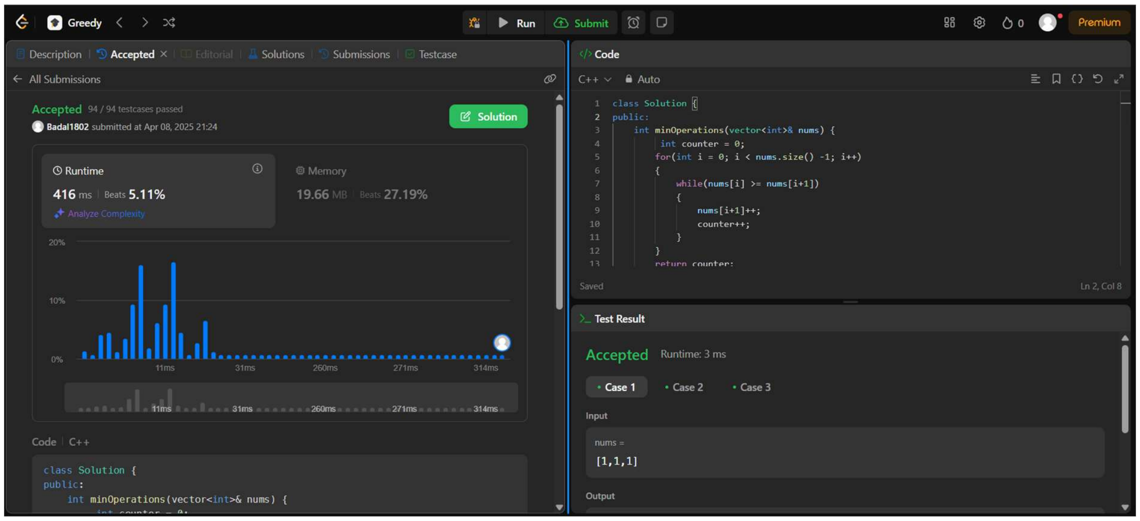
```
boxTypes =
[[1,3],[2,2],[3,1]]

truckSize =
```

Min Operations to Make Array Increasing

CODE:

```
class Solution {
public:
    int minOperations(vector<int>& nums) {
         int counter = 0;
        for(int i = 0; i < nums.size() -1; i++)
        {
            while(nums[i] >= nums[i+1])
            {
                nums[i+1]++;
                counter++;
            }
        }
        return counter;
    }
};
```

OUTPUT:



Greedy

Run Submit Premium

Description | Accepted | Editorial | Solutions | Submissions | Testcase

All Submissions

Accepted 94 / 94 testcases passed

Badal1802 submitted at Apr 08, 2025 21:24

Solution

Runtime

416 ms | Beats 5.11%

Analyze Complexity

Memory

19.66 MB | Beats 27.19%

Code | C++

```
class Solution {
public:
    int minOperations(vector<int>& nums) {
        int counter = 0;
```

Code

C++ Auto

```
class Solution {
public:
    int minOperations(vector<int>& nums) {
        int counter = 0;
        for(int i = 0; i < nums.size() -1; i++)
        {
            while(nums[i] >= nums[i+1])
            {
                nums[i+1]++;
                counter++;
            }
        }
        return counter;
```

Saved   Ln 2, Col 8

Test Result

Accepted Runtime: 3 ms

Case 1 Case 2 Case 3

Input

```
nums =
[1,1,1]
```

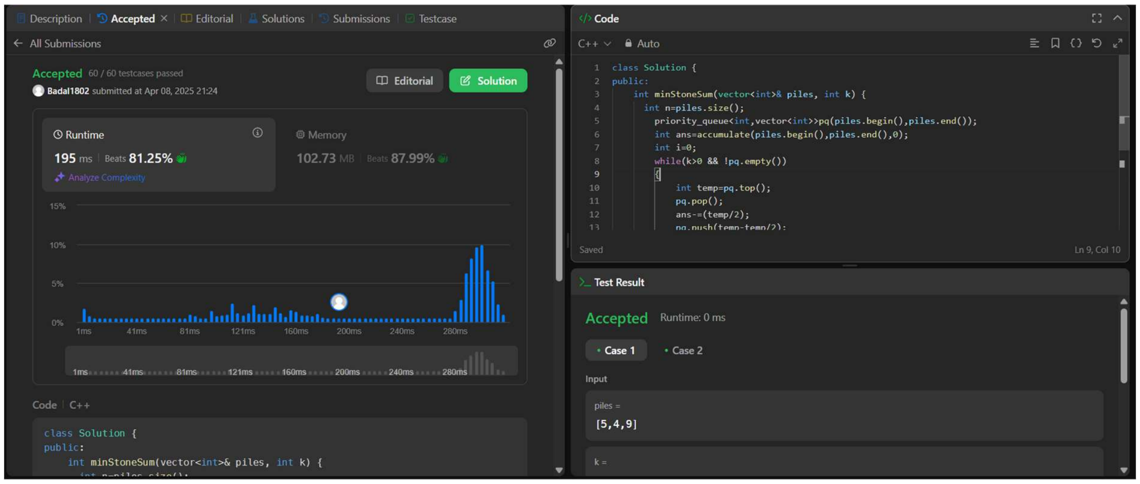
Output

Remove Stones to Maximize Total

CODE:

```
class Solution {
public:
    int minStoneSum(vector<int>& piles, int k) {
      int n=piles.size();
        priority_queue<int,vector<int>>pq(piles.begin(),piles.end());
        int ans=accumulate(piles.begin(),piles.end(),0);
        int i=0;
        while(k>0 && !pq.empty())
        {
            int temp=pq.top();
            pq.pop();

            ans-=(temp/2);

            pq.push(temp-temp/2);

            k--;

        }


        return ans;

    }

};
```

OUTPUT:



Accepted 60 / 60 testcases passed
Badal1802 submitted at Apr 08, 2025 21:24

Runtime
195 ms | Beats 81.25%

Memory
102.73 MB | Beats 87.99%

Code | C++

```
class Solution {
public:
    int minStoneSum(vector<int>& piles, int k) {
```

C++ Auto

```
class Solution {
public:
    int minStoneSum(vector<int>& piles, int k) {
      int n=piles.size();
        priority_queue<int,vector<int>>pq(piles.begin(),piles.end());
        int ans=accumulate(piles.begin(),piles.end(),0);
        int i=0;
        while(k>0 && !pq.empty())
        {
            int temp=pq.top();
            pq.pop();
            ans-=(temp/2);
            pq.push(temp-temp/2);
```

Ln 9, Col 10

Accepted Runtime: 0 ms

Input

piles =
[5,4,9]

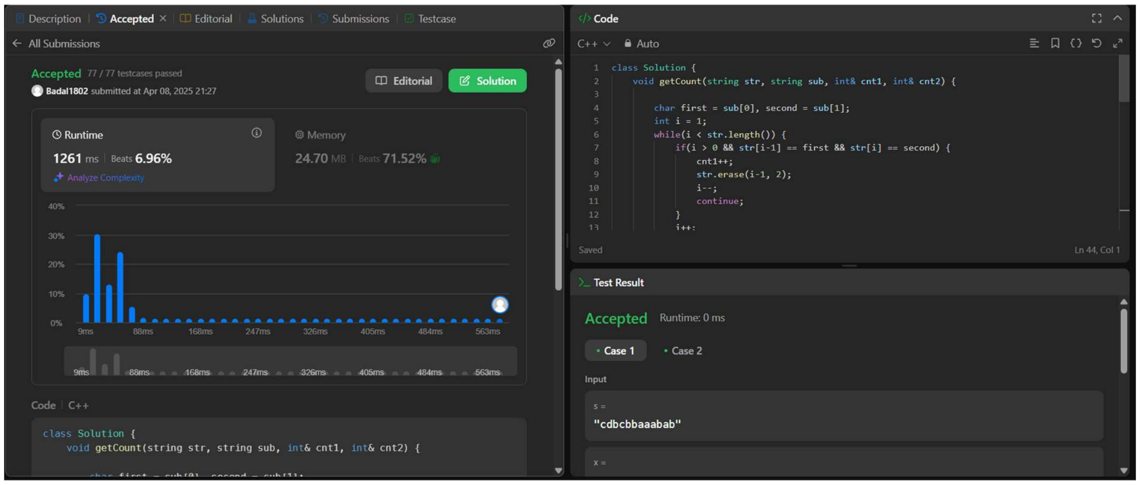
k =

Max Score from Removing Substrings

CODE:

```
class Solution {

    void getCount(string str, string sub, int& cnt1, int& cnt2) {


        char first = sub[0], second = sub[1];

        int i = 1;
        while(i < str.length()) {
            if(i > 0 && str[i-1] == first && str[i] == second) {
                cnt1++;
                str.erase(i-1, 2);
                i--;
                continue;
            }
            i++;
        }

        i = 1;
        while(i < str.length()) {
            if(i > 0 && str[i-1] == second && str[i] == first) {
                cnt2++;
                str.erase(i-1, 2);
                i--;
                continue;
            }
            i++;
        }
        return;
    }
public:
    int maximumGain(string s, int x, int y) {

        int mxABcnt = 0;
        int mxBAcnt = 0;
        int minBAcnt = 0;
        int minABcnt= 0;
        getCount(s, "ab", mxABcnt, minBAcnt);
        getCount(s, "ba", mxBAcnt, minABcnt);

        int operation1 = mxABcnt * x + minBAcnt * y;
        int operation2 = mxBAcnt * y + minABcnt * x;
        return max(operation1, operation2);
    }
};
```

OUTPUT:



Accepted 77 / 77 testcases passed
Badal1802 submitted at Apr 08, 2025 21:27

Runtime 1261 ms | Beats 6.96%
Memory 24.70 MB | Beats 71.52%

```
class Solution {
    void getCount(string str, string sub, int& cnt1, int& cnt2) {

        char first = sub[0], second = sub[1];
        int i = 1;
        while(i < str.length()) {
            if(i > 0 && str[i-1] == first && str[i] == second) {
                cnt1++;
                str.erase(i-1, 2);
                i--;
                continue;
            }
            i++;
```

Saved
Ln 44, Col 1

Test Result

Accepted Runtime: 0 ms

Case 1 Case 2

Input

s =
"cdbcbbaaabab"

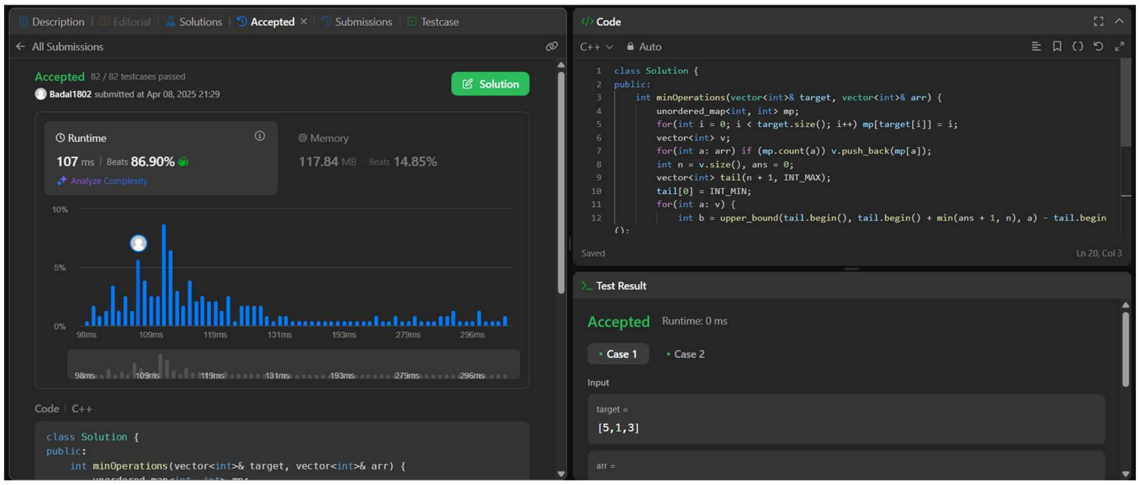
x =

Min Operations to Make a Subsequence

CODE:

```
class Solution {
public:
    int minOperations(vector<int>& target, vector<int>& arr) {
        unordered_map<int, int> mp;
        for(int i = 0; i < target.size(); i++) mp[target[i]] = i;
        vector<int> v;
        for(int a: arr) if (mp.count(a)) v.push_back(mp[a]);
        int n = v.size(), ans = 0;
        vector<int> tail(n + 1, INT_MAX);
        tail[0] = INT_MIN;
        for(int a: v) {
            int b = upper_bound(tail.begin(), tail.begin() + min(ans + 1, n), a) - tail.begin();
            if (b == 0 || (tail[b - 1] < a && tail[b] > a)) {
                tail[b] = a;
                ans = max(ans, b);
            }
        }
        return target.size() - ans;
    }
};
```

OUTPUT:



Accepted 82 / 82 testcases passed

Badal1802 submitted at Apr 08, 2025 21:29

Runtime 107 ms | Beats 86.90%

Memory 117.84 MB | Beats 14.85%

Code | C++

```
class Solution {
public:
    int minOperations(vector<int>& target, vector<int>& arr) {
```

```
class Solution {
public:
    int minOperations(vector<int>& target, vector<int>& arr) {
        unordered_map<int, int> mp;
        for(int i = 0; i < target.size(); i++) mp[target[i]] = i;
        vector<int> v;
        for(int a: arr) if (mp.count(a)) v.push_back(mp[a]);
        int n = v.size(), ans = 0;
        vector<int> tail(n + 1, INT_MAX);
        tail[0] = INT_MIN;
        for(int a: v) {
            int b = upper_bound(tail.begin(), tail.begin() + min(ans + 1, n), a) - tail.begin
();
```

Accepted Runtime: 0 ms

Input

target =
[5,1,3]

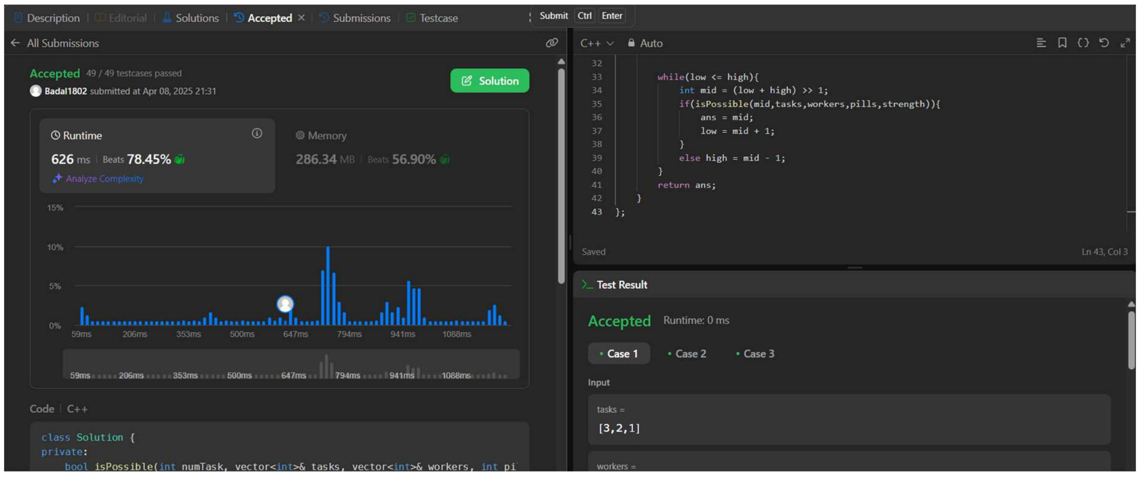
arr =

Max Number of Tasks You Can Assign

CODE:

```
class Solution {
private:
    bool isPossible(int numTask, vector<int>& tasks, vector<int>& workers, int pills, int
strength){
        multiset<int> ms(workers.end()-numTask,workers.end());


        for(int i=numTask-1; i>=0; i--){
            auto it = ms.end();
            it--;
            if(*it < tasks[i]){
                if(!pills) return false;
                it = ms.lower_bound(tasks[i]-strength);
                if(it == ms.end()) return false;
                pills--;
            }
            ms.erase(it);
        }


        return true;
    }
public:
    int maxTaskAssign(vector<int>& tasks, vector<int>& workers, int pills, int strength) {


        sort(tasks.begin(), tasks.end());
        sort(workers.begin(),workers.end());


        int n = tasks.size();
        int m = workers.size();


        int low = 0;
        int high = min(n,m);
        int ans = 0;


        while(low <= high){
            int mid = (low + high) >> 1;
            if(isPossible(mid,tasks,workers,pills,strength)){
                ans = mid;
                low = mid + 1;
            }
            else high = mid - 1;
        }
        return ans;
    }
};
```

OUTPUT:



Accepted 49 / 49 testcases passed

Badal1802 submitted at Apr 08, 2025 21:31

Runtime
626 ms | Beats 78.45%

Memory
286.34 MB | Beats 56.90%

Code | C++

```
class Solution {
private:
    bool isPossible(int numTask, vector<int>& tasks, vector<int>& workers, int pi
```

```
        while(low <= high){
            int mid = (low + high) >> 1;
            if(isPossible(mid,tasks,workers,pills,strength)){
                ans = mid;
                low = mid + 1;
            }
            else high = mid - 1;
        }
        return ans;
    }
};
```

Saved   Ln 43, Col 3

Test Result

Accepted Runtime: 0 ms

Case 1   Case 2   Case 3

Input

tasks =
[3,2,1]

workers =