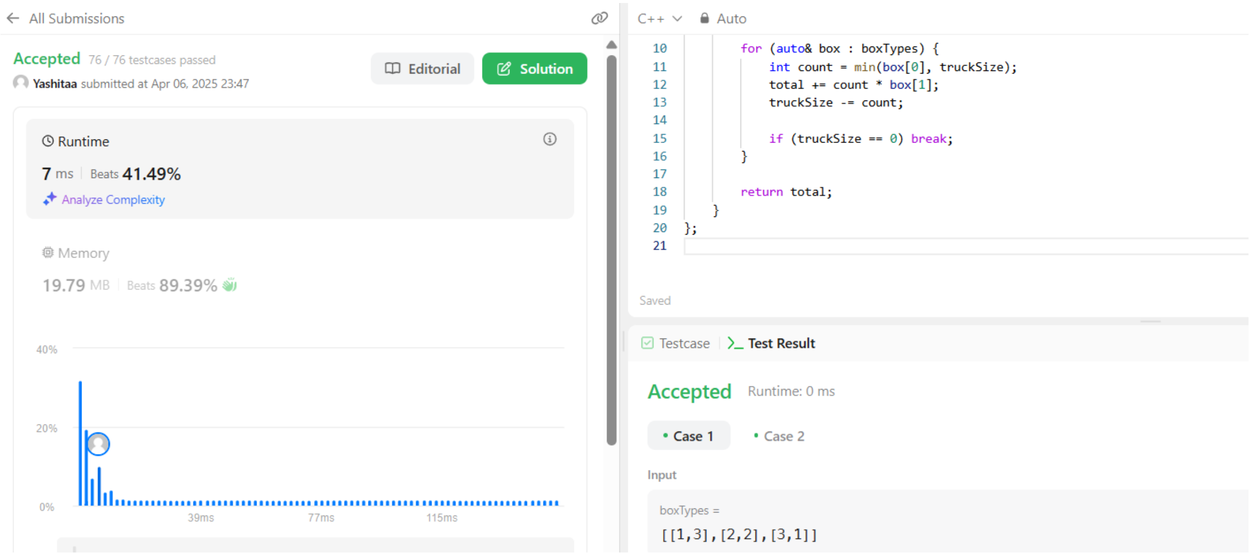
Name: Yashita

Uid: 22BCS15024

Section: FL_Iot 601 ‘A’
AP assignment 8

1. Max Units on a Truck

```
class Solution {
public:
    int maximumUnits(vector<vector<int>>& boxTypes, int truckSize) {
        sort(boxTypes.begin(), boxTypes.end(), [](const vector<int>& a, const 
vector<int>& b) {
            return a[1] > b[1];
        });

        int total = 0;

        for (auto& box : boxTypes) {
            int count = min(box[0], truckSize);
            total += count * box[1];
            truckSize -= count;

            if (truckSize == 0) break;
        }

        return total;
    }
};
```



Accepted 76 / 76 testcases passed

Yashitaa submitted at Apr 06, 2025 23:47

7 ms | Beats 41.49%

19.79 MB | Beats 89.39%

```
        for (auto& box : boxTypes) {
            int count = min(box[0], truckSize);
            total += count * box[1];
            truckSize -= count;

            if (truckSize == 0) break;
        }

        return total;
    }
};
```

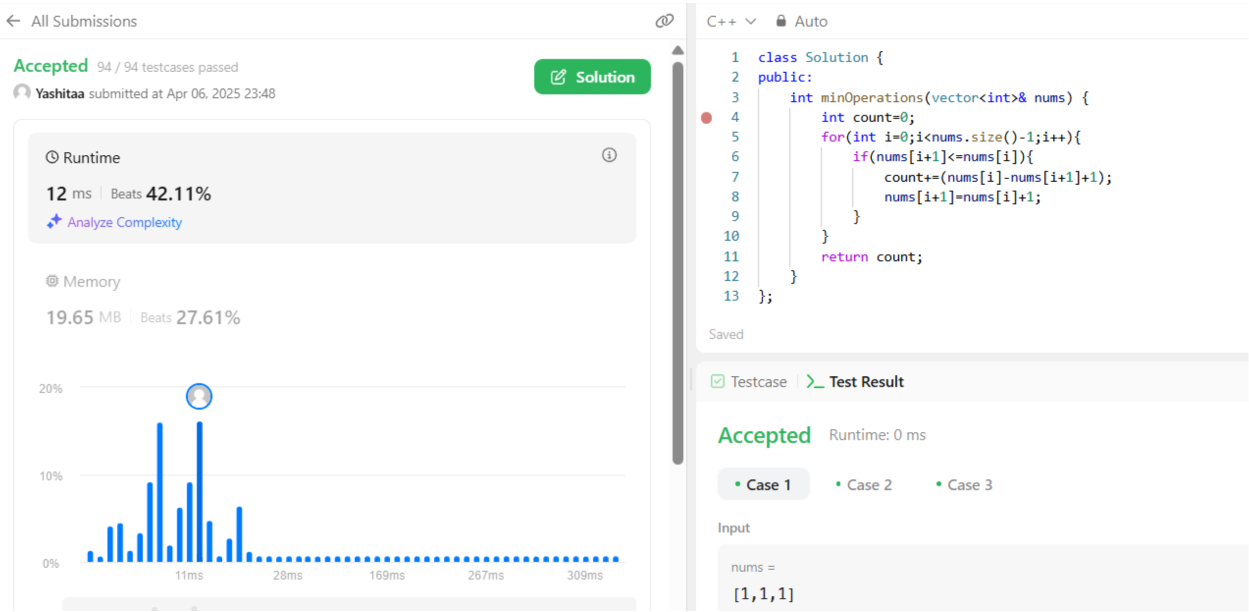
Accepted Runtime: 0 ms

Input

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

2. Minimum Operations to Make the Array Increasing

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



Accepted 94 / 94 testcases passed

Yashitaa submitted at Apr 06, 2025 23:48

Runtime: 12 ms | Beats 42.11%

Memory: 19.65 MB | Beats 27.61%

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

Accepted Runtime: 0 ms

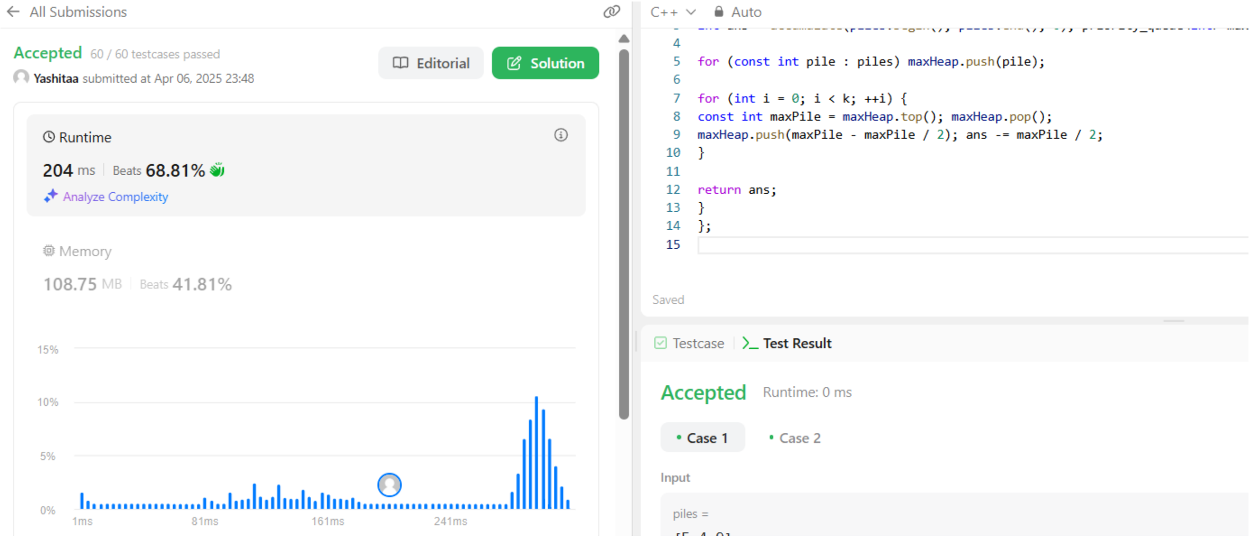
• Case 1 • Case 2 • Case 3

Input

nums =
[1,1,1]

3. Remove Stones to Minimize the Total

```
class Solution {
 public:
  int minStoneSum(vector<int>& piles, int k) {
    int ans = accumulate(piles.begin(), piles.end(), 0);
    priority_queue<int> maxHeap;

    for (const int pile : piles)
      maxHeap.push(pile);

    for (int i = 0; i < k; ++i) {
      const int maxPile = maxHeap.top();
      maxHeap.pop();
      maxHeap.push(maxPile - maxPile / 2);
      ans -= maxPile / 2;
    }
    return ans;
  }
};
```



← All Submissions

Accepted 60 / 60 testcases passed

Yashitaa submitted at Apr 06, 2025 23:48

Runtime

204 ms | Beats 68.81%

Memory

108.75 MB | Beats 41.81%

```
for (const int pile : piles) maxHeap.push(pile);

for (int i = 0; i < k; ++i) {
const int maxPile = maxHeap.top(); maxHeap.pop();
maxHeap.push(maxPile - maxPile / 2); ans -= maxPile / 2;
}

return ans;
}
};
```

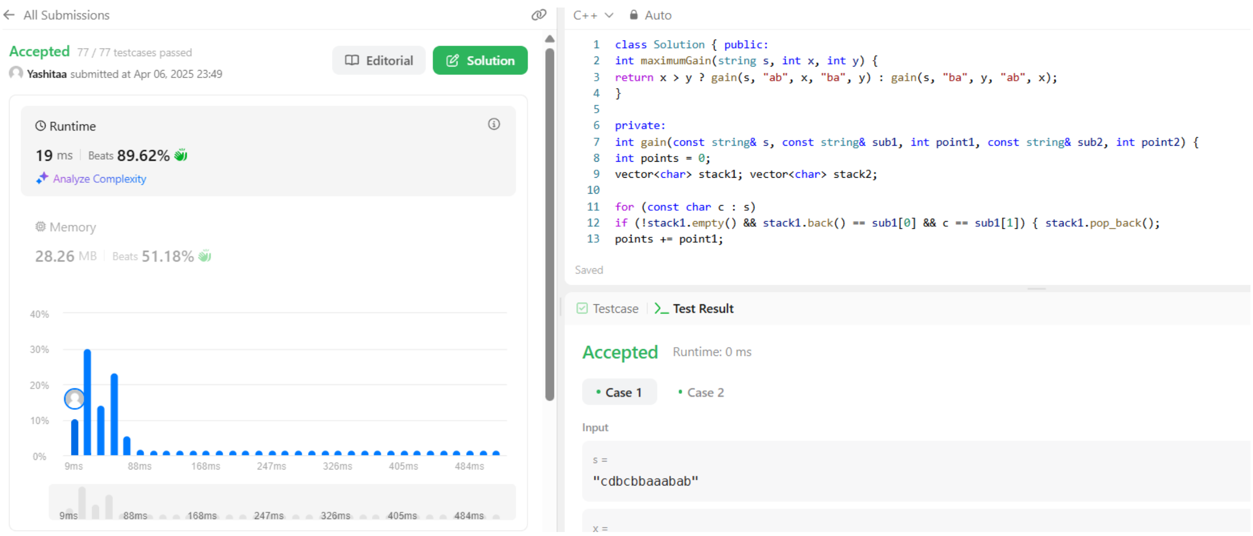
Accepted Runtime: 0 ms

Input

piles =

4. Maximum Score From Removing Substrings

```
class Solution {
 public:
  int maximumGain(string s, int x, int y) {
    return x > y ? gain(s, "ab", x, "ba", y) : gain(s, "ba", y, "ab", x);
  }

 private:
  int gain(const string& s, const string& sub1, int point1, const string& sub2,
           int point2) {
    int points = 0;
    vector<char> stack1;
    vector<char> stack2;
    for (const char c : s)
      if (!stack1.empty() && stack1.back() == sub1[0] && c == sub1[1]) {
        stack1.pop_back();
        points += point1;
      } else {
        stack1.push_back(c);
      }
    for (const char c : stack1)
      if (!stack2.empty() && stack2.back() == sub2[0] && c == sub2[1]) {
        stack2.pop_back();
        points += point2;
      } else {
        stack2.push_back(c);
      }
    return points;
  }
};
```



Accepted 77 / 77 testcases passed

Yashitaa submitted at Apr 06, 2025 23:49

19 ms | Beats 89.62%

28.26 MB | Beats 51.18%

```
class Solution { public:
int maximumGain(string s, int x, int y) {
return x > y ? gain(s, "ab", x, "ba", y) : gain(s, "ba", y, "ab", x);
}

private:
int gain(const string& s, const string& sub1, int point1, const string& sub2, int point2) {
int points = 0;
vector<char> stack1; vector<char> stack2;

for (const char c : s)
if (!stack1.empty() && stack1.back() == sub1[0] && c == sub1[1]) { stack1.pop_back();
points += point1;
```

Accepted Runtime: 0 ms

Input

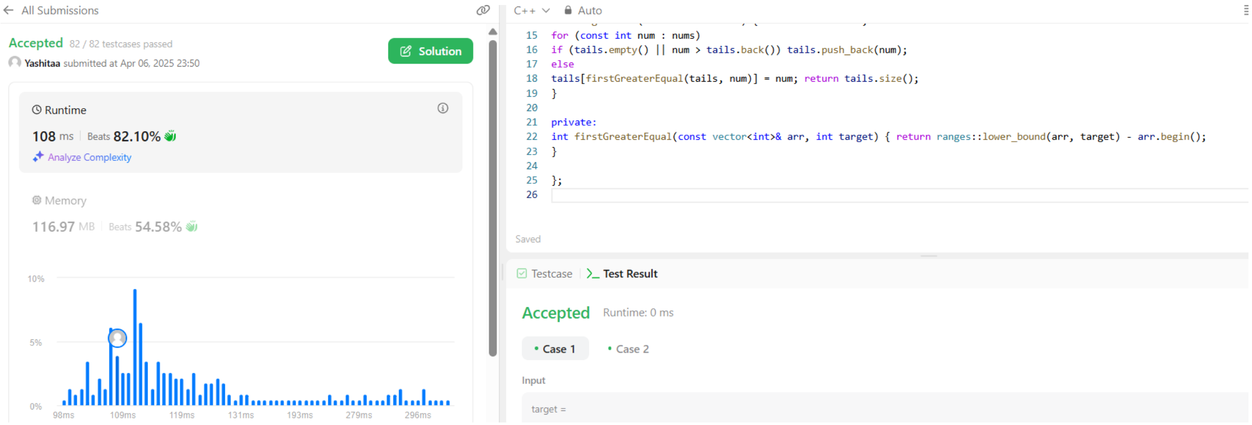
s =

"cdbcbbaaabab"

x =

5. Minimum Operations to Make a Subsequence

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

    for (int i = 0; i < target.size(); ++i)
      numToIndex[target[i]] = i;

    for (const int a : arr)
      if (const auto it = numToIndex.find(a); it != numToIndex.end())
        indices.push_back(it->second);

    return target.size() - lengthOfLIS(indices);
  }

 private:
  int lengthOfLIS(vector<int>& nums) {
    vector<int> tails;
    for (const int num : nums)
      if (tails.empty() || num > tails.back())
        tails.push_back(num);
      else
        tails[firstGreaterEqual(tails, num)] = num;
    return tails.size();
  }

 private:
  int firstGreaterEqual(const vector<int>& arr, int target) {
    return ranges::lower_bound(arr, target) - arr.begin();
  }
};
```



All Submissions

Accepted 82 / 82 testcases passed

Yashitaa submitted at Apr 06, 2025 23:50

Solution

Runtime

108 ms | Beats 82.10%

Analyze Complexity

Memory

116.97 MB | Beats 54.58%

C++ Auto

```
for (const int num : nums)
if (tails.empty() || num > tails.back()) tails.push_back(num);
else
tails[firstGreaterEqual(tails, num)] = num; return tails.size();
}

private:
int firstGreaterEqual(const vector<int>& arr, int target) { return ranges::lower_bound(arr, target) - arr.begin();
}

};
```

Saved

Testcase | Test Result

Accepted Runtime: 0 ms

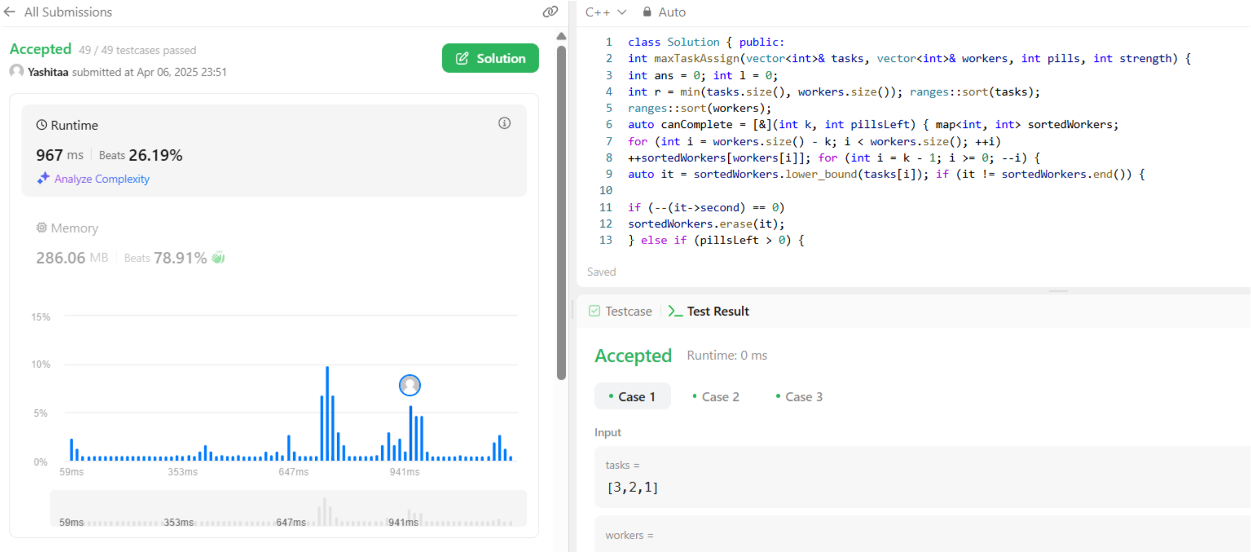
• Case 1 • Case 2

Input

target =

6. Maximum Number of Tasks You Can Assign

```
class Solution {
 public:
  int maxTaskAssign(vector<int>& tasks, vector<int>& workers, int pills,
                    int strength) {
    int ans = 0;
    int l = 0;
    int r = min(tasks.size(), workers.size());
    ranges::sort(tasks);
    ranges::sort(workers);
    auto canComplete = [&](int k, int pillsLeft) {
      map<int, int> sortedWorkers;
      for (int i = workers.size() - k; i < workers.size(); ++i)
        ++sortedWorkers[workers[i]];
      for (int i = k - 1; i >= 0; --i) {
        auto it = sortedWorkers.lower_bound(tasks[i]);
        if (it != sortedWorkers.end()) {
          if (--(it->second) == 0)
            sortedWorkers.erase(it);
        } else if (pillsLeft > 0) {
          it = sortedWorkers.lower_bound(tasks[i] - strength);
          if (it != sortedWorkers.end()) {
            if (--(it->second) == 0)
              sortedWorkers.erase(it);
            --pillsLeft;
          } else {
            return false;
          }
        } else {
          return false;
        }
      }

      return true;
    };

    while (l <= r) {
      const int m = (l + r) / 2;
      if (canComplete(m, pills)) {
        ans = m;
        l = m + 1;
      } else {
        r = m - 1;
      }
    }

    return ans;
  }
};
```



Accepted 49 / 49 testcases passed

Yashitaa submitted at Apr 06, 2025 23:51

Runtime

967 ms | Beats 26.19%

Memory

286.06 MB | Beats 78.91%

```cpp
class Solution { public:
int maxTaskAssign(vector<int>& tasks, vector<int>& workers, int pills, int strength) {
int ans = 0; int l = 0;
int r = min(tasks.size(), workers.size()); ranges::sort(tasks);
ranges::sort(workers);
auto canComplete = [&](int k, int pillsLeft) { map<int, int> sortedWorkers;
for (int i = workers.size() - k; i < workers.size(); ++i)
++sortedWorkers[workers[i]]; for (int i = k - 1; i >= 0; --i) {
auto it = sortedWorkers.lower_bound(tasks[i]); if (it != sortedWorkers.end()) {

if (--(it->second) == 0)
sortedWorkers.erase(it);
} else if (pillsLeft > 0) {
```

Saved

Accepted Runtime: 0 ms

Input

tasks =

[3,2,1]

workers =