

AP Experiment-8

Name: Arnav Jain

UID: 22BCS15161

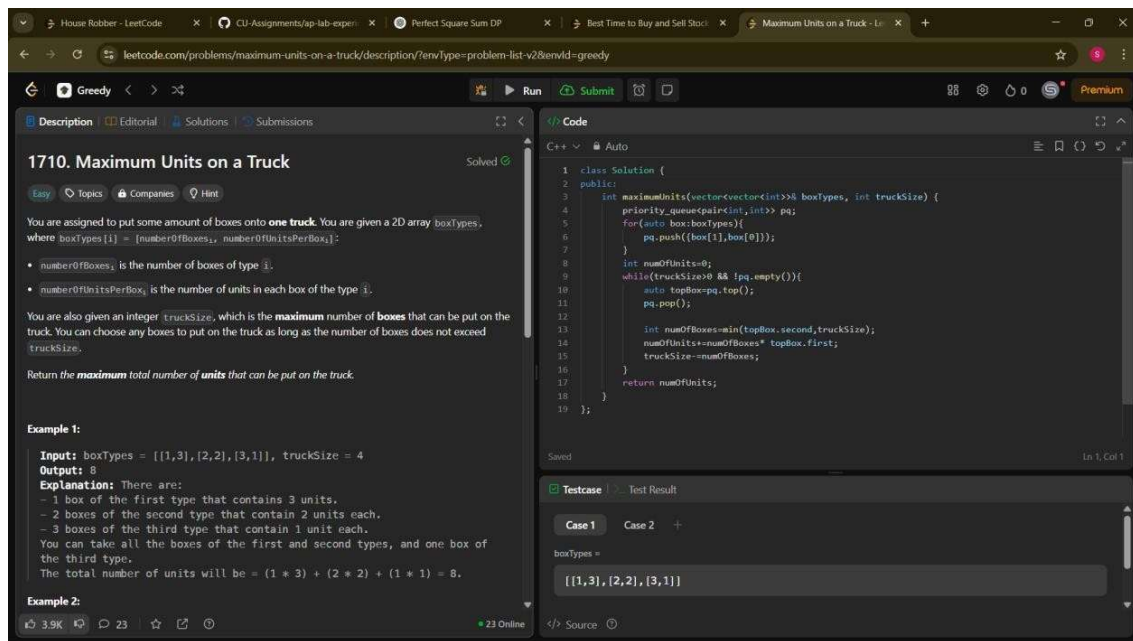
Section: 601-A

Q1. Maximum Units on a Truck <https://leetcode.com/problems/maximum-units-on-a-truck/description/?envType=problem-list-v2&envId=greedy>

CODE:

```
class Solution {
public:
    int maximumUnits(vector<vector<int>>& boxTypes, int truckSize) {
        priority_queue<pair<int,int>> pq;
        for(auto box:boxTypes){
            pq.push({box[1],box[0]});
        }
        int numOfUnits=0;
        while(truckSize>0 && !pq.empty()){
            auto topBox=pq.top();
            pq.pop();

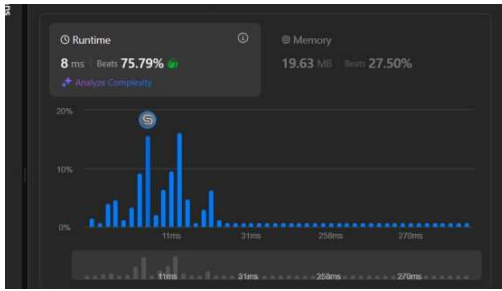
            int numOfBoxes=min(topBox.second,truckSize);
            numOfUnits+=numOfBoxes* topBox.first;
            truckSize-=numOfBoxes;
        }
        return numOfUnits;
    }
};
```



Q2. <https://leetcode.com/problems/minimum-operations-to-make-the-array-increasing/description/?envType=problem-list-v2&envId=greedy>

CODE:

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



Q3. <https://leetcode.com/problems/remove-stones-to-minimize-the-total/?envType=problem-list-v2&envId=greedy>

CODE:

```
class Solution {
```

```
public:
```

```
    int minStoneSum(vector<int>& piles, int k) {
        priority_queue<int> maxHeap;
```

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

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

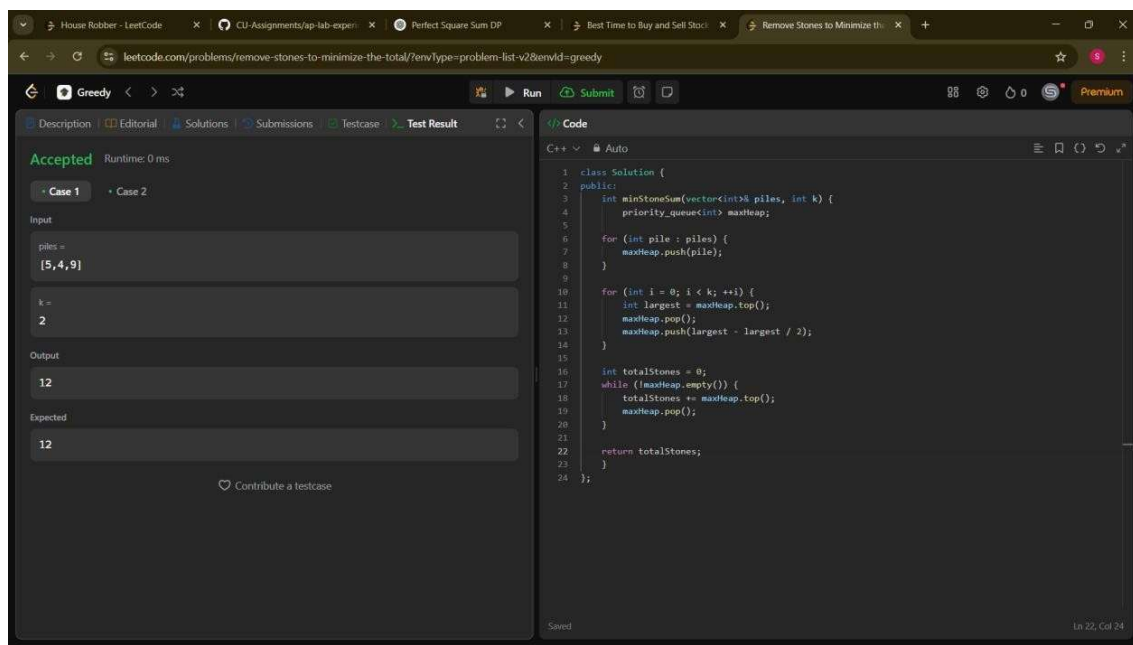
```
        int totalStones = 0;
```

```

while (!maxHeap.empty()) {
    totalStones += maxHeap.top();
    maxHeap.pop();
}

return totalStones;
}
};

```



Q4. <https://leetcode.com/problems/maximum-score-from-removing-substrings/?envType=problem-list-v2&envId=greedy>

CODE:

```

class Solution {
public:
    int maximumGain(string s, int x, int y) {
        int points = 0;

        while (true) {
            size_t pos_ab = s.find("ab");

```

```

size_t pos_ba = s.find("ba");

if (pos_ab != string::npos && (pos_ba == string::npos || x >= y)) {
    s.erase(pos_ab, 2);
    points += x;
} else if (pos_ba != string::npos) {
    s.erase(pos_ba, 2);
    points += y;
} else {
    break;
}
}

return points;
}

};

```

The screenshot displays the LeetCode interface for the problem "Maximum Score From Removing Substrings". The problem is marked as "Accepted" with a runtime of 0 ms. The input string is "cdbcbbaaabab", x is 4, and y is 5. The output is 19, which matches the expected result. The C++ code in the editor implements a greedy algorithm using a while loop to repeatedly remove the highest-scoring substring ("ab" or "ba") until no more can be found.

```

1 class Solution {
2 public:
3     int maximumGain(string s, int x, int y) {
4         int points = 0;
5
6         while (true) {
7             size_t pos_ab = s.find("ab");
8             size_t pos_ba = s.find("ba");
9
10            if (pos_ab != string::npos && (pos_ba == string::npos || x >= y)) {
11                s.erase(pos_ab, 2);
12                points += x;
13            } else if (pos_ba != string::npos) {
14                s.erase(pos_ba, 2);
15                points += y;
16            } else {
17                break;
18            }
19        }
20
21        return points;
22    }
23 };

```