

Name = Ishan Sharma

UID = 22BCS11144

Problem 1

```
major.cpp > findMajorityElement(const vector<int>&)
1  #include <iostream>
2  #include <vector>
3
4  using namespace std;
5
6  int findMajorityElement(const vector<int>& nums) {
7      int count = 0;
8      int candidate = 0;
9
10     for (int num : nums) {
11         if (count == 0) {
12             candidate = num;
13         }
14         count += (num == candidate) ? 1 : -1;
15     }
16
17     count = 0;
18     for (int num : nums) {
19         if (num == candidate) {
20             count++;
21         }
22     }
23
24     return candidate;
25 }
26
27 int main() {
28     int n;
29     cout << "Enter the number of elements in the array: ";
30     cin >> n;
31
32     vector<int> nums(n);
33     cout << "Enter the elements of the array: ";
34     for (int i = 0; i < n; ++i) {
35         cin >> nums[i];
36     }
37 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\ISHAM\OneDrive\Documents\coding> cd "c:
Enter the number of elements in the array: 5
Enter the elements of the array: 3 2 4 2 4
The majority element is: 4
PS C:\Users\ISHAM\OneDrive\Documents\coding> █
```

Problem 2

```
major.cpp  shape.cpp x container.cpp  happymods.cpp  minimumjobs.cpp
shape.cpp > generatePascalsTriangle(int)
1  #include <iostream>
2  #include <vector>
3
4  using namespace std;
5
6  vector<vector<int>> generatePascalsTriangle(int numRows) {
7      vector<vector<int>> triangle;
8
9      for (int i = 0; i < numRows; ++i) {
10         vector<int> row(i + 1, 1);
11
12         for (int j = 1; j < i; ++j) {
13             row[j] = triangle[i - 1][j - 1] + triangle[i - 1][j];
14         }
15
16         triangle.push_back(row);
17     }
18
19     return triangle;
20 }
21
22 int main() {
23     int numRows;
24     cout << "Enter the number of rows for Pascal's Triangle: ";
25     cin >> numRows;
26
27     vector<vector<int>> pascalsTriangle = generatePascalsTriangle(numRows);
28
29     cout << "Pascal's Triangle with " << numRows << " rows:" << endl;
30     for (const auto& row : pascalsTriangle) {
31         for (int num : row) {
32             cout << num << " ";
33         }
34         cout << endl;
35     }
36
37     return 0;
```

```
Enter the number of rows for Pascal's Triangle: 4
Pascal's Triangle with 4 rows:
1
1 1
1 2 1
1 3 3 1
PS C:\Users\ISHAM\OneDrive\Documents\coding>
```

Problem 3

```
container.cpp > maxArea(const vector<int>& height)
1  #include <bits/stdc++.h>
2
3  using namespace std;
4
5  int maxArea(const vector<int>& height) {
6      int maxWater = 0;
7      int left = 0, right = height.size() - 1;
8
9      while (left < right) {
10         int width = right - left;
11         int currentHeight = min(height[left], height[right]);
12         maxWater = max(maxWater, width * currentHeight);
13
14         if (height[left] < height[right]) {
15             ++left;
16         } else {
17             --right;
18         }
19     }
20
21     return maxWater;
22 }
23
24 int main() {
25     int n;
26     cout << "Enter the number of lines: ";
27     cin >> n;
28
29     vector<int> height(n);
30     cout << "Enter the heights of the lines: ";
31     for (int i = 0; i < n; ++i) {
32         cin >> height[i];
33     }
34
35     int result = maxArea(height);
36     cout << "The maximum amount of water a container can store is: " << result << endl;
37
38     return 0;
39 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\ISHAM\OneDrive\Documents\coding> cd "c:\Users\ISHAM\OneDrive\Documents\coding\"
Enter the number of lines: 3
Enter the heights of the lines: 2
3
4
The maximum amount of water a container can store is: 4
PS C:\Users\ISHAM\OneDrive\Documents\coding> |
```

Problem 4

```
major.cpp shape.cpp container.cpp happymods.cpp X minimumjobs.cpp
happymods.cpp > maxHappyGroups(int, vector<int>&)
1  #include <bits/stdc++.h>
2
3  using namespace std;
4
5  int maxHappyGroups(int batchSize, vector<int>& groups) {
6      vector<int> remainderCount(batchSize, 0);
7      int happyGroups = 0;
8
9      for (int group : groups) {
10         int remainder = group % batchSize;
11         if (remainder == 0) {
12             happyGroups++;
13         } else if (remainderCount[batchSize - remainder] > 0) {
14             happyGroups++;
15             remainderCount[batchSize - remainder]--;
16         } else {
17             remainderCount[remainder]++;
18         }
19     }
20
21     int remainingGroups = 0;
22     for (int i = 1; i < batchSize; ++i) {
23         remainingGroups += remainderCount[i];
24     }
25
26     happyGroups += remainingGroups / batchSize;
27
28     return happyGroups;
29 }
30
31 int main() {
32     int batchSize, n;
33     cout << "Enter the batch size: ";
34     cin >> batchSize;
35
36     cout << "Enter the number of groups: ";
37     cin >> n;
38
39     vector<int> groups(n);
40     cout << "Enter the size of each group: ";
41     for (int i = 0; i < n; ++i) {
42         cin >> groups[i];
43     }
44
45     int result = maxHappyGroups(batchSize, groups);
46     cout << "The maximum number of happy groups is: " << result << endl;
47
48     return 0;
49 }
50
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\ISHAM\OneDrive\Documents\coding> cd "c:\Users\ISHAM\OneDrive\Documents\coding\" ; i
Enter the batch size: 4
Enter the number of groups: 4
Enter the size of each group: 3 3 2 4
The maximum number of happy groups is: 1
PS C:\Users\ISHAM\OneDrive\Documents\coding>
```

Problem 5

```
major.cpp  shape.cpp  container.cpp  happymods.cpp  minimumjobs.cpp x
minimumjobs.cpp > isFeasible(vector<int>& jobs, vector<int>& workers, int idx, int maxTime) {
1  #include <bits/stdc++.h>
2
3  using namespace std;
4
5  bool isFeasible(vector<int>& jobs, vector<int>& workers, int idx, int maxTime) {
6      if (idx == jobs.size()) return true;
7
8      for (int i = 0; i < workers.size(); ++i) {
9          if (workers[i] + jobs[idx] <= maxTime) {
10             workers[i] += jobs[idx];
11             if (isFeasible(jobs, workers, idx + 1, maxTime)) return true;
12             workers[i] -= jobs[idx];
13         }
14
15         if (workers[i] == 0) break;
16     }
17
18     return false;
19 }
20
21 int findMinimumTime(vector<int>& jobs, int k) {
22     sort(jobs.rbegin(), jobs.rend());
23
24     int left = jobs[0], right = accumulate(jobs.begin(), jobs.end(), 0), result = right;
25
26     while (left <= right) {
27         int mid = left + (right - left) / 2;
28         vector<int> workers(k, 0);
29
30         if (isFeasible(jobs, workers, 0, mid)) {
31             result = mid;
32             right = mid - 1;
33         } else {
34             left = mid + 1;
35         }
36     }
37
38     return result;
39 }
40
41 int main() {
42     int n, k;
43     cout << "Enter the number of jobs: ";
44     cin >> n;
45
46     vector<int> jobs(n);
47     cout << "Enter the time required for each job: ";
48     for (int i = 0; i < n; ++i) {
49         cin >> jobs[i];
50     }
51
52     cout << "Enter the number of workers: ";
53     cin >> k;
54
55     int result = findMinimumTime(jobs, k);
56     cout << "The minimum possible maximum working time is: " << result << endl;
57
58     return 0;
59 }
60
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\ISHAM\OneDrive\Documents\coding> cd "c:\Users\ISHAM\OneDrive\Documents\coding\" ;  
}  
Enter the number of jobs: 4  
Enter the time required for each job: 8  
3  
5  
6  
Enter the number of workers: 4  
The minimum possible maximum working time is: 8  
PS C:\Users\ISHAM\OneDrive\Documents\coding> |
```