Experiment 10

Student Name: Riyan Reyaz	**UID:22BCS11986**
Branch: CSE	**Section/Group: NTPP 603/B**
Semester: 06	**Date of Performance: 18/04/2025**
Subject Name: AP Lab II	**Subject Code: 22CSP-351**

1. **Aim**:
   a) Pascal's Triangle
   b) Hamming Distance
   c) Task Scheduler

2. **Source Code:**

a.

```
class Solution {
 public:
  vector<vector<int>> generate(int numRows) {
    vector<vector<int>> ans;

    for (int i = 0; i < numRows; ++i)
      ans.push_back(vector<int>(i + 1, 1));

    for (int i = 2; i < numRows; ++i)
      for (int j = 1; j < ans[i].size() - 1; ++j)
        ans[i][j] = ans[i - 1][j - 1] + ans[i - 1][j];

    return ans;
  }
};
```

b.

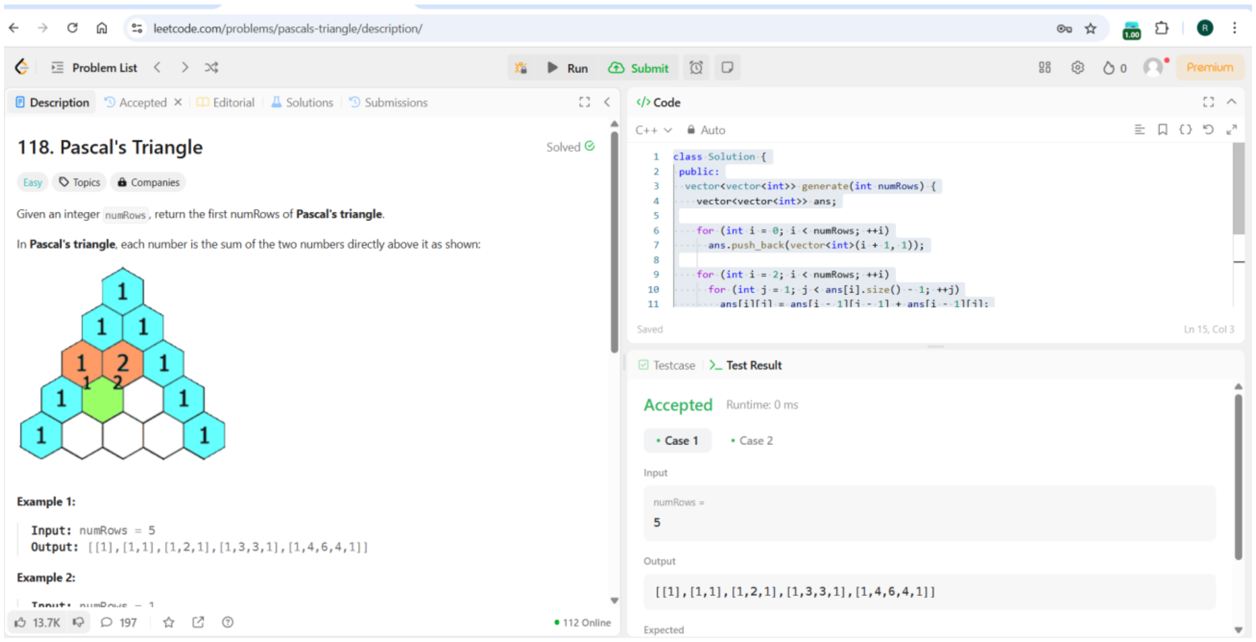
```
class Solution {
 public:
  int hammingDistance(int x, int y) {
    int ans = 0;

    while (x > 0 || y > 0) {
      ans += (x & 1) ^ (y & 1);
      x >>= 1;
      y >>= 1;
    }

    return ans;
  }
};
```

c.

```
class Solution {
 public:
  int leastInterval(vector<char>& tasks, int n) {
    if (n == 0)
      return tasks.size();

    vector<int> count(26);

    for (const char task : tasks)
      ++count[task - 'A'];

    const int maxFreq = ranges::max(count);
    const int maxFreqTaskOccupy = (maxFreq - 1) * (n + 1);
    const int nMaxFreq = ranges::count(count, maxFreq);
    return max(maxFreqTaskOccupy + nMaxFreq, static_cast<int>(tasks.size()));
  }
};
```

Screenshot of Outputs:

a.



118. Pascal's Triangle

Given an integer `numRows`, return the first numRows of **Pascal's triangle**.

In **Pascal's triangle**, each number is the sum of the two numbers directly above it as shown:

Example 1:

```
Input: numRows = 5
Output: [[1],[1,1],[1,2,1],[1,3,3,1],[1,4,6,4,1]]
```

Example 2:

```cpp
class Solution {
public:
  vector<vector<int>> generate(int numRows) {
    vector<vector<int>> ans;

    for (int i = 0; i < numRows; ++i)
      ans.push_back(vector<int>(i + 1, 1));

    for (int i = 2; i < numRows; ++i)
      for (int j = 1; j < ans[i].size() - 1; ++j)
        ans[i][j] = ans[i - 1][j - 1] + ans[i - 1][j];
```

Saved   Ln 15, Col 3

Accepted   Runtime: 0 ms

Input

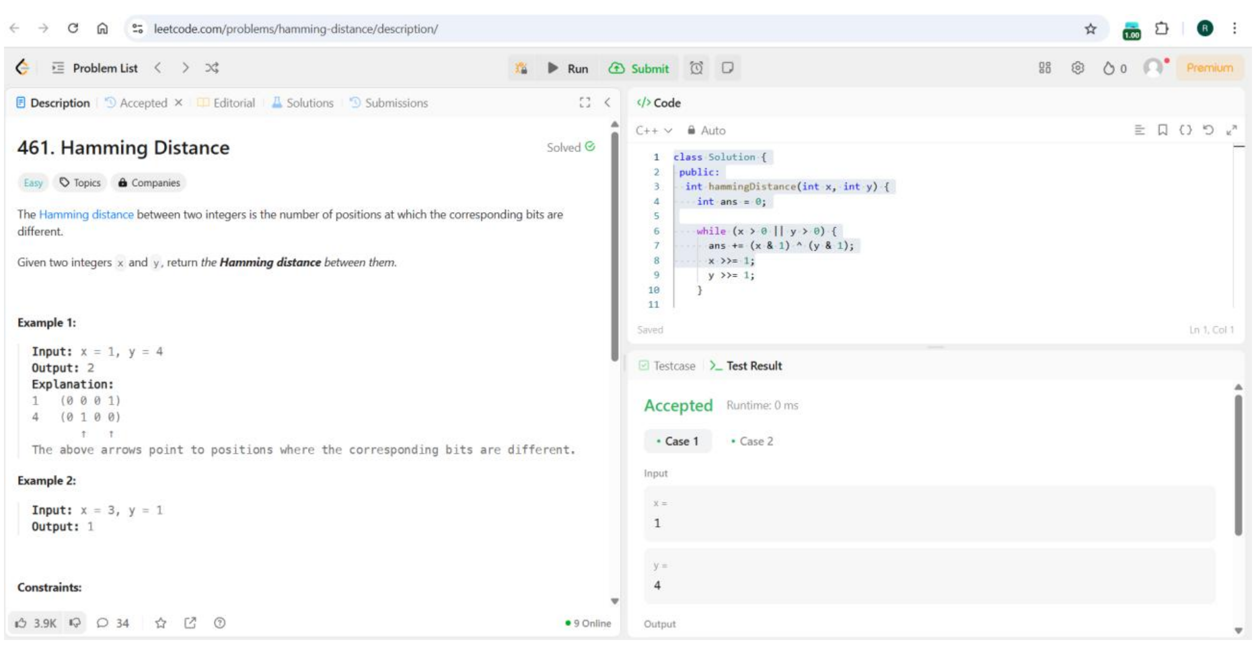
```
numRows =
5
```

Output

```
[[1],[1,1],[1,2,1],[1,3,3,1],[1,4,6,4,1]]
```

Expected

b.



461. Hamming Distance

The Hamming distance between two integers is the number of positions at which the corresponding bits are different.

Given two integers `x` and `y`, return *the **Hamming distance** between them.*

Example 1:

```
Input: x = 1, y = 4
Output: 2
Explanation:
1   (0 0 0 1)
4   (0 1 0 0)
       ↑   ↑
The above arrows point to positions where the corresponding bits are different.
```

Example 2:

```
Input: x = 3, y = 1
Output: 1
```

Constraints:

```cpp
class Solution {
public:
  int hammingDistance(int x, int y) {
    int ans = 0;

    while (x > 0 || y > 0) {
      ans += (x & 1) ^ (y & 1);
      x >>= 1;
      y >>= 1;
    }

```

Saved   Ln 1, Col 1

Accepted   Runtime: 0 ms

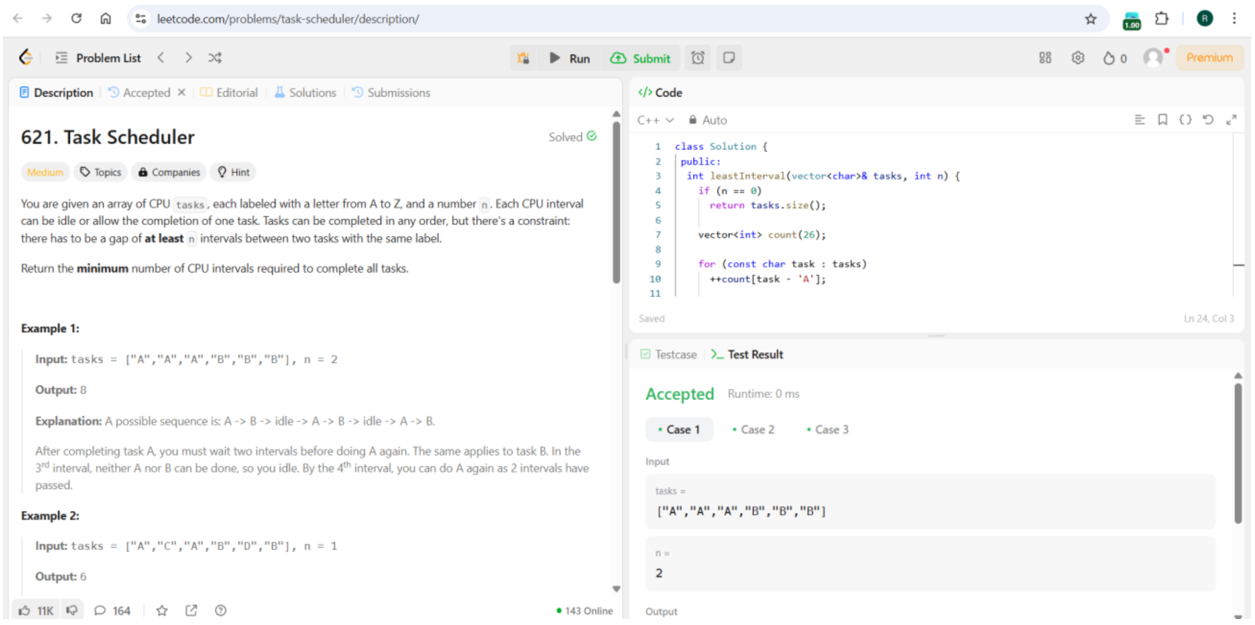
Input

```
x =
1
```

```
y =
4
```

Output

c.



621. Task Scheduler

Solved

You are given an array of CPU `tasks`, each labeled with a letter from A to Z, and a number `n`. Each CPU interval can be idle or allow the completion of one task. Tasks can be completed in any order, but there's a constraint: there has to be a gap of **at least** `n` intervals between two tasks with the same label.

Return the **minimum** number of CPU intervals required to complete all tasks.

Example 1:

Input: tasks = ["A","A","A","B","B","B"], n = 2

Output: 8

Explanation: A possible sequence is: A -> B -> idle -> A -> B -> idle -> A -> B.

After completing task A, you must wait two intervals before doing A again. The same applies to task B. In the 3rd interval, neither A nor B can be done, so you idle. By the 4th interval, you can do A again as 2 intervals have passed.

Example 2:

Input: tasks = ["A","C","A","B","D","B"], n = 1

Output: 6

C++ Auto

```cpp
class Solution {
 public:
  int leastInterval(vector<char>& tasks, int n) {
    if (n == 0)
      return tasks.size();

    vector<int> count(26);

    for (const char task : tasks)
      ++count[task - 'A'];

```

Saved   Ln 24, Col 3

Testcase | Test Result

Accepted   Runtime: 0 ms

Case 1   Case 2   Case 3

Input

tasks =
["A","A","A","B","B","B"]

n =
2

Output

3) Learning Outcomes:

1) **Learned about various miscellaneous problems.**