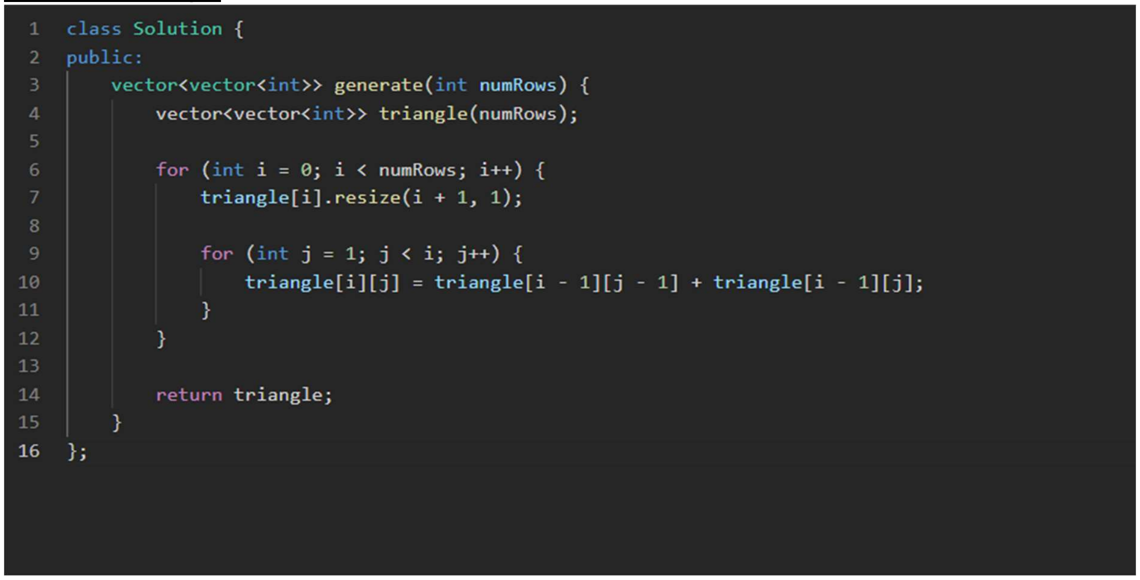
1. **Pascal's Triangle**:



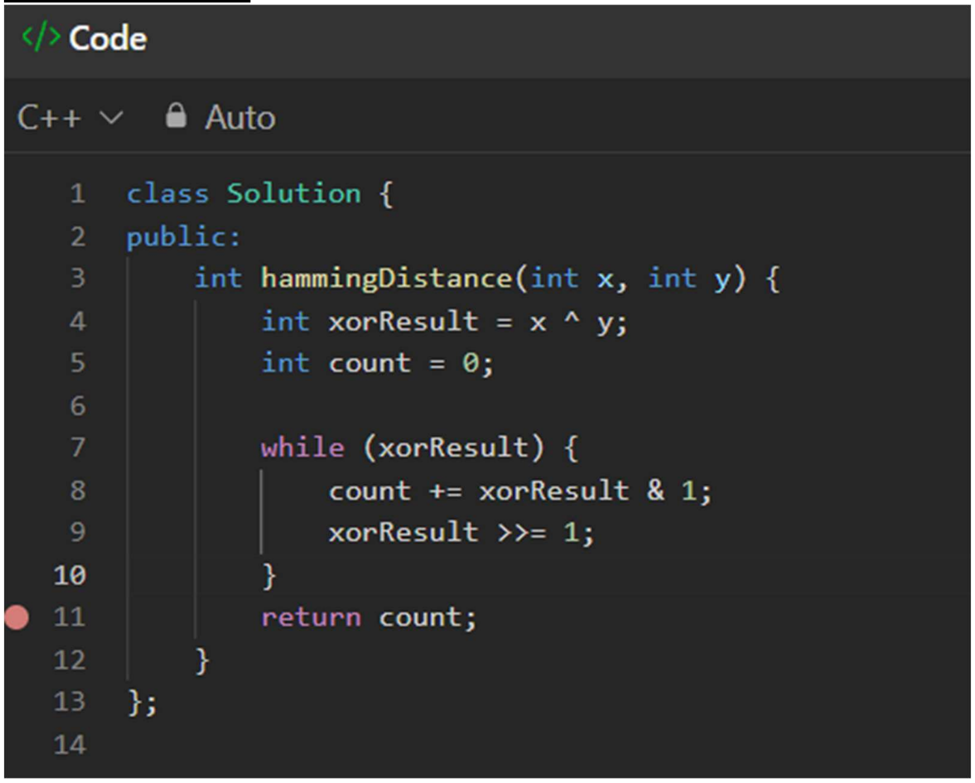
```cpp
class Solution {
public:
    vector<vector<int>> generate(int numRows) {
        vector<vector<int>> triangle(numRows);

        for (int i = 0; i < numRows; i++) {
            triangle[i].resize(i + 1, 1);

            for (int j = 1; j < i; j++) {
                triangle[i][j] = triangle[i - 1][j - 1] + triangle[i - 1][j];
            }
        }

        return triangle;
    }
};
```

2. **Hamming Distance**:



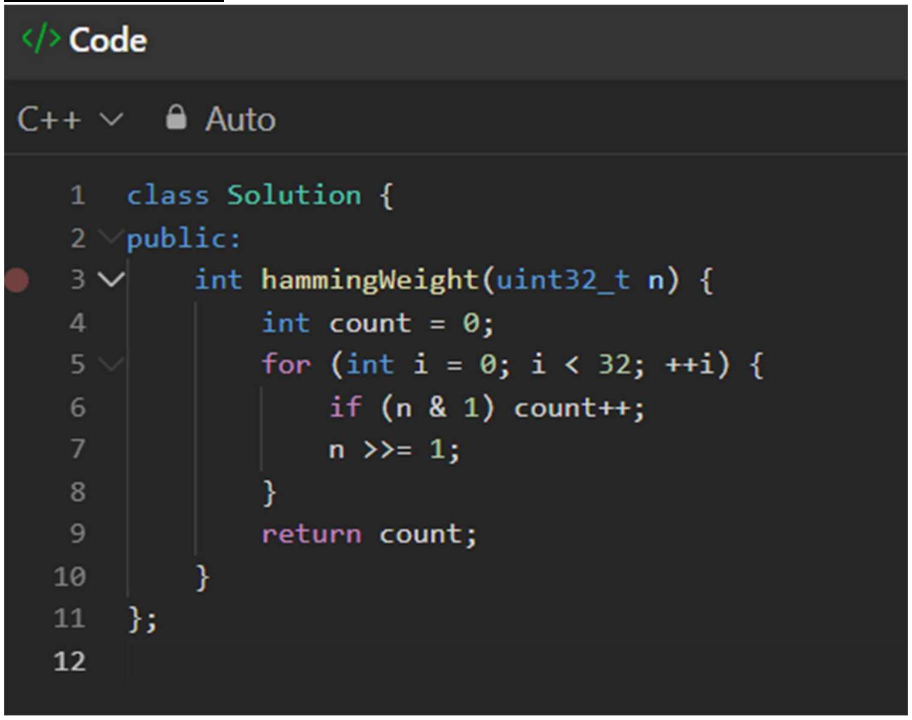
Code

C++ Auto

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

        while (xorResult) {
            count += xorResult & 1;
            xorResult >>= 1;
        }
        return count;
    }
};
```

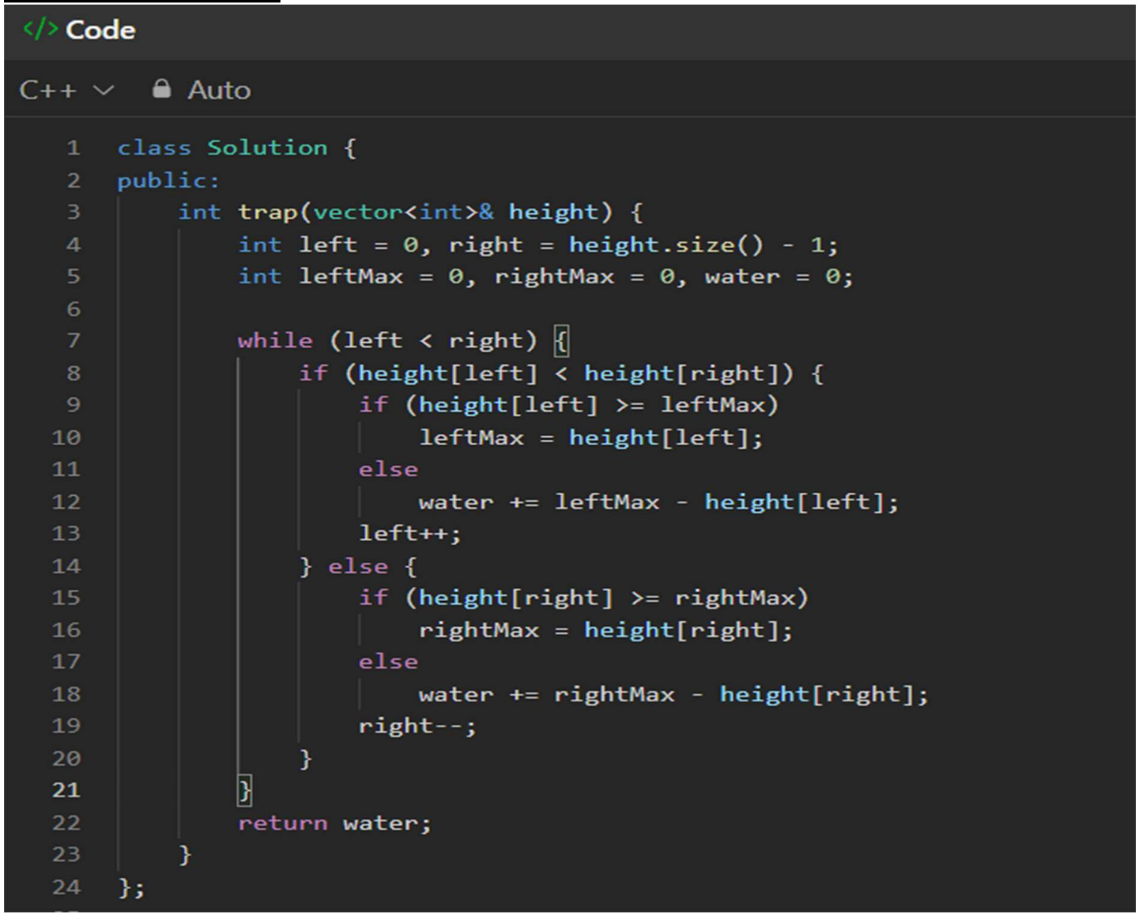
3. **Number of 1 Bits**:



```cpp
class Solution {
public:
    int hammingWeight(uint32_t n) {
        int count = 0;
        for (int i = 0; i < 32; ++i) {
            if (n & 1) count++;
            n >>= 1;
        }
        return count;
    }
};

```

4. **Trapping Rain Water**:



```cpp
class Solution {
public:
    int trap(vector<int>& height) {
        int left = 0, right = height.size() - 1;
        int leftMax = 0, rightMax = 0, water = 0;

        while (left < right) {
            if (height[left] < height[right]) {
                if (height[left] >= leftMax)
                    leftMax = height[left];
                else
                    water += leftMax - height[left];
                left++;
            } else {
                if (height[right] >= rightMax)
                    rightMax = height[right];
                else
                    water += rightMax - height[right];
                right--;
            }
        }
        return water;
    }
};
```