NAME : Yashita

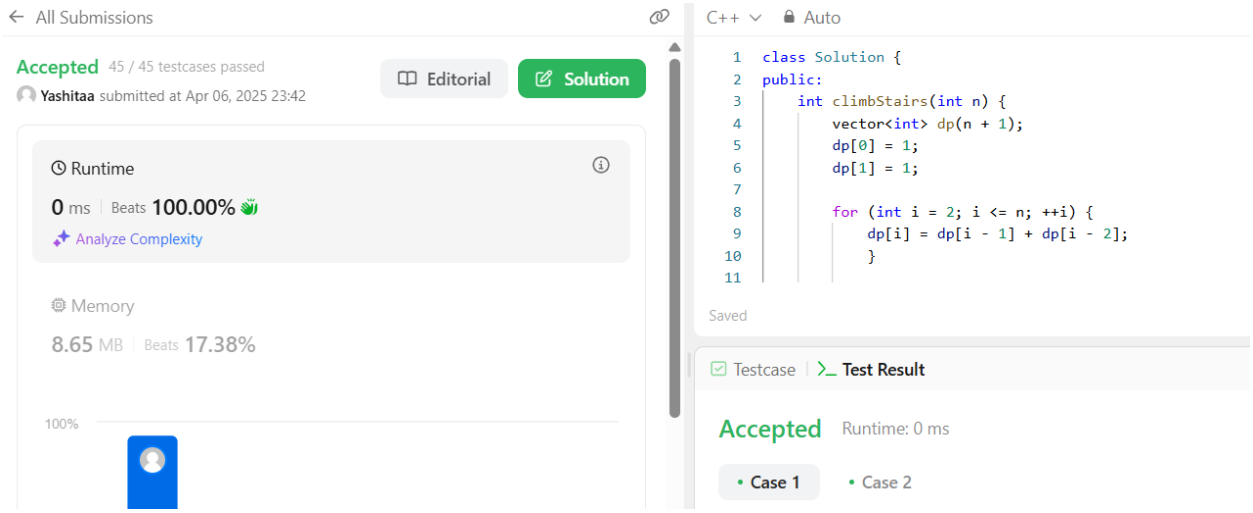
UID : 22BCS15024

SECTION : Fl_Iot 601 ‘A’

Ap experiment 7

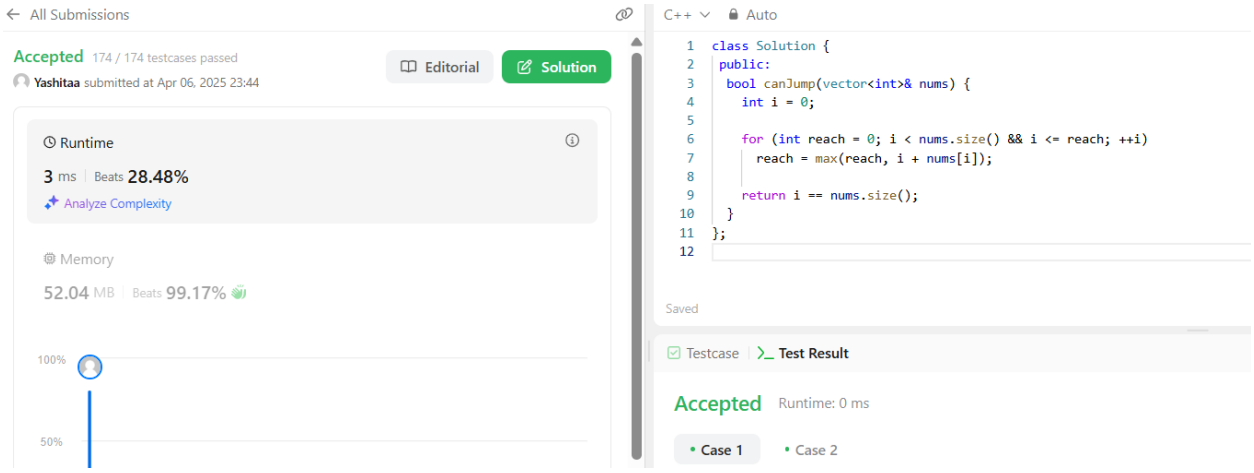
1. **Climbing Stairs**

```
class Solution {

public:

   int climbStairs(int n) {

      vector<int> dp(n + 1);

      dp[0] = 1;

      dp[1] = 1;


      for (int i = 2; i <= n; ++i) {

         dp[i] = dp[i - 1] + dp[i - 2];

         }


      return dp[n];

   }

};
```



2. **Jump Game**

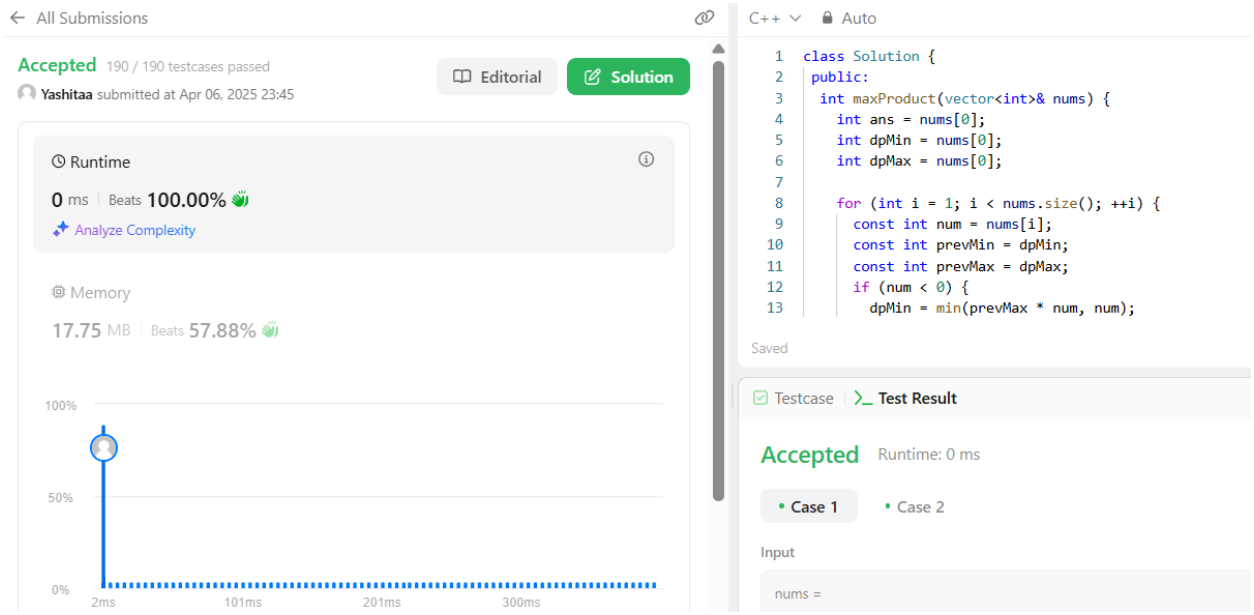
```
class Solution {
 public:
  bool canJump(vector<int>& nums) {
    int i = 0;

    for (int reach = 0; i < nums.size() && i <= reach; ++i)
      reach = max(reach, i + nums[i]);

    return i == nums.size();
  }
};
```



3. **Maximum Product Subarray**

```
class Solution {
 public:
  int maxProduct(vector<int>& nums) {
    int ans = nums[0];
    int dpMin = nums[0];
    int dpMax = nums[0];

    for (int i = 1; i < nums.size(); ++i) {
```

```
    const int num = nums[i];
    const int prevMin = dpMin;
    const int prevMax = dpMax;
    if (num < 0) {
      dpMin = min(prevMax * num, num);
      dpMax = max(prevMin * num, num);
    } else {
      dpMin = min(prevMin * num, num);
      dpMax = max(prevMax * num, num);
    }
    ans = max(ans, dpMax);
  }

  return ans;
 }
};
```



4. **Perfect Squares**

```
class Solution {
 public:
  int numSquares(int n) {
    vector<int> dp(n + 1, n);
    dp[0] = 0;
    dp[1] = 1;

    for (int i = 2; i <= n; ++i)
      for (int j = 1; j * j <= i; ++j)
        dp[i] = min(dp[i], dp[i - j * j] + 1);

    return dp[n];
  }
};
```