

Assignment-04

Student Name: Akash Pandey
Branch: CSE
Semester: 6th
Subject: Advance Programming Lab-II

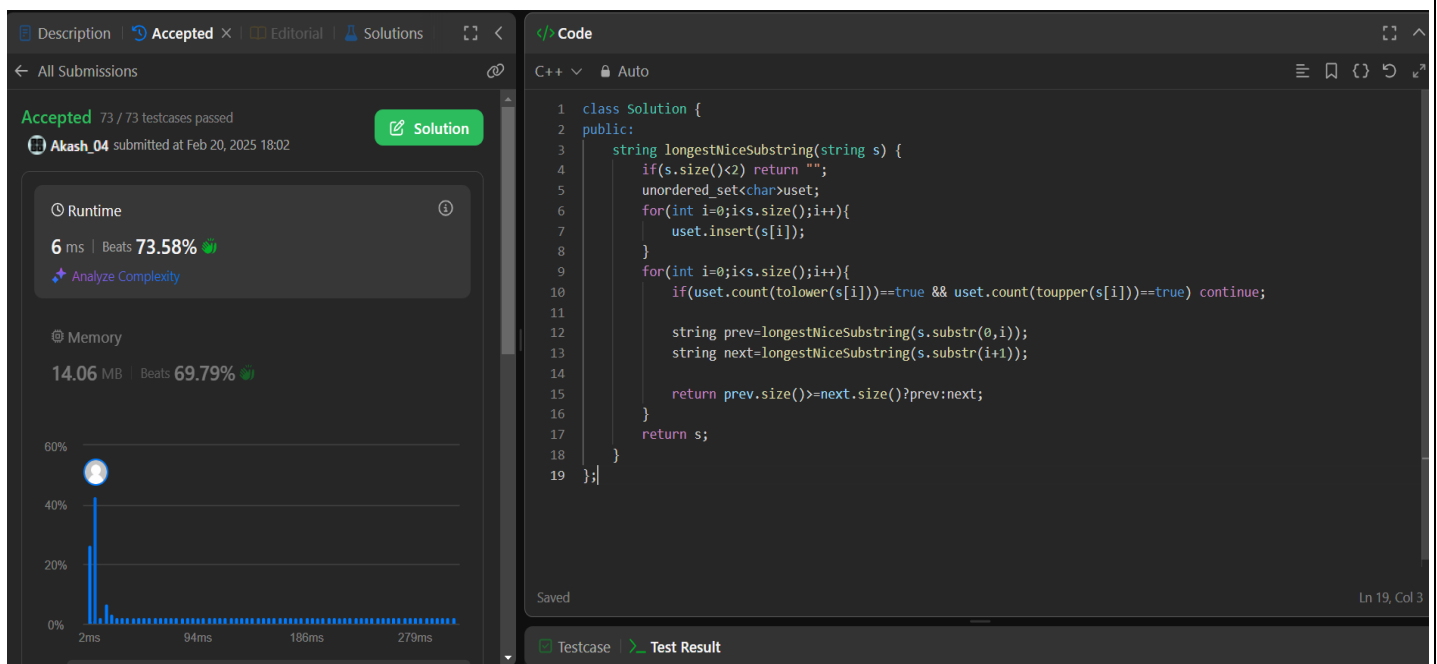
UID: 22BCS11135
Section: 22BCS_FL-601A
DOP: 20-02-25
Subject Code: 22CSP-351

1. Longest Nice Substring

```
class Solution {
public:
    string longestNiceSubstring(string s) {
        if(s.size()<2) return "";
        unordered_set<char>uset;
        for(int i=0;i<s.size();i++){
            uset.insert(s[i]);
        }
        for(int i=0;i<s.size();i++){
            if(uset.count(tolower(s[i]))==true && uset.count(toupper(s[i]))==true) continue;

            string prev=longestNiceSubstring(s.substr(0,i));
            string next=longestNiceSubstring(s.substr(i+1));

            return prev.size()>=next.size()?prev:next;
        }
        return s;
    }
};
```

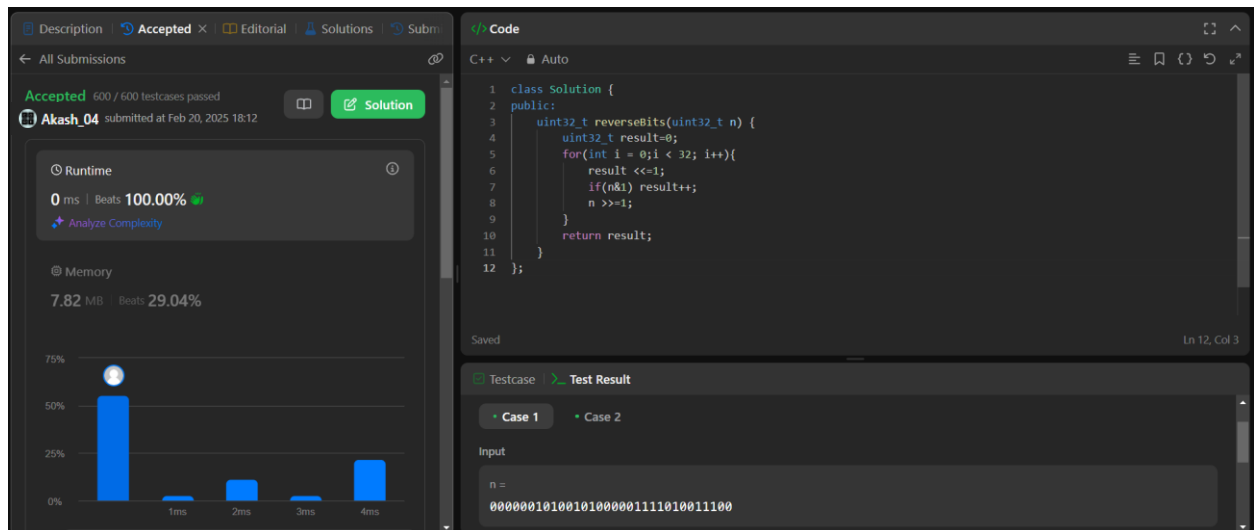


The screenshot shows a code editor interface with the following details:

- Top Bar:** Description, Accepted (73 / 73 testcases passed), Editorial, Solutions.
- Submission Info:** Akash_04 submitted at Feb 20, 2025 18:02. A green 'Solution' button is visible.
- Runtime Statistics:**
 - 6 ms | Beats 73.58%
 - Analyze Complexity button.
- Memory Statistics:**
 - 14.06 MB | Beats 69.79%
- Performance Graph:** A bar chart showing runtime performance across different test cases, with a peak at 2ms.
- Code Editor:**
 - Language: C++
 - Code content: The same C++ solution as provided in the text above.
 - Line numbers: 1 to 19.
 - Status: Saved.
 - Bottom bar: Testcase, Test Result.

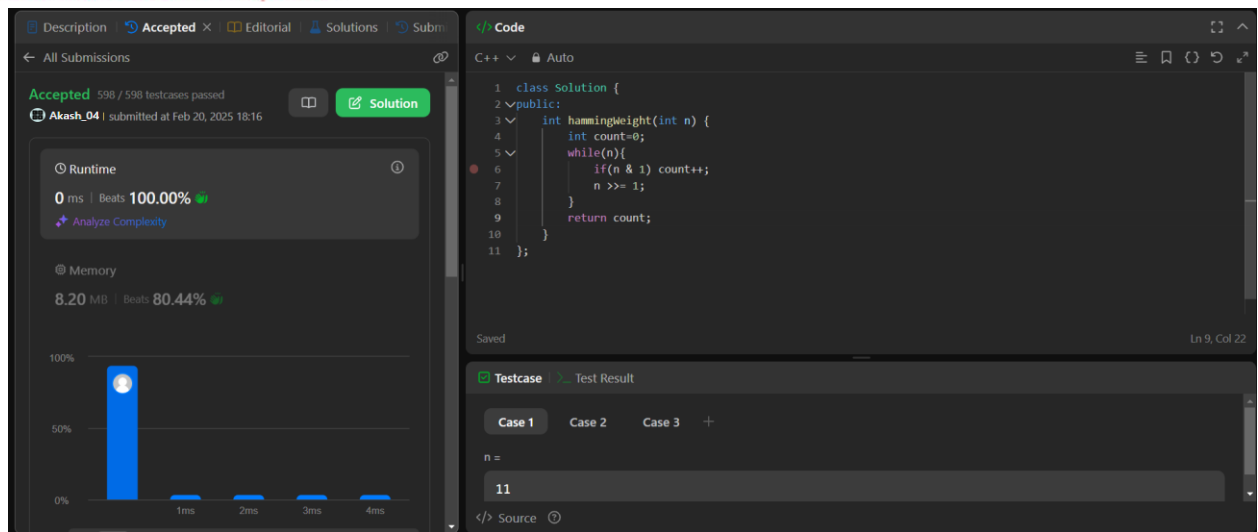
2. Reverse bits

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



3 Number of 1 bit

```
class Solution {
public:
    int hammingWeight(int n) {
        int count=0;
        while(n){
            if(n & 1) count++;
            n >>= 1;
        }
        return count;
    }
};
```

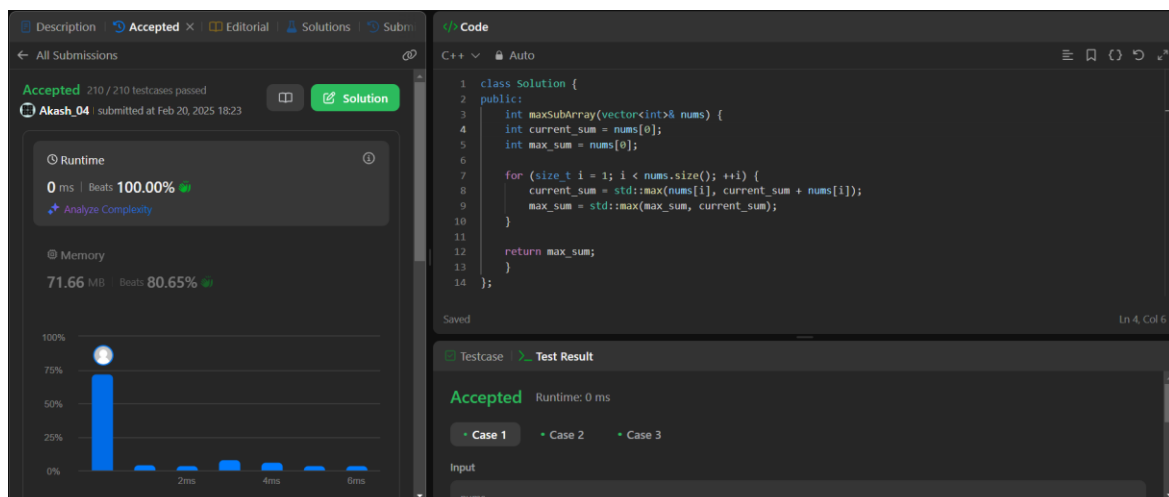


4. maximum Subarray

```
class Solution {
public:
    int maxSubArray(vector<int>& nums) {
        int current_sum = nums[0];
        int max_sum = nums[0];

        for (size_t i = 1; i < nums.size(); ++i) {
            current_sum = std::max(nums[i], current_sum + nums[i]);
            max_sum = std::max(max_sum, current_sum);
        }

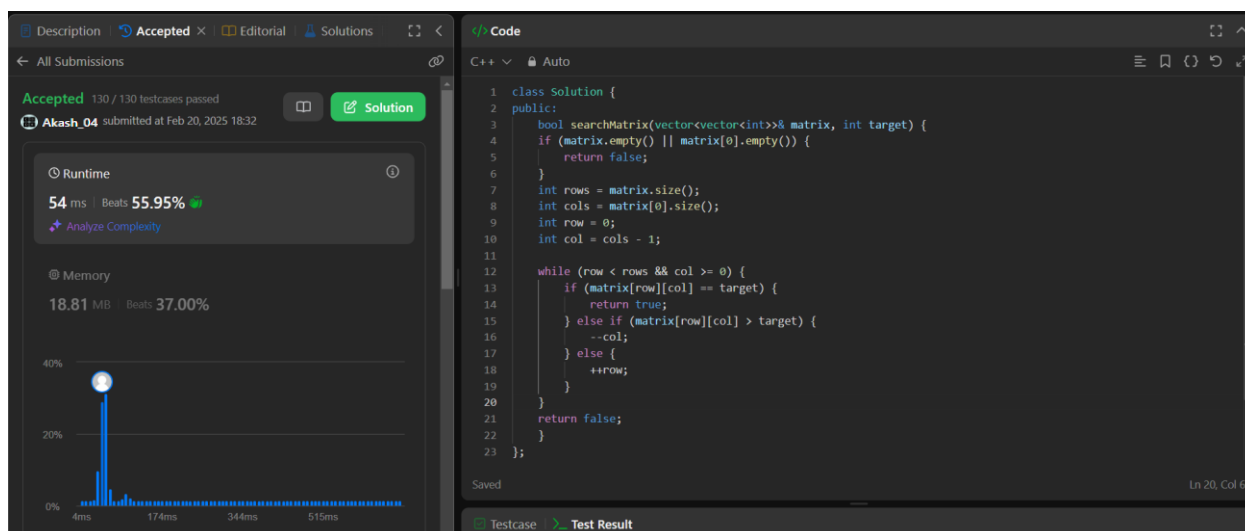
        return max_sum;
    }
};
```



5. Search a 2D Matrix

```
class Solution {
public:
    bool searchMatrix(vector<vector<int>>& matrix, int target) {
        if (matrix.empty() || matrix[0].empty()) {
            return false;
        }
        int rows = matrix.size();
        int cols = matrix[0].size();
        int row = 0;
        int col = cols - 1;

        while (row < rows && col >= 0) {
            if (matrix[row][col] == target) {
                return true;
            } else if (matrix[row][col] > target) {
                --col;
            } else {
                ++row;
            }
        }
        return false;
    }
};
```



6. Search Pow

```
class Solution {
public:
    int modPow(int a, int b, int mod) {
        int result = 1;
        a %= mod;
        while (b > 0) {
            if (b % 2 == 1) {
                result = (result * a) % mod;
            }
            a = (a * a) % mod;
            b /= 2;
        }
        return result;
    }
    int superPow(int a, vector<int>& b) {
        const int mod = 1337;
        int result = 1;
        for (int digit : b) {
            result = modPow(result, 10, mod) * modPow(a, digit, mod) % mod;
        }
        return result;
    }
};
```

