All Contests > In21-CS2023-Lab3 > Array Manipulation

Array Manipulation

Problem Submissions Leaderboard Discussions

Starting with a 1-indexed array of zeros and a list of operations, for each operation add a value to each the array element between two given indices, inclusive. Once all operations have been performed, return the maximum value in the array.

Example

$n = 10$

$queries = [[1, 5, 3], [4, 8, 7], [6, 9, 1]$

Queries are interpreted as follows:

```
    a b k
    1 5 3
    4 8 7
    6 9 1
```

Add the values of k between the indices a and b inclusive:

```
index->  1 2 3  4  5 6 7 8 9 10
        [0,0,0, 0, 0,0,0,0,0, 0]
        [3,3,3, 3, 3,0,0,0,0, 0]
        [3,3,3,10,10,7,7,7,0, 0]
        [3,3,3,10,10,8,8,8,1, 0]
```

The largest value is 10 after all operations are performed.

Function Description

Complete the function *arrayManipulation* in the editor below.

arrayManipulation has the following parameters:

- *int n* - the number of elements in the array
- *int queries[q][3]* - a two dimensional array of queries where each *queries[i]* contains three integers, *a*, *b*, and *k*.

Returns

- *int* - the maximum value in the resultant array

Input Format

The first line contains two space-separated integers n and m, the size of the array and the number of operations.
Each of the next m lines contains three space-separated integers a, b and k, the left index, right index and summand.

Constraints

- $3 \leq n \leq 10^7$

- $1 \le m \le 2 * 10^5$
- $1 \le a \le b \le n$
- $0 \le k \le 10^9$

Sample Input

```
5 3
1 2 100
2 5 100
3 4 100
```

Sample Output

```
200
```

Explanation

After the first update the list is `100 100 0 0 0`.
After the second update list is `100 200 100 100 100`.
After the third update list is `100 200 200 200 100`.

The maximum value is 200.

Contest ends in 2 hours

Submissions: 160
Max Score: 40
Difficulty: Hard

Rate This Challenge:
☆☆☆☆☆

More

C++

```
#include <bits/stdc++.h>

using namespace std;

string ltrim(const string &);
string rtrim(const string &);
vector<string> split(const string &);

/*
 * Complete the 'arrayManipulation' function below.
 *
 * The function is expected to return a LONG_INTEGER.
 * The function accepts following parameters:
 *  1. INTEGER n
 *  2. 2D_INTEGER_ARRAY queries
 */

long arrayManipulation(int n, vector<vector<int>> queries) {

}

int main()
{
    ofstream fout(getenv("OUTPUT_PATH"));

    string first_multiple_input_temp;
    getline(cin, first_multiple_input_temp);

    vector<string> first_multiple_input = split(rtrim(first_multiple_input_temp));

    int n = stoi(first_multiple_input[0]);

    int m = stoi(first_multiple_input[1]);

    vector<vector<int>> queries(m);

    for (int i = 0; i < m; i++) {
        queries[i].resize(3);

        string queries_row_temp_temp;
        getline(cin, queries_row_temp_temp);

        vector<string> queries_row_temp = split(rtrim(queries_row_temp_temp));

        for (int j = 0; j < 3; j++) {
            int queries_row_item = stoi(queries_row_temp[j]);

            queries[i][j] = queries_row_item;
        }
    }

    long result = arrayManipulation(n, queries);

    fout << result << "\n";

    fout.close();

    return 0;
}

string ltrim(const string &str) {
    string s(str);

    s.erase(
        s.begin(),
        find_if(s.begin(), s.end(), not1(ptr_fun<int, int>(isspace)))
    );

    return s;
}

string rtrim(const string &str) {
    string s(str);

    s.erase(
        find_if(s.rbegin(), s.rend(), not1(ptr_fun<int, int>(isspace))).base(),
        s.end()
    );

    return s;
}

vector<string> split(const string &str) {
    vector<string> tokens;

    string::size_type start = 0;
    string::size_type end = 0;

    while ((end = str.find(" ", start)) != string::npos) {
        tokens.push_back(str.substr(start, end - start));

        start = end + 1;
    }

    tokens.push_back(str.substr(start));

    return tokens;
}

```

Line: 1 Col: 1

Upload Code as File ☐ Test against custom input Run Code Submit Code

Interview Prep | Blog | Scoring | Environment | FAQ | About Us | Support | Careers | Terms Of Service | Privacy Policy |