Problem Solving (DS18)

هذا البرنامج التدريبي مُصاغ بعناية لتمكين المتدربين من تطوير قدراتهم الفكرية على غرار المبرمجين المحترفين، والتعاون بكفاءة ضمن فريق محترف في شركة "جمال تك" أو أي مؤسسة متعددة الجنسيات أخرى. نظرًا لأهمية اللغة الإنجليزية في بيئة العمل العالمية، يتم تقديم المحتوى التدريبي بالإنجليزية. لا يشترط إتقان اللغة بشكل كامل، لكن من الضروري امتلاك القدرة الكافية لفهم المتطلبات وتنفيذها بشكل فعّال. يُمكن للمتدربين استخدام مترجم جوجل أو الاستعانة بـ "شات جي بي تي" للتغلب على أية عقبات لغوية، المهم هو الفهم الدقيق للمطلوب وتحقيقه بنجاح.

لتعظيم الاستفادة من التدريب، يُنصح بمحاولة حل التمارين بشكل مستقل لمدة ساعة واحدة على الأقل قبل الرجوع إلى الحل المرفق في نهاية الملف.

قد يتضمن الحل كودًا برمجيًا غير مفسر بعد، والغرض من ذلك هو تشجيعك على محاولة فهم الأكواد البرمجية الجديدة التي لم تتعرض لها من قبل. هذه المهارة ضرورية في سوق العمل، حيث تتطور لغات البرمجة باستمرار ويظهر كل يوم لغات جديدة. ستواجه دائمًا أكوادًا لم تدرسها من قبل، ومن المهم أن تكون قادرًا على فهمها بنفسك دون الحاجة إلى دراسة مسبقة. يمكنك الاستعانة بمحرك البحث جوجل، أو استخدام ChatGPT، أو حتى اللجوء لأصدقائك للمساعدة. الهدف الأساسي هو أن تصل إلى فهم معنى كل كود بأي طريقة ممكنة لتتمكن من إيجاد موقعك في سوق العمل.

إن وجود كود برمجي غير مفسر يشكل تحديًا يتوجب عليك إيجاد حل له. هذا النوع من التدريبات يعد جزءًا أساسيًا من تدريبات 'Problem Solving'، التي تهدف إلى تمكينك من أداء عملك بفاعلية بغض النظر عن التحديات والعقبات. هذه القدرة على حل المشكلات هي ما يتمتع به العاملون في 'جمال تك'، ومن الضروري أن تطور في نفسك هذه المهارة لتصبح عضوًا فعالًا في فريق عمل 'جمال تك'.

Gammal Tech's Circular Challenge

Background

Gammal Tech, a pioneer in software development, is on a mission to save the planet. Known for its innovative approaches and advanced technologies, Gammal Tech has developed a unique data structure to model and solve complex environmental problems. Your task is to work with this structure to achieve a sustainable solution.

Problem Statement

Gammal Tech has devised a circular linked list to represent a series of environmental checkpoints around the globe. Each node in the list contains two values: the checkpoint ID (a unique integer) and the environmental health score (a non-negative integer). Due to recent environmental changes, Gammal Tech needs to update the health scores.

Your task is to write a program that can process queries to update and report the health scores of these checkpoints. The operations you need to implement are:

1. Update: Increase the health score of a checkpoint by a given value.
2. Report: Calculate the sum of health scores in the circular list.

Constraints

- 1 ≤ Number of checkpoints ≤ 10^5
- 0 ≤ Health score ≤ 10^6
- 1 ≤ Number of queries ≤ 10^4
- Checkpoint IDs are unique and sequentially assigned starting from 1

Input Format

- The first line contains two integers, N and Q, denoting the number of checkpoints and queries, respectively.
- The next line contains N integers, the initial health scores of the checkpoints in a circular order.
- The following Q lines describe the queries. Each query is either:
  - `1 x v` (to update checkpoint x by increasing its health score by v)
  - `2` (to report the sum of health scores)

Output Format

For each report query, print a single line containing the sum of health scores after considering all the updates so far.

Sample Input

```
5 3
10 20 30 40 50
1 3 10
1 4 20
2
```

Sample Output

```
150
```

Explanation

Initially, the sum of health scores is $10+20+30+40+50=150$. After the first update, checkpoint 3's score increases to $30+10=40$, and after the second update, checkpoint 4's score increases to $40+20=60$. However, the output is for the report query, which is before these updates are applied.

لتعظيم الاستفادة من التدريب، يُنصح بمحاولة حل التمرين بشكل مستقل لمدة ساعة واحدة على الأقل قبل الرجوع إلى الحل المرفق

C++ Solution:

```cpp
#include <iostream>
#include <vector>
using namespace std;

int main() {
    int N, Q;
    cin >> N >> Q;

    vector<int> healthScores(N);
    for (int i = 0; i < N; ++i) {
        cin >> healthScores[i];
    }

    while (Q--) {
        int type;
        cin >> type;
        if (type == 1) {
            int x, v;
            cin >> x >> v;
            healthScores[x - 1] += v;
        } else if (type == 2) {
            long long sum = 0;
            for (int score : healthScores) {
                sum += score;
            }
            cout << sum << endl;
        }
    }

    return 0;
}
```