Sparse Vector

(1 sec, 512mb)

Sparse Vector เป็นโครงสร้างข้อมูลแบบหนึ่ง ซึ่งมีการใช้ในลักษณะเดียวกับ Vector กล่าวคือเราสามารถอ้างถึงช่องต่าง ๆ ได้โดยใช้หมายเลข 0, 1, 2, ... อย่างไรก็ตามในการใช้งาน Sparse Vector นั้น จะมีแค่บางช่องที่มีข้อมูล โดยช่องส่วนใหญ่จะถือว่าไม่มีข้อมูลอยู่ภายใน ตัวอย่างเช่น เราอาจจะมี Sparse Vector v ที่เก็บข้อมูลประเภท int ที่เก็บข้อมูลเพียงต่อไปนี้

v[2] = 10, v[5] = 30, v[10] = 7, v[11] = 8, v[20] = 9

โดยจะถือว่าช่อง v[0], v[1], v[3], v[4], v[6], ... และช่องอื่น ๆ อีกหลาย ๆ ช่องที่ไม่ปรากฏข้างบนนี้ไม่มีข้อมูลอยู่ เราสามารถใช้ std::map<int,int> ในการเก็บ sparse vector นี้ได้ โดยให้ map นี้เก็บเฉพาะช่องที่มีข้อมูล

อย่างไรก็ตาม ในการใช้งาน sparse vector นั้น บางครั้งเราต้องทำการ insert ข้อมูลในลักษณะเดียวกับ vector กล่าวคือ insert โดยระบุตำแหน่งที่ต้องการ insert และต้องเลื่อนช่องใด ๆ ที่อยู่หลังจากช่องดังกล่าวด้วย

ตัวอย่างเช่น จาก sparse vector ข้างต้นนี้ หากเราทำการ insert ณ ช่องหมายเลข 5 ด้วยค่า 999 จะทำให้ sparse vector เรามีค่าเป็น

v[2] = 10, v[5] = 999, v[6] = 30, v[11] = 7, v[12] = 8, v[21] = 9

ให้สังเกตว่าช่องหมายเลข 5, 10, 11, 20 ถูกเลื่อนไปอยู่ช่องหมายเลข 6, 11, 12, 21 และหลังจากนั้น หากเราทำการ insert ณ ช่องหมายเลข 15 ด้วยค่า 444 จะทำให้มีค่าเป็น

v[2] = 10, v[5] = 999, v[6] = 30, v[11] = 7, v[12] = 8, v[15] = 444, v[22] = 9

จงเขียนฟังก์ชัน insert_into_sv(map<int,int> &v, int pos, int value) ซึ่งทำการ insert ข้อมูล value เข้าไป ณ ตำแหน่ง pos ของ sparse vector v

ข้อบังคับ

ในโจทย์ข้อนี้จะมี code เริ่มต้นมาให้แล้ว (แสดงอยู่ด้านล่างของโจทย์) ให้นิสิตเขียนโปรแกรมเพิ่มเติมลงไปในฟังก์ชัน insert_into_sv เท่านั้นโดยห้ามแก้ไขส่วนอื่น ๆ นอกจากนี้ ในฟังก์ชัน insert_into_sv นั้น ห้ามเรียกฟังก์ชันใด ๆ ที่มีการอ่านเขียนข้อมูลจากคีย์บอร์ดหรือจอภาพโดยเด็ดขาด (เช่น ห้ามเรียกใช้ cin, cout, scanf, printf, ฯลฯ) และห้ามสร้างตัวแปรแบบ static (ถ้าไม่รู้จักว่า static คืออะไร ก็ไม่ต้องกังวล) grader จะไม่ทำการตรวจสอบในเรื่องนี้ระหว่างการสอบ แต่จะมีการตรวจสอบอีกทีในภายหลัง หากเรียกใช้จะได้ 0 คะแนนทันที

คำอธิบายฟังก์ชัน main()

main จะอ่านข้อมูลมาสองบรรทัด ตามรูปแบบนี้

- บรรทัดแรกประกอบด้วยจำนวนเต็ม n และ main จะสร้าง map<int,int> v ที่ไม่มีข้อมูลใด ๆ ขึ้นมา
- หลังจากนั้นอีก n บรรทัดจะเป็นข้อมูลการเรียก insert_into_sv บรรทัดละ 1 ข้อมูล โดยที่แต่ละบรรทัดประกอบด้วยจำนวนเต็มสองตัว คือ pos และ value

  หลังจากนั้น main จะทำการพิมพ์ค่าใน v ออกมาทางหน้าจอ

ชุดข้อมูลทดสอบ

- 20% n <= 20
- 40% n <= 2000
- 40% n <= 150000

คำแนะนำ

โจทย์ข้อนี้ อาจจะ จำเป็นต้องใส่ข้อมูลหลาย ๆ ตัวลงใน map<int,int> การเรียก insert ของ map (หรือการใช้ [] ของ map) นั้นจะใช้เวลา O(log N) อย่างไรก็ตาม std::map นั้นมีฟังก์ชันการ insert อีกรูปแบบคือ iterator insert(iterator position, const value_type& val); โดยการเรียกฟังก์ชันนี้ เราจะต้องให้ position ซึ่งเป็นiterator ของ map ของเราที่ชี้ไปยังตัวที่ควรจะอยู่ถัดจากข้อมูลที่เรากำลัง insert เข้าไป

หากเราให้ position ที่ถูกต้องตามเงื่อนไขดังกล่าวแล้ว การเรียก insert ติด ๆ กัน K ครั้ง นั้นจะใช้เวลาเป็น O(n) แทนที่จะเป็น O(n log n) แต่ถ้าหากเราให้ position ที่ไม่ตรงตามเงื่อนไขไป การ insert ของ map นั้นจะทำงานได้ถูกต้อง แต่ใช้เวลาตามปรกติแทน ไม่ได้เร็วขึ้นอย่างไร

ขอให้ลองพิจารณาการใช้งานฟังก์ชัน std::map::insert ในรูปแบบนี้ด้วย

โค้ดเริ่มต้น

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

void insert_into_sv(map<int,int> &v, int pos, int value) {
  //your code here
}

int main() {
  ios_base::sync_with_stdio(false);cin.tie(0);
  int n;
  map<int,int> v;

  cin >> n;
  for (int i = 0;i < n;i++) {
    int a,b;
    cin >> a >> b;
    insert_into_sv(v,a,b);
  }

  cout << v.size() << "\n";
  for (auto &x : v) {
    cout << x.first << ": " << x.second << "\n";
  }
}
```