

```

#include<iostream>
#include<queue>
using namespace std;
struct data
{ string s; double b; int r; };
struct cmp:
public binary_function<data, data, bool>
{ bool operator()(data x, data y)
  {if(x.b < y.b) return true;
   if(x.b==y.b) if(x.r > y.r) return true;
   return false;
  }
};
priority_queue<data, vector<data>, cmp> Q;
int main()
{int c=0;
 int n; cin >> n;
 for(int i=1;i<=n;i++)
 { int m; cin >> m;
  for(int j=1;j<=m;j++)
  { data d;
   cin >> d.s >> d.b;
   d.r=c; c++;
   Q.push(d);
  }
  int k; cin >> k;
  for(int j=1; j<=k; j++) Q.pop();
 }
 cout << (Q.top()).s << endl;
}

```

Група С

Задача С1. РЕДИЦИ, автор Емил Келеведжиев

Да разгледаме редицата (3,1,4,2,4). Тя може да бъде разбита на две подредици – (3,4) и (1,2,4), които са строго растящи, т.е. елементите им са сортирани в строго нарастващ ред. *Разбиване* означава, че всяка от двете подредици трябва има поне по един елемент, обединението на елементите на двете подредици трябва да се състои от елементите на дадената редица и никой елемент от дадената редица не трябва едновременно да е елемент и на двете подредици. Забележете, че за редицата (4,8,1,5,3) такова разбиване не може да се направи. Напишете програма **SEQ**, която “познава” дали дадена редица от цели положителни числа може да бъде разбита на две строго растящи подредици.

Вход

На първия ред на стандартния вход е зададен броят N на редиците, които програмата трябва да провери, $N \leq 10$. Всеки от следващите N реда дефинира по една редица. Редът започва с броя M на елементите в съответната редица, $2 \leq M \leq 100\,000$. В същия ред, след един интервал, са зададени елементите на редицата, разделени един от друг с интервали. Всички елементи са цели положителни числа, не надминаващи 1 милиард.

Изход

На стандартния изход програмата трябва да изведе низ с дължина N , съставен от нули и единици (без разделящи интервали), с резултатите от направените проверки. За поредната редица низът трябва да съдържа цифрата 1, ако е възможно тази редица да се разбие на две строго растящи подредици, или цифрата 0, ако такова разбиване не е възможно.

ПРИМЕР

Вход	Изход
2	10
5 3 1 4 2 4	
5 4 8 1 5 3	

Решение

Наивната идея да образуваме всички възможни разбивания на редицата в две подредици и да проверяваме, дали получените подредици са строго растящи, ще доведе до алгоритъм с експоненциално нарастване на броя на стъпките, при увеличаване размерите на задачата. Алгоритъм с линейна сложност се получава по следния начин. Да допуснем, че успешно сме разбили началото (съставено от първите I елемента) на редицата и a и b са последните (максималните) елементи на двете подредици. За началото, съставено от първия елемент, такова разбиване е възможно и единствено – в едната подредица е първият елемент, а втората е празна. Максималният измежду a и b да означим с max . Да разгледаме $(I+1)$ -вия елемент c . Ако $c > max$, трябва да го причислим към подредицата на max . Ако $c < max$ – опитваме да го причислим към другата подредица. Ако това е възможно – получили сме разбиване на първите $I+1$ елемента. Ако с добавянето му там строгата монотонност на подредицата се нарушава, ясно е, че задачата няма решение.