Olympic pizza

Pizza is a widely-known traditional Italian food, consisting of oven-baked, flat bread covered with tomato sauce, melted mozzarella cheese and many other optional toppings. There are so many kinds of pizzas according to their shape (circular, rectangular or square), their thickness (from few millimeters to a couple of centimeters) and, most importantly, their toppings (cheese, ham, salami, sausage, fries, mushrooms, olives, etc., and any combination of them). That's why several pizza places (*pizzerias*) can serve such a large variety of pizzas!

Olympic pizza

Our crowded pizzeria at IOI needs your help to manage its orders and make its famous pizza. Its secret is to use very fresh ingredients and it needs frequent deliveries of new ingredients to achieve this. Because of frequent traffic jams in Italy, many deliveries arrive late, causing some ingredients to be unavailable at some times. When an ingredient is not available, the pizzeria cannot any make pizza that requires that ingredient.

Your task is to write a program that will receive notifications of new pizza orders and new ingredient deliveries. Then, it has to tell the pizza maker when she can bake a pizza: this has to be done as soon as all the needed ingredients are available.

Statement

Your task is to write a program that, receiving orders and deliveries, tells the pizza maker when she can bake pizzas as described above. Specifically, there are exactly 8 ingredients for topping a pizza and they are identified by the integers ranging from 0 to 7. The pizza orders are also numbered consecutively starting from 0 (which represents the first pizza order). Your program has to implement the following routines.

- `Init`() — it is called exactly once at the beginning of the execution (before any other routines) to indicate that the pizzeria can start receiving orders or deliveries.
- `Order`(N, A) — given a positive integer $N \leq 8$ and an array A of N integers (ranging from 0 to 7, with no two equal elements), it notifies your program that a pizza has been ordered and that it requires the N ingredients described by the elements of A (in arbitrary order).
- `Delivery`(I) — given an integer I between 0 and 7, it notifies your program that a delivery of *one portion* of ingredient I has been received, so it can be used for *one* pizza.

Your program has to call the following routine provided by the system to tell the pizza maker to bake a pizza:

- `Bake`(K) — given an integer $K \geq 0$, it indicates that the pizza for order K has to be baked.

You must call `Bake` as soon as its ingredients are ready: if there are more pizzas that can be baked, the one ordered first must be chosen. Some of the ordered pizzas might never be made because of the lack of some ingredients.

Example

In the following example, the left column reports the calls to your routines, while the right column reports the calls to the system routine that your program should make.

Your routines	System routine
`Init`()	
`Delivery`(1)	
`Delivery`(1)	
`Delivery`(1)	
`Delivery`(2)	
`Delivery`(2)	
`Order`(3, [1, 2, 3])	
`Delivery`(4)	
`Delivery`(4)	
`Order`(3, [1, 2, 4])	`Bake`(1)
`Delivery`(3)	`Bake`(0)
`Order`(4, [1, 2, 3, 4])	
`Delivery`(2)	

Subtask 1 [25 points]

The routines `Order` and `Delivery` will be called at most 100 times in total.

Subtask 2 [25 points]

The routines `Order` and `Delivery` will be called at most 5 000 times in total.

Subtask 3 [20 points]

The routines `Order` and `Delivery` will be called at most 100 000 times in total. Also, all delivery calls will happen before any order call.

Subtask 4 [30 points]

The routines `Order` and `Delivery` will be called at most 100 000 times in total.

Implementation details

You have to submit exactly one file, named `pizza.c`, `pizza.cpp` or `pizza.pas`, which implements the routines described above using the following signature.

C/C++ programs

```
void Init();
void Order(int N, int *A);
void Delivery(int I);
```

The `Bake` function will have the following signature:

```
void Bake(int K);
```

Pascal programs

```
procedure Init();
procedure Order(N : LongInt; var A : array of LongInt);
procedure Delivery(I : LongInt);
```

The `Bake` procedure will have the following signature:

```
procedure Bake(K : LongInt);
```

These routines must behave as described above. Of course you are free to implement other routines for their internal use. Your submissions must not interact in any way with standard input or standard output, nor with any other file.

Run-time limits

- Time limit: 1 second.
- Memory limit: 256 MiB.