IOI '14 P5 - Friend

Time limit: 1.0s   **Memory limit:** 256M

We build a social network from n people numbered $0, \ldots, n - 1$. Some pairs of people in the network will be friends. If person x becomes a friend of person y, then person y also becomes a friend of person x.

The people are added to the network in n stages, which are also numbered from 0 to $n - 1$. Person i is added in stage i. In stage 0, person 0 is added as the only person of the network. In each of the next $n - 1$ stages, a person is added to the network by a *host*, who may be any person already in the network. At stage i ($0 < i < n$), the host for that stage can add the incoming person i into the network by one of the following three protocols:

- `IAmYourFriend` makes person i a friend of the host only.
- `MyFriendsAreYourFriends` makes person i a friend of *each* person, who is a friend of the host at this moment. Note that this protocol does *not* make person a friend of the host.
- `WeAreYourFriends` makes person i a friend of the host, and also a friend of *each* person, who is a friend of the host at this moment.

After we build the network we would like to pick a *sample* for a survey, that is, choose a group of people from the network. Since friends usually have similar interests, the sample should not include any pair of people who are friends with each other. Each person has a *survey confidence*, expressed as a positive integer, and we would like to find a sample with the maximum total confidence.

Example

stage	host	protocol	friend relations added
1	0	`IAmYourFriend`	(1, 0)
2	0	`MyFriendsAreYourFriends`	(2, 1)
3	1	`WeAreYourFriends`	(3, 1), (3, 0), (3, 2)
4	2	`MyFriendsAreYourFriends`	(4, 1), (4, 3)
5	0	`IAmYourFriend`	(5, 0)

Initially the network contains only person 0. The host of stage 1 (person 0) invites the new person 1 through the `IAmYourFriend` protocol, hence they become friends. The host of stage 2 (person 0 again) invites person 2 by `MyFriendsAreYourFriends`, which makes person 1 (the only friend of the host) the only friend of person 2. The host of stage 3 (person 1) adds person 3 through `WeAreYourFriends`, which makes person 3 a friend of person 1 (the host) and people 0 and 2 (the friends of the host). Stages 4 and 5 are also shown in the table above. The final network is shown in the following figure, in which the numbers inside the circles show the labels of people, and the numbers next to the circles show the survey confidence. The sample consisting of people 3 and 5 has total survey confidence equal to $20 + 15 = 35$, which is the maximum possible total confidence.
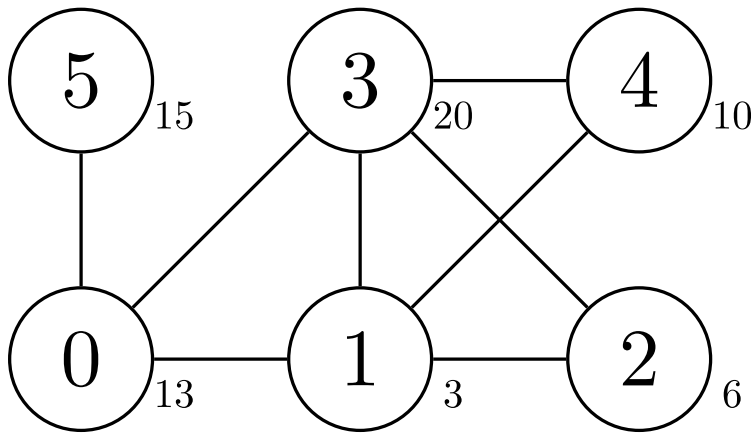
Task

Given the description of each stage and the confidence value of each person, find a sample with the maximum total confidence. You only need to implement the function `findSample`.

- `findSample(n, confidence, host, protocol)`
  - `n`: the number of people.
  - `confidence`: array of length n; `confidence[i]` gives the confidence value of person i.
  - `host`: array of length n; `host[i]` gives the host of stage i.
  - `protocol`: array of length n; `protocol[i]` gives the protocol code used in stage i ($0 < i < n$): `0` for `IAmYourFriend`, `1` for `MyFriendsAreYourFriends`, and `2` for `WeAreYourFriends`.
  - Since there is no host in stage 0, `host[0]` and `protocol[0]` are undefined and should not be accessed by your program.
  - The function should return the maximum possible total confidence of a sample.

Subtasks

subtask	points	n	confidence	protocols used
1	11	$2 \le n \le 10$	$1 \le \text{confidence} \le 1\,000\,000$	All three protocols
2	8	$2 \le n \le 1000$	$1 \le \text{confidence} \le 1\,000\,000$	Only `MyFriendsAreYourFriends`
3	8	$2 \le n \le 1000$	$1 \le \text{confidence} \le 1\,000\,000$	Only `WeAreYourFriends`
4	19	$2 \le n \le 1000$	$1 \le \text{confidence} \le 1\,000\,000$	Only `IAmYourFriend`
5	23	$2 \le n \le 1000$	All confidence values are 1	Both `MyFriendsAreYourFriends` and `IAmYourFriend`
6	31	$2 \le n \le 100\,000$	$1 \le \text{confidence} \le 10\,000$	All three protocols

Implementation details

Your submission should implement the subprogram described above, using the following signatures.

C/C++ program

```c
int findSample(int n, int confidence[], int host[], int protocol[]);
```

Pascal programs

```pascal
function findSample(n: longint, confidence: array of longint, host: array of longint;
protocol: array of longint): longint;
```