Friend

We bouwen een sociaal netwerk van n personen, genummerd van 0 tot en met $n-1$. Sommige paren in dit netwerk worden vrienden. Als persoon x vrienden wordt met persoon y, dan wordt persoon y ook vrienden met persoon x.

De mensen worden aan dit netwerk toegevoegd in n fases, die zelf ook genummerd zijn van 0 tot en met $n-1$. Persoon i wordt in fase i toegevoegd. In fase 0, wordt persoon 0 als de enige persoon tot dan toe in het netwerk opgenomen. In de volgende $n-1$ fases wordt telkens één persoon aan het netwerk toegevoegd door een *gastheer*. Dat kan iedereen zijn die al deel uitmaakt van het netwerk. In fase i ($0 < i < n$), kan de gastheer van die fase de nieuweling, persoon i het netwerk binnenhalen door gebruik te maken van één van de volgende drie protocollen:

- *IAmYourFriend* zorgt ervoor dat persoon i uitsluitend vrienden wordt met de gastheer.
- *MyFriendsAreYourFriends* zorgt ervoor dat persoon i vrienden wordt met *iedere* vriend (op dat moment) van de gastheer. Let op: In dit protocol wordt persoon i *geen* vriend van de gastheer.
- *WeAreYourFriends* zorgt ervoor dat persoon i vrienden wordt met de gastheer, en ook een vriend wordt van *iedere* vriend (op dat moment) van de gastheer.

Als het netwerk is gebouwd, gaan we op zoek naar een *steekproef* voor een onderzoek. Daarvoor wordt er een aantal mensen uit het netwerk gekozen. Omdat vrienden vaak een gemeenschappelijke belangstelling hebben, mogen in de steekproef geen mensen voorkomen die vrienden zijn.
Iedere persoon in het netwerk heeft een zekere *betrouwbaarheid* (Engels: confidence), uitgedrukt als een positieve integer. We zijn op zoek naar een steekproef met maximale totale betrouwbaarheid.

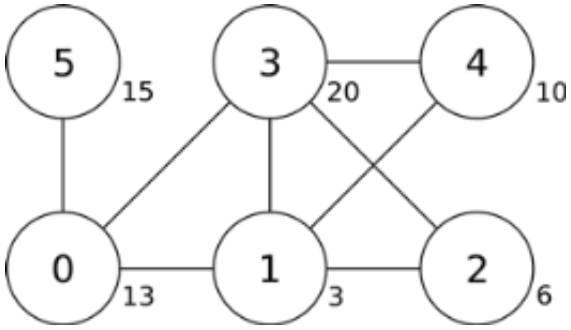
Voorbeeld

fase	gastheer	protocol	nieuwe vrienden
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)

In eerste instantie bestaat het netwerk alleen maar uit persoon 0.
De gastheer in fase 1 (persoon 0) nodigt persoon 1, met het protocol IAmYourFriend en zo worden ze vrienden.
De gastheer in fase 2 (weer persoon 0) nodigt persoon 2 uit met MyFriendsAreYourFriends.
Daardoor wordt persoon 1 (de enige vriend van persoon 0) nu de enige vriend van persoon 2.
De gastheer van fase 3 (persoon 1) nodigt persoon 3 uit met WeAreYourFriends. Nu raakt persoon 3

bevriend met persoon 1 (de gastheer) en met de vrienden van de gastheer, persoon 0 en persoon 2. In de tabel hierboven staan ook fases 4 en 5.

Het netwerk dat uiteindelijk ontstaat zie je hieronder. Het getal in elke cirkel geeft het nummer van de persoon aan; de getallen naast elke cirkel de betrouwbaarheid. De steekproef met personen 3 en 5 geeft de grootste totale betrouwbaarheid met een totaal van 20 + 15 = 35.



Opdracht

Je krijgt de beschrijving van de fases en de betrouwbaarheid van elke persoon. Zoek op basis hiervan een steekproef met maximale betrouwbaarheid. Je hoeft alleen de functie `findSample` te implementeren.

- `findSample(n, confidence, host, protocol)`
  - `n`: het aantal personen.
  - `confidence`: array van lengte n; `confidence[i]` geeft de betrouwbaarheid van persoon i.
  - `host`: array van lengte n; `host[i]` geeft de gastheer van fase i.
  - `protocol`: array van lengte n; `protocol[i]` geeft de code voor het protocol voor fase i ($0 < i < n$): 0 voor IAmYourFriend, 1 voor MyFriendsAreYourFriends, en 2 voor WeAreYourFriends.
  - Omdat er voor fase 0 geen gastheer is, zijn `host[0]` en `protocol[0]` niet gedefiniëerd en mogen ze niet gebruikt worden door je programma.
  - De functie moet als resultaat de maximaal mogelijke betrouwbaarheid van een steekproef opleveren.

Subtasks

In sommige subtasks wordt maar een deel van de protocollen gebruikt, zoals je kunt zien in onderstaande tabel.

subtask	punten	n	betrouwbaarheid	gebruikte protocollen
1	11	$2 \leq n \leq 10$	$1 \leq \text{confidence} \leq 1.000.000$	Alle drie protocollen
2	8	$2 \leq n \leq 1.000$	$1 \leq \text{confidence} \leq 1.000.000$	Alleen MyFriendsAreYourFriends
3	8	$2 \leq n \leq 1.000$	$1 \leq \text{confidence} \leq 1.000.000$	Alleen WeAreYourFriends
4	19	$2 \leq n \leq 1.000$	$1 \leq \text{confidence} \leq 1.000.000$	Alleen IAmYourFriend

subtask	punten	n	betrouwbaarheid	gebruikte protocollen
5	23	$2 \leq n \leq 1.000$	Alle betrouwbaarheden zijn 1	Zowel MyFriendsAreYourFriends en IAmYourFriend
6	31	$2 \leq n \leq 100.000$	$1 \leq \text{confidence} \leq 10.000$	All drie protocollen

Implementatie details

Je moet één bestand inzenden met de naam `friend.c`, `friend.cpp` of `friend.pas`. Dit bestand moet het subprogramma dat hierboven beschreven is implementeren, met de volgende signatures. Als je C/C++ gebruikt, moet je ook een header bestand `friend.h` includen.

C/C++ programma

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

Pascal programma

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

Voorbeeld grader

De voorbeeld grader leest de invoer in het volgende formaat:

- regel 1: `n`
- regel 2: `confidence[0]`, ..., `confidence[n-1]`
- regel 3: `host[1]`, `protocol[1]`, `host[2]`, `protocol[2]`, ..., `host[n-1]`, `protocol[n-1]`

De voorbeeld grader drukt de waarde van `findSample` af.