Przyjaciel (Friend)

Budujemy sieć społecznościową złożoną z n osób ponumerowanych 0, …, $n-1$. Pewne pary osób w sieci stają się przyjaciółmi. Jeśli osoba x staje się przejacielem osoby y, to osoba y również staje się przyjacielem osoby x.

Osoby są dodawane do sieci w n fazach, ponumerowanych od 0 do $n-1$. Osoba i jest dodawana do sieci w fazie i. W fazie 0 w sieci jest umieszczana tylko osoba 0. W każdej z następnych $n-1$ faz nowa osoba jest zapraszana do sieci przez *gospodarza*, którym może być dowolna osoba znajdująca się już w sieci. Osoba będąca gospodarzem w fazie i $(0 < i < n)$ może dodać do sieci osobę i zgodnie z jednym z trzech następujących protokołów:

- *IAmYourFriend* czyni osobę i przyjacielem gospodarza (i nikogo więcej).
- *MyFriendsAreYourFriends* czyni osobę i przyjacielem *każdej* osoby będącej przyjacielem aktualnego gospodarza. Zwróć uwagę, że w tym protokole osoba i *nie* zostaje przyjacielem gospodarza.
- *WeAreYourFriends* czyni osobę i przyjacielem aktualnego gospodarza oraz *wszystkich* osób będących jego aktualnymi przyjaciółmi.

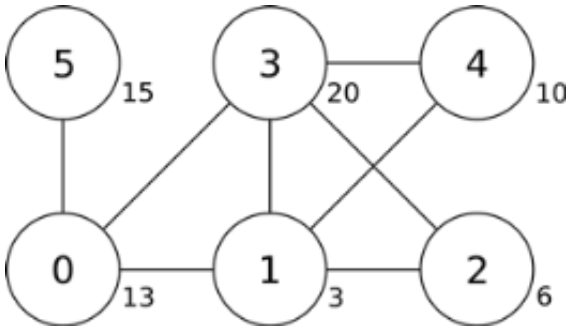
Po zbudowaniu sieci chcielibyśmy wybrać *próbkę* osób z sieci do badania jej własności. Ponieważ przyjaciele mają zazwyczaj podobne zainteresowania, w próbce nie może znaleźć się żadna para przyjaciół. Z każdą osobą związana jest jej *adekwatność* dla badań, wyrażona dodatnią liczbą całkowitą. Naszym celem jest znalezienie próbki o największej sumarycznej adekwatności.

Przykład

faza	gospodarz	protokół	dodane pary przyjaciół
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)

Początkowo w sieci jest tylko osoba 0. Gospodarz fazy 1 (osoba 0) zaprasza nową osobę 1 przy użyciu protokołu IAmYourFriend, co oznacza, że osoby 0 i 1 stają się przyjaciółmi. Gospodarz fazy 2 (ponownie osoba 0) zaprasza osobę 2 za pomocą MyFriendsAreYourFriends, w wyniku czego osoba 1 (jedyny przyjaciel gospodarza) staje się jedynym przyjacielem osoby 2. Gospodarz fazy 3 (osoba 1) dodaje do sieci osobę 3, wykonując WeAreYourFriends, czy czyni osobę 3 przyjacielem osoby 1 (gospodarza) oraz osób 0 i 2 (przyjaciół gospodarza). W tabeli powyżej zaprezentowano też fazy 4 i 5. Końcowa sieć jest przedstawiona na rysunku, na którym liczby w kółkach to etykiety osób, natomiast liczby obok kółek to ich adekwatności. Próbka składająca się z osób 3 i 5 ma sumaryczną

adekwatność równą 20+15 = 35, która jest największą możliwą sumaryczną adekwatnością próbki.



Zadanie

Mając dane opisy faz oraz adekwatności poszczególnych osób, znajdź próbkę o największej sumarycznej adekwatności. Twoje zadanie polega tylko na napisaniu funkcji `findSample`.

- `findSample(n, confidence, host, protocol)`
  - `n`: liczba osób.
  - `confidence`: tablica rozmiaru n; `confidence[i]` podaje adekwatność osoby i.
  - `host`: tablica rozmiaru n; `host[i]` podaje gospodarza w fazie i.
  - `protocol`: tablica rozmiaru n; `protocol[i]` podaje kod protokołu używanego w fazie i ($0 < i < n$): 0 oznacza IAmYourFriend, 1 oznacza MyFriendsAreYourFriends, natomiast 2 oznacza WeAreYourFriends.
  - Ponieważ faza 0 nie ma gospodarza, `host[0]` oraz `protocol[0]` nie są określone i nie powinny być wykorzystywane w Twoim programie.
  - Wynikiem funkcji powinna być największa możliwa sumaryczna adekwatność próbki.

Podzadania

W niektórych podzadaniach wykorzystuje się tylko część z protokołów, zgodnie z tabelą poniżej.

podzad.	liczba pkt	n	adekwatość	używane protokoły
1	11	$2 \leq n \leq 10$	$1 \leq \text{adekwatność} \leq 1,000,000$	Wszystkie trzy protokoły
2	8	$2 \leq n \leq 1,000$	$1 \leq \text{adekwatność} \leq 1,000,000$	Tylko MyFriendsAreYourFriends
3	8	$2 \leq n \leq 1,000$	$1 \leq \text{adekwatność} \leq 1,000,000$	Tylko WeAreYourFriends
4	19	$2 \leq n \leq 1,000$	$1 \leq \text{adekwatność} \leq 1,000,000$	Tylko IAmYourFriend
5	23	$2 \leq n \leq 1,000$	Wszystkie adekwatności są równe 1	Protokoły MyFriendsAreYourFriends i IAmYourFriend
6	31	$2 \leq n \leq 100,000$	$1 \leq \text{adekwatność} \leq 10,000$	Wszystkie trzy protokoły

Implementacja

Powinieneś zgłosić dokładnie jeden plik o nazwie `friend.c`, `friend.cpp` lub `friend.pas`. W pliku powinna znaleźć się implementacja funkcji opisanej powyżej, o następującej sygnaturze. W przypadku programu w C/C++ powinieneś także załączyć (*include*) plik nagłówkowy `friend.h`.

Programy w C/C++

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

Programy w Pascalu

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

Przykładowy program sprawdzający

Przykładowy program sprawdzający wczytuje dane w następującym formacie:

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

Przykładowy program sprawdzający wypisze na wyjście wynik funkcji `findSample`.