Friend (Amigos)

Construiremos una red social de n personas numeradas $0, \ldots, n-1$. Algunos pares de personas en la red serán amigos. Si la persona x se vuelve amiga de la persona y, entonces la persona y también se vuelve amiga de la persona x.

Las personas se agregan a la red social en n etapas, cada una de ellas numerada de 0 a $n-1$. La persona i es agregada en la etapa i. En la etapa 0 la persona 0 es agregada como la única persona en la red. En cada una de las siguientes $n-1$ etapas, una persona es agregada a la red por un *anfitrión*, el cual puede ser cualquier persona que ya se encuentra en la red. En la etapa i $(0 < i < n)$, el anfitrión de esa etapa puede agregar a la persona i a la red a través de uno de los siguientes tres protocolos:

- *YoSoyTuAmigo* hace que la persona i sea solamente amiga del anfitrión.
- *MisAmigosSonTusAmigos* hace que la persona i sea amiga de *cada* persona que sea amiga del anfitrión en ese momento. Nota que este protocolo *no* hace a la persona i amiga del anfitrión.
- *NosotrosSomosTusAmigos* hace que la persona i sea amiga del anfitrión, así como de *cada* persona que sea amiga del anfitrión en ese momento.

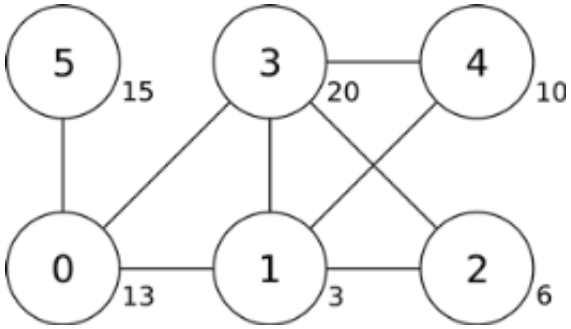
Despues de construir la red social queremos tomar una *muestra* para una encuesta, esto es, elegir un grupo de personas de la red. Como los amigos tienen generalmente gustos similares, la muestra no debe incluir ningún par de personas que sean amigas entre ellas. Cada persona tiene un nivel de *confianza* para la encuesta, expresado como un entero positivo, lo que queremos es encontrar la muestra con la confianza total máxima.

Ejemplo

etapa	anfitrión	protocolo	relaciones de amigos añadidas
1	0	YoSoyTuAmigo	(1, 0)
2	0	MisAmigosSonTusAmigos	(2, 1)
3	1	NosotrosSomosTusAmigos	(3, 1), (3, 0), (3, 2)
4	2	MisAmigosSonTusAmigos	(4, 1), (4, 3)
5	0	YoSoyTuAmigo	(5, 0)

Inicialmente la red contiene solo a la persona 0. El anfitrión de la etapa 1 (persona 0) invita a la nueva persona 1 a través del protocolo *YoSoyTuAmigo*, por lo cual se vuelven amigos. El anfitrión de la etapa 2 (persona 0 otra vez) invita a la persona 2 a través del protocolo *MisAmigosSonTusAmigos*, lo cual hace a la persona 1 (el único amigo del anfitrión) el único amigo de la persona 2. El anfitrión de la etapa 3 (persona 1) agrega a la persona 3 a través del protocolo *NosotrosSomosTusAmigos*, lo cual hace que a la persona 3 amiga de la persona 1 y de las personas 0 y 2 (Los amigos del anfitrión). Etapas 4 y 5 se muestran en la tabla de abajo. La red, al final de las 5 etapas, se muestra en la

siguiente figura, los números mostrados dentro de los circulos representan las etiquetas de las personas y los números cercanos a los circulos muestran la confianza de cada persona para la encuesta.
La muestra que se compone de las personas 3 y 5 tiene una confianza total igual a $20 + 15 = 35$, la cual es la confianza total máxima posible para una muestra dentro de esa red.



Problema

Dada la descripción de cada etapa y el valor de la confianza de cada persona, encuentra una muestra con la confianza total máxima. Debes implementar la función `findSample`.

- `findSample(n, confidence, host, protocol)`
  - `n`: la cantidad de personas.
  - `confidence`: un arreglo de longitud n; `confidence[i]` la confianza de la persona i.
  - `host`: un arreglo de longitud n; `host[i]` indica el anfitrión para la etapa i.
  - `protocol`: un arreglo de longitud n; `protocol[i]` da el codigo del protocolo usado en la etapa i $(0 < i < n)$: 0 para *YoSoyTuAmigo*, 1 para *MisAmigosSonTusAmigos*, y 2 para *NosotrosSomosTusAmigos*.
  - Al no haber anfitrión para la etapa 0, `host[0]` y `protocol[0]` no están definidos y no deben ser accedidos por tu programa.
  - La función debe devolver el valor de la confianza total máxima de la muestra.

Subproblemas

Algunos subproblemas usan sólo un subconjunto de los protocolos, como se muestra en la siguiente tabla.

subproblema	puntos	n	confianza	protocolos usados
1	11	$2 \leq n \leq 10$	$1 \leq$ confianza $\leq 1,000,000$	Los tres protocolos
2	8	$2 \leq n \leq 1,000$	$1 \leq$ confianza $\leq 1,000,000$	Sólo *MisAmigosSonTusAmigos*
3	8	$2 \leq n \leq 1,000$	$1 \leq$ confianza $\leq 1,000,000$	Sólo *NosotrosSomosTusAmigos*
4	19	$2 \leq n \leq 1,000$	$1 \leq$ confianza $\leq 1,000,000$	Sólo *YoSoyTuAmigo*
5	23	$2 \leq n \leq 1,000$	El valor de todas las confianzas es 1	Ambos *MisAmigosSonTusAmigos* y *YoSoyTuAmigo*

subproblema	puntos	n	confianza	protocolos usados
6	31	$2 \leq n \leq 100,000$	$1 \leq \text{confianza} \leq 10,000$	Los tres protocolos

Detalles de implementación

Debes enviar solamente un archivo, llamado `friend.c`, `friend.cpp` o `friend.pas`. Este archivo deberá implementar la función descrita abajo, usando los siguientes prototipos. Además deberás incluir el archivo `friend.h`en el caso de implementaciones en C/C++.

programa en C/C++

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

programa en Pascal

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

Evaluador de Ejemplo

El evaluador de ejemplo lee la entrada en el siguiente formato:

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

El evaluador de ejemplo imprime el valor devuelto por la función `findSample`.