Friend

Nosotros construiremos una red social de n personas numeradas de 0, ... , $n-1$. Algunos pares de personas en la red seran amigos. Si una persona x se convierte en un amigo de la persona y, entonces la persona y tambien se convierte en un amigo de la persona x.

Las personas se agregan a la red en n etapas, las cuales son tambien numeradas desde 0 hasta $n-1$. La persona i se adiciona en la etapa i. En la etapa 0, la persona 0 es adicionada como la unica persona de la red. En cada una de las proximas $n-1$ etapas, una persona es adicionada a la red por un *anfitrion*, quien puede ser cualquier persona que ya este en la red. En la etapa i ($0 < i < n$), el anfitrion para esa etapa puede adicionar la persona entrante i en la red por uno de los siguientes tres protocolos:

- *IamYourFriend* hace a la persona i amigo del anfitrion solmente.
- *MyFriendsAreYourFriends* hace a la persona i amigo de *cada* amigo del anfitrion. Note que este protocolo *no* hace a la persona i un amigo del anfitrion.
- *WeAreYourFriends* hace una persona i un amigo del anfitrion, y tambien amigo de *cada* amigo del anfitrion.

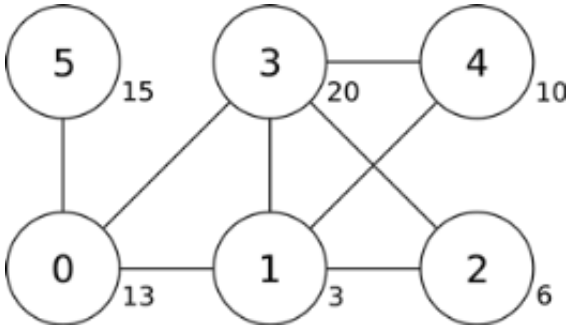
Despues que construimos la red a nosotros nos gustaria coger una *muestra* para una encuesta, o sea, elegir un grupo de la red. Ya que los amigos usualmente tienen similares intereses, en la muestra no debe incluir cualquier par de personas que sean amigos entre si. Cada persona tiene un grado de *confidencia* para la encuesta, expresado en un entero positivo, y le gustaria encontrar una red con el maximo total de confidencia.

Example

Etapa	anfitrion	protocolo	relaciones del amigo adicionada
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)

Inicialmente la red contiene solamente a la persona 0. El anfitrion de la etapa 1 (persona 0) invita la nueva persona 1 a traves del protocolo IamYourFriend, por tanto ellos se convierten en amigos. El anfitrion de la de la etapa 2 (persona 0 otra vez) invita a la persona 2 por MyFriendsAreYourFriends, lo cual hace a la persona 1 (el unico amigo del anfitrion) el unico amigo de la persona 2. El anfitrion de la etapa 3 (persona 1) adiciona la persona 3 a traves de WeAreYourFriends, lo cual hace a la persona 3 un amigo de la persona 1 (el anfitrion) y la persona 0 y 2 (los amigos del anfitrion). Las etapas 4 y 5 son tambien mostradas en la tabla de abajo. La red final aparece en la siguiente figura, en la cual los

numeros dentro de los circulos muestran las etiquetas de las personas, y los numeros proximos a los circulos muestran la confidencia para la encuesta. La muestra consiste de la persona 3 y 5 la cul tiene total de confidencia para la encuesta igual a 20 + 15 = 35, lo cual es la confidencia total maxima posible.



Tarea

Dada la descripcion de cada etapa y el valor de confianza de cada persona, encuentre una muestra confianza total maxima. Tu solamente necesitas implementar la funcion `findSample`.

- `findSample(n, confidence, host, protocol)`
  - `n`: el numero de personas.
  - `confidence`: arreglo de longitud n; `confidence[i]` contiene el valor de confidencia de la persona i.
  - `host`: arraeglo de tamano n; `host[i]` contiene el anfitrion la etapa i.
  - `protocol`: arreglo de tamno n; `protocol[i]` contiene el codigo de protocolo usdo en la etapa i ($0 < i < n$): 0 para IamYourFriend, 1 para MyFriendsAreYourFriends, y 2 para WeAreYourFriends.
  - Ya que no hay host en la etapa 0, `host[0]` y `protocol[0]` estan indindefinidos y no deben ser accedidos por su programa.
  - La funcion debe retornar el maximo total de confidencia posible de la muestra.

Subtareas

Algunas subtareas usan solamente un conjunto de protocolos, como se muestra en la siguiente tabla.

subtask	puntos	n	confidencia	protocolos usados
1	11	$2 \leq n \leq 10$	$1 \leq \text{confidence} \leq 1,000,000$	Todos los tres protocolos
2	8	$2 \leq n \leq 1,000$	$1 \leq \text{confidence} \leq 1,000,000$	Solamente MyFriendsAreYourFriends
3	8	$2 \leq n \leq 1,000$	$1 \leq \text{confidence} \leq 1,000,000$	Solamente WeAreYourFriends
4	19	$2 \leq n \leq 1,000$	$1 \leq \text{confidence} \leq 1,000,000$	Solamente IamYourFriend
5	23	$2 \leq n \leq 1,000$	Todos los valores de confidencia son 1	Both MyFriendsAreYourFriends and IamYourFriend

subtask	puntos	n	confidencia	protocolos usados
6	31	$2 \leq n \leq 100,000$	$1 \leq \text{confidence} \leq 10,000$	Todos los tres protocolos

Implementation details

Usted tiene que entregar exactamente un fichero, llamado `friend.c`, `friend.cpp` or `friend.pas`. El fichero debe implementar los subprogramas descritos arriba, usando las siguientes encabezados de funciones. Usted tambien necesita incluir u fichero encabezamiento `friend.h` para la implementacion en C/C++.

C/C++ program

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

Pascal programs

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

Sample grader

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

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

El ejemplo de grader will imprimira el valor de retorno de `findSample`.