Друзья

Создается социальная сеть, состоящая из n участников, пронумерованных 0, ... , $(n-1)$. Некоторые пары участников этой сети могут стать друзьями. Если участник x становится другом участника y, то участник y также становится другом участника x.

Участники добавляются в сеть за n этапов, которые также пронумерованы от 0 до $(n-1)$. Участник i добавляется на этапе i. На этапе 0 добавляется участник с номером 0 как единственный участник сети. На каждом из следующих $(n-1)$ этапов очередной участник добавляется в сеть *хозяином* этапа, которым может быть любой участник, уже добавленный в сеть. На этапе i ($0 < i < n$), хозяин этапа может добавить очередного участника i в сеть по одному из трех протоколов:

- *IAmYourFriend* делает участника i другом только хозяина этапа.
- *MyFriendsAreYourFriends* делает участника i другом *каждого* друга хозяина в этот момент. Заметьте, что этот протокол *не* делает участника i другом хозяина.
- *WeAreYourFriends* делает участника i другом хозяина в этот момент, а также другом *каждого* друга хозяина.

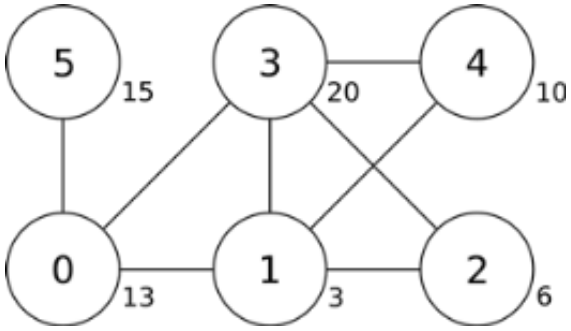
После того, как сеть создана, необходимо сделать *выборку* для опроса, то есть отобрать группу участников сети. Поскольку друзья обычно имеют общие интересы, эта выборка не должна содержать пары участников, являющихся друзьями. Каждый участник имеет некоторый *уровень доверия* в опросах, который задан положительным целым числом, и нужно сделать выборку участников с максимальным суммарным уровнем доверия.

Пример

Этап	Хозяин	Протокол	Добавленные пары друзей
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)

Вначале сеть содержит только участника с номером 0. Хозяин первого этапа (участник с номером 0) приглашает нового участника с номером 1, используя протокол IAmYourFriend, и они становятся друзьями. Хозяин второго этапа (снова участник с номером 0) приглашает второго участника, используя протокол MyFriendsAreYourFriends, который делает участника с номером 1 (единственный друг хозяина) единственным другом участника с номером 2. Хозяин третьего этапа (участник с номером 1) добавляет третьего участника, используя протокол WeAreYourFriends, что делает третьего участника другом первого участника (хозяина) и

участников с номерами 0 и 2 (они же друзья хозяина). Этапы 4 и 5 также показаны в таблице выше. Финальная сеть представлена на рисунке ниже, где число в кружке показывает номер участника, а число рядом с кружком показывает уровень доверия в опросах для этого участника. Выборка, состоящая из участников с номерами 3 и 5, имеет суммарный уровень доверия в опросах, который составляет 20 + 15 = 35, что является максимально возможным суммарным уровнем доверия.



Постановка задачи

Имея описание каждого этапа и уровень доверия в опросах каждого участника, необходимо найти выборку участников сети с максимальным суммарным уровнем доверия. Вы должны реализовать функцию `findSample`.

- `findSample(n, confidence, host, protocol)`
  - `n`: количество участников;
  - `confidence`: массив длины n; `confidence[i]` задает уровень доверия к участнику с номером i;
  - `host`: массив длины n; `host[i]` задает хозяина i-го этапа;
  - `protocol`: массив длины n; `protocol[i]` задает код протокола, используемого на i-ом этапе ($0 < i < n$): 0 — для IAmYourFriend, 1 — для MyFriendsAreYourFriends, 2 — для WeAreYourFriends;
  - поскольку на этапе 0 нет хозяина, и `host[0]` и `protocol[0]` не определены, то ваша программа не должна к ним обращаться;
  - функция должна возвращать максимально возможный суммарный уровень доверия для выборки участников.

Подзадачи

Некоторые подзадачи используют не все протоколы, как показано в таблице ниже.

Подзадача	Баллы	n	Уровень доверия (confidence)	Используемые протоколы
1	11	$2 \leq n \leq 10$	$1 \leq \text{confidence} \leq 1\,000\,000$	Все три протокола
2	8	$2 \leq n \leq 1\,000$	$1 \leq \text{confidence} \leq 1\,000\,000$	Только MyFriendsAreYourFriends
3	8	$2 \leq n \leq 1\,000$	$1 \leq \text{confidence} \leq 1\,000\,000$	Только WeAreYourFriends
4	19	$2 \leq n \leq 1\,000$	$1 \leq \text{confidence} \leq 1\,000\,000$	Только IAmYourFriend
5	23	$2 \leq n \leq 1\,000$	Все уровни доверия равны 1	Только MyFriendsAreYourFriends и IAmYourFriend
6	31	$2 \leq n \leq 100\,000$	$1 \leq \text{confidence} \leq 10\,000$	Все три протокола

Детали реализации

Вы должны послать ровно один файл, названный `friend.c`, `friend.cpp` или `friend.pas`. В этом файле должна быть реализована функция, описанная выше с указанными ниже прототипами. На языках C/C++ вы должны подключить заголовочный файл `friend.h`.

Язык C/C++

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

Язык Pascal

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

Пример проверяющего модуля

Предоставленный пример проверяющего модуля имеет следующий формат входных данных:

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

Предоставленный пример проверяющего модуля выведет значение, возвращаемое функцией `findSample`.