Задача Рамен

Вход	`stdin`
Изход	`stdout`

Има N приятели означени с $F_0, \ldots, F_{N-1}$ и N вида рамен означени с $R_0, \ldots, R_{N-1}$ в ресторанта за рамен. Всеки приятел F_i има определено предпочитание A_{ij} за рамен от вид R_j – колкото по-голямо е предпочитанието, толкова повече приятел F_i обича рамен R_j. Предпочитанията на всеки приятел са различни – т.е. $A_{ij} \neq A_{ij'}$ за $j \neq j'$. Разбира се, възможно е $A_{ij} < 0$.

Нека да предположим, че приятелите посещават ресторанта за рамен много пъти. По време на всяко посещение, няма двама приятели, които да ядат един и същ вид рамен (няма достатъчно налични от един и същ вид). Означаваме реда, в който приятелите вземат техния рамен, като $F_{\pi_0}, \ldots, F_{\pi_{N-1}}$, за някоя пермутация $\pi_0, \ldots, \pi_{N-1}$ на числата $0, \ldots, N-1$. Тогава приятел F_{π_0} ще вземе своя любим рамен (т.е. този, за който има най-голямо предпочитание), приятел F_{π_1} ще вземе своя любим рамен *с изключение на този, който е взет от* F_{π_0} и т.н. С други думи, F_{π_i} ще вземе любимия си рамен измежду тези, които не са взети от $F_{\pi_0}, \ldots, F_{\pi_{i-1}}$.

Качеството на определена пермутация π е сумата от предпочитанията на приятелите за вида рамен, който вземат. Това означава, че ако приятел F_i вземе рамен от вид R_{σ_i}, то качеството на π се получава като $\sum_{i=0}^{N-1} A_{i,\sigma(i)}$.

Вашата задача е да намерите пермутация π с максимално качество. Може да я намерите като правите експерименти с различни поръчки на рамен от приятелите (т.е. различни посещения на ресторанта). Трябва да намерите оптимална пермутация без да е необходимо да посещавате ресторанта прекалено много пъти.

Детайли по имплементацията

Трябва да напишете следните функции:

```
std::vector<int> find_order(int N);
```

Тук N е броят на приятелите. Функцията трябва да върне пермутация π на числата $0, \ldots, N-1$ с максимално качество. За тази цел може да извикате многократно следната функция (най-много 750 пъти):

```
std::vector<std::pair<int, int>> query(const std::vector<int>& order);
```

Функцията приема като параметър пермутацията π на числата $0, \ldots, N-1$ (означена с `order`). Тя връща списък с двойки $(\sigma(i), A_{i,\sigma(i)})$, които задават, че приятел F_i е поръчал рамен от вид $R_{\sigma(i)}$ при условие, че приятелите вземат своя рамен по реда, зададен от пермутацията π.

Ограничения

- $1 \leq N \leq 75$
- $|A_{ij}| \leq 2\,000\,000$
- Предвиденото от научния комитет решение се нуждае от най-много cN^k заявки за някакви константи $c, k \geq 1$. За да не претоварите клетата тестваща система, може да извикате функцията `query` най-много 750 пъти, което е повече от реалното поведение на предвиденото решение.

#	Точки	Ограничения
1	5	$N = 5$
2	15	$N = 15$
3	20	$N = 30$
4	20	$N = 45$
5	20	$N = 60$
6	20	$N = 75$

Пример

Действия на Вашата програма	Действия на комитета
	Извикване на `find_order(2)`.
`query({0, 1})`	
	`{{0, 9}, {1, 0}}`
`query({1, 0})`	
	`{{1, 5}, {0, 5}}`
`find_order(2)` връща `{1, 0}`.	

За примера има $N = 2$ приятели, със следните предпочитания $A_{i,j}$ за $N = 2$-та вида рамен:

A	0	1
0	9	5
1	5	0

В началото комитета извиква Вашата функция: `find_order(2)`. Тя пробва следните две поръчки: $\{0, 1\}$ и $\{1, 0\}$. Първата постига качество от $9 + 0 = 9$, докато втората постига качество от $5 + 5 = 10$. След това Вашата функция връща по-добрата от двете поръчки, която е пермутацията $\{1, 0\}$. Върната пермутация има максималното възможно качество, което се изисква.