Тропическа градина

Професорът по ботаника Сомхед често посещава най-голямата ботаническа градина в Тайланд, заедно с групи студенти. Ландшафтът в градината е направен с **N** фонтана (номерирани с 0, 1, 2, ..., **N**–1) и **M** пътеки. Всяка пътека свързва различна двойка от два различни фантана и по всяка пътека може да се върви в двете посоки. Всеки фонтан е свързан с поне една пътека. Сомхед с всяка група студенти може да започне разходката си от произволен фонтан.

Сомхед обича красивите тропически растения. Затова Сомхед иска да продължи от всяко кръстовище с групата студенти по най-красивата пътека, *освен* ако най-красивата пътека не е тази, по която са вървели преди да дойдат в кръстовището. В този случай те ще тръгнат по следващата по красота пътека. Само когато няма такава алтернатива, те ще се върнат обратно, използвайки отново пътеката, по която са дошли. Понеже Сомхед е професионален ботаник, той счита, че не съществуват две еднакво красиви пътеки.

Групите студенти не се интересуват много от растения, но биха искали да обядват в първокласния ресторант, намиращ се до фонтана **P**. Сомхед знае, че неговите студенти огладняват след преминаване по точно **K** пътеки (**K** може да е различно за различните групи студенти). Затова, той иска да пресметне колко са различните маршрути, които може да избере за всяка група, такива че:

- Всяка група може да тръгне от всеки фонтан.
- Последователните пътеки от маршрута се избират по описания по-горе начин.
- Всяка група трябва да завърши при фонтан номер **P**, след като е преминала по точно **K** пътеки.

Забележка: Възможно е една група да мине край фонтана P и по-рано по своя маршрут, но се иска разходката *да завърши* там.

Задача

При дадена информация за фонтаните и пътеките, вие трябва да намерите отговорите за **Q** групи от студенти, т.е. за **Q** стойности на **K**.

Напишете процедура **count_routes**(**N, M, P, R, Q, G**), която има следните параметри:

- **N** – брой на фонтаните. Фонтаните са номерирани от 0 до **N**–1.
- **M** – брой на пътеките. Пътеките са номерирани от 0 до **M**–1 и са дадени *в намаляващ* ред по красота: пътеката с номер **i** е по-красива от пътеката с номер **i**+1 за $0 \le \mathbf{i} < \mathbf{M}-1$.
- **P** – фонтанът, където се намира ресторантът.
- **R** – двумерен масив, задаващ пътеките. За $0 \le \mathbf{i} < M$, пътеката **i** свързва фонтани **R[i][0]** и **R[i][1]**. Напомняме, че всяка пътека свързва два различни фонтана и че няма две пътеки, които свързват една и съща двойка фонтани.
- **Q** – брой на групите от студенти.
- **G** – едномерен масив от цели числа, съдържащ стойностите на **K**. За $0 \le \mathbf{i} < \mathbf{Q}$, **G[i]** е броят на пътеките **K**, по които трябва да преминат студентите от **i**-тата група.

За $\mathbf{0} \le \mathbf{i} < \mathbf{Q}$ вашата процедура трябва да намери броя на възможните маршрути, всеки с точно **G[i]** пътеки, съответно при i-тата група за достигане на фонтана **P**. За всяка група **i**, вашата процедура трябва да извика процедурата **answer**(X), за да съобщи броя на маршрутите **X**. Отговорите трябва да се дават в същия ред, в който са дадени групите от студенти. Ако няма валиден маршрут, вашата процедура трябва да извика **answer**(0).

Примери

Пример 1

Разглеждаме случая, показан на Фигура 1, където **N**=6, **M=6**, **P=0**, **Q=1**, **G[0]=3** и

R=
1 2
0 1
0 3
3 4
4 5
1 5

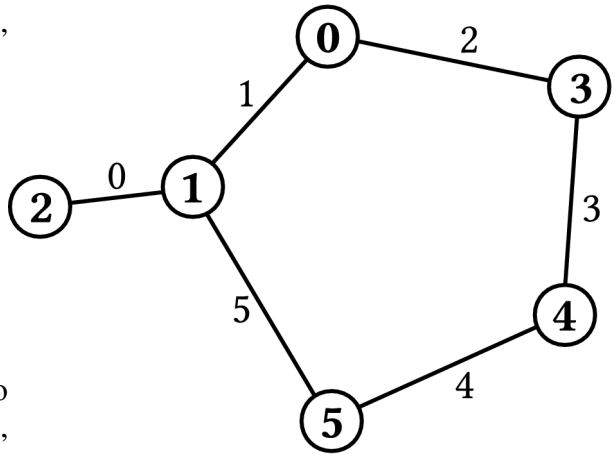

Figure 1.

Пътеките са дадени в намаляващ ред по красота. Така пътека 0 е по-красива от пътека 1, пътека 0 е най-красивата, пътека 1 е втората по красота и т.н.

Съществуват само два възможни коректни маршрута, които минават по 3 пътеки:

- $1 \rightarrow 2 \rightarrow 1 \rightarrow 0$ и
- $5 \rightarrow 4 \rightarrow 3 \rightarrow 0$.

Първият маршрут започва от фонтан 1. Най-красивата пътека от него води до фонтан 2. При фонтан 2, групата няма избор – трябва да се върне по същата пътека обратно при фонтан 1. Сега там групата ще пропусне пътека 0 и ще избере пътека 1 и така ще стигне до фонтан **P**=0. Така процедурата трябва да извика **answer**(2).

Пример 2

Разглеждаме случая, показан на Фигура 2, където **N=5**, **M=5**, **P=2**, **Q=2**, **G[0]=3, G[1]=1** и

R=
1 0
1 2
3 2
1 3
4 2

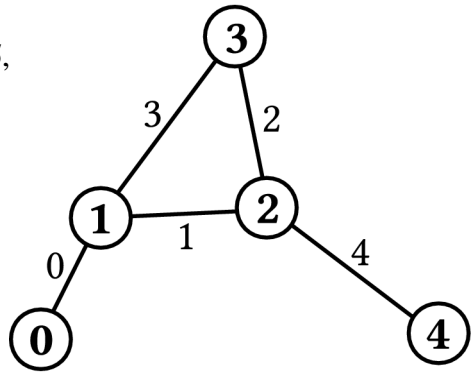

Figure 2.

За първата група студенти има само един валиден маршрут, с който се достига фонтан 2 с изминаване на 3 пътеки: $1 \rightarrow 0 \rightarrow 1 \rightarrow 2$.

За втората група студенти има два валидни маршрута, с които се достига фонтан 1 с изминаване на 1 пътека: $3 \rightarrow 2$, и $4 \rightarrow 2$.

Следователно правилната реализация на **count_routes** трябва първо да извика **answer**(1) за да отговори на въпроса за G[0], и след това да извика **answer**(2), за да отговори на въпроса за G[1].

Подзадачи

Подзадача 1 (49 точки)

- $2 \leq \mathbf{N} \leq 1\,000$
- $1 \leq \mathbf{M} \leq 10\,000$
- $\mathbf{Q} = 1$
- Всеки елемент на **G[0]** е цяло число между 1 и 100, включително.

Подзадача 2 (20 точки)

- $2 \leq \mathbf{N} \leq 150\,000$
- $1 \leq \mathbf{M} \leq 150\,000$
- $\mathbf{Q} = 1$
- Всеки елемент на **G[0]** е цяло число между 1 и 1 000 000 000, включително.

Подзадача 3 (31 точки)

- $2 \leq \mathbf{N} \leq 150\,000$
- $1 \leq \mathbf{M} \leq 150\,000$
- $1 \leq \mathbf{Q} \leq 2\,000$
- Всеки елемент на **G** е цяло число между 1 и 1 000 000 000, включително.

Детайли по реализацията

Ограничения

- Ограничение по време: 5 секунди
- Ограничение за памет: 256 MB
  Забележка: Не се задава явно ограничение за размера на стека. Паметта за стека зависи от общото ограничение за памет.

Интерфейс (API)

- Фолдер на имплементацията: garden/
- Да се имплементира от състезателя: garden.c или garden.cpp или garden.pas
- Интерфейс на състезателя: garden.h или garden.pas
- Интерфейс на грейдера: gardenlib.h или gardenlib.pas
- Примерен грейдер: grader.c или grader.cpp или grader.pas
- Вход на примерния грейдер: grader.in.1, grader.in.2, ...
  Забележка: Примерният грейдер чете входа си в следния формат:
  - Ред 1: **N**, **M** и **P**.
  - Редове 2 до **M**+1: описание на пътеките; т.е. ред i+2 съдържа R[i][0] и R[i][1], разделени с интервал, за $0 \leq i < M$.
  - Ред **M**+2: Q.
  - Ред **M**+3: масив G като редица от цели числа, разделени с интервали.
  - Ред **M**+4: масив от очаквани решения като редица от цели числа, разделени с интервали.
- Очакван изход за примерния грейдер: grader.expect.1, grader.expect.2, ...
  За тази задача, всеки един от тези файлове трябва да съдържа единствено текста "**Correct.**".