شهر جادویی

شهردار تاشکند می‌خواهد پارک تفریحی معروف «شهر جادویی» را از نو طراحی کند. به شما یک عدد صحیح مثبت K داده شده است و وظیفه‌ی شما طراحی یک پارک به صورت زیر است.

- یک عدد صحیح N انتخاب کنید که تعداد مکان‌های دیدنی را نشان می‌دهد. مکان‌های دیدنی با اعداد صحیح از 0 تا $N-1$ شماره‌گذاری شده‌اند.
- مسیرهای پیاده‌روی دوطرفه اضافه کنید که هر کدام بین یک جفت مکان **مشخص** قرار گیرند. می‌توان بیش از یک مسیر پیاده‌روی بین یک جفت مکان دیدنی وجود داشته باشد. **الزامی نیست که بتوانید با استفاده از مسیرهای پیاده‌روی بین هر جفت مکان دیدنی رفت و آمد کنید.**
- برای هر i با شرط $0 \le i < N$، به مکان i نوع $T[i]$ را اختصاص دهید. همچنین $2K$ نوع مختلف موجود است که از 0 تا $2K-1$ شماره‌گذاری شده‌اند. هر نوع باید حداقل به یک مکان اختصاص داده شود. مکان‌های مختلف ممکن است نوع یکسانی داشته باشند.

تحقیقات میدانی نشان می‌دهند که:

1. هر بازدیدکننده ترجیح می‌دهد از سه مکان بازدید کند، بدون اینکه از دو مکان با نوع یکسان به طور متوالی بازدید کند.
2. بازدیدکنندگان از مکان‌هایی که به مسیرهای پیاده‌روی زیادی متصل هستند، خوششان نمی‌آید.

شهردار می‌خواهد **همه‌ی** بازدیدکنندگان بالقوه را راضی کند و دو شرط برای طرح شما وضع می‌کند.

یک سه‌تایی مرتب از **نوع‌های** (t_1, t_2, t_3) به صورتی که $t_1 \neq t_2$ و $t_2 \neq t_3$ باشد را جالب می‌نامیم، اگر $0 \le t_1, t_2, t_3 < 2K$ باشد. توجه داشته باشید که t_1 ممکن است برابر با t_3 باشد. بنابراین $2K \cdot (2K-1)^2$ سه‌تایی جالب وجود دارد.

شرط ۱: برای هر سه‌تایی جالب (t_1, t_2, t_3) باید سه مکان a_1 و a_2 و a_3 (با شرط $0 \le a_1, a_2, a_3 < N$) وجود داشته باشد، به طوری که:

- نوع‌های a_1 و a_2 و a_3 با t_1 و t_2 و t_3 مطابقت دارند، یعنی $T[a_1] = t_1$ و $T[a_2] = t_2$ و $T[a_3] = t_3$.
- بین a_1 و a_2 یک مسیر پیاده‌روی وجود داشته باشد.
- بین a_2 و a_3 یک مسیر پیاده‌روی وجود داشته باشد.

وجود یا عدم وجود مسیر پیاده‌روی بین a_1 و a_3 اهمیتی ندارد. همچنین توجه داشته باشید که وقتی $t_1 = t_3$ است، مکان‌های a_1 و a_3 ممکن است یکسان باشند.

شرط ۲: هر مکان دیدنی می‌تواند حداکثر به K مسیر پیاده‌روی متصل شود.

این مسئله شامل 50 زیرمسئله‌ی **فقط-خروجی** با **امتیازدهی جزئی** است. هر زیرمسئله با مقدار مشخصی از K مطابقت دارد و شما باید پارکی طراحی کنید که تمام شرایط فوق را برای آن مقدار K برآورده کند. امتیاز شما به تعداد مکان‌های دیدنی موجود در راه‌حل شما بستگی دارد: مکان‌های دیدنی کمتر، امتیاز بالاتری به همراه دارند.

جزئیات پیاده‌سازی

دو روش برای ارسال راه‌حل شما وجود دارد و می‌توانید برای هر زیرمسئله از هر یک از آن‌ها استفاده کنید:

- **فراخوانی تابع**
- **فایل خروجی**

برای ارسال راه‌حل خود از طریق **فراخوانی تابع**، باید تابع زیر را پیاده‌سازی کنید.

```
std::pair<std::vector<int>, std::vector<std::pair<int, int>>> construct(int K)
```

- K : نصف تعداد نوع‌های مکان‌ها و همچنین حداکثر تعداد مجاز مسیرهای پیاده‌روی متصل به هر مکان دیدنی.
- این تابع دقیقاً یک بار برای هر زیرمسئله فراخوانی می‌شود.

این تابع باید یک جفت (T, E) که یک پارک تفریحی را توصیف می‌کند، برگرداند. فرض کنید M تعداد مسیرهای پیاده‌روی در پارک شما باشد.

- T : آرایه‌ای به طول N که نوع مکان‌ها را توصیف می‌کند.
- E : آرایه‌ای به طول M که مسیرهای پیاده‌روی را توصیف می‌کند. برای هر $0 \le j < M$، $E[j] = (U[j], V[j])$ نشان‌دهنده‌ی یک مسیر پیاده‌روی دو طرفه بین مکان $U[j]$ و مکان $V[j]$.

برای ارسال راه‌حل خود از طریق **فایل خروجی**، یک فایل متنی با قالب زیر ایجاد و ارسال کنید:

```
N M
T[0] T[1] ... T[N-1]
U[0] V[0]
U[1] V[1]
...
U[M-1] V[M-1]
```

توجه داشته باشید که راه‌حل شما باید محدودیت‌های زیر را برآورده کند تا **معتبر** در نظر گرفته شود:

- $N \le 2000$
- $0 \le T[i] < 2K$ برای هر $0 \le i < N$.
- $0 \le U[j], V[j] < N$ و $U[j] \ne V[j]$ برای هر $0 \le j < M$.
- شرط ۱ و شرط ۲ باید رعایت شوند.

محدودیت‌ها

- $1 \le K \le 50$

امتیازدهی

تعداد 50 زیرمسئله وجود دارد؛ یکی برای هر عدد صحیح K از 1 تا 50. یعنی برای زیرمسئله‌ی i ام $(1 \le i \le 50)$، مقدار $K = i$ است.

هر زیرمسئله طبق جدول زیر دارای امتیاز S و اندازه‌ی پارک هدف P است.

Subtasks	S	P
1	1	2
2	8	12
3	9	24
4	9	40
5	9	50
6 - 10	4	$12 \cdot K$
11 - 12	3	$12 \cdot K$
13 - 50	1	$12 \cdot K$

برای هر زیرمسئله‌ای اگر راه‌حل شما یک پارک تفریحی معتبر را توصیف نکند، امتیاز راه‌حل شما 0 خواهد بود (در CMS به عنوان `Output isn't correct` گزارش می‌شود).

در غیر این صورت، امتیاز شما بر اساس N و پارامترهای S و P به شرح زیر محاسبه می‌شود:

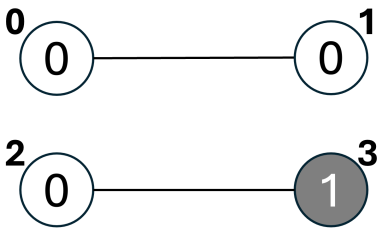
Condition	Score
$N \leq P$	S
$P < N \leq 2P$	$(0.4 + 0.3 \cdot \frac{2P-N}{P}) \cdot S$
$2P < N \leq 2000$	$(0.1 + 0.3 \cdot \frac{2P}{N}) \cdot S$

مثال

فراخوانی زیر را در نظر بگیرید.

```
construct(1)
```

در این مثال $K = 1$ است، بنابراین $2K = 2$ نوع مکان وجود دارد. شکل زیر یک راه‌حل معتبر با $N = 4$ مکان دیدنی و $M = 2$ مسیر پیاده‌روی را نشان می‌دهد. مکان‌های دیدنی $0, 1, 2$ از نوع 0 هستند و مکان دیدنی 3 از نوع 1 است.



دو سه‌تایی جالب وجود دارد:

- برای نوع سه‌تایی $(0, 1, 0)$ می‌توانیم $(a_1, a_2, a_3) = (2, 3, 2)$ را انتخاب کنیم.
- برای نوع سه‌تایی $(1, 0, 1)$ می‌توانیم $(a_1, a_2, a_3) = (3, 2, 3)$ را انتخاب کنیم.

این یعنی شرط ۱ برقرار است.

این تابع می‌تواند جفت $([0,0,0,1],[(0,1),(2,3)])$ را برگرداند. توجه داشته باشید که راه‌حل ارائه‌شده برای این مثال ممکن است برای $K=1$ بهینه نباشد.

ارزیاب نمونه

قالب ورودی:

```
K
```

قالب خروجی:

```
N M
T[0] T[1] ... T[N-1]
U[0] V[0]
U[1] V[1]
...
U[M-1] V[M-1]
```

توجه داشته باشید که خروجی ارزیاب نمونه با قالب مورد نیاز برای فایل خروجی مطابقت دارد.