

Tivoli

Borgmesteren af Tashkent ønsker at ombygge den berømte forlystelsespark Tivoli. Du er givet et positivt heltal K og din opgave er at udforme en park der opfylder følgende krav:

- Vælg et heltal N , antallet af forlystelser, og afmærk forlystelserne med heltal fra 0 til $N - 1$.
- For hvert i hvor $0 \leq i < N$, tildel forlystelse i typen $T[i]$. Der er $2K$ forskellige typer nummereret fra 0 til $2K - 1$. Hver type skal blive tildelt til mindst en forlystelse. Forskellige forlystelser kan have den samme type.
- Tilføj en flisegang mellem par af **forskellige** forlystelser. Du kan gå begge veje på disse flisegange. **Det er ikke nødvendigt at parken er konstrueret så man kan gå mellem alle par af forlystelser ved brug af flisegange.**

Ny forskning viser at:

1. Alle besøgende foretrækker at besøge præcis tre forlystelser uden at besøge to forlystelser af den samme type lige efter hinanden.
2. Besøgende kan ikke lide når forlystelserne er slutpunkter til for mange flisegange.

For at opfylde **alle** eventuelle besøgendes ønsker, har borgmesteren opsat følgende krav til din udformning.

Vi siger at en ordnet triple af **typer** (t_1, t_2, t_3) hvor $0 \leq t_1, t_2, t_3 < 2K$ er **interessant** hvis $t_1 \neq t_2$ og $t_2 \neq t_3$. Vær opmærksom på at t_1 godt kan være lig med t_3 . Der er derfor $2K \cdot (2K - 1)^2$ forskellige interessante tripler.

Betingelse 1: For hver interessant triple (t_1, t_2, t_3) , skal der eksistere tre forlystelser a_1, a_2, a_3 (med $0 \leq a_1, a_2, a_3 < N$) sådan at:

- Typerne af a_1, a_2, a_3 svarer til typerne t_1, t_2, t_3 . Det vil sige at, $T[a_1] = t_1$, $T[a_2] = t_2$ og $T[a_3] = t_3$.
- Der er en flisegang mellem a_1 og a_2 .
- Der er en flisegang mellem a_1 og a_3 .

Det er ligegyldigt hvorvidt der er en flisegang mellem a_1 og a_3 . Bemærk også at når $t_1 = t_3$, så må forlystelserne a_1 og a_3 gerne være den samme.

Betingelse 2: Hver forlystelse kan være slutpunkt til **Højst** K flisegange.

Denne opgave består af 50 **output-only** delopgaver med **delpoint**. Hver delopgave svarer til en specifik værdi af K , og du skal udforme en park der opfylder alle betingelserne for det givne K .

Antallet af point du får afhænger af hvor mange forlystelser der er i din løsning - færre forlystelser vil give flere eller det samme antal point.

Implementerings detaljer

Der er to måder at indsende din løsning og du må bruge den ene, den anden eller begge for hver delopgave:

- **Funktionskald**
- **Output fil**

For at indsende din løsning med **Funktionskald** skal du implementere følgende funktion

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

- K : halvdelen af antallet af typer af forlystelser, og også det maksimale antal af flisegange, som hver forlystelse kan være et slutpunkt til.
- Denne funktion kaldes præcis en gang for hver delopgave.

Denne funktion skal returnere et par (T, E) der beskriver forlystelsesparken. Lad M være antallet af flisegange i din park.

- T : en liste af længde N der beskriver typen af forlystelserne.
- E : en liste af længde M der beskriver flisegangene. For hver $0 \leq j < M$, udgør $E[j] = (U[j], V[j])$ en flisegang mellem de to forskellige forlystelser $U[j]$ og $V[j]$. Det er ikke tilladt at have flere flisegange mellem de samme par af forlystelser.

For at indsende din løsning som en **output fil**, skal du lave og indsende en tekst fil i følgende format:

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

Bemærk at din løsning skal overholde følgende begrænsninger og skal være anset som **gyldig**:

- $N \leq 2000$
- $0 \leq T[i] < 2K$ for alle $0 \leq i < N$, og alle typer skal være tildelt mindst en forlystelse.
- $0 \leq U[j], V[j] < N$ og $U[j] \neq V[j]$ for alle $0 \leq j < M$.
- Betingelse 1 and 2 skal være overholdt.

Begrænsninger

- $1 \leq K \leq 50$

Point

Der er 50 delopgaver, én for hvert heltal K fra 1 til 50. For delopgave i ($1 \leq i \leq 50$), er i værdien af K .

Hver delopgave har pointtal S og et mål for antallet af forlystelser P , som vist i følgende tabel:

Delopgave	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	$12 \cdot K$

Der gælder for enhver delopgave at hvis din løsning ikke beskriver en gyldig forlystelsespark, så bliver pointallet for din løsning 0 (tilbageporteret som `Output isn't correct` i CMS)

Ellers bliver dine point beregnet med udgangspunkt i N og parametrene S og P som vist herunder:

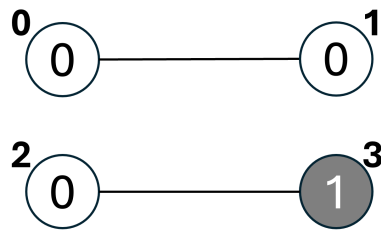
Betingelser	Point
$N \leq P$	S
$P < N \leq 2P$	$(0.4 + 0.3 \cdot \frac{2P-N}{P}) \cdot S$
$2P < N \leq 2000$	$(0.1 + 0.3 \cdot \frac{2P}{N}) \cdot S$

Eksempel

Overvej:

```
construct (1)
```

I det her eksempel er $K = 1$, så der er $2K = 2$ typer af forlystelser. Figuren nedenfor viser en gyldig løsning med $N = 4$ forlystelser og $M = 2$ flisegange. Forlystelse 0, 1, 2 er af type 0 og forlystelse 3 er af type 1.



Der er to interessante tripler:

- For type triplen(0, 1, 0) kan vi vælge $(a_1, a_2, a_3) = (2, 3, 2)$.
- For type triplen (1, 0, 1) kan vi vælge $(a_1, a_2, a_3) = (3, 2, 3)$.

Det betyder at betingelse 1 er overholdt.

Funktionen returnere parret $([0, 0, 0, 1], [(0, 1), (2, 3)])$. Observer at den givne løsning for dette eksempel måske ikke er optimalt for $K = 1$.

Eksempel Grader

Input format:

K

Output format:

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

Læg mærke til at outputtet af eksempel graderen overholder det ønsket format for ouput filen.