Krokodilváros

Krokodilvárosban **N** terem van, amelyeket **M** kétirányú, ismert hosszú folyosó köt össze. A folyosók két végpontja különböző és két terem között legfeljebb egy folyosó van. A termek közül K olyan van, amely kijáratot tartalmaz. A 0-s teremből indulunk és a lehető leggyorsabban ki kell jutni valamelyik kijáraton.

A haladás során azonban bizonyos folyosókat blokkolhatnak, de egyszerre csak egyet. A következő blokkolásnál az előző felszabadul.

A folyosók hálózata olyan, hogy létezik olyan stratégia, amellyel ki lehet jutni, bárhogyan is blokkolnak lépésenként egy-egy folyosót. Egy ilyen stratégiával számolhatod ki azt az időt, ami alatt biztosan ki lehet jutni a városból!

Feladat

Írj travel_plan(N,M,R,L,K,P) függvényt:

- **N** – a termek száma. A termeket **0**–tól **N-1**–ig sorszámozzuk.
- **M** – a folyosók száma. A folyosókat **0**–tól **M-1**–ig sorszámozzuk.
- **R** – a folyosókat leíró kétdimenziós tömb. Az **i.** folyosó az **R[i][0]** és az **R[i][1]** két különböző terem között halad (**0 ≤ i < M**), két terem között legfeljebb egy folyosó van.
- **L** – a folyosók hosszát leíró egydimenziós tömb. Az **1 ≤ L[i] ≤ 1 000 000 000** az **R[i][0]**-**R[i][1]** folyosó hossza, azaz az eljutás ideje az egyik teremből a másikba, illetve vissza.
- **K** – a kijáratok száma (**1 ≤ K < N**).
- **P** – a K különböző kijáratot leíró egydimenziós tömb.
  A P[i] az i. kijárat terem sorszáma (0 ≤ i < K). A **0**-ás biztosan nem kijárat.

A függvényed azt a legkisebb **T** időt adja meg, ami alatt a 0-s teremből biztosan el lehet jutni valamelyik kijárathoz!

A nem kijárati termekhez legalább 2 folyosó csatlakozik. A megoldás értéke: **T ≤ 1 000 000 000**.

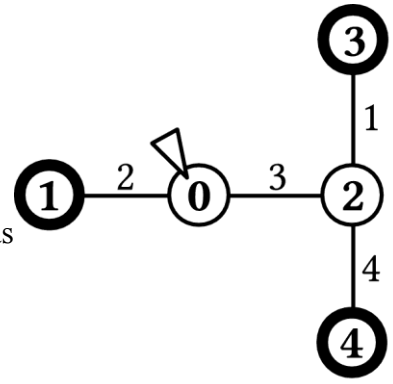
1. példa

N=5, **M=4**, **K=3**,

R=
0 1
0 2
3 2
2 4

L=
2
3
1
4

P=
1
3
4



1. ábra

A kijárati termeket az ábrákon vastag körökkel jelöljük. A megoldás menete:

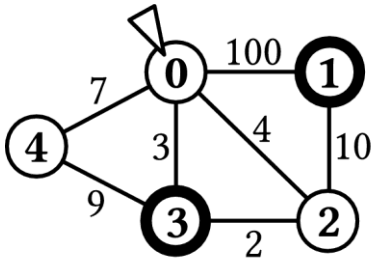
- Ha a 0-ban vagyunk, akkor menjünk az 1-be, de ha az a folyosó blokkolva van, akkor a 2-be!
- Ha a 2-ben vagyunk, akkor menjünk a 3-ba, de ha az a folyosó blokkolva van, akkor a 4-be!

Legrosszabb esetben 3+4=7 időegység alatt lehet kijutni, tehát a függvényed értéke 7 legyen!

2. példa

N=5, **M=7**, **K=2**,

	0 2		4		
	0 3		3		
	3 2		2		
R=	2 1	**L=**	10	**P=**	1
	0 1		100		3
	0 4		7		
	3 4		9		



2. ábra

A megoldás menete:

- Ha a 0-ban vagyunk, akkor menjünk a 3-ba, de ha a folyosó blokkolva van, akkor a 2-be!
- Ha a 2-ben vagyunk, akkor menjünk a 3-ba, de ha a folyosó blokkolva van, akkor az 1-be!
- Jó stratégia esetén a 4-be nem fogsz lépni.

Legrosszabb esetben 14 időegység alatt ki lehet jutni, azaz a travel_plan függvény értéke **14**.

Tesztek

1. eset (46 pont)

- **$3 \leq N \leq 1\,000$**.
- A folyosókból álló hálózat fa. Azaz **M** = **N**-1 és bármely **i** és **j** terem között pontosan 1 út van.
- Minden kijárati teremhez pontosan 1 folyosó csatlakozik.
- A nem kijárat termekhez legalább 3 folyosó csatlakozik.

2.eset (43 pont)

- **$3 \leq N \leq 1\,000$**.
- **$2 \leq M \leq 100\,000$**.

3. eset (11 pont)

- **$3 \leq N \leq 100\,000$**.
- **$2 \leq M \leq 1\,000\,000$**.

Határok

- Időlimit: 2 másodperc
- Memórialimit: 256 MB
  Megjegyzés: A verem méretre nincs külön korlát.

Interfész (API)

- A megoldás könyvtára: crocodile/
- A megvalósítandó modul: crocodile.c vagy crocodile.cpp vagy crocodile.pas
- Saját interfész: crocodile.h vagy crocodile.pas
- Értékelő interfész: crocodile.h vagy crocodilelib.pas
- Minta értékelő: grader.c vagy grader.cpp vagy grader.pas és crocodilelib.pas
- Minta bemenetek: grader.in.1, grader.in.2, ...

  Megjegyzés: A minta értékelő a bemenetet a következő formában olvassa:

  - **1**. sor: N, M és K.
  - **2**…**M+1**. sor: Az i+2. sorban az R[i][0], R[i][1] és az L[i] van, egy-egy szóközzel elválasztva (**$0 \leq i < M$**).
  - **M+2**. sor: a kijárati helyek – P[0], P[1],..., P[K-1] egész számok.
  - **M+3**: a megoldás értéke.

- A minta bemenetre elvárt kimenetek: grader.expect.1, grader.expect.2, ..., amelyekben a „Correct.” szövegnek kell lenni.