PA - Tema 2
- Polițiști, Programatori și Scandal -

Responsabili:
Ioan Popescu, Andreea-Nicoleta Duțu, Claudia Preda,
Andrei Preda, Matei Hriscu, Radu Nichita, Theo Pruteanu

Deadline soft: **22.05.2024 23:55**
Deadline hard: **29.05.2024 23:55**

CUPRINS

1 PROBLEMA 1: NUMĂRARE

1.1 Enunț

Nostalgic de vremea când încă era student, Adrian se întreabă cât de similare sunt traseele de antrenament ale Academiei de Poliție cu drumurile din orașul București.

Formal, se dau două grafuri orientate aciclice, fiecare cu câte N noduri și M muchii. Vi se cere să determinați numărul de lanțuri elementare de la 1 la N comune în ambele grafuri, modulo $10^9 + 7$.

Spunem despre un șir de noduri $a_1, a_2, \ldots, a_k$ că este un lanț elementar de la 1 la N dacă:

- $a_1 = 1$, $a_k = N$
- pentru orice $i < k$ există o muchie orientată de la a_i la a_{i+1}
- $a_i \neq a_j$ oricare ar fi i și j.

Două lanțuri $a_1, a_2, \ldots, a_k$ și $b_1, b_2, \ldots, b_q$ sunt comune dacă $k = q$ și $a_i = b_i$, oricare ar fi i.

1.2 Date de intrare

Pe prima linie din fișierul **numarare.in** se vor afla două numere naturale N și M separate prin câte un spațiu, reprezentând numărul de noduri, și numărul de muchii.

Pe următoarele M linii vor fi câte două numere naturale x și y, reprezentând faptul că există o muchie orientată de la x la y în primul graf.

Pe următoarele M linii vor fi câte două numere naturale x, y, reprezentând faptul că există o muchie orientată de la x la y în al doilea graf.

1.3 Date de ieșire

În fișierul **numarare.out** veți scrie numărul de lanțuri elementare comune de la 1 la N în cele două grafuri modulo $10^9 + 7$.

1.4 Restricții și precizări

- $1 \leqslant N \leqslant 10^5$, numărul de noduri
- $1 \leqslant M \leqslant 2 * 10^5$, numărul de muchii
- $1 \leqslant x < y \leqslant N$, unde (x, y) reprezintă o muchie

1.5 Testare și punctare

- Punctajul maxim este de 35 puncte.

- Timpul de execuție:
  - C/C++: **2 s**
  - Java: **3 s**
- Sursa care conține funcția **`main`** trebuie obligatoriu denumită: **`numarare.c`**, **`numarare.cpp`** sau **`Numarare.java`**.

1.6 Exemple

numarare.in	numarare.out	Explicație
4 5 1 2 1 3 2 3 2 4 3 4 1 2 1 4 2 3 2 4 3 4	2	Există două lanțuri elementare comune de la 1 la 4, anume $1 \to 2 \to 3 \to 4$ și $1 \to 2 \to 4$.

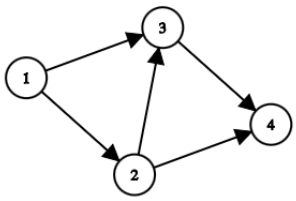

Figura 1: Primul graf

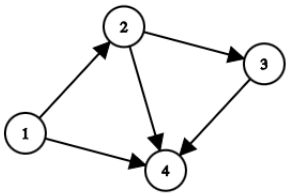

Figura 2: Al doilea graf

2 PROBLEMA 2: TRENURI

2.1 Enunț

Profund impresionat de trenurile din Elveția (și pentru că și-a plictisit toți prietenii vorbind despre ele), Bogdan Urzescu a decis să meargă într-o călătorie lungă.

Ca orice programator care se respectă, Bogdan a descărcat o listă de M curse directe cu trenul între diverse orașe. Pentru că el este preocupat de criptomonede, vă cere ajutorul cu următoarea problemă: știind orașul din care pleacă și orașul în care acesta își dorește să ajungă, spuneți-i numărul maxim de orașe distincte pe care le poate vizita. Se garantează că Bogdan poate ajunge la destinație.

Este important de menționat că nu există niciun ciclu în rețeaua de trenuri, adică, pornind din orice oraș, nu se poate ajunge din nou în acel oraș, indiferent de cursele alese.

2.2 Date de intrare

Pe prima linie din fișierul **trenuri.in** se vor afla denumirile orașului din care Bogdan va pleca și orașul în care Bogdan vrea să ajungă, separate printr-un spațiu.

Pe a doua linie se va afla un singur număr natural M, reprezentând numărul de trasee directe.

Pe următoarele M linii se vor afla câte două șiruri de caractere x și y, reprezentând faptul că există o cursă directă de la orașul x la orașul y.

2.3 Date de ieșire

În fișierul **trenuri.out** veți afișa un singur număr, reprezentând numărul maxim de orașe distincte care pot fi vizitate.

2.4 Restricții și precizări

- $1 \leqslant M \leqslant 2 * 10^5$
- Prin o cursă orientată de la x la y se înțelege că putem ajunge din orașul x în orașul y folosind acea cursă, dar nu și invers.
- Numele orașelor sunt formate doar din litere mici ale alfabetului englez.
- Numele orașelor sunt reale.

2.5 Testare și punctare

- Punctajul maxim este de 40 puncte.
- Timpul de execuție:
  - C/C++: **3 s**
  - Java: **4 s**
- Sursa care conține funcția **`main`** trebuie obligatoriu denumită: **`trenuri.c`**, **`trenuri.cpp`** sau **`Trenuri.java`**.

2.6 Exemple

`trenuri.in`	`trenuri.out`	Explicație

`bucuresti timisoara`
`4`
`bucuresti sibiu`
`sibiu timisoara`
`sibiu cluj`
`cluj timisoara`
	`4`	Există două posibilități de a ajunge în Timișoara: București → Sibiu → Timișoara și București → Sibiu → Cluj → Timișoara. Al doilea traseu trece prin 4 orașe, fiind și numărul maxim de orașe care pot fi vizitate.

3 PROBLEMA 3: DRUMURI OBLIGATORII

3.1 Enunț

Se dă un graf orientat cu costuri pe muchii și 3 noduri x, y, z. Se cere să se găsească o submulțime de muchii a căror sumă este minimă astfel încât să existe cel puțin un drum care pleacă din nodul x și cel puțin un drum care pleacă din nodul y, ambele ajungând în nodul z.

3.2 Date de intrare

Pe prima linie din fișierul **drumuri.in** se vor afla două numere naturale, N, respectiv M, separate prin câte un spațiu, reprezentând numărul de noduri, respectiv numărul de muchii.

Pe următoarele M linii se vor afla câte 3 numere a, b, c separate prin câte un spațiu reprezentând faptul că există o muchie orientată de la a la b de cost c.

Pe ultima linie se vor afla 3 numere naturale, x, y, z, cu semnificația din enunț.

3.3 Date de ieșire

În fișierul **drumuri.out** veți scrie un singur număr întreg reprezentând suma minimă a unei submulțimi de muchii care să respecte cerințele din enunț.

3.4 Restricții și precizări

- $1 \leqslant N \leqslant 10^5$
- $1 \leqslant M \leqslant 2 * 10^5$
- $1 \leqslant c \leqslant 10^9$
- Se garantează că există cel puțin un drum de la x la z și cel puțin un drum de la y la z în graful inițial.

3.5 Testare și punctare

- Punctajul maxim este de 40 puncte.
- Timpul de execuție:
  - C/C++: **2.5 s**
  - Java: **3.5 s**
- Sursa care conține funcția **main** trebuie obligatoriu denumită: **drumuri.c**, **drumuri.cpp** sau **Drumuri.java**.

3.6 Exemple

drumuri.in	drumuri.out	Explicație
5 7 1 3 6 1 4 5 2 4 6 2 5 5 4 5 5 5 3 2 5 4 5 2 1 3	13	Dacă alegem muchiile $(1,3)$, cu costul 6, $(2,5)$ cu costul 5 și $(5,3)$ cu costul 2, obținem o submulțime de muchii cu costul 13, și putem ajunge atât din nodul 2 cât și din nodul 1 în nodul 3.

4 PROBLEMA 4: SCANDAL (BONUS)

4.1 Enunț

Pentru a evita viitoarele scandaluri, Andrei și Adrian au decis să stabilească niște reguli pe care le vor folosi atunci când organizează o petrecere. Cei doi au în total N prieteni comuni, și au venit cu M reguli de tipul (x, y, c) care pot să însemne următoarele lucruri:

- dacă $c = 0$, atunci cel puțin unul dintre prietenii x sau y trebuie să participe la petrecere.
- dacă $c = 1$, atunci dacă x nu participă la petrecere, nici y nu va participa la petrecere.
- dacă $c = 2$, atunci dacă y nu participă la petrecere, nici x nu va participa la petrecere.
- dacă $c = 3$, atunci cel puțin unul dintre x și y nu participă la petrecere.

Să se determine o listă posibilă de invitați care pot fi chemați la petrecere fără să se încalce niciuna dintre reguli. Dacă există mai multe soluții se poate afișa oricare, indiferent de numărul de invitați (soluția nu trebuie să conțină număr minim sau număr maxim de invitați).

4.2 Date de intrare

Pe prima linie din fișierul **`scandal.in`** se vor afla două numere N și M separate prin câte un spațiu, reprezentând numărul de prieteni, respectiv numărul de reguli.

Pe următoarele M linii se vor afla câte un triplet de numere x, y, c cu semnificația din enunț.

4.3 Date de ieșire

În fișierul **`scandal.out`** veți scrie pe prima linie un singur număr, numărul de invitați la petrecere. Apoi, pe câte o linie, se va afișa câte un invitat.

4.4 Restricții și precizări

- $1 \leqslant N \leqslant 10^5$, numărul de prieteni
- $1 \leqslant M \leqslant 2 * 10^5$, numărul de reguli
- $1 \leqslant x, y \leqslant N$
- $0 \leqslant c \leqslant 3$

4.5 Testare și punctare

- Punctajul maxim este de **25** puncte.
- Timpul de execuție:
  - C/C++: **3 s**
  - Java: **4.5 s**
- Sursa care conține funcția **`main`** trebuie obligatoriu denumită: **`scandal.c`**, **`scandal.cpp`** sau **`Scandal.java`**.

4.6 Exemple

scandal.in	scandal.out	Explicație
5 7 2 3 3 2 4 3 2 5 3 3 1 3 4 3 0 4 5 1 5 1 2	1 4	Datorită regulii $(4, 3, 0)$, trebuie să invităm pe măcar unul dintre 4 și 3. Alegem să îl invităm pe 4. Observăm că doar cu 4 invitat respectăm toate regulile, deci este o soluție validă. Alte soluții valide ar fi să îi invităm pe 1 și 4 sau pe 1, 4, și 5, etc.

5 PUNCTARE

- Punctajul temei este de 150 puncte, distribuit astfel:
  - Problema 1: **35p**
  - Problema 2: **40p**
  - Problema 3: **40p**
  - Problema 4: **25p**
  - 5 puncte vor fi acordate pentru comentarii și README.
  - 5 puncte vor fi acordate automat de checker pentru coding style. Totuși, la corectarea manuala se pot aplica **depunctări de până la 20 de puncte** pentru **coding style neadecvat**.

Punctajul pe README, comentarii și coding style este condiționat de obținerea unui punctaj strict pozitiv pe cel puțin un test.

Se poate obține un bonus de **25p** rezolvând problema bonus **Scandal**. Acordarea bonusului **NU** este condiționată de rezolvarea celorlalte teste/probleme. În total se pot obține 150 de puncte (**NU** se trunchiază).

Pentru detalii puteți să vă uitați și peste regulile generale de trimitere a temelor.

- O temă care **NU** compilează va fi punctată cu 0.
- O temă care **NU** trece niciun test pe vmchecker va fi punctată cu 0.
- Vor exista mai multe teste pentru fiecare problemă în parte. Punctele pe teste sunt independente, punctajul pe un anumit test nefiind condiționat de alte teste.
- Fiecare problemă va avea o limită de timp pe test (precizată pe pagina cu enunțul). Dacă execuția programului pe un test al acelei probleme va dura mai mult decât limita de timp, veți primi automat 0 puncte pe testul respectiv și execuția va fi întreruptă.
- În fișierul `README.md` va trebui **să descrieți soluția** pe care ați ales-o pentru fiecare problemă, **să precizați complexitatea** pentru fiecare și alte lucruri pe care le considerați utile de menționat.

5.1 Checker

- Arhiva se va trimite pe VMchecker, unde tema se va testa folosind un set de teste private.
- Pentru testarea locală, aveți disponibil un set de teste publice (de aceeași dificultate) pe pagina cu resurse a temei.
- Arhiva pe care o primiți (cu scheletul temei) conține fișierul `README.checker.md`, referitor la funcționarea checker-ului. Vă recomandăm să îl citiți. Checkerul este strict cu lucruri precum numele surselor voastre, numele regulilor din `Makefile` etc.
- **Punctajul pe teste** este cel de pe VMchecker și se acordă rulând tema doar cu testele private.

- Checkerul verifică doar existența unui README cu denumire corectă (`README.md`) și conținut nenul. **Punctajul final pe README și comentarii** se acordă la corectarea manuală a temei.
- La corectarea manuală se poate depuncta pentru **erori de coding style** care nu sunt semnalate de checker.
- Corectorii își rezervă dreptul de a scădea puncte pentru orice problemă găsită în implementare, dacă vor considera acest lucru necesar.
- Pentru citirea în Java se recomandă folosirea `BufferedReader` (în link găsiți o clasă custom pentru citire, asemănătoare cu `Scanner`).

6 FOLOSIRE CHATGPT

- Folosirea ChatGPT, Copilot sau a oricărui model de limbaj sau tool (denumit in continuare **LLM**) ce vă poate ajuta la rezolvarea temei, cu idei sau cod, este puternic descurajată, dar nu interzisă.
- În cazul in care folosiți LLM-uri, trebuie să specificați acest lucru în README, precum și modul în care acestea au fost folosite (ex: ce prompt-uri ați folosit), pentru fiecare problemă pentru care au fost folosite tool-uri.
- Pentru fiecare problemă rezolvată folosind un LLM, pentru care a fost specificat în README acest lucru, se va aplica o penalizare de **33%** din punctajul acelei probleme.
- În cazul în care o problemă este rezolvată folosind un LLM, dar nu este specificat acest lucru in README, acest lucru se va considera **încercare de copiere** și se va sancționa conform regulamentului.

7 FORMAT ARHIVĂ

- Temele pot fi testate automat pe VMchecker. Acesta suportă temele rezolvate în C/C++ și Java.
- Arhiva cu rezolvarea temei trebuie să fie **`.zip`**, și va conține:
  - Fișierul/fișierele sursă
  - Fișierul **`Makefile`**
  - Fișierul **`README.md`**
- **ATENȚIE!** Tema va fi compilată și testată **DOAR pe Linux**.
- **ATENȚIE!** Pentru cei ce folosesc C/C++ **NU** este permisă compilarea cu opțiuni de optimizare a codului (`-O1`, `-O2`, etc.).
- **ATENȚIE!** Orice nerespectare a restricțiilor duce la pierderea punctajului (după regulile de mai sus).

8 LINKS

- Regulament general PA
- Google C++ Style Guide
- Google Java Style Guide
- Debugging și Structuri de Date
- Coding Tips pentru teme la PA (OCW)