Tema 2 – Proiectarea algoritmilor

Deadline: 27.05.2022
Responsabili: David Iancu, Mihai Nan

Problema 1 – Redundant (30p)

Se dă un graf orientat G cu N noduri si M muchii. Ne dorim să selectăm doar un număr nimim de muchii, astfel încât accesibilitatea în graf de la un nod la altul să nu se schimbe. De exemplu, dacă avem 3 noduri și muchii de la nodul 1 la nodul 2, de la nodul 2 la nodul 3, respectiv de la nodul 1 la nodul 3, este suficient să pastraă doar primele 2 muchii, ultima muchie ($1 \rightarrow 3$) fiind redundantă, deoarece putem ajunge de la 1 la 3 folosind cele 2 muchii intermediare. **Atenție, muchiile pot să nu existe în graful original.**

Precizări

Fișierul de intrare „reduntant.in” conține cele 2 numere N și M, urmate de M muchii. În fișierul ”redundant.out” trebuie să scrieți un singur număr – numărul minim de muchii care trebuie păstrate.

Restricții:

$1 \leq N \leq 10000$
$1 \leq M \leq 20000$

Exemplu

Input
4 6
1 2
2 3
3 4
1 3
1 4
2 4

Output
3

Explicație

Există drumuri de la 1 la 2, de la 1 la 3, de la 1 la 4, de la 2 la 3, de la 2 la 4 și de la 3 la 4. Dacă păstrăm doar muchiile 1->2, 2->3 și 3->4 putem ajunge de la orice nod la un nod cu număr mai mare (dar nu ne putem întoarce înapoi).

Problema 2 – Arbori de acoperire (30p)

Se dă un graf neorientat cu N noduri și M muchii. Dintre aceste muchii, ne interesează în special 2 seturi de muchii, raportate la arborii minimi de acoperire posibili (se știe că pot exista mai mulți arbori minimi de acoperire). Primul set de muchii constă în acele muchii care apar în toți arborii minimi de acoperire care se pot forma considerând graful inițial. Al doilea set de muchii constă în acele muchii care apar în cel puțin un arbore de acoperire posibil (dar nu în toate). Să se afișeze cele 2 seturi de muchii.

Precizări

Fișierul de intrare „arbori.in” va conține descrierea unui graf cu costuri – numerele N și M, apoi M muchii de tipul (a,b,c), reprezentând faptul că există muchie de la a la b cu cost c.
Fișierul de ieșire „arbori.out” va conține pe prima linie numărul de elemente din cele 2 mulțimi, mai întâi n1, cardinalul mulțimii cu muchii care apar în toți arborii de acoperire posibili, apoi n2, cardinalul muțimii formate din muchii care apar în cel puțiin un arbore de acoperire posibil, dar nu în toate. Pe următoarele n1 linii se vor afla muchiile din prima mulțime, iar pe următoarele n2 muchiile din a doua mulțime. Muchiile trebuie afișate în ordinea citirii din fișier.

Restricții

$1 \leq N \leq 5000$
$1 \leq M \leq 1000000$
$1 \leq c \leq 1000$

Exemplu

Input

5 7
1 2 1
2 3 1
3 4 2
4 1 2
1 5 3
4 5 3
2 5 6

Output
2 4
1 2
2 3
3 4
4 1
1 5
4 5

Explicație
Se pot forma 4 arbori minimi posibili de acoperire:
1. folosind muchiile (1, 2),(2, 3),(3, 4),(1, 5)
2. folosind muchiile (1, 2),(2, 3),(4, 1),(1, 5)
3. folosind muchiile (1, 2),(2, 3),(3, 4),(4, 5)
4. folosind muchiile (1, 2),(2, 3),(4, 1),(4, 5).

Problema 3 – cicluri

Se dă un graf orientat cu N noduri și M muchii. Cerința este foarte simplă – care este ciclul cu cel mai ieftin cost care se poate forma? Se cere să se afișeze doar costul acestuia.

Precizări

Din fișierul ”cicluri.in” se citește un graf orientat cu costuri, identic cu citirea de la problema 2. În fișierul „cicluri.out” se afișează un singur număr – ciclul cu costul cel mai mic.

Restricții

$1 \leq N \leq 500$
$1 \leq M \leq 1000$

Exemplu

Input

5 6
1 2 1
2 3 10
3 1 11
2 4 2
4 5 3
5 1 4

Output

10

Explicație:

Putem forma 2 cicluri – 1,2,3,1 și 1,2,4,5,1. Al doilea ciclu are un cost mai mic, 10, față de costul primului, care este 22.

Precizări generale:

Se acceptă doar soluții în C/ C++.
Aveți nevoie de un makefile cu regulile build, run-p1, run-p2, run-p3 și clean.
Este necesar un fișier de README, în care să explicați modul în care ați gândit algoritmii (doar ideea, fără cod). Punctajul pe README este de 10p, în total (3p pe problemă + 1 punct dacă aveți toate problemele abordate).
Temele vor fi testate anti-plagiat. Nu dați codul colegilor voștri!
Timpul de rulare este de 1 secundă/ test. Temele vor fi testate și pe vmchecker. Checkerul urmează să apară în scurt timp.
Rezolvările se vor încărca atât pe vmchecker, cât și pe Moodle și Teams.