Subiectul 1

Se dă un graf neorientat conex cu n>3 vârfuri și m>n muchii.
Informațiile despre graf se citesc din fișierul graf.in cu următoarea structură:

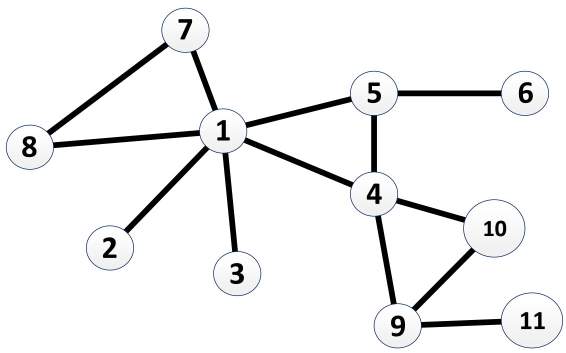
- pe prima linie sunt n și m
- pe următoarele m linii sunt câte 2 numere naturale reprezentând extremitățile unei muchii

Se citeşte de la tastatură un vârf v.

a) Să se afişeze muchiile critice care sunt incidente în v, dacă există (altfel se va afişa mesajul “nu exista”). O(**m**)

b) Să se afişeze listele de adiacență ale unui arbore parțial T al lui G în care vârful v are gradul cu 1 mai mic decât îl are în G: $d_T(v) = d_G(v) - 1$, dacă un astfel de arbore există O(**m**)

graf.in	Iesire pe ecran dacă se citește pentru v valoarea 1 (nu contează ordinea în care se afișează informațiile; soluția la b) nu este unică)
11 13 1 2 1 3 1 4 1 5 4 5 5 6 1 7 7 8 1 8 4 9 9 10 10 4 9 11	muchii critice: 1 2 1 3 Arbore: 1: 2 3 4 5 7 2: 1 3: 1 4: 1 9 10 5: 1 6 6: 5 7: 1 8 8: 7 9: 4 11 10: 4 11: 9



Subiectul 2

Se citesc informații despre un graf **orientat fără circuite** G din fișierul `graf.in`.
Fișierul are următoarea structură:

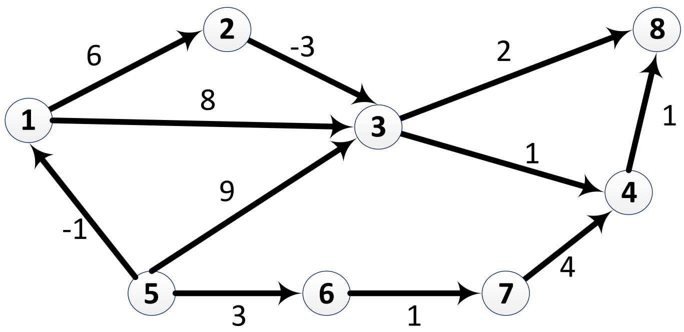
- pe prima linie sunt două numere reprezentând numărul de vârfuri n (n>4) și numărul de arce m ale grafului
- pe următoarele m linii sunt câte 3 numere întregi reprezentând extremitatea inițială, extremitatea finală și costul unui arc din graf (costul unui arc poate fi și **negativ**).
- pe ultima linie este un nod sursă s.

Spunem că un vârf y este accesibil din x în G dacă există un drum de la x la y.

a) Să se determine centralitatea vârfului s: cc(s) = suma distanțelor de la s la vârfurile t accesibile din s (=sum{d(s,t) | t este accesibil din s}). **Complexitate O(n+m)**

b) Să se afișeze un drum de cost maxim în G. **Complexitate O(n+m)**

graf.in	Iesire pe ecran (nu este unică)
8 11	18
1 2 6	5 3 8
2 3 -3	
1 3 8	
3 8 2	
3 4 1	
4 8 1	
5 1 -1	
5 3 9	
5 6 3	
6 7 1	
7 4 4	
1	



Explicații:

d(1,2) = 6, d(1,3)=3, d(1,4) = 4, d(1,8) = 5 => cc(1) =6+3+4+5= 18

Drumul 5,3,8 are cel mai mare cost dintre toate drumurile din graf

Subiectul 3

Propuneți un algoritm bazat pe algoritmul Ford-Fulkerson / Edmonds Karp pentru rezolvarea următoarei probleme.

n angajați, numerotați 1,..., n sunt membri în m proiecte, numerotate 1,..., m. Pentru fiecare angajat se cunosc proiectele în care este membru. Pentru fiecare proiect trebuie ales un coordonator. Știind că un angajat poate coordona cel mult 2 proiecte, să se desemneze, dacă este posibil, câte un coordonator pentru fiecare proiect. Datele despre membri proiectelor se vor citi dintr-un fișier cu următoarea structură:

- pe prima linie sunt numerele naturale n și m
- pe următoarele linii sunt perechi de numere naturale i j cu $i \in \{1,\dots, n\}$ și $j \in \{1,\dots, m\}$ cu semnificația: angajatul i este membru în proiectul j.

Dacă se pot desemna coordonatori pentru toate proiectele, rezultatul se va afișa sub forma prezentată în exemplul de mai jos. Altfel se va afișa mesajul “nu este posibil” și numărul maxim de proiecte pentru care se poate desemna un coordonator.

Complexitate $O(n^2m)$ (pot fi cel mult mn perechi i j cu semnificația: angajatul i este membru în proiectul j)

Observație: Putem modela problema cu un graf bipartit angajați-proiecte. Dacă un angajat poate coordona cel mult un proiect, atunci problema se reduce la a determina un cuplaj de cardinal maxim în graful bipartit angajați-proiecte și a verifica dacă orice vârf de tip proiect este saturat. Se acorda 1p dacă se rezolva problema pe cazul în care că un angajat poate coordona cel mult un proiect.

angajati.in	Iesire pe ecran (solutia nu este unica)
4 7 1 1 2 1 1 2 2 3 3 3 3 4 4 1 4 3 1 5 1 6 1 7 4 5 3 6 2 4	1 2 1 7 2 1 2 3 3 4 3 6 4 5 Dacă eliminăm linia cu perechea 4 5 din datele de intrare (angajatul 4 nu mai este membru în proiectul 5) problema nu mai are soluție, se poate desemna coordonator pentru maxim 6 proiecte