Subiectul 1 (3 puncte)

Se dă un arbore ponderat T cu n>3 vârfuri și un vârf s.
Informațiile despre arbore se citesc din fișierul ***graf.in*** cu structura:

- pe prima linie este n
- pe următoarele n linii sunt listele de adiacență ale lui G; linia i începe cu gradul vârfului i și apoi conține vecinii vârfului i, fiecare vecin fiind urmat de costul muchiei:

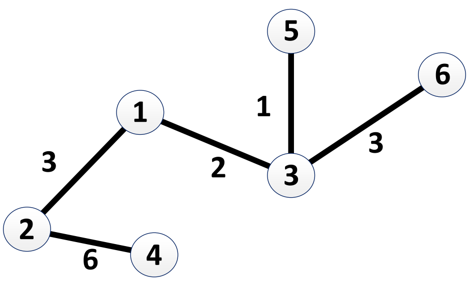
<grad> <vecin1> <cost_muchie_de_la_i_la_vecin1> <vecin2> <cost_muchie_de_la_i_la_vecin2> etc

- pe ultima linie este vârful s

(vârfurile sunt numerotate de la 1)
Pentru un lanț P în T definim capacitatea lanțului P ca fiind capacitatea minimă a unei muchii din P.

a) Să se afișeze pentru fiecare vârf v capacitatea unicului lanț elementar de la s la v (sub forma v: capacitate lanț) **Complexitate O(n)**
b) Să se afișeze care este capacitatea minimă a unui lanț cu o extremitate în s și să se afișeze un astfel de lanț

graf.in	***Iesire pe ecran***
6 **2** 2 3 3 2 **2** 1 3 4 6 **3** 1 2 5 1 6 3 **1** 2 6 **1** 3 1 **1** 3 3 1 Explicații: informațiile de pe linia 2 2 3 3 2 corespunzătoare vârfului 1 se citesc astfel: varful 1 are gradul 2; primul său vecin este 2 și costul muchiei de la el la 2 este 3: al doilea sau vecin este 3, iar costul muchiei de la el la 3 este 2	1: 0 2: 3 3: 2 4: 3 5: 1 6: 2 Capacitatea minima este 1 pentru lantul 1 3 5



Subiectul 2 (3 puncte)

Se citesc informații despre un graf **orientat** ponderat **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, **m>n**
- pe următoarele m linii sunt câte 3 numere pozitive reprezentând extremitatea inițială, extremitatea finală și costul unui arc din graf
- pe următoarea linie (a (m+2)-a linie) din fișier sunt un număr natural k (0<k<n) și un șir de k vârfuri reprezentând vârfurile sursă ale grafului $s_1,\ldots,s_k$
- pe ultima linie a fișierului sunt două vârfuri t_1 și t_2, reprezentând vârfurile destinație ale grafului (distincte de vârfurile sursă de pe linia anterioară).

Notăm cu $S=\{s_1,\ldots,s_k\}$ mulțimea vârfurilor sursă din G și cu $T=\{t_1, t_2\}$ mulțimea vârfurilor destinație din G. Spunem că un vârf y este accesibil din x în G dacă există un drum de la x la y.

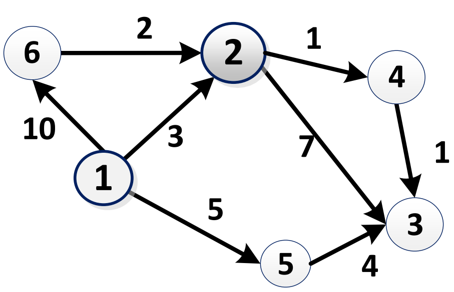
a) Să se verifice dacă graful dat este fără circuite și să se afișeze un mesaj corespunzător.

b) Să se determine un drum de cost maxim care începe cu un vârf din S și se termină cu un vârf din T. Dacă nu există un astfel de drum se va afișa un mesaj corespunzător: niciun vârf destinație nu este accesibil dintr-un vârf sursă.

Complexitate O(m)

Exemplu

graf.in	Iesire pe ecran
6 8 1 2 3 1 6 10 6 2 2 2 4 1 4 3 1 5 3 4 1 5 5 2 3 7 2 1 3 2 5	drum maxim 1 6 2



S = {1, 3}
T = {2, 5}

Subiectul 3 (3 puncte)

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

Pentru n proiecte, numerotate 1,..., n s-au înscris m studenți numerotați 1,..,m, fiecare student depunând o listă de opțiuni cu proiectele la care vrea să participe.

a) Dat un număr k de la tastatură, să de determine o listă de k asocieri proiect – student prin care k studenți diferiți sunt asociați la k proiecte diferite **Complexitate O(km)**

b) Să se determine, dacă există, o modalitatea de a asocia toți studenții la proiecte astfel încât un student să fie asociat la exact 2 proiecte, iar la un proiect să fie asociați exact 2 studenți și să se afișeze o astfel de modalitate sub forma prezentată în exemplul de mai jos. Altfel se va afișa mesajul “nu este posibil”. **Complexitate O(nm)**

Datele despre proiecte și studenți 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 $i \in \{1,\ldots, n\}$ și $j \in \{1,\ldots, m\}$ cu semnificația: studentul j s-a înscris la proiectul i.

graf.in	Iesire pe ecran (solutia nu este unica)
4 4 1 1 1 2 1 3 2 1 2 2 3 1 3 3 3 4 4 3 4 4 (primul este indicele proiectului, al doilea al studentului)	a) pentru k=2 asocieri proiect - student 1 1 2 2 b) asocieri proiect-student 1 1 1 2 2 1 2 2 3 3 3 4 4 3 4 4