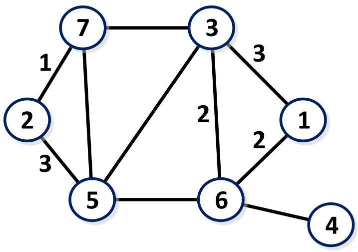
1. (**1p**) Adăugați ponderi - numere naturale **mai mari decât 3** - pe muchiile grafului din figura de mai jos care nu au încă ponderi, astfel încât graful să aibă exact doi arbori parțiali de cost minim (justificați).



2. (**1p**) Care dintre următoarele afirmații sunt adevărate? Justificați (complexitatea algoritmilor studiați se presupune cunoscută, nu trebuie demonstrată în justificare)

a) Algoritmul Bellman-Ford are complexitate mai bună decât algoritmul lui Dijkstra pentru grafuri orientate ponderate cu ponderi pozitive

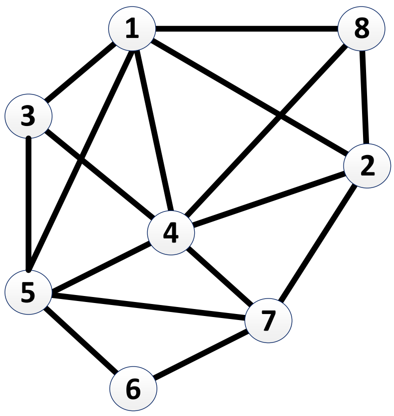
b) O ordonare (sortare) topologică a vârfurilor unui graf orientat fără circuite se termină cu un vârf cu grad extern 0

c) Un graf neorientat conex cu n vârfuri are cel puțin n muchii

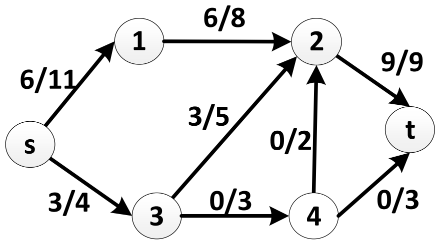
d) Un graf eulerian poate avea mai multe componente conexe

3. (1p) a) Fie G un graf neorientat conex cu gradul maxim al unui vârf 6. Care este numărul maxim de culori folosite de algoritmul Greedy de colorare a vârfurilor lui G prezentat la curs, dacă vârfurile sunt ordonate folosind strategia Smallest First? Justificați.

b) Exemplificați (cu explicații) algoritmul Greedy de colorare cu vârfurile ordonate folosind strategia Smallest First pentru graful următor.



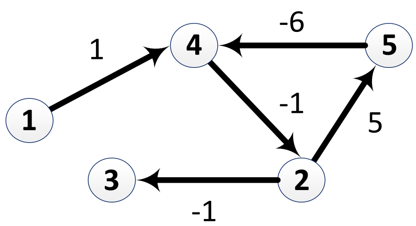
4. (1,5p) Definiți noțiunile de flux, tăietură minimă și lanț nesaturat/drum de creștere. Ilustrați pașii algoritmului Ford-Fulkerson pentru rețeaua din figura următoare (unde pe un arc e sunt trecute valorile f(e)/c(e) reprezentând flux/capacitate), pornind de la fluxul indicat și alegând la fiecare pas un s-t lanț f-nesaturat de lungime minimă (algoritmul Edmonds-Karp)**.** Indicați o tăietură (s-t tăietură) minimă în rețeaua (se vor indica vârfurile din bipartiție, arcele directe, arcele inverse). Mai există și o altă s-t tăietură minimă în această rețea? Justificați răspunsurile.



5. (2p) Fie G = (V, E, w) un graf orientat ponderat, cu ponderi numere întregi și s un vârf în G. Considerăm algoritmul lui Bellman Ford descris în următorul pseudocod:

```
pentru fiecare u∈V executa
     d[u] = infinit; tata[u]=0
d[s] = 0
pentru i = 1, |V|-1 executa
     pentru fiecare uv ∈ E executa
          daca d[u]+w(u,v)<d[v] atunci
               d[v] = d[u]+w(u,v)
               tata[v] = u
```

Considerăm graful următor.



La finalul execuției pseudocodului de mai sus pentru acest graf, s=1 și arcele considerate in ordinea (1,4), (5,4) (4,2), (2,5), (2,3) vectorul d are elementele 0 , -8, -9, -7, -3 iar vectorul tata este 0, 4, 2, 5, 2.

Adăugați în pseudocod instrucțiunile necesare pentru ca algoritmul să testeze existența unui circuit cu cost negativ în graf accesibil din s (=pentru care există un drum de la s la un vârf al său) și, în caz afirmativ, să afișeze unul, și ilustrați-le pe graful dat ca exemplu (cu explicații).

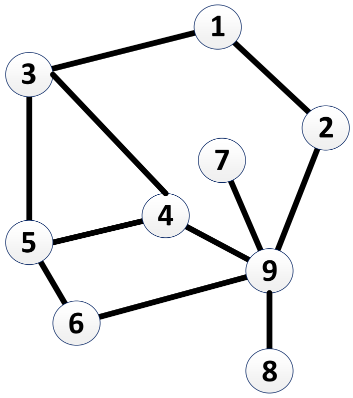
6. (1p) Este corect următorul algoritm de determinare a unui arbore parțial de cost minim al unui graf conex ponderat G = (V, E, w)? Justificați (fără a apela în justificare la modul de funcționare al altor algoritmi; rezultatele folosite trebuie demonstrate și trebuie explicat modul în care se folosesc)

`T = (V, E = ∅) – inițial V conține toate vârfurile și nu conține nicio muchie`

`pentru i = 1, |V|-1`

1. **`Alege o componentă conexă C al lui T care conține vârful i`**
2. **`Alege o muchie de cost minim e cu o extremitate în C și cealaltă nu și adaugă e la T`**

7. (**1,5p)**.a) Indicați fețele hărții următoare și gradul fiecărei fețe.



b) Fie M=(V, E, F) o hartă conexă cu n>3 vârfuri și m muchii. Arătați că dacă orice vârf din M are gradul 3 și orice față are gradul 3 sau 6 atunci sunt exact 4 fețe de grad 3.