Laborator 10

Exerciții

1. **Diametru**[1]

Se dă un arbore neorientat cu `N` noduri numerotate de la `1` la `N`. Să se găsească un diametru al arborelui, adică o cale care nu conține niciun nod de două ori şi care are lungimea maximă (exprimată în număr de muchii).

Date de intrare

Fişierul de intrare diametru.in conține pe prima linie numărul de noduri, `N`. Pe următoarele `N - 1` linii se află câte o pereche de noduri `u v` indicând că între nodurile `u` şi `v` există o muchie.

Date de ieşire

În fişierul de ieşire diametru.out se va tipări, pe prima linie, diametrul `D`. Pe a doua linie se vor tipări `D + 1` noduri distincte care, parcurse în ordine, formează o cale de `D` muchii. Dacă există mai multe soluții, oricare este acceptabilă.

Restricții

$2 \le N \le 100.000$.

2. **Dijkstra**[2]

Se dă un graf orientat ponderat cu `n` noduri – în care fiecare arc are asociat un cost, număr natural strict pozitiv, şi un nod `p`. Să se determine, folosind algoritmul lui Dijkstra, costul minim al drumului de la `p` la fiecare nod al grafului.

Date de intrare

Fişierul de intrare dijkstra.in conține pe prima linie numerele `n p`, iar următoarele linii câte un triplet `i j c`, cu semnificația: există arcul `(i j)` şi are costul `c`.

Date de ieşire

Fişierul de ieşire dijkstra.out va conține pe prima linie n numere, separate prin exact un spațiu, al `i`-lea număr reprezentând costul drumului minim de la `p` la `i`. Dacă nu există drum de la `p` la un anumit vârf, costul afişat va fi `-1`.

[1] http://varena.ro/problema/diametru
[2] https://www.pbinfo.ro/probleme/588/dijkstra

Restricții

$1 \le n \le 100$.

Costul unui arc va fi mai mic decât `1000`.

Costul unui drum este egal cu suma costurilor arcelor care îl compun.

3. **Traveling**[3]

Ivan Cel Rapid trebuie să plătească din fonduri proprii călătoria spre locul unde urmează să se desfăşoare următorul concurs de programare. El dispune doar de `S` euro. Din acest motiv, el a verificat cu mare grijă programul mijloacelor de transport în comun şi bineînțeles prețurile.

Vom nota cu `1` locul de plecare, cu `N` locul unde urmează să se desfaşoare concursul şi cu `2`, `3`, `...`, `N - 1` celelalte oraşe prin care ar putea să treacă. Ivan a găsit `M` curse de forma: cursa între oraşele `v` şi `w` durează `t` ore şi are tariful de `e` Euro în ambele sensuri. Pot exista mai multe curse între oraşele `v` şi `w`. Acestea pot varia atât ca durată cât şi ca tarif.

Să se scrie un program care găseşte o călătorie de la oraşul `1` la oraşul `N` la un tarif mai mic sau egal cu `S`. Dacă există mai multe variante de a călători, programul va afişa varianta în care timpul petrecut pe drum este minim.

Date de intrare

Fişierul de intrare traveling.in conține pe prima linie numerele întregi pozitive `S`, `N` şi `M`. Fiecare din următoarele `M` linii conține `4` numere întregi: `v`, `w`, `t` şi `e` (descriind o cursă).

Date de ieşire

În fişierul de ieşire traveling.out se va găsi durata călătoriei. Dacă nu se găseşte nicio variantă de călătorie la un tarif mai mic sau egal cu S, programul o să afişeze -1.

Restricții

$S \le 2.000$, $N \le 3.000$, $M \le 5.000$,

$1 \le v \le N$, $1 \le w \le N$, $1 \le t \le 100$, $1 \le e \le 100$.

4. **Drum**[4]

Se dă un graf orientat cu `N` vârfuri, numerotate de la `1` la `N`. Se cere să se răspundă la `Q` întrebări de forma: "*există un drum optim de la `1` la `N` care să treacă prin `X`?*"

Date de intrare

Fişierul de intrare drum.in conține pe prima linie `N` = numărul de varfuri ale grafului, `M` = numărul arcelor sale şi `Q` = numărul de întrebări, cele trei numere fiind separate prin câte un spațiu. Pe următoarele `M` linii se află câte două numere naturale cuprinse între `1` şi `N` reprezentând extremitățile arcelor. Pe ultimele `Q` linii ale fişierului se află câte un număr `X`, care defineşte o întrebare de tipul celor descrise mai sus.

[3] http://varena.ro/problema/traveling
[4] http://varena.ro/problema/drum

Date de ieşire

În fişierul de ieşire drum.out vor fi scrise pe primele `Q` linii răspunsurile la întrebări. În cazul în care vârful `X` aparține unui drum optim, în fişierul de ieşire se va scrie `1`, iar în caz contrar `0`.

Restricții

$1 \le N \le 100.000$, $1 \le M \le 200.000$, $1 \le Q \le 100.000$.

5. **Dijkstra2**[5]

Dijkstra este un cetățean model al comunității în care trăiește. El își ajută fiecare vecin aflat în necaz. Astăzi, Vlad îi cere ajutorul și îl roagă să livreze câte un pachet fiecărui vecin de-al lor.

Știind că sunt `n` case în această comunitate, iar distanțele dintre ele variază, Dijkstra vă roagă să realizați un program care să afișeze costul drumului minim dintre casa `p`, casa lui Vlad, și casele vecinilor. În cazul în care nu există drum până la unul dintre vecini, se va afișa `-1`.

Date de intrare

Programul citește din fișier numărul `n`, numărul de case din comunitate, `m`, numărul de drumuri dintre case, `p`, numărul casei lui Vlad, iar pe următoarele `m` linii se vor afla `x`, `y` și `d` cu semnificația că există drum între casele `x` și `y` de lungime `d`.

Date de ieșire

Programul va afișa `n` numere separate printr-un spațiu, al `i`-lea număr reprezentând distanța minimă de la casa lui Vlad la casa `i`.

Restricții

$1 \le p \le n \le 100.000$, $1 \le m \le 250.000$.

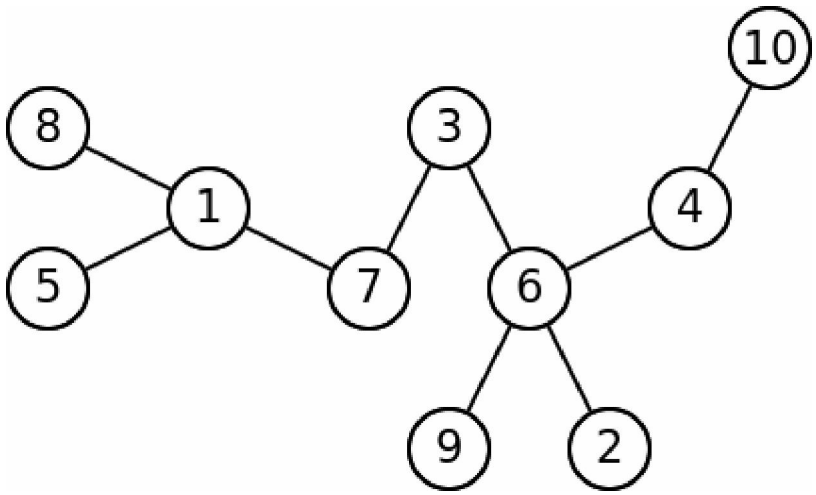
Cele `m` distanțe citite vor fi mai mici decât `20.000`

6. **Capitala**[6]

Imperiul Roman se clatină sub atacurile barbarilor! Harta imperiului conține `N` oraşe şi drumuri de legătură. Oricare drum poate fi parcurs într-o zi. Din eficienţă, romanii au construit numărul minim necesar de drumuri pentru a putea călători între oricare două oraşe. Oraşele de la marginea imperiului se numesc avanposturi şi au un singur drum de legătură.

Pentru a proteja administrația, romanii doresc să mute capitala într-un oraş cât mai depărtat de avanposturi: distanța de la acel oraş până la cel mai apropiat avanpost trebuie să fie maximă. Ajutați-i să găsească toate oraşele care maximizează această distanță.

[5] https://www.pbinfo.ro/probleme/1887/dijkstra2
[6] http://varena.ro/problema/capitala



Date de intrare

Fişierul de intrare capitala.in conţine pe prima linie numărul `N` de oraşe. Pe următoarele `N-1` linii sunt date perechi de numere `x y` cu semnificaţia că între oraşele `x` şi `y` există un drum. Oraşele sunt numerotate de la `1` la `N`.

Date de ieşire

În fişierul de ieşire capitala.out se vor tipări, pe prima linie, distanţa maximă până la un avanpost şi numărul de oraşe care au această distanţă, separate printr-un spaţiu. Pe linia a doua se vor tipări, în ordine crescătoare, numerele oraşelor care obţin această distanţă, separate prin spaţii.

Restricţii

$3 \le N \le 100.000$.

7. **Parc**[7]

Parcul oraşului este alcătuit din `N` intersecţii, numerotate de la `1` la `N`, unite între ele prin `M` alei bidirecţionale, fiecare având o anumită lungime. Într-o intersecţie precizată `C` se organizează un concert; de asemenea, unele intersecţii, precizate şi ele, reprezintă porţi de intrare în parc, accesul fiind posibil doar prin aceste porţi.

Gigel poate ajunge cu maşina la oricare dintre aceste porţi, dar vă roagă să alegeţi pentru el acea poartă pentru care distanţa până la intersecţia `C` este minimă. Dacă există mai multe porţi cu această proprietate se va determina poarta cu numărul de ordine mai mic.

Date de intrare

Fişierul de intrare parc.in conţine pe prima linie numerele `N M C`; următoarele `M` linii câte un triplet `i j L`, cu semnificaţia: există alee între intersecţia `i` şi intersecţia `j` şi are lungimea `L`. Următoarea linie conţine numărul de porţi `P`; ultima linie conţine `P` numere diferite, reprezentând porţile.

Date de ieşire

Fişierul de ieşire parc.out va conţine pe prima linie numărul de ordine al porţii alese.

[7] https://www.pbinfo.ro/probleme/593/parc

Restricții

$1 \le n \le 100$.

Lungimea unei alei va fi mai mică decât 1000.

Între oricare două intersecții există drum, direct sau prin intermediul altor intersecții

8. **Festivaluri**[8]

Tudor este foarte indecis, deoarece a fost chemat la `R` festivaluri și puterea lui fizică nu îi permite să ajungă la toate. În orașul în care locuiește sunt `M` străzi unidirecționale și `N` intersecții numerotate cu numere de la `1` până la `N`. Festivalurile au loc în `R` intersecții. El pornește din intersecția cu numărul `z`.

Pentru a ajunge dintr-o intersecție în alta, folosește străzile. Când parcurge o stradă, el consumă o anumită energie, care diferă de la stradă la stradă.

După terminarea fiecărui festival, Tudor se va reîntoarce la casa lui, adică la intersecția cu numărul `z`, costul drumului de această dată fiind `0`, pornind din nou la următorul festival.

Întrucât este un om foarte dedicat muzicii, Tudor vrea să participe la cât mai multe festivaluri, dar fără să-și depășească puterea lui fizică `P`.

Determinați numărul maxim de festivaluri la care poate participa.

Date de intrare

Fișierul de intrare festivaluri.in va conține pe prima linie numerele `N, M, P, z, R`. Pe următoarele `M` linii se vor afla câte `3` numere, reprezentând intersecția de unde începe strada, intersecția unde se termină strada și energia consumată pentru a parcurge strada. Pe următorul rând, se vor afla cei `R` indici ai intersecțiilor unde se vor organiza festivalurile.

Date de ieșire

Fișierul de ieșire festivaluri.out va conține pe prima linie numărul `cnt`, reprezentând numărul maxim de festivaluri la care poate participa Tudor.

Restricții

$1 \le N, M, z, R \le 100$, $1 \le P \le 10.000$.

Numerele de pe cele `m+1` linii a fișierului de intrare vor fi mai mici decât `100`.

Este posibil ca plecând din intersecția `z` să nu se poată ajunge în toate intersecțiile.

9. **Graph**[9]

Călinuța tocmai a găsit o foaie de hârtie pe care este desenat un graf orientat aciclic cu `N` noduri şi `M` arce, fiecare arc având o distanță de valoare întreagă.

Dându-se `N`, `M` şi cele `M` arce cu distanțele dintre ele, trebuie să calculați pentru Călinuța distanța minimă dintre fiecare două noduri.

[8] https://www.pbinfo.ro/probleme/1895/festivaluri
[9] https://www.pbinfo.ro/probleme/146/graph

Date de intrare

Fişierul de intrare graph.in conţine pe prima linie numerele naturale `N` şi `M` separate printr-un singur spaţiu, iar pe următoarele `M` linii se află câte `3` numere `A`, `B` şi `C`, reprezentând un arc de la `A` către `B` de lungime `C`.

Date de ieşire

Fişierul de ieşire graph.out conţine `N` linii cu câte `N` valori separate prin câte un spaţiu, reprezentând matricea drumurilor minime. Dacă nu există drum de la un nod `a` la un nod `b`, valoarea care trebuie afişată este "`x`" (fără ghilimele).

Restricţii

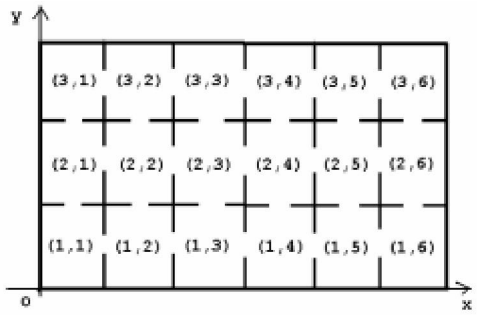
$1 \le n \le 1500$, $1 \le m \le 30000$

Numerele nodurilor sunt valori cuprinse între `1` şi `N`.

Lungimea arcelor aparţine intervalului `[-1000; 1000]`.

10. **Zâna**[10]

Castelul Zânei Spiriduşilor este construit pe o suprafaţă de teren dreptunghiulară şi are $N \times M$ camere identice, de formă pătratică, dispuse câte `M` pe direcţia `Ox` şi câte `N` pe direcţia `Oy` ca în desenul alăturat în care `N = 3` şi `M = 6`. Din fiecare cameră se poate intra în orice cameră învecinată, cameră ce are un perete comun cu aceasta. Fiecare cameră este identificată prin coordonatele sale, ca în figură.



În castel trăiesc `K` spiriduşi împreună cu Zâna lor. Fiind în curând aniversarea zilei de naştere a Zânei, fiecare spiriduş a pregătit câte un cadou pe care îl ascunde, nevăzut de ceilalţi, într-una din camerele castelului. Tradiţia acestei sărbători impune următoarele reguli:

- În căutarea cadourilor, Zâna porneşte din camera de coordonate `(1, 1)`. Ea se deplasează prin camerele castelului cât timp în aceste camere nu se află niciun cadou.
- Căutarea se încheie în momentul în care Zâna intră într-o cameră în care se află cel puţin un cadou. Zâna va primi toate cadourile aflate în această cameră iar restul cadourilor din celelalte camere vor dispărea.

[10] https://www.pbinfo.ro/probleme/957/zana

Cerință

Scrieți un program care să citească din fișierul zana.in numerele naturale N, M, K şi cele K perechi de numere naturale reprezentând coordonatele camerelor în care spiriduşii au ascuns cadourile, şi care să determine:

- numărul maxim X de cadouri pe care le poate primi Zâna în urma respectării regulilor;
- numărul Y de camere în care poate ajunge Zâna respectând regulile, camere ce conțin fiecare câte X cadouri.

Date de intrare

Fişierul zana.in conține pe prima linie cele trei numere naturale: N, M, K, separate prin câte un spațiu. Pe fiecare din următoarele K linii, câte una pentru fiecare spiriduş, sunt scrise câte două numere naturale: I, J, separate printr-un spațiu, reprezentând coordonatele camerei în care spiriduşul curent a ascuns cadoul.

Date de ieşire

Fişierul zana.out va conține două linii. Pe prima linie se va scrie numărul natural X reprezentând numărul maxim de cadouri pe care le poate primi Zâna conform tradiției. Pe cea de-a doua linie se va scrie numărul natural Y, reprezentând numărul camerelor în care poate ajunge Zâna şi care conțin fiecare câte X cadouri.

Restricții

K, M, N, I, J sunt numere naturale nenule.

$2 \le N, M \le 100$, $10 \le K \le 510$, $1 \le I \le N$, $1 \le J \le M$.

Se garantează că nu există cadouri în căsuța (1, 1).

11. **Roboțel**[11]

Tudor a primit un joc educațional numit "*Roboțel*" cu ajutorul căruia va învăța punctele cardinale *Nord*, *Est*, *Sud*, *Vest*. Jocul constă dintr-un roboțel care se deplasează pe o tablă de forma unei matrici pătratice, împărțită în R linii şi R coloane. Fiecare căsuță, aflată la intersecția dintre o linie şi o coloană, este fie căsuță „*liberă*", fie căsuță „*semnalizator*", caz în care este etichetată cu una din literele N, E, S, V, reprezentând 4 sensuri posibile de deplasare. Când roboțelul ajunge într-o „*căsuță semnalizator*", el îşi schimbă sensul de deplasare astfel:

- Dacă căsuța este etichetată cu N atunci roboțelul se va deplasa în continuare de jos în sus;
- Dacă căsuța este etichetată cu E atunci roboțelul se va deplasa în continuare de la stânga la dreapta;
- Dacă căsuța este etichetată cu S atunci roboțelul se va deplasa în continuare de sus în jos;
- Dacă căsuța este etichetată cu V atunci roboțelul se va deplasa în continuare de la dreapta la stânga.

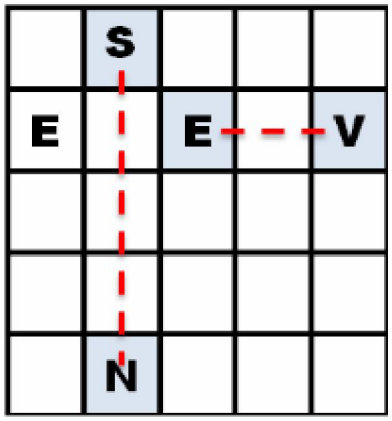
Două căsuțe semnalizator formează o pereche „*blocantă*" dacă:

[11] http://varena.ro/problema/robotel

- Se află pe aceeaşi linie şi conţin literele `E` şi `V`, căsuţa cu `E` are coloana mai mică decât a celei etichetate cu `V` şi între ele, pe aceeaşi linie nu există alte căsuţe semnalizatoare.
- Se află pe aceeaşi coloană şi conţin literele `S` şi `N`, căsuţa cu `S` are linia mai mică decât a celei etichetate cu `N` şi între ele, pe aceeaşi coloană nu există alte căsuţe semnalizatoare.

În figura 1, de exemplu, sunt 2 perechi blocante: perechea `(1,2) (5.2)` şi perechea `(2,3) (2,5)`.

Figura 1



Roboţelul porneşte din căsuţa `(1,1)`, aflată pe prima linie şi prima coloană şi dacă aceasta este liberă, se deplasează, în cadrul primei linii, de la stânga la dreapta. În cazul în care căsuţa de pornire `(1,1)` este semnalizator, atunci roboţelul se va deplasa pe direcţia indicată de litera cu care este etichetată. Considerând că roboţelul se deplasează pe tablă, el se opreşte doar în următoarele situaţii:

- Roboţelul intră într-o căsuţă liberă aflată pe prima sau ultima linie, respectiv prima sau ultima coloană, caz în care dacă s-ar menţine sensul deplasării actuale roboţelul ar părăsi tabla;
- Roboţelul intră într-o „*căsuţă semnalizator*” a unei perechi blocantă şi se va opri în cealaltă căsuţă a perechii.

De exemplu, în Figura 2, roboţelul ajunge în căsuţa liberă `(3,5)` unde se opreşte. În Figura 3, roboţelul se va opri în căsuţa `(4,1)` deoarece dacă ar schimba sensul spre `Est`, ar reveni în ultima căsuţă semnalizator vizitată, `(4,3)`.

Roboţelul înaintează o căsuţă într-un pas, în sensul de deplasare.

Figura 2

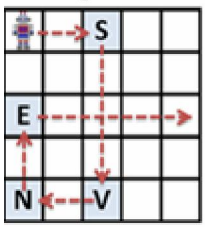
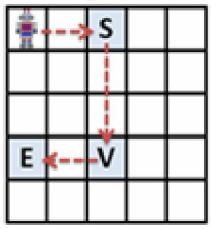


Figura 3



Cerinţe

Scrieţi un program care, cunoscând numărul `R` de linii şi coloane şi cele `K` căsuţe semnalizator, determină:

- Toate perechile blocante de pe tablă.
- Numărul de paşi efectuaţi pe fiecare sens în parte: *Nord*, *Est*, *Sud* şi *Vest*.

Date de intrare

Fişierul de intrare robotel.in conţine pe prima linie numărul natural `P` reprezentând cerinţa din problemă care trebuie rezolvată, pe a doua linie, separate printr-un spaţiu numărul natural `R` şi numărul natural `K`, iar pe următoarele `K` linii, două numere naturale şi un caracter, separate prin câte un spaţiu reprezentând, în ordine, linia, coloana şi litera unei căsuţe semnalizator.

Date de ieşire

Dacă valoarea lui `P` este `1`, se va rezolva doar cerinţa `1)`. Fişierul de ieşire robotel.out va conţine perechile blocante, pentru fiecare pereche de căsuţe afişându-se `4` numere naturale separate printr-un spaţiu, reprezentând, în ordine, linia, coloana primei căsuţe semnalizator, respectiv linia şi coloana celei de-a doua căsuţe semnalizator. Perechile de căsuţe vor fi afişate pe linii ordonat după regula: o pereche `L1 C1 L2 C2` va fi afişată înaintea perechii `L3 C3 L4 C4` dacă `L1 < L3` sau `L1 = L3` şi `C1 < C3`, adică se va afişa mai întâi perechea cu prima căsuţă având linia mai mică decât a primei căsuţe din cealaltă pereche sau la linii egale, va fi afişată perechea cu coloana mai mică. Dacă nu există astfel de perechi de căsuţe, în fişierul de ieşire se va afişa valoarea `0`.

Dacă valoarea lui `P` este `2`, se va rezolva doar cerinţa `2)`. Fişierul de ieşire robotel.out va conţine `4` numere naturale separate printr-un spaţiu, reprezentând, în ordine, numărul de paşi parcurşi de roboţel în sensurile *Nord*, *Est*, *Sud* şi *Vest*.

Restricţii

$2 \le R \le 200$ şi $1 \le K \le R \times R$.

O căsuţă semnalizator conţine o singură literă.

Pentru cerinţa `2` se garantează că în toate testele deplasarea roboţelului se opreşte!