MODALITATEA DE DESFĂȘURARE A TESTULUI DE LABORATOR LA DISCIPLINA "PROGRAMAREA ALGORITMILOR"

- Testul de laborator la disciplina "Programarea algoritmilor" se va desfășura în ziua de **08.01.2022**, între orele 9^{30} și 12^{00}, astfel:
  - **09^{30} – 10^{00}**: efectuarea prezenței studenților
  - **10^{00} – 11^{30}**: desfășurarea testului
  - **11^{30} – 12^{00}**: verificarea faptului că sursele trimise de către studenți au fost salvate pe platformă

- Testul se va desfășura pe platforma MS Teams, iar pe tot parcursul desfășurării lui studenții trebuie să fie conectați pe canalul dedicat **cursului** de "Programarea algoritmilor" corespunzător seriei lor.
- În momentul efectuării prezenței, fiecare student trebuie să aibă pornită camera video în MS Teams și să prezinte buletinul sau cartea de identitate. Dacă dorește să-și protejeze datele personale, studentul poate să acopere codul numeric personal și/sau adresa!
- În timpul desfășurării testului studenții pot să închidă camera video, dar trebuie să o deschidă dacă li se solicită acest lucru de către un cadru didactic!

- Testul va conține **3 subiecte**, iar un subiect poate să aibă mai multe cerințe.

- Rezolvarea unui subiect se va realiza într-un singur fișier sursă Python (.py), indiferent de numărul de cerințe, care va fi încărcat/atașat ca răspuns pentru subiectul respectiv.

- Numele fișierului sursă Python trebuie să respecte următorul șablon: ***grupa_nume_prenume_subiect.py***. De exemplu, un student cu numele Popescu Ion Mihai din grupa 131 trebuie să denumească fișierul care conține rezolvarea primului subiect astfel: **131_Popescu_Ion_Mihai_1.py.**

- La începutul fiecărui fișier sursă Python se vor scrie, sub forma unor comentarii, următoarele informații: numele și prenumele studentului, grupa sa și enunțul subiectului rezolvat în fișierul sursă respectiv. Dacă un student nu reușește să rezolve deloc un anumit subiect, totuși va trebui să încarce/atașeze un fișier sursă Python cu informațiile menționate anterior!

- Toate rezolvările (fișierele sursă Python) trimise de către studenți vor fi verificate din punct de vedere al similarității folosind un software specializat, iar eventualele fraude vor fi sancționate conform Regulamentului de etică și profesionalism al FMI (http://old.fmi.unibuc.ro/ro/pdf/2015/consiliu/Regulament_etica_FMI.pdf).

Subiect 1

[4 p.] Fișierul text *text.in* conține, pe mai multe linii, un text în care cuvintele sunt despărțite prin spații și semnele de punctuație uzuale. Să se scrie în fișierul text *text.out* literele din textul dat care au frecvențele relative nenule, precum și frecvențele lor relative, conform modelului din exemplul de mai jos. Frecvența relativă a unei litere într-un text este egală cu raportul dintre frecvența sa în textul respectiv și numărul total de litere din acel text. Literele vor fi scrise în fișierul *text.out* în ordinea descrescătoare a frecvențelor lor, iar în cazul unor frecvențe egale vor fi ordonate alfabetic. Nu se va face distincție între litere mici și litere mari.

Exemplu:

text.in	text.out
Ana are multe pere si mere, dar are mai multe mere decat pere si prune mai multe decat pere.	e: 0.250 a: 0.125 r: 0.125 m: 0.097 t: 0.069 i: 0.056 p: 0.056 u: 0.056 d: 0.042 l: 0.042 c: 0.028 n: 0.028 s: 0.028

Textul din fișierul de intrare *text.in* conține 72 de litere mari și mici, iar litera 'e' apare de 18 ori în textul dat, deci frecvența relativă a literei 'e' este egală cu 18 / 72 = 0.250.

Subiectul 2

Fișierul "**date.in**" are n linii cu următoarea structură: pe linia i sunt prezente, separate prin câte un spațiu, n numere naturale reprezentând elementele de pe linia i dintr-o matrice, ca în exemplul de mai jos.
Liniile și coloanele unei matrice se presupun numerotate de la 0.

a) [0,25p] Scrieți o funcție **citire_matrice** care citește numerele din fișierul "**date.in**" și returnează o matrice de dimensiuni n x n formată din aceste numere.

b) [1,5p] Scrieți o funcție care primește ca parametri: o matrice (listă de liste), un caracter ch care poate primi valoarea "c" sau "d" și doi parametri x și y cu valoare implicită 0 .
Funcția va modifica matricea primită ca parametru astfel:

- Dacă al doilea parametru - caracterul ch - primește la apel valoarea "c", funcția interschimbă coloana x cu coloana y.
- Dacă al doilea parametru - caracterul ch - primește la apel valoarea "d", funcția nu va primi la apel decât 2 parametri și trebuie să interschimbe elementele de pe diagonala principală cu elementele de pe diagonala secundară.

c) [1,25p] Folosind apeluri ale funcției definite la punctul **b)**, **oglindiți** matricea returnată de funcția de la punctul a) după coloana de pe poziția [n / 2] și apoi interschimbați elementele de pe diagonala principală cu cele de pe diagonala secundară. După oglindire și interschimbare, să se parcurgă matricea în **zig-zag** pe linii și să se afișeze șirul obținut în fișierul "**date.out**" ca în exemplu. **Se cunoaște faptul că n este impar**.
Explicație suplimentară : Parcurgerea în zig-zag pe linii se va face de sus în jos, astfel:

- prima linie se parcurge de la stânga la dreapta,
- a doua linie se parcurge de la dreapta la stânga,
- a treia linie se parcurge de la stânga la dreapta etc.

date.in	**după oglindire + interschimbarea diagonalelor**	**date.out**
1 4 3 2 5 6 9 8 7 10 15 14 13 12 11 16 19 18 17 20 21 24 23 22 25	1 2 3 4 5 10 9 8 7 6 11 12 13 14 15 20 19 18 17 16 21 22 23 24 25	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25

Subiect 3

Se consideră o rețea în plan formată din puncte unite prin linii numite legături (muchii). Fiecare punct are coordonatele întregi, iar o legătură are asociată o culoare (un șir de caractere fără spații reprezentând numele culorii, de exemplu: roșu, verde, albastru) și o etichetă (un șir care poate conține spații). Un punct cu coordonatele x și y va fi notat (x,y). Se consideră fișierul text *legaturi.in* care conține informații despre o astfel de rețea, fiecare linie conținând informații despre o legătură sub forma:

[x1,y1] [x2,y2] culoare eticheta

unde [x1,y1] și [x2,y2] sunt punctele între care există legătură, iar ***culoare*** și ***etichetă*** reprezintă culoarea și eticheta asociate legăturii. Vom spune că legătura este între punctele [x1,y1] și [x2,y2], vom numi punctele [x1,y1] și [x2,y2] capetele legăturii (legătura este neorientată și bidirecțională) și cele două puncte sunt vecine. Un exemplu de fișier de acest tip este următorul:

legaturi.in

```
[1,2] [1,3] rosu legatura 1
[1,2] [1,4] albastru legatura 2
[1,3] [2,6] rosu legatura 3
[2,6] [2,7] albastru legatura 4
[2,7] [3,8] rosu legatura 5
```

a) [2 p.] Să se memoreze datele din fișier într-o singură structură astfel încât să se **răspundă eficient** la cerințele de la punctele următoare (interogarea și modificarea informațiilor despre o legătură dintre două puncte date, determinarea vecinilor unui punct).

b) [1 p.] Scrieți o funcție **insereaza_legatura** care primește 5 parametri:

- în primul parametru se transmite structura în care s-au memorat datele la cerința a)
- următorii 2 parametri sunt tupluri cu 2 elemente reprezentând capetele unei legături
- ultimii 2 parametri sunt șiruri de caractere reprezentând culoarea și eticheta unei legături.

Dacă există deja o legătură în rețea între punctele primite ca parametru funcția va returna **False**, altfel funcția va adăuga această legătură în rețea (modificând structura trimisă ca parametru) și va returna **True**. Să se apeleze funcția pentru punctele (1,3) și (2,7), culoarea negru și eticheta "legatura noua" și să se afișeze legăturile memorate în structura obținută în același format în care s-au dat și datele în fișier (nu contează ordinea în care se vor afișa legăturile).

ieșire

```
[1,2] [1,3] rosu legatura 1
[1,2] [1,4] albastru legatura 2
[1,3] [2,6] rosu legatura 3
[2,6] [2,7] albastru legatura 4
[2,7] [3,8] rosu legatura 5
[1,3] [2,7] negru legatura noua
```

c) [1 p.] Scrieți o funcție **vecini** care primește ca parametri (în această ordine): structura în care s-au memorat datele la cerința a) și un număr variabil de tupluri cu 2 elemente reprezentând puncte din rețea. Funcția va returna o listă cu acele puncte *p* din rețea cu proprietatea că există legătură de la *p* la orice punct primit ca parametru (vecinii comuni pentru punctele primite ca parametru). Punctele din lista returnată vor fi ordonate descrescător după a doua coordonată. Apelați funcția pentru punctele (2,7) și (1,2) și afișați rezultatul obținut.

ieșire

```
[(3, 8), (1, 3)]
```