Structuri de date
Tema 1

1. Se consideră un vector V cu n elemente numere naturate de cel puţin 2 cifre. Să se construiască un al doilea vector W, cu acelaşi număr de elemente şi care pe fiecare poziţie i să aibă media cifrelor elementului $V[i]$.(1p)

2. Se consideră un vector *numere* care conţine *nr* numere naturale. Să se verifice care dintre cifrele de la 0 la 9 apare cel mai des în vector şi de câte ori apare. (1p)
   Exemplu: $numere = \{12, 231, 9012, 34, 8123, 22, 507\}$. Atunci cifra care apare cel mai des este 2 şi apare de 6 ori.

3. Se consideră două mulţimi de numere naturale A şi B, memorate în doi vectori distincţi (într-o mulţime fiecare element este unic!). Să se construiască un al tri-lea vector, care conţine elementele din $A \cap B$. (1p)

4. Se consideră un vector de n puncte. Fiecare punct este o structură cu două câmpuri, reprezentând coordonatele spaţiale (x, y). Să se afişeze perechile de puncte care: se află la distanţa cea mai mare şi la distanţa cea mai mică unul de celălat. Se consideră distanţa euclidiană. Se cere utilizarea unei structuri coordonata, cu campurile X şi Y, cu o funcţie membră distanţă, care returnează distanţa de la punctul curent la un punct dat ca parametru. (2p)
   Exemplu: $puncte = \{(3, 7), (2, 1), (6, 11), (0, 2)\}$, atunci perechea de puncte aflate la cea mai mare distanţă unul faţă de celălalt este $\{(6, 11), (0, 2)\}$, iar perechea de puncte cele mai apropiate este $\{(2, 1), (0, 2)\}$.

5. Se consideră o structură *fractie*, cu câmpurile de tip int *numarator* şi *numitor*. Această structură dispune de o metodă *reductie*, care reduce fracţia (ex: 12/30 devine 2/5), de funcţii pentru operaţii aritmetice, de funcţii de comparare. De asemenea o funcţie de transformare în număr zecimal (ex: $2/5 = 0.4$). Să se se citească un vector de fracţii. Fiecare dintre acestea să se reducă. Să se sorteze vectorul cât mai eficient şi să se calculeze suma elementelor sale. Scrieţi funcţii de citire/ afişare pentru fracţii. Punctajul maxim se acordă pentru rezolvarea completă, elegantă şi eficientă.(4p)

6. Se consideră o matrice pătratică de dimensiuni $n \times n$. Să se determine câte elemente prime se află sub diagonala principlaă. De asemenea să se interschimbe cele două diagonale. (1p)

7. Un profesor a studiat structura relaţiilor dintre elevii săi. Pentru a reprezenta această structură, profesorul a numerotat elevii de la 1 la n şi a construit o matrice pătratică cu n linii astfel: $a(i,j) = 1$ dacă elevul i îl agreează pe elevul j şi 0 altfel. Se consideră că fiecare elev se agreează pe sine însuşi. (1p)

   a. Afişaţi pe ecran elevul (elevii dacă sunt mai mulţi) care are (au) cei mai mulţi prieteni şi câţi prieteni are (au). Se consideră prieteni doi elevi care se agreează reciproc.

   b. Există vreun elev pentru care nu există niciun alt elev pentru care sentimentele sunt reciproce (se agreează/ nu se agreează reciproc)?

   c. Afişaţi elevii care nu sunt agreeaţi de nimeni

   Exemplu: se consideră 6 elevi şi matricea de prietenie următoare

$$\begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}$$

   a. Elevul 1 are cei mai mulţi prieteni (2) şi anume pe elevul 2 şi pe elevul 5.

   b. Elevul 3 se află în situaţia de nereciprocitate. El agreează elevii 4 şi 5, dar este agreeat de elevii 1 şi 2.

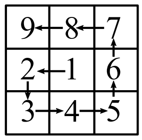
   c. Elevul care nu este agreeat de nimeni este elevul 6.

8. Se consideră un număr natural cu n cifre. Să se taie p cifre, $p < n$, astfel încât numărul rămas să fie cel mai mare posibil. Nu este permisă schimbarea ordinii cifrelor în număr. Se stochează numărul într-un vector.(3p).
   Exemplu: pentru numărul $nr = 129670020$ cu $n = 9$, dacă se taie $p = 5$ cifre, numărul minim care se obţine este: 9720.

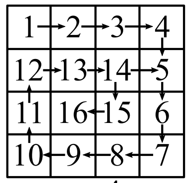
9. Să se scrie într-o matrice pătratică numerele de la 1 la n^2 în spirală: (2p)

   - pentru n impar începând din centru
   - pentru n par începând din colţul stânga-sus

   Exemplu:



n=3



n=4

10. Se consideră o matrice *matr* cu *nrow* linii şi *ncol* coloane ale cărei elemente sunt numere naturale formate dintr-o singură cifră. Se consideră că fiecare linie reprezintă un număr în baza $base \in \{2, 3, 4, \ldots, 10\}$. Să plaseze numerele transformate în baza zece într-un vector *numbers* şi să se afişeze acest vector. (1p)
    Exemplu: Se consideră $base = 2$ şi matricea

$$\begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

    Atunci $numbers = \{28, 21, 6, 19, 1\}$.

11. Se consideră o matrice *mat* cu dimensiunile $m \times n$. Să se afişeze indicii acelor coloane, care au toate elementele nenule şi să se construiască o a doua matrice *rezult*, care să aibă exact coloanele din *mat* găsite anterior.(1p)

12. Se consideră o matrice pătratică A de dimensiuni $n \times n$, subdiagonală. O matrice se numeşte subdiagonală dacă toate elementele aflate deasupra diagonalei principale sunt nule. **Observaţie:** suma şi produsul a două matrice subdiagonale sunt tot matrice subdiagonale.

    a. Să se transforme partea utilă a matricii (adică elementele de pe diagonala principală şi de sub diagonala principală) într-un vector. (1p)

    b. Să se scrie un algoritm care citeşte 2 matrice subdiagonale A şi B, le transformă conform (a) în doi vectori V_a şi V_b şi apoi calculează produsul $C = AB$ al celor două matrice folosind doar vectorii V_a şi V_b. (3p)

    Exemplu: Se consideră matricea subdiagonală

$$A = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 2 & 3 & 0 & 0 & 0 \\ 4 & 1 & 3 & 0 & 0 \\ 1 & 9 & 0 & 2 & 0 \\ 1 & 0 & 1 & 2 & 2 \end{pmatrix}$$

a. Partea utilă este $V_a = \{1, 2, 3, 4, 1, 3, 1, 9, 0, 2, 1, 0, 1, 2, 2\}$

b. Dacă se consideră matricea

$$B = \begin{pmatrix} 2 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 2 & 0 & 0 \\ 1 & 1 & 2 & 3 & 0 \\ 2 & 0 & 0 & 1 & 2 \end{pmatrix}$$

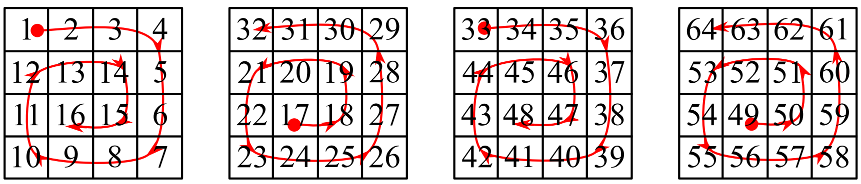
Atunci $V_b = \{2, 1, 1, 0, 1, 2, 1, 1, 2, 3, 2, 0, 0, 1, 2\}$, iar produsul celor două matrice este $V_c = \{2, 7, 3, 9, 4, 6, 13, 11, 4, 6, 8, 3, 6, 8, 4\}$ care reprezintă matricea

$$C = \begin{pmatrix} 2 & 0 & 0 & 0 & 0 \\ 7 & 3 & 0 & 0 & 0 \\ 9 & 4 & 6 & 0 & 0 \\ 13 & 11 & 4 & 6 & 0 \\ 8 & 3 & 6 & 8 & 4 \end{pmatrix}$$

13. Se consideră un cub de dimensiune $n \times n \times n$ care conţine numerele de la 1 la n^3 dispuse în modul următor:

- pe fiecare strat impar ordonate în spirală începând din colţul stânga sus, în sensul acelor de ceasornic
- pe fiecare strat par ordonate în spirală, pornind de la elementul aflat deasupra ultimului element parcurs pe stratul anterior şi în sens invers acelor de ceasornic.

Exemplu: pentru $n = 4$



strat 1 strat 2 strat 3 strat 4

a. Dacă se citeşte un număr p, să se determine poziţia în cub - adică coordonatele x - linia, y - coloana, z - strat, fără a construi efectiv cubul. (2.5p)

b. Să se determine numărul de numere prime de pe fiecare faţă laterală a cubului. (2.5p)

Observaţie: Punctajul maxim se obţine pentru consum de memorie minim şi eficienţă maximă, evitând construcţia efectivă a cubului (care este $O(n^3)$). Precizaţi complexitatea pentru fiecare punct în parte.

În exemplul de mai sus:

a. Numărul 45 are coordonatele $(2, 2, 3)$.

b. Numerele prime de pe feţele laterale sunt reprezentate în pătratele gri din figura de mai jos

61	62	63	64
36	35	34	33
29	30	31	32
4	3	2	1

fata 1

58	59	60	61
39	38	37	36
26	27	28	29
7	6	5	4

fata 2

55	56	57	58
42	41	40	39
23	24	25	26
10	9	8	7

fata 3

64	53	54	55
33	44	43	42
32	21	22	23
1	12	11	10

fata 4

De exemplu, pentru suprafaţa de joc din figura c) numărul de pătrate accesibile (marcate cu X) de pe suprafaţă este 11. Pătratele rămase albe nu sunt atacate de nici o dama. Pentru suprafaţa de joc cu $N = 6$, $D = 3$ şi $K = 4$ un exemplu de suprafaţă de joc este figura d).

Condiţii necesare pentru acceptarea temei:

1. Este obligatorie utilizarea alocării dinamice a memoriei pentru vectori şi matrice. Folosiţi sintaxa C++.

2. Dacă doriţi să folosiţi containerul *vector* atunci

   - trebuie ataşată în plus o minidocumentaţie legată de *vector*, care să conţină: - modul de alocare/dealocare a memoriei pentru vector, modul de funcţionare a *push_back*, rezervarea de memorie atunci când se cunoaşte numărul de elemente care se vor introduce în vector, diferenţa între metodele specifice *size*() şi *capacity*(), accesul la elemente (diferenţa dintre operatorul [ ] şi metoda at()), iniţializarea unui vector.
   - în program se cere rezervarea memoriei de la început, acolo unde se cunoaşte numărul de elemente citite, astfel încât să se evite realocări care nu sunt necesare.

3. NU sunt permise variabile globale.

4. SE CERE separarea citirii datelor de rezolvarea problemei! Citiţi matricile din fişier!

5. Menţionaţi la începutul programului (în comentarii) complexitatea algoritmului (fără citire afişarea rezultatelor).

Evaluare: Rezolvaţi la alegere probleme. Fiecare problemă are alături punctajul aferent. Se acordă pentru această temă suma punctajelor problemelor rezolvate, dar maxim nota 10. Un punct este din oficiu. Se tine cont de criteriile generale de evaluare, prezentate în lista de criterii de pe platformă de e-learning de la prima unitate de învăţare.