Nr. 252 / 15.01.2026

Avizat,
Inspector Școlar General Adjunct,
Prof. Răzvan Delcea VASILE

SUBIECTE PROPUSE PENTRU
EXAMENUL DE ATESTAT PROFESIONAL LA INFORMATICĂ
Programare Pascal/C/C++
Matematică informatică, intensiv informatică 2025-2026

1. Se citește un vector A cu n (n<=1000) elemente numere naturale din fișierul atestat.txt. Ordonați crescător elementele prime și plasați-le la începutul vectorului și apoi descrescător pe cele neprime, în a doua parte a vectorului. Afișați în fișierul ieșire.txt vectorul ordonat ca în cerință.

Atestat.txt

7

33 13 77 19 5 34 100

Ieșire.txt

5 13 19 100 77 34 33

---

2. Se citește din fișierul atestat.txt o matrice pătratică cu n linii și n coloane (n<=100) cu elemente numere naturale din intervalul [0,1000]. Calculați sumele celor 4 zone ale matricii în care este aceasta împărțită de cele două diagonale și afișați în fișierul ieșire.txt litera corespunzatoare zonei cu suma maximă.
Exemplu:

Atestat.txt

4
3 4 1 6
3 4 2 1
5 6 5 9
2 4 3 6
ieșire.txt

E

---

3. Se citește din fișierul atestat.txt o matrice nXm cu elemente întregi. Ștergeți din matrice liniile care nu au elementele ordonate strict crescător sau strict descrescător și afișați în fișierul ieșire.txt matricea rezultată.
Exemplu:

Atestat.txt
4 4
2 1 3 4
6 10 11 13
6 5 4 2
4 3 2 44

Ieșire.txt
6 10 11 13
6 5 4 2

4. Se citesc din fişierul “atestat txt”, de pe prima linie, 2 numere n şi m (1≤n,m≤50). De pe următoarele 2 linii se citesc elementele a două şiruri de lungime n, respectiv m, , numere întregi, de maxim 9 cifre, care sunt ordonate crescător. Să se scrie un program care să construiască în memorie un şir care conţine rezultatul interclasării celor două şiruri. Şirul rezultat se va afişa în fişierul ieșire.txt

Atestat.txt

2 3

4 6

1 2 7

Iesire.txt

1 2 3 4 6 7

---

5. Pe prima linie a fișierului atestat.txt se găsesc numere naturale nenule. Se cere

a. Pentru fiecare număr impar din fișier să se afișeze cel mai mare număr care se poate forma din cifrele lui.

b. Să se verifice daca numerele din fișierul de intrare reprezintă o progresie aritmetică. În caz afirmativ să se afișeze primul element și rația, altfel să se afișeze un mesaj corespunzător.

Atestat.txt

31 38 45 52 59 66

Ieșire.txt

31 38 54 52 95 66

Primul termen este 31 și rația 7

---

6. Fie un tablou bidimensional cu n linii și n coloane. Se cere

a. Să se afișeze transpusa matricei.

b. Să se afișeze câte perechi de numere prime între ele sunt pe diagonala secundară a matricei inițiale.

Datele de intrare se vor citi din fișierul atestat.txt, astfel

-pe prima linie se găsește n

Pe următoarele n linii se găsesc câte n numere ce reprezintă elementele matricei.

Atestat.txt

4

1 2 3 4

5 6 7 8

9 10 11 12

13 14 15 16

Ieșire.txt

1 5 9 13

2 6 10 14

3 7 11 15

4 8 12 16

5 perechi de numere prime sunt pe diagonala secundară a matricei inițiale

---

7. Prima linie a fișierului atestat.txt conține două numere naturale m și n (1<=n, m<=100) iar următoarele m linii câte n numere întregi cu maxim 4 cifre fiecare, separate prin câte un spațiu. Se cere să se afișeze în fișierul ieșire.txt câte din cele m linii din fișier sunt simetrice . Spunem că o linie a fișierului este simetrică dacă elementele egal depărtate de capetele liniei respective sunt egale (primul element de pe linie este egal cu ultimul element al liniei, al doilea cu penultimul etc.)

Exemplu

Atestat.txt are următorul conținut

3 5

15 3 0 3 15

5 9 1 9 5

6 8 20 8 3

Ieșire.txt

2

---

8. Fișierele atestat1.txt și atestat2.txt conțin fiecare numele a 7 persoane, câte un nume pe fiecare linie, fiecare nume având cel mult 15 litere. Știind că în fiecare fișier numele sunt memorate în ordine alfabetică, scrieți un program C++ care să citească din cele două fișiere și să afișeze în fișierul ieșire.txt toate numele din cele două fișiere în ordine alfabetică, separate printr-un spațiu.

Exemplu

Atestat1txt are următorul conținut

Ana

Dana

Daniel

Ene

Mara

Nae

Paul

Exemplu

Atestat2txt are următorul conținut

Angi

Cora

Dora

Horia

Oana

Paul

Tibi

Ieșire.txt

Ana Angi Cora Dana Daniel Dora Ene Horia Mara Nae Oana Paul Paul Tibi

---

9. Fișierul atestat.txt conține pe prima linie un număr natural n(1<=n<=10000) iar pe a doua linie n numere reale separate prin câte un spațiu. Fiecare număr real este format din cel mult 10 cifre, inclusiv partea zecimală. Scrieți programul C++ ce determină cifrele ce nu apar în scrierea nici unui număr real din fișier. Cifrele se vor afișa în fișierul ieșire.txt în ordine crescătoare , separate prin câte un spațiu. În cazul în care toate cifrele sunt utilizate în scrierea numerelor din fișier se va afișa mesajul NICI UNA.

Exemplu

Atestat.txt are următorul conținut

-1.23 36 22.57 208

Ieșire.txt

4 9

---

10. Fișierul atestat.txt conține cel mult 1000 de numere întregi separate prin spații. Numerele din fișier au valori cuprinse între -30000 și 30000. Să se determine și să se afișeze în fișierul ieșire.txt cel mai mare număr din fișier precum și numărul de apariții ale acestuia.

Exemplu

Atestat.txt are următorul conținut

2 7 12 3 8 12 9 5

Ieșire.txt

12 2

---

11. Scrieți un program care creează fișierul text iesire.txt ce conține o linie unică având în ordine descrescătoare toate numerele pare mai mici sau egale cu n, unde n este un număr natural citit de la tastatură(n<=1000).Numerele scrise în fișier vor fi separate prin câte un spațiu.

Se citește n=11

iesire.txt

10 8 6 4 2 0

---

12. Fișierul atestat.txt conține pe prima linie un număr natural nenul (n<=100) și pe următoarea linie n numere reale pozitive ordonate crescător separate prin câte un spațiu.

Să se scrie un program care citește din fișier numărul n și care determină numărul minim de intervale [x, x+1] cu x număr natural a căror reuniune include toate numerele reale din fișier.

Atestat.txt

6

2.3 2.8 5.1 5.7 5.9 6.3

Iesire.txt

3

13. Pe prima linie a fișierului atestat.txt se află un număr natural nenul mai mic sau egal cu 1000 și pe a doua linie un șir de n numere naturale, despărțite prin câte un spațiu fiecare număr fiind format din cel mult patru cifre. Scrieți un program care citește numerele din fișier și afișează mesajul DA dacă elementele pare sunt în ordine crescătoare ,iar elementele impare sunt în ordine descrescătoare și mesajul NU în caz contrar.

Atestat.txt

7

10 1133 12 331 42 1354 221

Afișează DA

---

14. Din fișierul atestat.txt se citesc două numere naturale n și m (n, m<=100) și apoi citește o matrice cu n linii și m coloane, cu elemente numere naturale din intervalul [0,10000].

a) Afișați pe prima linie a fișierului ieșire.txt elementul maxim din matrice.

b) Afișați pe a doua linie a fișierului ieșire.txt câte dintre liniile matricei au elementele în ordine strict crescătoare.

atestat.txt

8 6

1 2 3 4 5 6

4 5 6 7 9 10

3 2 3 2 3 2

9 5 4 3 2 1

4 5 3 4 4 4

9 8 7 6 5 2

3 5 4 3 3 3

Ieșire.txt

10

3

15. Se citesc din fișierul atestat.txt, de pe 2 linii consecutive, 2 numere mari scrise în baza 10, fiecare având cel mult 50 de cifre. Să se scrie un program care să calculeze suma lor, folosind şiruri în care se memorează cifrele numerelor. Rezultatul se va afişa în fişierul ieșire.txt.

16. Se citesc din fişierul atestat.txt , de pe 2 linii consecutive, 2 numere mari scrise în baza 10, fiecare având cel mult 50 de cifre. Să se scrie un program care să compare numerele , folosind şiruri în care se memorează cifrele numerelor. Rezultatul se va afişa în fişierul ieșire.txt.

---

17. Din fișierul atestat.txt se citește un număr n mai mic decât 2000000000 si apoi se citesc n numere naturale. Afișați în fișierul iesire.txt lungimea celei mai lungi secvențe din numerele citite care are proprietatea ca începe si se termina cu aceeași valoare si nu mai conține acea valoare (în afara de primul si ultimul element al secvenței).
Exemplu:atestat.txt
14
3 2 4 3 4 2 3 4 5 6 7 2 5 5
iesire.txt
7
Explicație: cea mai lunga secvență care respecta cerința este 2 3 4 5 6 7 2

---

18. Se citește din fișierul atestat.txt un număr n și un număr q, q din intervalul [2,9]. Verificați dacă n este corect scris în baza q. Să se afișeze da în fișierul ieșire.txt sau nu în caz contrar.

Atestat.txt

1372 9

Ieșire.txt

Da

---

19. Se citește din fișierul atestat.txt un tablou unidimensional cu n <=100 componente numere naturale. Să se afișeze pe primul rând în fișierul ieșire.txt cifrele numărului celui mai mare care se poate obține din cifrele elementului minim și ale celui maxim din vector.

b. Să se afișeze pe al doilea rând din fișierul ieșire câte numere din tablou sunt pătrate perfecte.

Atestat.txt

6

271 109 28 713 14 36

Ieșire.txt

74311

Tabloul are 1 pătrat perfect

---

20. Se citeşte din fişierul atestat.txt, de pe prima linie, un număr n (1≤n≤50). De pe următoarea linie se citesc elementele unui şir de lungime n, numere întregi, de maxim 9 cifre. Să se scrie un program care să realizeze sortarea crescătoare a elementelor şirului folosind metoda inserţiei. Rezultatul se va afişa în fişierul ieșire.txt.

Exemplu:

Atestat.txt

7

3 8 4 1 9 2 4

Ieșire.txt

1 2 3 4 4 8 9

---

21. Se citesc din fişierul atestat.txt , de pe prima linie, 2 numere n şi m (1≤n,m≤50). De pe următoarele 2 linii se citesc elementele a două şiruri de lungime n, respectiv m, , numere întregi, de maxim 9 cifre, care sunt ordonate crescător. Să se scrie un program care să construiască în memorie un şir care conţine rezultatul interclasării celor două şiruri. Şirul rezultat se va afişa în fişierul ieșire.txt.

Exemplu:

Atestat.txt

5 4

1 3 4 7 8

1 2 3 6

Ieșire.txt

1 1 2 3 3 4 6 7 8

---

22. Se citesc din fişierul atestat.txt, de pe prima linie, 2 numere n şi m (1≤n,m≤50). De pe următoarele două linii se citesc elementele a două şiruri de lungime n, respectiv m, numere întregi, de maxim 9 cifre. Elementele şirurilor sunt distincte două cate două. Scrieţi un program, care să construiască în memorie un şir ce va conţine intersecţia celor două şiruri. Rezultatul se va afişa în fişierul ieșire.txt.

Exemplu:

Atestat.txt

5 4

1 3 4 7 8

1 2 3 6

Ieșire.txt

1 3

---

23. Se citesc din fişierul atestat.txt , de pe prima linie, 2 numere n şi m (1≤n,m≤50). De pe următoarele două linii se citesc elementele a două şiruri de lungime n, respectiv m, numere întregi, de maxim 9 cifre. Elementele şirurilor sunt distincte două cate două. Să se scrie un program, care să construiască în memorie un şir ce va conţine reuniunea celor două şiruri. Rezultatul se va afişa în fişierul ieșire.txt.

Exemplu:

Atestat.txt

5 4

1 3 4 7 8

1 2 3 6

Ieșire.txt

1 2 3 4 6 7 8

---

24. Se citeşte din fişierul atestat.txt, de pe prima linie, un număr n (1≤n≤50). De pe următoarea linie se citesc elementele unui şir de lungime n, numere întregi, de maxim 9 cifre. Să se scrie un program care să verifice dacă şirul citit conţine numai elemente distincte.

Exemplu:

Atestat.txt

5

1 3 4 7 8

Ieșire.txt

Da

---

25. Se citeşte din fişierul atestat.txt, de pe prima linie, un număr n (1≤n≤50). De pe următoarea linie se citesc elementele unui şir de lungime n, numere întregi, de maxim 9 cifre. Să se scrie un program care să realizeze sortarea crescătoare a elementelor şirului folosind metoda selecţiei. Rezultatul se va afişa în fişierul ieșire.txt.

Exemplu:

Atestat.txt

7

3 8 4 1 9 2 4

Ieșire.txt

1 2 3 4 4 8 9

---

26. Se consideră un șir de n numere întregi distincte n<=20 și două numere întregi a,b.

Să se afișeze pe ecran toate elementele șirului care nu sunt în intervalul închis ce se poate forma cu numerele a și b.

b. Să se ordoneze crescător numai elementele situate între elementul minim și elementul maxim din șirul dat și să se afișeze pe ecran șirul obținut după ordonare.

atestat.txt

7

3 1 8 7 2 4 -1

5 3

Ieșire.txt

1 8 7 2 -1

3 1 8 2 4 7 -1

27. Se citeşte din fişierul atestat.txt, de pe prima linie, un număr n (1≤n≤50). De pe următoarea linie se citesc elementele unui şir de lungime n, numere întregi, de maxim 9 cifre. Să se scrie un program care să realizeze sortarea crescătoare a elementelor şirului folosind metoda numărării. Se va ţine cont de faptul că în şir poate să apară un element şi de mai multe ori. Rezultatul se va afişa în fişierul ieșire.txt.

Exemplu:

Atestat.txt

7

3 8 4 1 9 2 4

Ieșire.txt

1 2 3 4 4 8 9

---

28. Se citeşte din fişierul atestat.txt, de pe prima linie, un număr natural n (1≤n≤50). De pe următoarea linie se citesc n numere naturale de maxim 9 cifre.. Să se scrie un program care să calculeze cel mai mare divizor comun al celor n numere de pe linia a doua a fişierului. Rezultatul se va afişa în fişierul ieșire.txt.

Exemplu:

Atestat.txt

5

14 8 22 4 32

Ieșire.txt

2

---

29. Din fisierul numere.în se citeste un numar natural n (n<=100000) si apoi n numere naturale cu cel mult 9 cifre fiecare. Afisati în fisierul numere.out cea mai lunga secventa de cifre identice care se obtine prin lipirea celor n numere. Daca exista mai multe secvente de lungime maxima, atunci se va afisa cea mai din dreapta.
numere.în
12
36 611 1111 12 11000000 0 0 0 0 0 3333 43219
numere.out
00000000000

30. Din fisierul date.în se citeste un numar natural n si apoi se citesc n numere naturale cu cel mult 2 cifre fiecare. Afisati în fisierul date.out lungimea celei mai lungi secvente din numerele citite care are proprietatea ca incepe si se termina cu aceeasi valoare.
Exemplu:
date.în
13
3 2 13 10 2 10 12 6 7 5 10 2 13
date.out
11
Explicatie: cea mai lunga secventa ceruta incepe si se termina cu 2 si contine 11 numere.

31:
Fișierul **`atestat.in`** conține pe prima linie un număr natural **`n, 2≤n≤100`**, iar pe cea de-a doua linie **`n`** numere naturale de cel mult **`9`** cifre fiecare, separate prin câte un spațiu.

Se consideră subprogramele:

- **`p_cifra`** (implementat recursiv) cu un singur parametru **`y`**, număr natural de cel mult 9 cifre. Subprogramul returnează cifra semnificativă (prima cifră) a numărului **`y`**.
- **`sortare`** cu doi parametri: **`v`** un tablou unidimensional cu cel mult **`100`** de componente care

memorează fiecare câte un număr natural de cel mult **9** cifre și **n** numărul efectiv de componente ale tabloului **v, 2≤n≤100.** Subprogramul ordonează descrescător elementele tabloului **v**.

Cerințe:

a. Scrieți definiția completă a subprogramului **p_cifra**;

b. Scrieți definiția completă a subprogramului **sortare**;

c. Scrieți un program care citește datele din fișierul **atestat.in** și, utilizând apeluri utile ale subprogramelor **p_cifra** și **sortare**, determină și scrie în fișierul **atestat.out, ordonate descrescător,** valorile aflate pe cea de-a doua linie a fișierului **atestat.in** care au cifra semnificativă un număr pătrat perfect. În cazul în care nu există astfel de numere, programul va scrie în fișierul **atestat.out** mesajul **"nu exista".**

Exemplu:

atestat.in	atestat.out
9 19 25 5632 9872 48903 33 17634 90 3452	48903 17634 9872 90 19

32

Prin *înjumătățirea* unui număr natural se înțelege înlocuirea fiecărei cifre pare cu jumătatea ei. De exemplu, prin înjumătățirea numărului **5622** se obține numărul **5311**.

Fișierul **atestat.in** conține pe prima linie un număr natural **n** (**2≤n≤100**), iar pe a doua linie, un șir de **n** numere naturale cu cel mult **9** cifre fiecare.

Se consideră subprogramele:

- **verif** care are un singur parametru **x** (număr natural cu maxim **9** cifre) și returnează valoarea **1** dacă toate cifrele numărului **x** sunt pare sau valoarea **0**, în caz contrar.
- **modif** care are ca unic parametru numărul natural **x**. Subprogramul înjumătățește valoarea lui **x** (conform definiției de mai sus) și furnizează numărul modificat prin intermediul aceluiași parametru.

Cerințe:

a. Să se scrie definiția completă a subprogramului **verif**;

b. Să se scrie definiția completă a subprogramului **modif**;

c. Să se scrie un program care citește din fișierul **atestat.in** numărul **n** și cele **n** elemente ale tabloului unidimensional **v** și, folosind apeluri utile ale subprogramelor **verif** și **modif**, determină înjumătățirea (conform definiției de mai sus) a elementelor tabloului care au toate cifrele pare. Programul scrie pe prima linie a fișierului **atestat.out** elementele tabloului modificat. Elementele tabloului care conțin cel puțin o cifră impară nu se modifică.

Exemplu:

atestat.in	atestat.out
5 63 8644 1024 102 2048	63 4322 1024 102 1024

33

Fișierul **atestat.in** conține pe prima linie un număr natural **n**, **2≤n≤100** și pe a doua linie **n** numere naturale cu cel puțin **2** și cel mult **6** cifre, separate printr-un spațiu.

Se consideră subprogramele:

- **inversareCifre** cu un parametru **x**, prin intermediul căruia primește un număr natural format din cel mult **6** cifre. Subprogramul modifică valoarea lui **x**, inversând ordinea cifrelor lui, cu excepția primei cifre care rămâne în aceeași poziție. De exemplu, pentru valoarea **21754** a parametrului **x**, în urma executării subprogramului, valoarea furnizată prin parametrul **x** va fi **24571**.
- **nrDivizori** cu un parametru **x**, prin intermediul căruia primește un număr natural nenul, format din cel mult **6** cifre. Subprogramul returnează numărul divizorilor parametrului **x**.

Cerințe:

a. Să se scrie definiția completă a subprogramului **inversareCifre;**
b. Să se scrie definiția completă a subprogramului **nrDivizori;**
c. Să se scrie un program care citește din fișierul **atestat.in** numărul **n** și cele **n** numere naturale, iar apoi, folosind apeluri utile ale subprogramelor **inversareCifre** și **nrDivizori**, modifică fiecare număr din șir care are mai mult de **4** divizori, inversând ordinea tuturor cifrelor lui, cu excepția primei cifre care rămâne în aceeași poziție și scrie în fișierul **atestat.out**, pe prima linie, toate numerele din șirul modificat. Dacă nu există numere cu mai mult de **4** divizori se va scrie în fișier, mesajul **"nu au fost facute modificari"**.

Exemple:

atestat.in	atestat.out
6 245 1763 23 1876 218 492873	254 1763 23 1678 218 492873
6 23 6 9 17 25 101	nu au fost facute modificari

34:

În fișierul **atestat.in**, pe prima linie se află un număr natural **n (1≤n≤100)**, iar pe a doua linie se află **n** numere naturale distincte cu cel mult 4 cifre fiecare. În fișier există cel puțin un număr care are cifre de parități diferite.

Se consideră subprogramele:

- **sterge** cu trei parametri: **v**, un tablou unidimensional cu maxim **100** de elemente, numere naturale cu cel mult **4** cifre fiecare, **n** un număr natural **(1≤n≤100)** care reprezintă numărul efectiv de elemente ale tabloului primit prin intermediul parametrului **v**, **x** un număr natural cu cel mult **4** cifre. Subprogramul șterge, în cazul în care găsește, elementul cu valoarea **x** din tabloul **v**, actualizând corespunzător valoarea parametrului **n**. Tabloul modificat este furnizat tot prin parametrul **v**.
- **cif** cu un parametru **n,** număr natural cu maxim **4** cifre. Subprogramul verifică dacă numărul **n** are toate cifrele de aceeași paritate și returnează valoarea **1** altfel returnează valoarea **0**.

Cerințe:

a. Scrieți definiția completă a subprogramului **sterge**;
b. Scrieți definiția completă a subprogramului **cif**;
c. Scrieți un program care citește din fișierul **atestat.in** un număr natural **n**, ce reprezintă numărul de elemente ale unui tablou unidimensional și **n** numere naturale distincte,

reprezentând elementele tabloului. Programul șterge din tablou toate numerele care au cifrele de aceeași paritate, folosind apeluri utile ale subprogramelor **sterge** și **cif**. Elementele tabloului modificat se scriu, separate prin câte un spațiu, pe prima linia a fișierului **atestat.out.** În cazul în care nu s-a găsit niciun astfel de număr, în fișierul **atestat.out** se scrie mesajul **"nu exista".**

Exemplu

atestat.in	atestat.out
7 37 132 7 2785 86 490 18	132 2785 490 18

35:

Fișerul **atestat.in** conține pe prima linie un număr natural **n, 2≤n≤100**, iar pe a doua linie **n** numere reale.

Se consideră subprogramele:

- **citeste** cu doi parametri: **v,** un tablou unidimensional cu cel mult **100** elemente numere reale și **n,** un număr natural **(2≤n≤100)**. Subprogramul citește din fișierul **atestat.in** și furnizează prin cei doi parametri numărul de elemente **n** și cele **n** elemente ale tabloului unidimensional **v**.

- **pozmax** cu trei parametri care primește prin intermediul parametrului **v** un tablou unidimensional cu cel mult **100** de elemente numere reale, prin parametrii **p1** și **p2** **(1<=p1, p2<=n)** primește două poziții din **v** și returnează poziția pe care se află valoarea maximă a elementelor din **v**, situate pe poziții cuprinse între **p1** și **p2**, inclusiv**.**

Cerințe:

a. Scrieți definiția completă a subprogramului **citeste**;
b. Scrieți definiția completă a subprogramului **pozmax**;
c. Scrieți un program care, utilizând apeluri utile ale subprogramelor **citeste** și **pozmax**, citește datele din fișierul **atestat.in** și scrie în fișierul **atestat.out,** pe prima linie, separate prin câte un spațiu, elementele tabloului unidimensional în ordine descrescătoare .

Exemplu:

atestat.in	atestat.out
5 2.4 1.9 8.1 4.4 10.25	10.25 8.1 4.4 2.4 1.9

36

Fişierul **atestat.in** conţine cel mult **100** de numere naturale cu cel mult patru cifre, toate numerele fiind scrise pe o singură linie, separate prin câte un spaţiu. *Valorile din fişier sunt ordonate descrescător.*

Se consideră subprogramele:

- **construire** cu doi parametri: **v**, un tablou unidimensional cu elemente numere naturale și **n,** un număr natural **(2≤n≤100)** reprezentând numărul de elemente ale tabloului **v**. Subprogramul citește numerele din fișierul **atestat.in** și furnizează, prin intermediul

parametrului **v,** un tablou unidimensional ce va conţine doar acele numere din fişier care au exact trei cifre, precum şi numărul de elemente ale acestuia, prin parametrul **n**. *Fişierul conţine cel putin un număr cu 3 cifre.*

- **cautare** cu trei parametri: **v** un tablou unidimensional cu elemente numere naturale, **n** un număr natural **(2≤n≤100)** reprezentând numărul de elemente ale tabloului **v** și **x** un număr natural. Subprogramul returnează, utilizând algoritmul de căutare binară, poziţia pe care se găsește valoarea **x** în vectorul **v** sau valoarea **-1**, în caz contrar.

Cerinţe:

a. Scrieţi definiţia completă a subprogramului **construire**;

b. Scrieţi definiţia completă a subprogramului **cautare**;

c. Scrieţi un program care, utilizând apeluri utile ale subprogramelor **construire** şi **cautare**, scrie în fișierul **atestat.out** poziţia în vectorul v pe care se găseşte un număr natural x, cu exact trei cifre, citit de la tastatură sau mesajul "nu exista" dacă numărul x nu se află printre elementele tabloului v.

Exemplu:

atestat.in	**atestat.out**
1204 991 234 102 79 **și de la tastatură se citește pentru x valoarea 105**	**nu exista**

37

Fişierul **atestat.in** conţine pe prima linie un număr natural **n** (**2≤n≤20**)**,** iar pe următoarele **n** linii, câte **n** numere naturale cu cel mult **6** cifre, separate printr-un spațiu.

Se consideră subprogramele:

- **elimColoana** cu trei parametri: **n (n≤20)** număr natural, **a,** tablou bidimensional cu **n** linii și **n** coloane, cu elemente numere naturale și **k (k≤n)** număr natural, reprezentând un indice de coloană din matricea **a**. Subprogramul elimină coloana de indice **k** din tabloul bidimensional **a**.
- **cifreImpare** cu un singur parametru **x**, număr natural cu cel mult **6** cifre, verifică dacă toate cifrele numărului **x** sunt impare, caz în care returnează valoarea **1**, altfel returnează valoarea **0**.

Cerințe:

a. Să se scrie definiția completă a subprogramului **elimColoana;**

b. Să se scrie definiția completă a subprogramului **cifreImpare;**

c. Să se scrie un program care citește din fișierul **atestat.in** un număr **n** și elementele unui tablou bidimensional, cu **n** linii și **n** coloane și, folosind apeluri utile ale subprogramelor **elimColoana** și **cifreImpare**, elimină coloana care are proprietatea că numărul de elemente alcătuite doar din cifre impare, este egal cu indicele coloanei. De exemplu, dacă pe coloana cu indicele **k** există **k** numere formate doar din cifre impare, atunci această coloană va fi eliminată. Dacă există mai multe coloane cu această proprietate, se va elimina doar coloana cu indicele cel mai mic. În fișierul **atestat.out,** se scrie matricea obținută în urma eliminării făcute conform cerinței; fiecare linie a tabloului se va scrie pe o linie a fișierului, iar elementele de pe aceeași linie, separate printr-un spațiu. Dacă niciuna dintre coloanele matricei nu va fi eliminată, în fișier se va scrie mesajul **"matrice nemodificata".**

Exemplu:

atestat.in	atestat.out	Explicații
4 12345 57 2 39 561 8 379 5 1157 9 32 199 595 3410 69 11	12345 2 39 561 379 5 1157 32 199 595 69 11	• coloana 1 are 2 elemente cu toate cifrele impare, deci nu va fi ștearsă • coloana 2 are 2 elemente cu toate cifrele impare, deci va fi ștearsă. Celelalte coloane rămân nemodificate.
4 34 57 2 39 561 8 379 52 112 92 32 199 2 3410 79 11	matrice nemodificată	

38

O matrice pătratică se numește *inferior triunghiulară* dacă are toate elementele aflate (strict) deasupra diagonalei principale egale cu **0**. Determinantul unei matrice inferior triunghiulare este egal cu produsul elementelor aflate pe diagonala principală a matricei.

Fișierul **atestat.in** conține pe prima linie un număr natural **n** (**2≤n≤20**), iar pe următoarele **n** linii câte **n** numere reale ce reprezintă elementele unei matrice pătratice, de dimensiune **n**.

Se consideră subprogramele:

- **inftr** cu doi parametri: un tablou bidimensional **a** cu elemente numere reale (maxim **20** de linii și **20** de coloane) și un număr natural **n** (**2≤n≤20**) ce reprezintă dimensiunea efectivă a tabloului **a**. Subprogramul returnează valoarea **1** dacă tabloul reprezintă o matrice inferior triunghiulară sau valoarea **0**, în caz contrar.
- **produs** (implementat recursiv) cu **doi** parametri: un tablou bidimensional **a** cu elemente numere reale (maxim **20** de linii și **20** de coloane) și un număr natural **n** (**2≤n≤20**) ce reprezintă dimensiunea efectivă a tabloului **a**. Subprogramul returnează produsul elementelor aflate pe diagonala principală a tabloului **a**.

Cerințe:

a. Să se scrie definiția completă a subprogramului **inftr**;
b. Să se scrie definiția completă a subprogramului **produs**;
c. Să se scrie un program care citește din fișierul **atestat.in** dimensiunea și elementele unei matrice pătratice și, folosind apeluri utile ale subprogramelor **inftr** și **produs**, scrie pe prima linie a fișierului **atestat.out** mesajul **"da"**, dacă matricea este inferior triunghiulară sau mesajul **"nu"**, în caz contrar. Pe a doua linie a fișierului se afișează valoarea determinantului matricei (dacă este inferior triunghiulară) sau mesajul **"nedeterminat"** (dacă matricea nu este inferior triunghiulară).

Exemplu:

atestat.in	atestat.out
3 3 0 0 1 2 0 1 0 1	da 6

39

Subiectul nr. 9:
Fișierul **atestat.in** conține pe prima linie un număr natural **n, 2≤n≤20**, iar pe următoarele **n** linii, câte **n** numere naturale nenule cu cel mult **9** cifre fiecare, separate prin câte un spațiu.

Se consideră subprogramele:

- **prim** cu un singur parametru **x**, număr natural nenul cu cel mult **9** cifre. Subprogramul returnează valoarea **1** dacă **x** este număr prim, respectiv valoarea **0** dacă **x** nu este număr prim;
- **contorizare** (implementat recursiv) cu un singur parametru **y**, număr natural nenul cu cel mult **9** cifre. Subprogramul returnează numărul de cifre pare ale lui **y**.

Cerințe:

a. Scrieți definiția completă a subprogramului **prim**;
b. Scrieți definiția completă a subprogramului **contorizare**;
c. Scrieți un program care citește datele din fișierul **atestat.in** și, utilizând apeluri utile ale subprogramelor **prim** și **contorizare,** determină și scrie în fișierul **atestat.out,** pe prima linie, numărul prim din matrice care are în scrierea sa cele mai multe cifre pare, iar pe a doua linie numărul de cifre pare. Dacă în matrice există mai multe valori prime cu același număr maxim de cifre pare, programul va scrie în fișier cea mai mică dintre aceste valori și numărul cifrelor pare care apar în scrierea sa. Dacă în matrice nu există niciun număr prim sau dacă numerele prime nu au cifre pare, programul va scrie în fișierul **atestat.out** mesajul **"nu exista"**.

Exemplu:

atestat.in
```
5
12  23  28   29  90
2  54  101  121  7
1096  41  9  607  102
220  34  32 421  6
```

atestat.out
```
421  2
```

40
În fișierul **atestat.in**, pe prima linie, se află două numere **n** și **m (2≤n,m≤20)** reprezentând numărul de linii respectiv numărul de coloane ale unui tablou bidimensional, iar pe următoarele **n** linii se află câte **m** numere naturale mai mici sau egale cu 200 ce reprezintă elementele tabloului.

Se consideră subprogramele:

- **fibo** care primește prin intermediul parametrului **n** un număr natural **(1≤n≤200)** și furnizează prin intermediul parametrului **x** o valoare naturală reprezentând cel mai mic număr mai mic sau egal cu **n** care face parte din șirul lui Fibonacci.
- **divprim** care primește prin intermediul parametrului **a** un număr natural **(2≤a≤200)** și returnează cel mai mic divizor prim al valorii parametrului **a**.

Cerințe:

a. Scrieți definiția completă a subprogramului **fibo;**
b. Scrieți definiția completă a subprogramului **divprim;**
c. Scrieți programul care citește datele din fișierul **atestat.in** și, folosind apeluri utile ale subprogramelor **fibo** și **divprim,** scrie în fișierul **atestat.out** media aritmetică a elementelor din tablou care sunt numere prime și fac parte din șirul lui Fibonacci; în cazul în care nu s-a găsit niciun astfel de număr se scrie în fișier mesajul **"nu exista"**.

Exemplu:

```
atestat.in
3 4
 9    7  21   4
13    8   2   6
28    3  10   5
```

```
atestat.out
5.75

Explicație: numerele prime care fac
parte din șirul lui Fibonaci sunt 13,
3, 5
```

41

În fișierul **atestat.in**, pe prima linie se află un număr **n (2≤n≤20)** reprezentând numărul de linii și de coloane ale unui tablou bidimensional, iar pe următoarele **n** linii se află câte **n** numere naturale, cu cel mult **9** cifre fiecare, reprezentând elementele tabloului.

Se consideră subprogramele:

- **oglindit**, care primește prin intermediul parametrului **x** un număr natural **(1≤x≤10^9)** și returnează valoarea valoarea obținută prin oglindirea lui **x**. De exemplu, dacă valoarea lui **x** este 123, subprogramul va returna valoarea 321.
- **maxim**, cu trei parametri care primește prin intermediul parametrului **a** o matrice pătratică cu cel mult **20** de linii și **20** de coloane, prin intermediul parametrului **n** un număr natural reprezentând numărul efectiv de linii și coloane ale matricei **a** și furnizează prin parametrul **p** cel mai mare palindrom care se află pe una dintre diagonalele matricei **a**, respectiv **-1**, dacă pe diagonalele matricei nu se află niciun palindrom.

Cerințe:

a. Scrieți definiția completă a subprogramului **oglindit;**
b. Scrieți definiția completă a subprogramului **maxim;**
c. Scrieți programul care citește datele din fișierul **atestat.in** și scrie în fișierul **atestat.out**, folosind apeluri utile ale subprogramelor **oglindit** și **maxim**, cel mai mare palindrom care se află pe una dintre diagonalele matricei, iar în cazul în care nu s-a găsit niciun astfel de număr se scrie în fișier mesajul **"nu exista"**.

Exemplu:

```
atestat.in
3
 9    7   21
33    8    2
22    3   10
```

```
atestat.out
22

Explicație: numerele palindrom care se află pe
una dintre diadonalele matricei sunt: 9, 8, 22
```

42

Subiectul nr. 12:

În fișierul **atestat.in,** pe prima linie se află un număr **n (2≤n≤20)** reprezentând numărul de linii și de coloane ale unui tablou bidimensional, iar pe următoarele **n** linii se află câte **n** numere naturale de maxim 6 cifre, ce reprezintă elementele tabloului.

Se consideră subprogramele:

- **interschimbL** cu patru parametri: **a,** reprezentând un tablou bidimensional cu cel mult 20 de linii și 20 de coloane; **n,** reprezentând numărul de linii, respectiv numărul de coloane; **k1** și **k2,** două numere naturale, reprezentând numărul de ordine a două linii din tablou. Subprogramul interschimbă elementele de pe linia **k1** cu elementele de pe linia **k2**.
- **interschimbC** cu patru parametri: **a,** reprezentând un tablou bidimensional cu cel mult 20

de linii și 20 de coloane; **n,** reprezentând numărul de linii, respectiv numărul de coloane; **c1** și **c2,** două numere naturale, reprezentând numărul de ordine a două coloane din tablou. Subprogramul interschimbă elementele de pe coloana **c1** cu elementele de pe coloana **c2**.

Cerințe:

a. Scrieți definiția completă a subprogramului **interschimbL;**
b. Scrieți definiția completă a subprogramului **interschimbC;**
c. Scrieți un program care citește din fisierul **atestat.in** numărul **n**, tabloul bidimensional cu **n*n** elemente și, folosind apeluri utile ale subprogramelor **interschimbL** și **interschimbC** ordonează crescător elementele diagonalei principale prin interschimarea liniilor și a coloanelor. Tabloul modificat se va scrie în fișierul **atestat.out,** fiecare linie a tabloului se va scrie pe o linie din fișier, iar elementele de pe aceeași linie vor fi separate printr-un spațiu.

Exemplu:

atestat.in	atestat.out
6	
9 1 1 1 1 1	1 2 2 2 2 2
3 8 3 3 3 3	3 4 3 3 3 3
4 4 7 4 4 4	6 6 5 6 6 6
6 6 6 5 6 6	4 4 4 7 4 4
3 3 3 3 4 3	3 3 3 3 8 3
2 2 2 2 2 1	1 1 1 1 1 9

43

Fișierul **atestat.in** conține pe prima linie un număr natural nenul **n, 0<n<10**, iar pe fiecare dintre următoarele **n** linii, câte o propoziție. Fiecare propoziție este formată din maximum **255** de caractere, litere mici ale alfabetului englez și spații.

Se consideră subprogramele:

- **vocale** cu un singur parametru: **prop** o propoziție formată din maximum **255** de caractere, litere mici ale alfabetului englez și spații. Subprogramul returnează numărul de vocale conținute de propoziția **prop.** Se consideră vocale literele: **a,e,i,o,u.**
- **cuvant** cu doi parametri: **prop** prin intermediul căruia primește o propoziție formată din maximum **255** de caractere, litere mici ale alfabetului englez și spații și **cuv** prin intermediul căruia furnizează primul cel mai lung cuvânt din propoziția **prop.**

Cerințe:

a. Scrieți definiția completă a subprogramului **vocale**;
b. Scrieți definiția completă a subprogramului **cuvant**;
c. Scrieți un program care citește datele din fișierul **atestat.in** și, utilizând apeluri utile ale subprogramelor **vocale** și **cuvant**, construiește în memorie o propozitie în care primul cuvânt este primul cel mai lung cuvânt din prima propoziție, al doilea cuvânt este primul cel mai lung cuvânt din propoziția a doua etc. și scrie în fișierul **atestat.out,** pe prima linie, propoziția obținută iar pe a doua linie numărul de vocale care apar în această propoziție. În propoziția nou formată, cuvintele sunt separare printr-un singur spațiu. Dacă propoziția obținută nu conține vocale, pe linia a doua a fișierului programul va scrie mesajul **"fara vocale"**.

Exemplu:

atestat.in	atestat.out
3 noi doi si voi a sosit fata pe care o asteptam in vacanta mergem la munte	noi asteptam vacanta 8

44

Fișierul **atestat.in** conține pe prima linie un număr natural **n** (**1≤n≤100**), iar pe următoarele **n** linii câte un cuvânt format din maxim **20** de caractere, litere mici și mari ale alfabetului englez.

Se consideră subprogramele:

- **nrlit** care primește ca parametru un șir de caractere **s** și returnează numărul literelor mari din șirul **s**.
- **trans** care are ca parametru șirul de caractere **s**, format numai din litere mari sau mici. Subprogramul are rolul de a transforma șirul **s** astfel încât prima literă să fie literă mare iar restul literelor să fie litere mici.

Cerințe:

a. Să se scrie definiția completă a subprogramului **nrlit**;
b. Să se scrie definiția completă a subprogramului **trans**;
c. Să se scrie un program care citește din fișierul **atestat.in** numărul **n** și cele **n** cuvinte și scrie pe prima linie a fișierului **atestat.out**, separate prin câte un spațiu, cuvintele transformate prin apelul subprogramului **trans**, iar pe a doua linie a fișierului, numărul total de litere mari din fișierul **atestat.in,** prin apelul subprogramului **nrlit**.

Exemplu:

atestat.in	atestat.out
4 Vara CARe a tReCuT	Vara Care A Trecut 7

45

Fișierul **atestat.in** conține, dispuse pe mai multe linii, cel mult un milion de caractere (litere mari și mici ale alfabetului englez, cifre și caractere speciale).

Se consideră subprogramele:

- **cifra** cu un singur parameru: **c** de tip caracter. Subprogramul returnează valoarea **1** dacă **c** este caracter cifră**,** respectiv valoarea **0** dacă **c** nu este caracter cifră.
- **numar** cu doi parametri: **n** un număr întreg cu cel mult **9** cifre și **cif** o cifră. Subprogramul determină modificarea valorii parametrului **n** prin lipirea cifrei **cif** la sfârșitul numărului. De exemplu, dacă **n** are valoarea **12**, în urma alelului **numar(n,3)** , **n** va avea valoarea **123**.

Cerințe:

a. Scrieți definiția completă a subprogramului **cifra**;
b. Scrieți definiția completă a subprogramului **numar**;
c. Scrieți un program care citește datele din fișierul **atestat.in** și, utilizând apeluri utile ale

subprogramelor **cifra** și **numar**, determină și scrie pe prima linie a fișierului **atestat.out** numărul obținut din cifrele care apar în fișierul **atestat.in**. Acest număr va conține, de la stânga spre dreapta, mai întâi cifrele impare în ordine crescătoare și apoi cifrele pare, așezate în ordine descrescătoare. Numărul obținut are cifrele distincte două câte două. În cazul în care, în fișierul **atestat.in** nu există cifre, programul va scrie în fișierul **atestat.out** mesajul **"nu exista"**.

Exemplu:

atestat.in	atestat.out
w e r r t y u h f d s d f h y i u h n 2 7 6 5 6 6 6 0 0 d f g h j k l a s d r v b g t t t y y a a q w v c 3 5 8 *) n & ! n s	3578620

46

În fișierul **atestat.in,** pe prima linie se află un număr natural **n (2≤n≤30),** iar pe următoarele **n** linii câte un cuvânt format din cel mult **100** de litere mici ale alfabetului englez.

Se consideră subprogramele:

- **citire** cu doi parametri: **cuv** un tablou bidimensional care poate reține cel mult **30** de cuvinte, pe rânduri diferite, fiecare cuvânt cu o lungime maximă de **100** de caractere; **n** reprezentând numărul liniilor din tablou. Subprogramul citește cuvintele din fișierul **atestat.in** și furnizează datele prin cei doi parametri definiți mai sus;
- **stergere** cu doi parametri: **s** prin care primește un șir de cel mult **100** de caractere și **c**, prin care primește un caracter. Subprogramul determină modificarea șirului **s**, eliminând toate aparițiile caracterului **c** și returnează numărul ștergerilor efectuate.

Cerințe:

a. Scrieți definiția completă a subprogramului **citire**;
b. Scrieți definiția completă a subprogramului **stergere**;
c. Scrieți un program care, folosind apeluri utile ale subprogramelor **citire** și **stergere**, citește datele din fișierul **atestat.in** și scrie în fișierul **atestat.out**, pe prima linie, toate literele comune celor **n** cuvinte. Fiecare literă se va scrie o singură dată, iar literele vor fi separate printr-un spațiu. Dacă nu există litere comune între cele **n** cuvinte, se va scrie în fișierul **atestat.out** mesajul **"nu exista"**.

Exemplu:

atestat.in	atestat.out	Explicație
4 evantai variabile vacante revoltator	a e v	Literele se afișează nu neapărat în această ordine

47

În fișierul **atestat.in** se află un text format din cel mult **50** de cuvinte, fiecare cuvânt având cel mult **30** de litere mici ale alfabetului englez. Cuvintele sunt separate printr-un singur spațiu.

Se consideră subprogramele:

- **prefixe** cu un singur parametru **s**, reprezentând adresa unui șir de cel mult **30** de caractere. Subprogramul afișează toate prefixele șirului **s** în ordinea crescătoare a lungimii lor. De exemplu, prefixele șirului **examen** sunt**: e ex exa exam exame examen**.
- **sufixe** cu un singur parametru **s**, reprezentând adresa unui șir de cel mult **30** de caractere. Subprogramul afișează toate sufixele șirului **s** în ordinea crescătoare a lungimii lor; De exemplu, sufixele șirului **examen** sunt**: n en men amen xamen examen.**

Cerințe:

a. Scrieți definiția completă a subprogramului **prefixe**;
b. Scrieți definiția completă a subprogramului **sufixe**;
c. Scrieți un program care citește cuvintele din fișierul **atestat.in,** determină primul cuvânt de lungime minimă și ultimul cuvânt de lungime maximă și folosind apeluri utile ale subprogramelor **prefixe** și **sufixe**, scrie în fișierul **atestat.out**: pe prima linie, primul cuvânt de lungime minimă, urmat de toate prefixele acestui cuvânt, separate prin câte un spațiu; pe a doua linie, ultimul cuvânt de lungime maximă, urmat toate sufixele acestui cuvânt, separate prin câte un spațiu.

Exemplu:

atestat.in	atestat.out
atestat carte recreatie vacanta sport spectacol	carte c ca car cart carte spectacol l ol col acol tacol ctacol ectacol pectacol spectacol

48

Fișierul **atestat.in** conține pe prima linie un număr natural **n**, **2≤n≤100**, iar pe următoarele **n** linii, separate prin câte un spațiu, câte două numere întregi **x** și **y**, de cel mult **9** cifre fiecare, reprezentând numărătorul (**x**) și numitorul (**y**) unei fracții algebrice.

Declararea alăturată este utilizată pentru a memora numărătorul și numitorul unei fracții algebrice, în această ordine.

```
struct fractie
  {  int x,y; };
```

Se consideră subprogramele:

- **cmmdc** - cu doi parametri **a** și **b,** două numere întregi de cel mult **9** cifre fiecare. Subprogramul returnează cel mai mare divizor comun al numerelor **a** și **b**.
- **suma** - cu doi parametri **f** și **g,** de tip fracție. Subprogramul returnează suma celor două fracții.

Cerințe:

a. Scrieți definiția completă a subprogramului **cmmdc**;
b. Scrieți definiția completă a subprogramului **suma**;
c. Scrieți un program care citește datele din fișierul **atestat.in** și, utilizând apeluri utile ale subprogramelor **cmmdc** și **suma,** scrie pe prima linie a fișierului **atestat.out** suma fracțiilor ireductibile din fișierul **atestat.in**, sub forma **x/y** (fracție ireductibilă), iar pe următoarele linii, scrie fracțiile ireductibile sub forma **x/y**, câte una pe linie. Dacă fișierul **atestat.in** nu conține fracții ireductibile, în fișierul **atestat.out** programul va scrie mesajul **"nu exista"**.

Exemplu:

atestat.in	atestat.out
7 2 3 4 16 7 15 3 7 12 28 34 68 9 20 5 20	169/84 2/3 7/15 3/7 9/20

49

Pentru memorarea unui număr complex se utilizează structura alăturată. Astfel, câmpul **a** reprezintă partea reală, iar câmpul **b** reprezintă partea imaginară a unui număr complex.

```
struct complexi
  { float a,b;};
```

Se consideră următoarele subprograme:

- **modul** care primește prin intermediul parametrului **x** un număr complex și returnează modulul acestui număr complex
- **suma** care primește prin intermediul parametrilor **x** și **y** două numere complexe și returnează suma lor.

Cerințe:

a. Să se scrie definiția completă a subprogramului **modul;**

b. Să se scrie definiția completă a subprogramului **suma;**

c. Scrieți programul care citește din fișierul **atestat.in** un număr natural **n (1≤n≤25),** ce reprezintă numărul de elemente ale unui tablou unidimensional cu elemente numere complexe, iar de pe următoarele **n** linii câte două numere reale **a** și **b,** reprezentând partea reală și partea imaginară a numerelor complexe ale tabloului și scrie în fișierul **atestat.out,** folosind apeluri utile ale subprogramelor **modul** și **suma,** suma numerelor complexe care au modulul un număr întreg. În cazul în care nu s-a găsit niciun astfel de număr, în fișierul **atestat.out** se va scrie mesajul **"nu exista"**.

Exemplu:

atestat.in	atestat.out
4 3.0 4.0 1.5 2.7 23.0 9.4 6.0 8.0	9 12

50

Fişierul **atestat.in** conține pe prima linie un număr natural **n (2≤n<100),** iar pe fiecare din următoarele **n** linii, separate prin câte un spaţiu, câte două numere reale reprezentând coordonatele carteziene ale unui punct din plan.

Pentru memorarea coordonatelor carteziene (abscisa și ordonata) ale unui punct din plan se va utiliza declararea alăturată.

```
struct punct
{ float x,y; };
```

Se consideră subprogramele:

- **distanta** cu doi parametri de tipul punct (definit mai sus) prin intermediul cărora primeşte coordonatele a două puncte din plan şi returnează distanța dintre cele două puncte.
- **arie** cu doi parametri de tipul punct (definit mai sus) prin intermediul cărora primește coordonatele carteziene a două puncte din plan reprezentând vârfuri opuse ale unui dreptunghi cu laturile paralele cu axele Ox și Oy şi returnează aria dreptunghiului.

Cerinţe:

a. Scrieţi definiţia completă a subprogramului **distanta ;**

b. Scrieţi definiţia completă a subprogramului **arie;**

c. Scrieţi un program care citeşte de pe prima linie a fișierului **atestat.in** un număr natural **n (2≤n<100)**, iar de pe următoarele linii coordonatele celor **n** puncte din plan. Cel puțin două dintre punctele din fișier determină un segment care **nu** este paralel cu axele. Prin apeluri utile ale subprogramelor **distanta** şi **arie**, programul va calcula aria maximă a unui dreptunghi cu laturile paralele cu axele Ox și Oy, care are o diagonală determinată de două dintre punctele citite din fișier. Programul va scrie pe ecran aria maximă și numerele de ordine ale punctelor care determină diagonala dreptunghiului de arie maximă.

Exemplu:

atestat.in	Se va afișa pe ecran
4	4 1 3
1 1	Explicație: aria maximă care se poate obține este 4 și
2 3	este aria dreptunghiului care are o diagonală formată
3 3	din punctele de coordonate (1,1) și (3,3), adică
4 2	punctele cu numerele de ordine 1 și 3

51. Se consideră un graf neorientat **G** cu **n** vârfuri **(n∈N, 2≤n≤30)** etichetate cu numerele distincte: **1,2,...,n**.

Fişierul **atestat.in** conţine mai multe linii. Pe prima linie a fişierului este scris numărul natural **n** reprezentând numărul de vârfuri ale grafului **G,** iar pe următoarele linii, perechi de numere naturale, separate prin câte un spaţiu, reprezentând muchiile distincte ale grafului **G**.

Se consideră subprogramele:

- **nrElem1** cu trei parametri: **n,**un număr natural, **n≤30**, **v**, un tablou unidimensional cu **n** elemente **0** sau **1** și **s,** un număr natural. Subprogramul determină numărul de elemente egale cu **1** din tabloul unidimensional **v** și furnizează acest rezultat prin intermediul parametrului **s**.
- **sumaElem** cu doi parametri: **n** un număr natural nenul (**2≤n≤30**) și **a** un tablou bidimensional, pătratic, cu **n** linii și **n** coloane și elemente numere naturale. Subprogramul returnează suma elementelor tabloului bidimensional **a**.

Cerințe:

a. Să se scrie definiția completă a subprogramului **nrElem1**
b. Să se scrie definiția completă a subprogramului **sumaElem**
c. Să se scrie un program care citește din fișierul **atestat.in** numărul **n** și, de pe următoarele linii, muchiile grafului. Folosind apeluri utile ale subprogramelor **nrElem1** și **sumaElem**, programul determină nodurile grafului de grad minim, elimină aceste noduri din graf și scrie în fișierul **atestat.out** numărul de muchii ale subgrafului obținut după eliminare.

Exemple:

Exemplul 1

atestat.in

```
6
6 1
2 5
5 4
3 4
6 2
5 6
2 4
6 4
```

atestat.out

```
6
```

Exemplul 2

atestat.in

```
4
1 2
1 3
2 4
3 4
```

atestat.out

```
0
```

Subiectul nr. 52:

Fișierul **atestat.in** conține pe prima linie un text format din maxim **100** de caractere, litere mici ale alfabetului englez și spații. Textul este format din cuvinte de maxim **20** de caractere, separate printr-un singur spațiu.

Se consideră subprogramele:

- **verif** care are ca parametru un șir de caractere **s,** format din maxim **20** de caractere, litere mici ale alfabetului englez. Subprogramul returnează valoarea **1** dacă șirul **s** conține cel puțin două consoane pe poziții consecutive și valoarea **0**, în caz contrar.
- **nrvoc** care are ca parametru șirul de caractere **s** format din maxim **20** de caractere și returnează numărul de vocale din șirul **s**.

Cerințe:

a. Să se scrie definiția completă a subprogramului **verif**;
b. Să se scrie definiția completă a subprogramului **nrvoc**;
c. Să se scrie un program care citește textul din fișierul **atestat.in** și, folosind apeluri utile ale subprogramelor **verif** și **nrvoc**, scrie în fișierul **atestat.out**, câte unul pe linie, cuvintele din text care conțin cel puțin trei vocale și două consoane alăturate. Dacă textul nu conține niciun cuvânt cu proprietățile cerute, în fișier se va scrie mesajul **"nu exista"**.

Exemplu:

```
atestat.in
atestat.out
biblioteca este deschisa in fiecare zi
biblioteca
deschisa
```

Subiectul nr. 53:

În fișierul **atestat.in** se află mai multe linii, fiecare linie conține câte un cuvânt alcătuit din litere mici ale alfabetului englez și cifre mai mici sau egale cu 4, codificat astfel: vocalele a, e, i, o, u au fost înlocuite, în ordine, cu cifrele 0, 1, 2, 3, 4, iar fiecare consoană a fost înlocuită cu caracterul aflat pe poziția anterioară în codul ASCII. Fișierul conține cel puțin un cuvânt.

Se consideră subprogramele:

- **construieste** care are ca parametru un șir cu cel mult **200** de caractere. Subprogramul citește cuvintele din fișierul **atestat.in** și construiește în memorie textul codificat folosind cuvintele citite. Textul codificat va avea cuvintele separate prin câte un spațiu și va fi furnizat prin parametru;
- **decodifica** care are ca parametru un șir cu cel mult **200** de caractere. Subprogramul decodifică textul ținând cont de regulile descrise în enunț. Textul decodificat va fi furnizat prin parametru.

Cerințe:

a. Scrieți definiția completă a subprogramului **construieste;**

b. Scrieți definiția completă a subprogramului **decodifica;**

c. Scrieți un program care, utilizând apeluri utile ale subprogramelor **construieste** și **decodifica**, citește datele din fișierul **atestat.in** și scrie pe linii diferite ale fișierului **atestat.out** textul codificat și textul obținut după decodificare.

Exemplu:

```
atestat.in
1w0l1m
c1
0s1rs0s
atestat.out
1w0l1m c1 0s1rs0s
examen de atestat
```

Subiectul nr. 54:

Fişierului **atestat.in** conţine, pe prima linie, un număr natural **n (2≤n≤20)**, iar pe următoarele **n** linii elementele unui tablou bidimensional **a,** cu **n** linii şi **n** coloane.

Se consideră subprogramele:

- **citire** cu doi parametri: **a**, şi **n**, care determină, în urma apelului, citirea numerelor din fişierul **atestat.in** şi furnizarea, prin intermediul parametrului **a,** a unui tablou bidimensional cu **n*n** componente numere naturale din mulţimea **{0,1}**, reprezentând matricea de adiacenţă asociată unui graf neorientat;

- **suma** care primeşte prin intermediul parametrului **v** un tablou unidimensional (elementele fiind numere naturale), iar prin intermediul parametrului **k** numărul de elemente din tabloul unidimensional **v**. Subprogramul returnează suma tuturor elementelor tabloului unidimensional **v**.

Cerinţe:

a. Scrieţi definiţia completă a subprogramului **citire.**

b. Scrieţi definiţia completă a subprogramului **suma.**

c. Scrieţi un program care, utilizând apeluri utile ale subprogramelor **citire** și **suma**, scrie în fişierul **atestat.out** etichetele nodurilor izolate sau, în situaţia în care nu există astfel de noduri, se va scrie mesajul **"nu există noduri izolate"**.

Exemple:

atestat.in	**atestat.out**
4 **0 0 1 0** **0 0 0 0** **1 0 0 0** **0 0 0 0**	**2 4**
4 **0 1 1 0** **1 0 0 1** **1 0 0 0** **0 1 0 0**	**nu exista noduri izolate**