

SUBIECTELE PROBEI PRACTICE PENTRU
EXAMENUL DE ATESTARE A COMPETENȚELOR PROFESIONALE A ABSOLVENȚILOR
CLASELOR DE MATEMATICĂ-INFORMATICĂ ȘI MATEMATICĂ-INFORMATICĂ,
INTENSIV INFORMATICĂ

PROGRAMARE

SPECIALIZAREA MATEMATICĂ-INFORMATICĂ INTENSIV INFORMATICĂ

Subiectul nr. 1:

Fișierul `atestat.in` conține pe prima linie un număr natural n , $2 \leq n \leq 100$, 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 , $2 \leq n \leq 100$. 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
9
19 25 5632 9872 48903 33 17634 90
3452
```

```
atestat.out
48903 17634 9872 90 19
```

Subiectul nr. 2:

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 ($2 \leq n \leq 100$), 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
5
63 8644 1024 102 2048
```

```
atestat.out
63 4322 1024 102 1024
```

Subiectul nr. 3:

Fișierul **atestat.in** conține pe prima linie un număr natural **n**, $2 \leq n \leq 100$ 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

Subiectul nr. 4:

În fișierul `atestat.in`, pe prima linie se află un număr natural n ($1 \leq n \leq 100$), 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 paritate 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 ($1 \leq n \leq 100$) 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:

- Scrieți definiția completă a subprogramului **sterge**;
- Scrieți definiția completă a subprogramului **cif**;
- 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 linie 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`

7
37 132 7 2785 86 490 18

`atestat.out`

132 2785 490 18

Subiectul nr. 5:

Fișierul `atestat.in` conține pe prima linie un număr natural n , $2 \leq n \leq 100$, 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 ($2 \leq n \leq 100$). 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** ($1 \leq p1, p2 \leq n$) 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:

- Scrieți definiția completă a subprogramului **citeste**;
- Scrieți definiția completă a subprogramului **pozmax**;
- 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

5
2.4 1.9 8.1 4.4 10.25

atestat.out

10.25 8.1 4.4 2.4 1.9

Subiectul nr. 6:

Fișierul **atestat.in** conține cel mult 100 de numere naturale cu cel mult patru cifre, toate numerele fiind scrise pe o singură linie, separate prin câte un spațiu. *Valorile din fișier sunt ordonate descrescător.*

Se consideră subprogramele:

- **construire** cu doi parametri: **v**, un tablou unidimensional cu elemente numere naturale și **n**, un număr natural ($2 \leq n \leq 100$) 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 conține doar acele numere din fișier care au exact trei cifre, precum și numărul de elemente ale acestuia, prin parametrul **n**. *Fișierul conține cel puțin un număr cu 3 cifre.*
- **cautare** cu trei parametri: **v** un tablou unidimensional cu elemente numere naturale, **n** un număr natural ($2 \leq n \leq 100$) reprezentând numărul de elemente ale tabloului **v** și **x** un număr natural. Subprogramul returnează, utilizând algoritmul de căutare binară, poziția pe care se găsește valoarea **x** în vectorul **v** sau valoarea -1, în caz contrar.

Cerințe:

- Scrieți definiția completă a subprogramului **construire**;
- Scrieți definiția completă a subprogramului **cautare**;
- Scrieți un program care, utilizând apeluri utile ale subprogramelor **construire** și **cautare**, scrie în fișierul **atestat.out** poziția în vectorul **v** pe care se găsește 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:

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

Subiectul nr. 7:

Fișierul `atestat.in` conține pe prima linie un număr natural n ($2 \leq n \leq 20$), 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 ($n \leq 20$) număr natural, a , tablou bidimensional cu n linii și n coloane, cu elemente numere naturale și k ($k \leq n$) 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:

- Să se scrie definiția completă a subprogramului **elimColoana**;
- Să se scrie definiția completă a subprogramului **cifreImpare**;
- 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 subprogramelelor **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 nemodificată**".

Exemplu:

<code>atestat.in</code>	<code>atestat.out</code>	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	matrice nemodificată	

112	92	32		
	199			
2	3410	79	11	

Subiectul nr. 8:

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 ($2 \leq n \leq 20$), 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 ($2 \leq n \leq 20$) 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 ($2 \leq n \leq 20$) ce reprezintă dimensiunea efectivă a tabloului **a**. Subprogramul returnează produsul elementelor aflate pe diagonala principală a tabloului **a**.

Cerințe:

- Să se scrie definiția completă a subprogramului **inftr**;
- Să se scrie definiția completă a subprogramului **produs**;
- 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:

<code>atestat.in</code>	<code>atestat.out</code>
3	da
3 0 0	6
1 2 0	
1 0 1	

Subiectul nr. 9:

Fișierul `atestat.in` conține pe prima linie un număr natural n , $2 \leq n \leq 20$, 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:

- Scrieți definiția completă a subprogramului **prim**;
- Scrieți definiția completă a subprogramului **contorizare**;
- 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

Subiectul nr. 10:

În fișierul **atestat.in**, pe prima linie, se află două numere **n** și **m** ($2 \leq n, m \leq 20$) 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 ($1 \leq n \leq 200$) 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 ($2 \leq a \leq 200$) și returnează cel mai mic divizor prim al valorii parametrului **a**.

Cerințe:

- Scrieți definiția completă a subprogramului **fibo**;
- Scrieți definiția completă a subprogramului **divprim**;
- 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 Fibonacci sunt 13, 2, 3, 5

Subiectul nr. 11:

În fișierul **atestat.in**, pe prima linie se află un număr n ($2 \leq n \leq 20$) 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 ($1 \leq x \leq 10^9$) și returnează 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 diagonalele matricei sunt: 9, 8, 22

Subiectul nr. 12:

În fișierul `atestat.in`, pe prima linie se află un număr n ($2 \leq n \leq 20$) 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:

- Scrieți definiția completă a subprogramului **interschimbL**;
- Scrieți definiția completă a subprogramului **interschimbC**;
- Scrieți un program care citește din fișierul `atestat.in` numărul n , tabloul bidimensional cu $n \times n$ elemente și, folosind apeluri utile ale subprogramelor **interschimbL** și **interschimbC** ordonează crescător elementele diagonalei principale prin interschimbarea 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:

<code>atestat.in</code>	<code>atestat.out</code>
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

Subiectul nr. 13:

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 propoziție 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 separate 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
3
noi doi si voi
a sosit fata pe care o asteptam
in vacanta mergem la munte
```

```
atestat.out

noi asteptam vacanta
8
```

Subiectul nr. 14:

Fișierul **atestat.in** conține pe prima linie un număr natural n ($1 \leq n \leq 100$), 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
4
Vara Care A Trecut
```

```
atestat.out
Vara Care A Trecut
```

Vara
CARE
a
tReCuT

7

Subiectul nr. 15:

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. 16:

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 parametru: **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 alăturării **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**

```
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
```

atestat.out

```
3578620
```

Subiectul nr. 17:

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**

```
1w011m
c1
```

atestat.out

```
1w011m c1 0s1rs0s
examen de atestat
```

0s1rs0s

Subiectul nr. 18:

În fișierul `atestat.in`, pe prima linie se află un număr natural n ($2 \leq n \leq 30$), 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:

- Scrieți definiția completă a subprogramului **citire**;
- Scrieți definiția completă a subprogramului **stergere**;
- 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
4
evantai
variabile
vacante
revoltator
```

```
atestat.out
a e v
```

Explicație
Literele se afișează nu neapărat în această ordine

Subiectul nr. 19:

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 exam 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:

- Scrieți definiția completă a subprogramului **prefixe**;
- Scrieți definiția completă a subprogramului **sufixe**;
- 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	carte c ca car cart carte
recreatie	spectacol l ol col acol tacol ctacol ectacol
vacanta sport	pectacol spectacol
spectacol	

Subiectul nr. 20:

Fișierul **atestat.in** conține pe prima linie un număr natural **n**, $2 \leq n \leq 100$, 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:

- Scrieți definiția completă a subprogramului **cmmdc**;
- Scrieți definiția completă a subprogramului **suma**;
- 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:

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

Subiectul nr. 21:

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:

- Să se scrie definiția completă a subprogramului **modul**;
- Să se scrie definiția completă a subprogramului **suma**;
- Scrieți programul care citește din fișierul **atestat.in** un număr natural **n** ($1 \leq n \leq 25$), 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:

<code>atestat.in</code>	<code>atestat.out</code>
4	9 12
3.0 4.0	
1.5 2.7	
23.0 9.4	
6.0 8.0	

Subiectul nr. 22:

Fișierul `atestat.in` conține pe prima linie un număr natural n ($2 \leq n < 100$), iar pe fiecare din următoarele n linii, separate prin câte un spațiu, câte două numere reale reprezentând coordonatele carteziane ale unui punct din plan.

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

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

Se consideră subprogramele:

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

Cerințe:

- a. Scrieți definiția completă a subprogramului **distanța** ;
- b. Scrieți definiția completă a subprogramului **arie** ;
- c. Scrieți un program care citește de pe prima linie a fișierului `atestat.in` un număr natural n ($2 \leq n < 100$), 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 **distanța** și **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 diagonală dreptunghiului de aria maximă.

Exemplu:`atestat.in``4``1 1``2 3``3 3``4 2`

Se va afișa pe ecran

`4 1 3`

Explicație: aria maximă care se poate obține este 4 și este aria dreptunghiului care are o diagonală formată din punctele de coordonate (1,1) și (3,3), adică punctele cu numerele de ordine 1 și 3

Subiectul nr. 23:

În fișierul `atestat.in`, pe prima linie se află un număr natural n ($2 \leq n \leq 100$) reprezentând numărul de noduri ale unui arbore, iar pe a doua linie se află n numere naturale reprezentând vectorul de tați al arborelui. Nodurile arborelui sunt etichetate de la 1 la n .

Se consideră subprogramele:

- **radacina** cu doi parametri: v un vector cu cel mult 100 de elemente numere naturale reprezentând vectorul de tați al unui arbore și n un număr natural ($2 \leq n \leq 100$) reprezentând numărul efectiv de elemente din v . Subprogramul returnează eticheta nodului rădăcină al arborelui.
- **construieste** cu trei parametri care primește prin intermediul parametrului v un vector cu cel mult 100 de elemente numere naturale, n un număr natural ($2 \leq n \leq 100$) reprezentând numărul efectiv de elemente din v și a o matrice pătratică cu n linii și n coloane. Subprogramul construiește în memorie și furnizează prin parametrul a matricea de adiacență a arborelui reprezentat prin vectorul de tați v .

Cerințe:

- Scrieți definiția completă a subprogramului **radacina**;
- Scrieți definiția completă a subprogramului **construieste**;
- Scrieți un program care, utilizând apeluri utile ale subprogramelor **radacina** și **construieste**, citește datele din fișierul `atestat.in` și scrie pe prima linie a fișierului `atestat.out` nodul rădăcină al arborelui, iar pe următoarele n linii câte n valori din mulțimea $\{0, 1\}$ separate prin câte un spațiu, reprezentând matricea de adiacență a arborelui.

Exemplu:

`atestat.in`

```
4
3 4 4 0
```

`atestat.out`

```
4
0 0 1 0
0 0 0 1
1 0 0 1
0 1 1 0
```

Subiectul nr. 24:

Se consideră un graf neorientat G cu n vârfuri ($n \in \mathbb{N}$, $2 \leq n \leq 30$) etichetate cu numerele distincte: $1, 2, \dots, n$.

Fișierul `atestat.in` conține mai multe linii. Pe prima linie a fișierului este scris numărul natural n reprezentând numărul de vârfuri ale grafului G , iar pe următoarele linii, perechi de numere naturale, separate prin câte un spațiu, reprezentând muchiile distincte ale grafului G .

Se consideră subprogramele:

- **nrElem1** cu trei parametri: n , un număr natural, $n \leq 30$, 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 ($2 \leq n \leq 30$) și **a** un tablou bidimensional, pătratic, cu **n** linii și **n** coloane și elemente numere naturale. Subprogramul returnează suma elementelor tabloului bidimensional **a**.

Cerințe:

- Să se scrie definiția completă a subprogramului **nrElem1**
- Să se scrie definiția completă a subprogramului **sumaElem**
- 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		Exemplul 2	
atestat.in	atestat.out	atestat.in	atestat.out
6	6	4	0
6 1		1 2	
2 5		1 3	
5 4		2 4	
3 4		3 4	
6 2			
5 6			
2 4			
6 4			

Subiectul nr. 25:

Fișierului **atestat.in** conține, pe prima linie, un număr natural **n** ($2 \leq n \leq 20$), iar pe următoarele **n** linii elementele unui tablou bidimensional **a**, cu **n** linii și **n** coloane.

Se consideră subprogramele:

- **citire** cu doi parametri: **a**, și **n**, care determină, în urma apelului, citirea numerelor din fișierul **atestat.in** și furnizarea, prin intermediul parametrului **a**, a unui tablou bidimensional cu **n*n** componente numere naturale din mulțimea $\{0, 1\}$, reprezentând matricea de adiacență asociată unui graf neorientat;
- **suma** care primește 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**.

Cerințe:

- Scrieți definiția completă a subprogramului **citire**.
- Scrieți definiția completă a subprogramului **suma**.
- Scrieți un program care, utilizând apeluri utile ale subprogramelor **citire** și **suma**, scrie în fișierul **atestat.out** etichetele nodurilor izolate sau, în situația în care nu există astfel de noduri, se va scrie mesajul "nu există noduri izolate".

Exemple:

atestat.in	atestat.out
4	2 4
0 0 1 0	

0 0 0 0

1 0 0 0

0 0 0 0

4

0 1 1 0

1 0 0 1

1 0 0 0

0 1 0 0

nu exista noduri izolate

Colectivul de autori:

Bălașa Filonela	- Colegiul Național „Grigore Moisil”
Gebăilă Gilda Grațiela	- Colegiul Național „Mihai Viteazul”
Popa Simona Mihaela	- Colegiul Național „Gheorghe Lazăr”
Petrișor Valiana Felicia	- Colegiul Național Bilingv „George Coșbuc”
Danciu Alina	- Colegiul Național „Ion Creangă”
Glaje Mihaela Denisa	- Colegiul Național „Grigore Moisil”
Bușe Constanța Elena	- Colegiul Național „Ion Neculce”

Coordonator:

Ștefania Penea – inspector școlar pentru Informatică și Tehnologia Informației și a Comunicațiilor, Inspectoratul Școlar al Municipiului București