

VIZAT,  
INSPECTOR ȘCOLAR GENERAL,  
PROF. VALERIA GHERGHE

**SUBIECTELE PROBEI PRACTICE PENTRU**  
**EXAMENUL DE ATESTAT PROFESIONAL LA INFORMATICĂ, 2017**  
**PROGRAMARE-INTENSIV**

1. În fișierul **date.in** pe prima linie este scris un număr natural nenul **n**,  $1 \leq n \leq 100$ , iar pe cea de-a doua linie **n** numere naturale de cel mult **9** cifre fiecare, separate prin câte un spațiu. Folosind subprogramele: **S1**, care primește prin intermediul parametrului **x**, un număr natural de cel mult **9** cifre și returnează prima (cea mai semnificativă) cifră a sa și subprogramul **S2** căruia i se transmit ca parametrii (**a**, 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 **a**,  $n < 101$ ) și care ordonează descrescător elementele vectorului **a**.

**Cerințe:**

Scrieți un program principal **Pascal/C/C++** care, folosind apeluri utile ale subprogramelelor **S1** și **S2**, să citească datele din fișierul **date.in** și care să scrie în fișierul **date.out** pe linii separate numerele aflate pe cea de-a doua linie a fișierului **date.in** care au prima cifră egală cu ultima, în ordine descrescătoare. În cazul în care nu există astfel de numere, programul va scrie în fișierul **date.out** mesajul **nu exista** pe prima linie a acestuia.

**Exemplu.** Dacă fișierul **date.in** are conținutul alăturat, programul va scrie în fișierul **date.out** următoarele:

123456781  
5415  
88  
7

6  
18 415 7 88 100 123456781

2. Se consideră un graf neorientat **G** cu **n** noduri ( $n \in \mathbb{N}$ ,  $2 < n < 20$ ) etichetate cu numerele distincte: 1,2,...,n. Fișierul **date.in** conține două linii. Pe prima linie a fișierului este scris numărul **n** de vârfuri ale grafului **G**, iar pe următoarea linie, **n** numere naturale, separate prin câte un spațiu, reprezentând gradele nodurilor grafului **G**.

Scrieți un program **Pascal/C/C++**, care determină și afișează pe ecran mesajul **“COMPLET”**, dacă graful **G** este graf complet și mesajul **“OARECARE”** în orice altă situație.

**Exemplu.** Dacă fișierul **date.in** are conținutul alăturat, programul va afișa pe ecran: **OARECARE**

6  
5 5 5 4 5 4

3. Fișierul *Numere.txt* conține pe prima linie un număr natural  $n$  ( $n < 10000$ ), iar pe linia următoare un șir de  $n$  numere naturale distincte cu maximum 4 cifre fiecare, separate prin câte un spațiu. Se cere:

- Afișați în fișierul text *Rezultat.out*, pe prima linie, poziția pe care s-ar găsi primul element din șirul aflat pe linia a doua a fișierului, dacă șirul ar fi ordonat crescător. Numerotarea pozițiilor elementelor în cadrul șirului este de la 1 la  $n$ . Alegeți un algoritm de rezolvare eficient din punct de vedere al memoriei utilizate și al timpului de executare.
- Verificați dacă primul număr de pe a doua linie din fișier este număr prim și afișați un mesaj corespunzător, pe a doua linie a fișierului de ieșire (**DA** sau **NU**). Veți folosi un subprogram care primește ca parametru un număr natural  $x$ , de cel mult 4 cifre și returnează **TRUE** dacă numărul  $x$  este prim sau **FALSE** în caz contrar.

**Exemplu:** Dacă fișierul *Numere.txt* are următorul conținut:

6

267 13 45 628 7 79

Fișierul *Rezultat.out* va avea următorul conținut:

5

NU

deoarece numărul 267 ar ocupa poziția a cincea în șirul ordonat crescător (7 13 45 79 267 628) și 267 nu este număr prim.

4. Se citesc de la tastatură un număr natural nenul  $n$ , ( $n < 10$ ) și un număr natural  $k$  ( $k \leq 9$ ).

a) Construiți o matrice pătratică  $a$ , unde  $a[i,j]$  este al  $i+j-1$  lea număr prim (dacă numerotăm indicii matricei 1.. $n$ ) sau al  $i+j+1$  lea număr prim (dacă numerotăm indicii matricei 0.. $n-1$ ). Afișați tabloul bidimensional astfel construit în fișierul *Matrice.out*, pe primele  $n$  linii ale fișierului, elementele unei linii fiind separate de un singur spațiu. Utilizați un subprogram prim, care primește ca și parametru un număr natural  $x$ , și returnează valoarea 1 dacă numărul transmis ca parametru este prim, și 0 în caz contrar.

b) Afișați pe următoarea linie în fișierul de ieșire, în ordine crescătoare toate elementele tabloului astfel construit care au exact  $k$  cifre distincte. Dacă nu există astfel de numere în tabloul construit afișați în fișier mesajul NU EXISTĂ.

**Exemplu:** dacă  $n=4$  și  $k=2$

Fișierul *Matrice.out* va avea următorul conținut:

2 3 5 7

3 5 7 11

5 7 11 13

7 11 13 17

13 17

OBS. Pentru  $n=4$  și  $k=4$  pe ultima linie din fișier se va afișa NU EXISTĂ

Considerăm că numărul 2 este primul număr prim.

5. Se citește de la tastatură două numere naturale  $n$  ( $1 \leq n \leq 10$ ) și  $k$  ( $0 < k \leq 10$ ).
- a) Construieți un tablou bidimensional cu  $n$  linii și  $n$  coloane, astfel încât parcurgând liniile matricei de sus în jos și de la stânga la dreapta se obțin, în prima linie primele  $n$  numere ale șirului Fibonacci în ordine **crescătoare**, în linia a doua următoarele  $n$  numere ale șirului Fibonacci în ordine **descrescătoare**, în linia a treia următoarele  $n$  numere ale acestui șir în ordine **crescătoare**, și așa mai departe, ca în exemplu. Elementele șirului Fibonacci se obțin astfel: primul element este **0**, al doilea este **1**, iar elementele următoare se obțin însumând cele două elemente care preced elementul curent. Astfel, primele **16** elemente ale acestui șir sunt: **0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610**. Programul afișează în fișierul **Matrice.out** matricea obținută, câte o linie a matricei pe câte o linie a fișierului, elementele fiecărei linii fiind separate prin câte un spațiu.
- b) Pe următoarea linie în fișier afișați produsul indicilor coloanelor pe care exista cel puțin  $k$  elemente pare.

**Exemplu:** dacă se citește de la tastatură  $n=4$  și  $k=2$

Fișierul **Matrice.out** va avea următorul conținut:

```
0 1 1 2
13 8 5 3
21 34 55 89
610 377 233 144
8
```

Pe ultima linie în fișier se va afișa **8** (deoarece coloanele **1, 2 și 4** conțin cel puțin două elemente pare).

6. Se definește șirul lui Fibonacci:  $f_1=0$ ,  $f_2=1$ ,  $f_n=f_{n-1}+f_{n-2}$ ,  $n \geq 3$ . Fișierul **atestat6.in** conține pe prima linie un număr natural  $n$  de cel mult 9 cifre. Să se descompună numărul natural  $n$  în sumă de termeni nenuli ai șirului Fibonacci, numărul termenilor din sumă trebuie să fie minim. Se va utiliza o funcție pentru determinarea celui de-al  $n$ -lea termen al șirului lui Fibonacci.

**Exemple:**

pentru  $n=8$  se va afișa  $5+3$ ;

pentru  $n=24$  se afișează  $21+3$ .

7. În fișierul **atestat7.in** se află pe prima linie maxim un milion de numere naturale de cel mult 2 cifre fiecare. Scrieți programul C/C++ care citește numerele din fișierul **atestat7.in** și determină și afișează pe ecran care dintre numerele citite apare de cele mai puține ori în fișier. Se va utiliza un algoritm eficient din punct de vedere al spațiului de memorie utilizat și al timpului de executare.

**Exemplu :**

<b>atestat7.in</b>	se afișează
<b>5 3 1 6 3 1 3 6 1 3</b>	<b>5</b>

8. În fișierul **atestat8.in** se află pe prima linie un număr natural  $n$  ( $n \leq 9$ ), iar pe a doua linie se află  $n$  numere cifre zecimale  $a_1, a_2, \dots, a_n$ . Se cere să se calculeze suma  $S = a_1a_2\dots a_n + a_2a_3\dots a_{n-1}a_n a_1 + a_3\dots a_n a_1 a_2 + \dots + a_n a_1 a_2 a_3 \dots a_{n-1}$ . Se va folosi un subprogram pentru permutarea circulară la stânga, cu o cifră, a cifrelor unui număr.

**Exemplu:** fie  $n=3$ ;  $a_1=1$ ;  $a_2=3$ ;  $a_3=7$ ;  $S=137+371+713=1221$

9. În fișierul **atestat9.in** se află un șir de cuvinte, câte unul pe linie. Să se scrie un program care citește cuvintele din fișier și le afișează în fișierul **atestat9.out** în ordine crescătoare a lungimilor lor. Dacă există două sau mai multe cuvinte cu aceeași lungime se vor ordona alfabetic. Pentru ordonare se va scrie un subprogram în care se va utiliza unul din algoritmii de sortare studiați.

**Exemplu:**

atestat9.in	atestat9.out
Mere	Ana
Ana	are
are	mere

10. Subprogramul **sum** primește prin intermediul parametrului  $n$  ( $1 < n < 30$ ) dimensiunea unui tablou bidimensional pătratic, prin intermediul parametrului  $a$  tabloul bidimensional de numere reale ( $a_{ij}$  cu  $1 \leq i \leq n$ ,  $1 \leq j \leq n$ ) și prin intermediul parametrului  $k$  un număr natural nenul ( $1 < k \leq 2 * n$ ). El returnează prin intermediul parametrului  $s$  suma tuturor elementelor  $a_{ij}$  cu proprietatea că  $i+j=k$ . Scrieți programul care citește de la tastatură un tablou de numere reale cu  $n$  linii și  $n$  coloane și afișează suma elementelor din tablou aflate strict deasupra diagonalei principale a tabloului, folosind apeluri ale subprogramului **sum**, definit conform cerinței.

**Exemplu:**

Date de intrare	se afișează
<b>n=4 k=5</b>	20
<b>2 7 3 1</b>	
<b>6 2 3 1</b>	
<b>0 3 1 5</b>	
<b>3 1 6 3</b>	

11. Se citesc de la tastatură două numere naturale nenule  $p$  și  $q$  ( $3 < p < q \leq 999999999$ ). Să se determine, dacă există, un număr prim  $x$  care aparține intervalului închis  $[p, q]$  pentru care valoarea expresiei  $|q+p-2*x|$  este minimă. S-a folosit notația  $|x|$  pentru modulul numărului  $x$ . Dacă nu există un astfel de număr, se va afișa valoarea 0, iar dacă există mai multe, se va afișa cel mai mic dintre ele.

Pentru rezolvarea problemei se va utiliza un subprogram prim care primește prin intermediul parametrului  $n$  un număr natural ( $n > 1$ ) și care returnează 1 dacă numărul  $n$  este prim sau 0, în caz contrar.

**Exemplu:** pentru  $p=6$  și  $q=18$ , dintre numerele prime 7, 11, 13 și 17, se va afișa 13 deoarece  $|18+7-2*13| < |18+7-2*11| < |18+7-2*17| < |18+7-2*7|$ .

**12.** Subprogramul **nrcar** primește prin intermediul parametrului  $s$  un șir cu cel mult 200 de caractere și prin parametrul  $c$  un caracter. El returnează prin intermediul parametrului  $p$  un număr natural reprezentând numărul de apariții ale caracterului  $c$  în șirul  $s$ . Scrieți programul care citește de la tastatură un șir de caractere (litere mari și mici ale alfabetului englez și cifre). Se cere să se determine numărul total de vocale din șirul dat, folosind apeluri ale subprogramului **nrcar**. Rezultatul se va afișa în fișierul text **atestat12.out**.

**Exemplu :**

Pentru  $s = \text{"Informatica01Atestat2016"}$  se va afișa 8 (deoarece sunt 8 vocale în șir)

**13.** Fișierul text **atestat13.in** conține pe primul rând un număr natural  $n$  ( $n > 1$ ) și apoi pe următorul rând  $n$  numere reale  $x_1, x_2, \dots, x_n$ . Să se determine câte dintre cele  $n$  numere citite se află în intervalul închis determinat de numerele  $x_1$  și  $x_n$ .

**Exemplu:**

pentru  $n=6$  și numerele 5, 6.35, 4.5, -10, -8, -9, se afișează valoarea 4 (deoarece patru dintre numerele date, cele subliniate, se află în intervalul determinat de numerele 5 și -9).

**14.** Fișierul text **atestat14.txt** conține pe prima linie două numere naturale  $n$  și  $m$  ( $0 < m < n < 5000$ ), pe cea de a doua linie  $n$  numere naturale  $a_1, a_2, \dots, a_n$  ( $0 \leq a_i \leq 9$ ), iar pe cea de a treia linie  $m$  numere naturale  $b_1, b_2, \dots, b_m$  ( $0 \leq b_i \leq 9$ ).

Scrieți un program care citește datele din fișier, verifică dacă șirul  $b$  se poate obține din șirul  $a$  și afișează pe ecran un mesaj corespunzător. Se va utiliza un algoritm eficient din punct de vedere al timpului de executare și al spațiului de memorie utilizat.

**Exemplu:**

<b>Atestat14.txt</b>	se afișează mesajul
7 5 2 4 1 6 2 1 3 6 1 2 3 2	"b se poate obtine din a"

15. Scrieți un program care citește din fișierul text **atestat15.in** un șir **v** de maxim 1.000.000 de numere naturale formate din exact două cifre fiecare și afișează *distanța* maximă care există între două elemente egale ale șirului. Definim *distanța* dintre două elemente  $v_i$  și  $v_j$  prin modulul diferenței indicilor celor două elemente,  $|j-i|$ . Dacă șirul conține doar elemente distincte, se va afișa valoarea 0.

**Exemplu:**

Atestat15.in	se afișează	Explicații
10 14 <b>12</b> 10 10 14 15 10 12 90 12 13 15 80 <b>12</b> 90	12	( $v_3=v_{15}=12$ , $15-3=12$ )

16. Pe prima linie a fișierului **atestat16.in** se găsește un număr natural  $n$ ,  $n \leq 1000$ , iar a doua linie un șir cu  $n$  numere naturale cu cel mult 9 cifre fiecare, separate prin câte un spațiu. Scrieți un program care scrie în fișierul **atestat16.out** pe prima linie numărul de valori prime din fișierul dat, iar pe următoarea linie toate numerele prime din șir, separate prin câte un spațiu, în ordinea apariției lor în fișier. Pentru determinarea numerelor prime se va utiliza un subprogram **prim** cu un singur paramentru  $x$  număr natural ( $0 \leq x \leq 1000000000$ ), care verifică dacă numărul transmis prin parametrul  $x$  este număr prim.

**Exemplu:**

atestat16.in	atestat16.out
8 12 13 2 123 5 41 77 23	5 13 2 5 41 23

17. Scrieți un program care citește de la tastatură un număr natural  $n$ ,  $2 \leq n \leq 10$  și construiește o matrice cu  $n$  linii și  $n$  coloane care va conține pe fiecare linie câte o permutare a mulțimii  $\{1, 2, \dots, n\}$ , astfel încât pe linii diferite să avem permutări diferite conform exemplului de mai jos. Matricea astfel obținută se va afișa în fișierul text **atestat17.out**, câte o linie a matricei pe câte o linie a ecranului.

**Exemplu:**

tastatură	ecran
5	1 2 3 4 5 2 3 4 5 1 3 4 5 1 2 4 5 1 2 3 5 1 2 3 4

18. Pe prima linie a fișierului **atestat18.in** se găsește un număr natural  $n$ ,  $n \leq 100$ , iar a doua linie conține un șir cu **maximum 100000000** numere naturale, separate prin câte un spațiu. Să se scrie în fișierul **atestat18.out** toate numerele din șir care sunt termeni din Șirul lui Fibonacci. Se va folosi un subprogram **fib** care verifică dacă un număr natural, dat ca parametru de intrare, este termen în șirul lui Fibonacci ( $f_1=0$ ,  $f_2=1$ ,  $f_n=f_{n-1}+f_{n-2}$ , pentru  $n>2$ ).

Exemplu:

<b>atestat18.in</b>	<b>atestat18.out</b>
3 4 7 1 10 21 13 5 2 12	3 1 21 13 5 2

19. Se citesc din fișierul **atestat19.in** două numere naturale  $a$  și  $b$  cu cel mult 9 cifre nenule fiecare. Scrieți un program care scrie în fișierul **atestat19.out** cel mai mic număr natural care se poate forma cu toate cifrele celor două numere citite.

Exemplu:

<b>atestat19.in</b>	<b>Atestat19.out</b>
856331	11123335678
17321	

20. Pe prima linie a fișierului **atestat20.in** se găsește un număr natural  $n$ ,  $n \leq 100$ , iar a doua linie conține un șir cu  $n$  numere naturale cu cel mult nouă cifre fiecare, separate prin câte un spațiu. Scrieți un program care scrie în fișierul **atestat20.out** toate numerele din șir pentru care suma cifrelor este divizibilă cu 3. Se va folosi o funcție recursivă **sumcif** care calculează și returnează suma cifrelor parametrului de intrare  $x$ .

Exemplu:

<b>atestat20.in</b>	<b>atestat20.out</b>
7	51 231 24
124 51 231 7 24 31 5	

21. Scrieți un program care citește de pe prima linie a fișierului **atestat21.in** un număr natural  $n$  iar de pe următoarele  $n$  linii un tablou bidimensional cu  $n$  linii și  $n$  coloane conținând numere naturale și care modifică matricea în felul următor: toate elementele liniilor care conțin valoarea minimă vor fi mărite cu valoarea maximă din matrice. Scrieți în fișierul **atestat21.out** matricea astfel obținută.

Exemplu:

atestat21.in	atestat22.out
4	
2 4 3 1	7 9 8 6
2 2 4 4	2 2 4 4
3 2 1 2	8 7 6 7
5 3 5 2	5 3 5 2

22. Fișierul text **atestat22.in** conține informații despre mai mulți elevi, sub o formă nestructurată. Informațiile sunt dispuse pe linii de maxim 200 de caractere și pot conține CNP-uri valide. Știi că CNP-ul unei persoane este un șir de 13 cifre consecutive, scrieți un program care determină și scrie în fișierul text **atestat21.out**, pe linii distincte, toate CNP-urile extrase din text. Dacă nu există nici un astfel de șir, se va scrie în fișier valoarea 0.

Exemplu:

atestat22.in	atestat22.out
Popesu George, 14 ani, 1020412342334; Gina Badea - 1031102343435, Republicii 7; Dana Marian: 2040405358687, fara nota, 2030609987654 - Janina Nalbu	1020412342334  1031102343435  2040405358687  2030609987654
atestat22.in	atestat22.out
Popesu George, 14 ani, 102412342334; Gina Badea - 10311023435, Republicii 7; Dana Marian: 204/040/5358687, fara nota	0

23. Spunem că un număr natural  $x$  este **rotund** dacă există un număr natural nenul  $k$ , mai mic strict decât numărul de cifre al lui  $x$ , astfel încât prin permutarea circulară a cifrelor numărului cu  $k$  poziții la dreapta, să se obțină numărul inițial. Scrieți un program care citește din fișierul text **atestat23.in**, de pe prima linie un număr natural  $n$ , apoi de pe linia doi  $n$  numere naturale din intervalul  $[10, 10^9]$ , separate prin câte un spațiu. Programul determină și afișează pe prima linie a ecranului câte numere rotunde sunt în fișier, iar pe linia a doua numerele rotunde în ordinea în care apar în fișierul **atestat23.in**, separate prin câte un spațiu. Dacă fișierul nu conține numere rotunde se va afișa valoarea 0.



Exemplu:

<b>atestat23.in</b>	<b>Ecran</b>
5 12 3232 123 144144 77	3 3232 144144 77
<b>atestat23.in</b>	<b>Ecran</b>
3 11231 45678 232	0

**24.** Scrieți un program eficient din punct de vedere al timpului de execuție, care generează și scrie în fișierul text **atestat24.txt**, pe prima linie, separate prin câte un spațiu, toate **palindroamele-munte** de nouă cifre (un palindrom are aspect de munte dacă cifrele sale sunt strict crescătoare până la jumătatea numărului. EX. 123454321). Se va scrie un subprogram care verifică dacă un număr este palindrom.

Pe a doua linie în fișier se va scrie numărul de palindroame-munte generate.

**25.** Din fișierul **vector.in** se citește de pe prima linie un număr natural **n**, iar de pe a doua linie se citesc **n** elemente numere naturale. Să se calculeze CMMDC al elementelor vectorului citit, folosind metoda **Divide et Impera**.

**26.** Se citește un șir de numere întregi din fișierul **nr.txt**. Elementele șirului se găsesc toate pe un rând în fișier, separate prin spații. Folosind metoda **Divide et Impera**, să se determine simultan suma elementelor pare și produsul elementelor impare.

**27.** Fișierul numere.txt conține un șir de numere întregi, scrise toate pe un singur rând, separate prin spații. Să se creeze o listă liniară simplu înlănțuită care să conțină numai valorile divizibile cu 3 din fișierul dat și să se afișeze pe ecran lista astfel obținută.

**28.** Se dă un grup format din **n** persoane, care se cunosc sau nu între ele. De la tastatură se introduc **m** perechi de numere întregi  $(x,y)$  cu semnificația "persoana  $x$  cunoaște pe persoana  $y$ ". Relația de cunoștință nu este neapărat reciprocă. Numim **celebritate**, o persoană care este cunoscută de către toate celelalte persoane din grup, dar ea nu cunoaște pe nici un alt membru al grupului. Să se determine dacă în grup există o astfel de celebritate.

29. Din fișierul text *clasa.in* se citește de primul rând un număr natural  $n$ , iar de pe următoarele  $n$  rânduri se citesc următoarele informații despre fiecare elev din clasă: numele, prenumele și media. Să se afișeze în fișierul *clasa.out* elevii din clasă ordonați descrescător după medie.
30. Se dă un arbore binar cu  $n$  noduri prin vectorii de descendenți  $S$  și  $D$  citați de la tastatură. Afișați pe ecran pe rânduri separate: frunzele arborelui, vârfurile cu un singur descendent direct și vârfurile cu doi descendenți direcți.
31. Se citește dintr-un fișier text un număr natural  $n$ ,  $n \geq 1$  și apoi cele  $n$  elemente numere reale distincte ale unui vector. Fără a ordona efectiv vectorul să se determine ce poziție ar ocupa ultimul element din vector, dacă acesta ar fi supus unui criteriu de ordonare.
32. Se dă un vector  $v$  cu  $n$  elemente numere naturale. Să se afișeze pentru fiecare pereche de elemente consecutive din  $v$ , cel mai mic număr cuprins în intervalul determinat de cele două valori, care este divizibil cu un număr natural  $k$ , citit de la tastatură, sau 0 dacă nu există un astfel de număr. Valorile cerute vor fi scrise în fișierul text 'fisa18.txt', fiecare pe câte o linie a fișierului.
33. Fișierul text 'fisa18.txt' conține pe prima linie un număr real pozitiv  $x$ , care are cel mult trei cifre la partea întreagă și cel mult cinci cifre la partea zecimală. Scrieți un program care afișează pe ecran, separate printr-un spațiu, două numere naturale, al căror raport este egal cu  $x$  și a căror diferență absolută este minimă.
34. Se dă fișierul text 'f19\_2.in', care conține pe prima sa linie un număr natural nenul  $n$ , iar pe următoarele  $n$  linii câte un număr natural format din cel mult opt cifre. Să se afișeze într-un fișier text 'f19\_2.out', pe câte o linie a fișierului, câte un număr din fișierul dat, urmat de caracterul ":" și de cifrele care apar în scrierea acestuia, în ordine crescătoare, separate prin câte un spațiu.
35. Fișierul text 'f19\_3.txt' conține pe prima linie un număr natural nenul  $n$ ,  $n \leq 100$ , iar pe următoarea linie  $n$  numere naturale nenule, de maximum patru cifre, reprezentând elementele unui vector. Să se verifice dacă elementele vectorului dat reprezintă sau nu o permutare a mulțimii  $\{1, 2, 3, \dots, n\}$ , afișându-se un mesaj corespunzător.
36. Se dă un vector  $v$  cu  $n$  elemente numere naturale, cu cel mult șase cifre fiecare. Să se scrie într-un fișier text, pe câte o linie a fișierului, pentru fiecare număr din cele  $n$  date, cel mai apropiat număr de acesta, care începe și se termină cu aceeași cifră, mai mare decât numărul respectiv.
37. Se citesc 2 șiruri de caractere. Să se construiască un șir ce conține primele 3 caractere din fiecare șir citit. Ex:  $a = \text{"Informatica"}; b = \text{"aplicata"}; c = \text{"Infapl"}$
38. Se citesc numele și nota de la informatică a doi elevi. Să se afișeze numele elevilor în ordinea descrescătoare a notei de la informatică. Dacă amândoi elevii au aceeași notă, se va afișa primul, cel care este alfabetic mai mic.
39. Se consideră un text cu cel mult 70 de caractere (litere mici ale alfabetului englez și spații), în care cuvintele sunt separate prin unul sau mai multe spații. Înaintea primului cuvânt și după ultimul cuvânt nu există spații. Scrieți un program care citește de la tastatură un text de tipul menționat mai sus

și afișează pe ecran numărul de cuvinte în care apare litera a. Exemplu: pentru textul : "voi sustine examenul la informatica" se va afișa valoarea 3.

**40.** Se citește un șir de maxim 20 caractere ce nu conține caractere albe. Să se elimine toate consoanele și să se afișeze șirul.

**41.** Fișierul atestat.txt conține un text scris cu litere mari pe una sau mai multe linii. Se cere:  
a) Să se afișeze litera ( literele) care apare de cele mai multe ori;  
b) Să se afișeze vocalele din text.

**42.** Fișierul atestat.in conține două linii. Pe prima linie este scris un număr natural nenul  $n$ , ( $5 < n < 30$ ). Pe cea de-a doua linie a fișierului sunt scrise  $n$  numere naturale separate prin câte un spațiu, formate fiecare din cel mult 9 cifre, reprezentând un șir de  $n$  numere naturale.

**43.** Să se scrie un program în limbajul Pascal/C/C++, care:

- să afișeze pe ecran, în linie, numerele din șir, separate prin câte un spațiu;
- să afișeze pe ecran, pe linii diferite, cel mai mic număr  $a$  și cel mai mare număr  $b$  din șirul dat;
- să scrie în fișierul atestat.out cel mai mare divizor comun al numerelor  $a$  și  $b$ , determinate la punctul b).

Notă: Programul va conține cel puțin un subprogram definit de utilizator.

Exemplu:

atestat.in		Date de ieșire:
6	a)	123 55 372 3465 242 44
123 55 372 3465 242 44	b)	44 3465
	c)	Fișierul atestat.out conține: 11

**44.** Fișierul atestat.in conține două linii. Pe prima linie este scris un număr natural nenul  $n$ , ( $5 < n < 30$ ). Pe cea de-a doua linie a fișierului sunt scrise  $n$  numere naturale separate prin câte un spațiu, formate fiecare din cel mult 9 cifre, reprezentând un șir de  $n$  numere naturale.

Să se scrie un program în limbajul Pascal/C/C++, care:

- să afișeze pe ecran, în linie, toate numerele din șir, separate prin câte un spațiu;
- să afișeze pe ecran, în linie, toate numerele din șir formate numai din cifre pare (dacă nu există astfel de numere în șir se va afișa mesajul "NU EXISTĂ NUMERE NUMAI CU CIFRE PARE");
- să citească de la tastatură două numere naturale nenule  $p1$  și  $p2$  ( $1 < p1 < p2 < n$ ), să ordoneze descrescător numerele din șir situate între pozițiile  $p1$  și  $p2$ , inclusiv, și să scrie noul șir în fișierul atestat.out, pe o linie, numerele separându-se prin câte un spațiu.

Notă: Programul va conține cel puțin un subprogram definit de utilizator.

Exemplu: de la tastatură se citesc: p1=2 și p2=4

atestat.in		Date de ieșire:
6	a)	1233 22 1785 56 15657 457
1233 22 1785 56 15657 457	b)	22
		Fișierul atestat.out conține:
	c)	1233 1785 56 22 15657 457

- 45.** Fișierul atestat.in conține două linii. Pe prima linie este scris un număr natural nenul  $n$ , ( $5 < n < 30$ ). Pe cea de-a doua linie a fișierului sunt scrise  $n$  numere reale separate prin câte un spațiu, reprezentând un șir de  $n$  numere reale.

Să se scrie un program în limbajul Pascal/C/C++, care:

- să afișeze pe ecran, în linie, toate numerele din șir, separate prin câte un spațiu;
- să afișeze pe următoarea linie a ecranului, media aritmetică a numerelor negative din șir, cu o precizie de 2 zecimale (dacă șirul nu conține numere negative se va afișa 0);
- să citească de la tastatură două numere naturale nenule  $p1$  și  $p2$  ( $1 < p1 < p2 < n$ ), să ordoneze crescător numerele din șir situate între pozițiile  $p1$  și  $p2$ , inclusiv, și să scrie noul șir în fișierul atestat.out, pe o linie, numerele separându-se prin câte un spațiu.

Notă: Programul va conține cel puțin un subprogram definit de utilizator.

Exemplu: de la tastatură se citesc: p1=2 și p2=4

atestat.in		Date de ieșire:
6	a)	-56.765 2.3 4.56 -1.2 -1.8 3
-56.765 2.3 4.56 -1.2 -1.8 3	b)	-19.92
		Fișierul atestat.out conține:
	c)	-56.765 -1.2 2.3 4.56 -1.8 3

46. Fișierul `atestat.in` conține două linii. Pe prima linie este scris un număr natural nenul  $n$ , ( $5 < n < 30$ ). Pe cea de-a doua linie a fișierului sunt scrise  $n$  numere naturale separate prin câte un spațiu, formate fiecare din cel mult 4 cifre, reprezentând un șir de  $n$  numere naturale distincte.

Să se scrie un program în limbajul Pascal/C/C++, care:

- să afișeze pe ecran, în linie, toate numerele din șir, separate prin câte un spațiu;
- să afișeze pe ecran, pe linii diferite, cel mai mic număr din șir și poziția acestuia;
- să scrie în fișierul `atestat.out`, pe o linie, separate prin câte un spațiu, toate numerele *perfecte* din șirul dat (dacă nu există astfel de numere, se va afișa mesajul "NU EXISTĂ NUMERE PERFECTE"). Un număr este *perfect* dacă este egal cu suma divizorilor lui pozitivi, exceptându-l pe el însuși, de exemplu:  $6 = 1+2+3$ .

Notă: Programul va conține cel puțin un subprogram definit de utilizator.

Exemplu:

atestat.in		Date de ieșire:
6	a)	28 11 81 496 6 100
28 11 81 496 6 100	b)	6 5
	c)	Fișierul <code>atestat.out</code> conține: 28 496 6

47. Fișierul `atestat.in` conține două linii. Pe prima linie este scris un număr natural nenul  $n$ , ( $5 < n < 30$ ). Pe cea de-a doua linie a fișierului sunt scrise  $n$  numere naturale separate prin câte un spațiu, formate fiecare din cel mult 4 cifre, reprezentând un șir de  $n$  numere naturale. Șirul conține cel puțin două numere pare.

Să se scrie un program în limbajul Pascal/C/C++, care:

- să afișeze pe ecran, în linie, toate numerele din șir, separate prin câte un spațiu;
- să afișeze pe următoarea linie a ecranului, media aritmetică a tuturor numerelor pare din șir ;
- să scrie în fișierul `atestat.out`, pe o linie, separate prin câte un spațiu, numerele de tip palindrom din șirul dat (dacă nu există astfel de numere, se va afișa mesajul "NU EXISTĂ NUMERE PALINDROM"). Un număr este palindrom dacă numărul citit de la stânga la dreapta este egal cu numărul citit de la dreapta la stânga, de exemplu: **33, 141, 2552**.

Notă: Programul va conține cel puțin un subprogram definit de utilizator.

Exemplu:

atestat.in		Date de ieșire:
6	a)	2552 56 32 444 46 1221

2552	56	32	444	46	1221	b)	626	
							Fișierul <b>atestat.out</b> conține:	
							c)	2552 444 1221

**48.** Din fișierul `date.in` se citeste un număr natural  $n$  ( $n \leq 100$ ) și apoi o matrice pătratică cu  $n$  linii și  $n$  coloane cu elemente numere naturale. Identificați numerele prime de pe diagonala secundară a matricii și ștergeți atât linia cât și coloana pe care se află acestea. Afișați matricea rezultată în fișierul `date.out`. Se vor scrie și apelurile funcțiilor pentru:

- citirea matricii
- afișarea matricii
- verificarea dacă un număr este prim
- ștergerea unei linii dintr-o matrice
- ștergerea unei coloane dintr-o matrice

Exemplu:

`date.in`

```
5
1 2 3 4 6
3 2 4 5 7
6 8 3 8 6
8 7 8 9 9
8 2 6 6 8
```

`date.out`

```
1 6
8 8
```

**49.** Se citeste un număr natural  $n$  cu cel mult 9 cifre. Scrieți o funcție care calculează și returnează cel mai mare număr care se poate forma cu cifrele lui  $n$ . Pe lângă funcția cerută se vor scrie și apelurile funcțiilor pentru:

- transformarea unui număr în vector de cifre
- ordonarea descrescătoare a unui vector
- transformarea din vector de cifre în număr.

Exemplu: Cu cifrele numărului 3426096 ce mai mare număr care se poate forma este 9664320.

Subprogramul sub primește prin intermediul parametrului  $n$  (cel mult 9) dimensiunea unei matrice pătrate, prin intermediul parametrului  $a$  matricea pătratică de dimensiune  $n$  cu elementele numere întregi și prin intermediul parametrului  $k$  un număr natural nenul din intervalul  $[2, 2n]$ . Subprogramul sub calculează și returnează suma tuturor elementelor  $a[i][j]$  cu proprietatea că  $i+j=k$ .

**50.** Scrieți programul care citește un număr natural  $n$  și un tablou de numere întregi cu  $n$  linii și  $n$  coloane și afișează suma elementelor din tablou aflate strict deasupra diagonalei secundare a tabloului, folosind apeluri ale subprogramului sub, definit conform cerinței.

Exemplu:

$n=4$

1 6 3 1

6 1 3 1

1 3 1 6

3 1 6 1

se va afișa 18

**51.** Fișierul DATE.IN conține pe prima linie două numere naturale nenule  $n$  și  $m$ . Scrieți un program care memorează în fișierul text DATE.OUT toate numerele prime din intervalul deschis  $(n, m)$ . Numerele se scriu în ordine crescătoare, câte 10 numere pe fiecare linie a fișierului.

Ex.

DATE.IN

87 241

DATE.OUT

89 97 101 103 107 109 113 127 131 137

139 149 151 157 163 167 173 179 181 191

193 197 199 211 223 227 229 233 239

**INSPECTOR ȘCOLAR PENTRU INFORMATICĂ,  
PROF. DANIELA IOANA TĂTARU**