

SUBIECTUL al II-lea

(40 de puncte)

1. Algoritmul alăturat este reprezentat în pseudocod.

S-a notat cu $a \% b$ restul împărțirii numărului natural a la numărul natural nenul b și cu $[c]$ partea întreagă a numărului real c .

- Scriveți ce se afișează dacă se citește numărul 10. (6p.)
- Scriveți două valori distincte care pot fi citite astfel încât, în urma executării algoritmului, pentru fiecare dintre acestea, numărul de valori egale cu 1 afișate să fie 4. (6p.)
- Scriveți programul C/C++ corespunzător algoritmului dat. (10p.)
- Scriveți în pseudocod un algoritm echivalent cu cel dat, înlocuind adecvat prima structură repetitivă **cât timp...execută** cu o structură de tip **pentru...execută**. (6p.)

```

citește n
(număr natural,  $n \geq 2$ )
i ← 1
cât timp i ≤ n execută
    m ← i
    cât timp m % 2 = 0 execută
        m ← [m/2]
    dacă m ≠ i atunci
        scrie m, ' '
    i ← i + 1

```

- Un graf neorientat cu 6 noduri, numerotate de la 1 la 6, are muchiile [1,2], [2,3], [2,4], [2,5], [4,5], [4,6], [5,6]. Scrieți listele de adiacență ale unui graf parțial al său care să fie conex și fără cicluri. (6p.)
- Variabilele c și i sunt de tip întreg, iar variabila s permite memorarea unui șir de cel mult 20 de caractere. Se citesc de la tastatură 10 cuvinte, formate din litere mici ale alfabetului englez și separate prin Enter. Scrieți secvența de mai jos, înlocuind punctele de suspensie astfel încât, în urma executării secvenței obținute, variabila c să memoreze valoarea 1 dacă există printre cuvintele citite cel puțin unul format din două litere și care să conțină o vocală și o consoană, sau valoarea 0 altfel. Se consideră vocale literele a, e, i, o, u .

Exemplu: dacă se citesc cuvintele alăturate, variabila c are valoarea 1.

```

c = .....;
for(i=1; i<=10; i++) {  cin>>s;  |  scanf("%s", s);
                        .....
}

```

dacă
au
plecat
el
nu
primește
și
înghetata
de
fragi

(6p.)

SUBIECTUL al III-lea

(30 de puncte)

- La un laborator sunt studiate aglomerările de fulgi de nea formate din câte **nouă** cristale de **patru** tipuri diferite date (notate cu 1, 2, 3 sau 4), astfel încât din fiecare tip să existe cel puțin câte un cristal. O astfel de aglomerare de fulgi a fost reprezentată printr-un număr natural, în care fiecare cifră reprezintă tipul unui cristal. Subprogramul **fulg** are un parametru, n , prin care primește un număr natural ($n \in [0, 10^9]$). Subprogramul returnează valoarea 1, dacă prin n este reprezentată o aglomerare de fulgi de nea dintre cele studiate, sau 0 în caz contrar. Scrieți în C/C++ definiția completă a subprogramului.

Exemplu: dacă $n=112243413$ subprogramul returnează 1, iar dacă $n=12314$ sau $n=112253513$ sau $n=112243457$ sau $n=111122223$, subprogramul returnează 0. (10p.)

- Fiind date două numere naturale a și b , numim valoare **generată** de a și b un număr obținut din a prin alipirea la stânga sau la dreapta sa a cifrelor lui b , în ordinea în care apar în acesta.

Exemplu: dacă $a=123$ și $b=45$, se pot obține două numere generate de acestea: 12345 și 45123.

Scrieți un program C/C++ care citește de la tastatură două numere naturale din intervalul [2, 50], m și n , și construiește în memorie un tablou bidimensional cu m linii, numerotate de la 1 la m , și n coloane, numerotate de la 1 la n , în care fiecare element este egal cu cea mai mică valoare generată de numerele de ordine ale liniei, respectiv coloanei pe care se află.

Programul afișează pe ecran tabloul obținut, fiecare linie a tabloului pe câte o linie a ecranului, elementele fiecărei linii fiind separate prin câte un spațiu.

Exemplu: pentru $m=5$ și $n=4$ se obține tabloul alăturat.

(10p.)

11	12	13	14
12	22	23	24
13	23	33	34
14	24	34	44
15	25	35	45

- Fișierul **numere.in** conține un șir de cel mult 10^6 numere naturale din intervalul [0, 99]. Numerele din fișier sunt separate prin câte un spațiu.

Se cere să se determine primul și ultimul număr din șir care conțin cea mai mare cifră ce apare în scrierea numerelor din fișier. Numerele determinate se afișează pe ecran, în ordinea apariției lor în șir, separate printr-un spațiu. Dacă nu există două astfel de numere pe poziții distincte, se afișează pe ecran mesajul **nu exista**. Proiectați un algoritm eficient din punctul de vedere al timpului de executare și al spațiului de memorie utilizat.

Exemplu: dacă fișierul conține numerele 34 5 38 30 87 70 11 8 82 25 se afișează pe ecran 38 82, dacă fișierul conține numerele 34 5 38 30 87 70 11 8 38 25 se afișează pe ecran 38 38, iar dacă fișierul conține numerele 34 5 38 30 se afișează pe ecran **nu exista**.

- Descrieți în limbaj natural algoritmul proiectat, justificând eficiența acestuia. (2p.)

- Scrieți programul C/C++ corespunzător algoritmului proiectat. (8p.)