2019 CLUJ

**UNIVERSITATEA BABEŞ-BOLYAI**
**FACULTATEA DE MATEMATICĂ ŞI INFORMATICĂ**

## Concurs Mate-Info - model Model 2

**Proba scrisă la Informatică**

**În atenţia concurenţilor:**

1. Se consideră că indexarea şirurilor începe de la 1.
2. Problemele tip grilă (Partea A) pot avea unul sau mai multe răspunsuri corecte. Răspunsurile trebuie scrise de candidat pe foaia de concurs (nu pe foaia cu enunţuri). Obţinerea punctajului aferent problemei este condiţionată de identificarea tuturor variantelor de răspuns corecte şi numai a acestora.
3. Pentru problemele din Partea B se cer rezolvări complete pe foaia de concurs.
   a. Rezolvările se vor scrie în *pseudocod* sau *într-un limbaj de programare* (*Pascal/C/C++*).
   b. Primul criteriu în evaluarea rezolvărilor va fi ***corectitudinea*** algoritmului, iar apoi ***performanţa*** din punct de vedere al *timpului de executare* şi al *spaţiului de memorie utilizat*.
   c. ***Este obligatorie descrierea şi justificarea*** (sub) algoritmilor înaintea rezolvărilor. Se vor scrie, de asemenea, ***comentarii*** pentru a uşura înţelegerea detaliilor tehnice ale soluţiei date, a semnificaţiei identificatorilor, a structurilor de date folosite etc. Neîndeplinirea acestor cerinţe duce la pierderea a 10% din punctajul aferent subiectului.
   d. Nu se vor folosi funcţii sau biblioteci predefinite (de exemplu: *STL*, funcţii predefinite pe şiruri de caractere).

## Partea A (60 puncte)

### A.1. Oare ce face? (6 puncte)

Se consideră subalgoritmul `alg(x, b)` cu parametrii de intrare două numere naturale ***x*** şi ***b*** ($1 \le x \le 1000$, $1 < b \le 10$).

```
Subalgoritm alg(x, b):
   s ← 0
   CâtTimp x > 0 execută
      s ← s + x MOD b
      x ← x DIV b
   SfCâtTimp
   returnează s MOD (b - 1) = 0
SfSubalgoritm
```

Precizaţi efectul acestui subalgoritm.

A. calculează suma cifrelor reprezentării în baza ***b*** a numărului natural ***x***
B. verifică dacă suma cifrelor reprezentării în baza ***b*** - 1 a numărului ***x*** este divizibilă cu ***b*** - 1
C. verifică dacă numărul natural ***x*** este divizibil cu ***b*** - 1
D. verifică dacă suma cifrelor reprezentării în baza ***b*** a numărului ***x*** este divizibilă cu ***b*** - 1

### A.2. Ce se afişează? (6 puncte)

Se consideră următorul program:

**Varianta C++/C**

```
int sum(int n, int a[], int s){
   s = 0;
   int i = 1;
   while(i <= n){
      if(a[i] != 0) s += a[i];
      ++i;
   }
   return s;
}


int main(){
   int n = 3, p = 0, a[10];
   a[1] = -1; a[2] = 0; a[3] = 3;
   int s = sum(n, a, p);
   cout << s << ";" << p; // printf("%d;%d", s, p);
   return 0;
}
```

**Varianta Pascal**

```
type vector = array[1..10] of integer;

function sum(n:integer; a:vector; s:integer):integer;
  var i : integer;
  begin
     s := 0; i := 1;
     while (i <= n) do
        begin
           if (a[i] <> 0) then s := s + a[i];
           i := i + 1;
        end;
     sum := s;
  end;
var n, p, s : integer;
          a : vector;
begin
   n := 3; a[1] := -1; a[2] := 0; a[3] := 3; p := 0;
   s := sum(n, a, p);
   writeln(s, ';', p);
end.
```

Care este rezultatul afişat în urma execuţiei programului?

A. 0;0
B. 2;0
C. 2;2
D. Niciun rezultat nu este corect



### A.3. Expresie logică (6 puncte)

Se consideră următoarea expresie logică (`X OR Z) AND (NOT X OR Y`). Alegeţi valorile pentru ***X***, ***Y***, ***Z*** astfel încât evaluarea expresiei să dea rezultatul `TRUE`:

A. ***X*** ← FALSE; ***Y*** ← FALSE; ***Z*** ← TRUE;
B. ***X*** ← TRUE; ***Y*** ← FALSE; ***Z*** ← FALSE;
C. ***X*** ← FALSE; ***Y*** ← TRUE; ***Z*** ← FALSE;
D. ***X*** ← TRUE; ***Y*** ← TRUE; ***Z*** ← TRUE;

### A.4. Calcul (6 puncte)

Fie subalgoritmul `calcul(a, b)` cu parametrii de intrare ***a*** şi ***b*** numere naturale, $1 \le a \le 1000$, $1 \le b \le 1000$.

```
1.    Subalgoritm calcul(a, b):
2.       Dacă a ≠ 0 atunci
3.          returnează calcul(a DIV 2, b + b) + b * (a MOD 2)
4.       SfDacă
5.       returnează 0
6.    SfSubalgoritm
```

Care din afirmaţiile de mai jos sunt false?

A. dacă ***a*** şi ***b*** sunt egale, subalgoritmul returnează valoarea lui ***a***
B. dacă ***a*** = 1000 şi ***b*** = 2, subalgoritmul se autoapelează de 10 ori
C. valoarea calculată şi returnată de subalgoritm este ***a*** / 2 + 2 * ***b***
D. instrucţiunea de pe linia 5 nu se execută niciodată

### A.5. Identificare element (6 puncte)

Se consideră şirul (1, 2, 3, 2, 5, 2, 3, 7, 2, 4, 3, 2, 5, 11, ...) format astfel: plecând de la şirul numerelor naturale, se înlocuiesc numerele care nu sunt prime cu divizorii lor proprii, fiecare divizor ***d*** fiind considerat o singură dată pentru fiecare număr. Care dintre subalgoritmi determină al ***n***-lea element al acestui şir (***n*** - număr natural, $1 \le n \le 1000$)?

A.
```
Subalgoritm identificare(n):
   a ← 1, b ← 1, c ← 1
   CâtTimp c < n execută
      a ← a + 1, b ← a, c ← c + 1, d ← 2
      f ← false
      CâtTimp c ≤ n şi d ≤ a DIV 2 execută
         Dacă a MOD d = 0 atunci
            c ← c + 1, b ← d, f ← true
         SfDacă
         d ← d + 1
      SfCâtTimp
      Dacă f atunci
         c ← c - 1
      SfDacă
   SfCâtTimp
   returnează b
SfSubalgoritm
```

B.
```
Subalgoritm identificare(n):
   a ← 1, b ← 1, c ← 1
   CâtTimp c < n execută
      c ← c + 1, d ← 2
      CâtTimp c ≤ n şi d ≤ a DIV 2 execută
         Dacă a MOD d = 0 atunci
            c ← c + 1, b ← d
         SfDacă
         d ← d + 1
      SfCâtTimp
      a ← a + 1, b ← a
   SfCâtTimp
   returnează b
SfSubalgoritm
```

C.
```
Subalgoritm identificare(n):
   a ← 1, b ← 1, c ← 1
   CâtTimp c < n execută
      a ← a + 1, d ← 2
      CâtTimp c < n şi d ≤ a execută
         Dacă a MOD d = 0 atunci
            c ← c + 1, b ← d
         SfDacă
         d ← d + 1
      SfCâtTimp
   SfCâtTimp
   returnează b
SfSubalgoritm
```

D.
```
Subalgoritm identificare(n):
   a ← 1, b ← 1, c ← 1
   CâtTimp c < n execută
      b ← a, a ← a + 1, c ← c + 1, d ← 2
      CâtTimp c ≤ n şi d ≤ a DIV 2 execută
         Dacă a MOD d = 0 atunci
            c ← c + 1, b ← d
         SfDacă
         d ← d + 1
      SfCâtTimp
   SfCâtTimp
   returnează b
SfSubalgoritm
```

### A.6. Factori primi (6 puncte)

Fie subalgoritmul `factoriPrimi(n, d, k, x)` care determină cei ***k*** factori primi ai unui număr natural ***n***, începând căutarea factorilor primi de la valoarea ***d***. Parametrii de intrare sunt numerele naturale ***n***, ***d*** şi ***k***, iar parametrii de ieşire sunt şirul ***x*** cu cei ***k*** factori primi ($1 \le n \le 10000$, $2 \le d \le 10000$, $0 \le k \le 10000$).

```
Subalgoritm factoriPrimi(n, d, k, x):
   Dacă n MOD d = 0 atunci
      k ← k + 1
      x[k] ← d
   SfDacă
   CâtTimp n MOD d = 0 execută
      n ← n DIV d
   SfCâtTimp
```

```
    Dacă n > 1 atunci
        factoriPrimi(n, d + 1, k, x)
    SfDacă
SfSubalgoritm
```

Stabiliți de câte ori se autoapelează subalgoritmul `factoriPrimi(n, d, k, x)` prin execuția următoarei secvențe de instrucțiuni:

```
n ← 120
d ← 2
k ← 0
factoriPrimi(n, d, k, x)
```

A. de 3 ori
B. de 5 ori
C. de 9 ori
D. de același număr de ori ca și în cadrul secvenței de instrucțiuni:

```
n ← 750
d ← 2
k ← 0
factoriPrimi(n, d, k, x)
```

**A.7. Oare ce face? (6 puncte)**

Se consideră subalgoritmul `expresie(n)`, unde ***n*** este un număr natural ($1 \leq n \leq 10000$).

```
Subalgoritm expresie(n):
    Dacă n > 0 atunci
        Dacă n MOD 2 = 0 atunci
            returnează -n * (n + 1) + expresie(n - 1)
        altfel
            returnează n * (n + 1) + expresie(n - 1)
        SfDacă
    altfel
        returnează 0
    SfDacă
SfSubalgoritm
```

Precizați forma matematică a expresiei $E(n)$ calculată de acest subalgoritm:

A. $E(n) = 1*2 - 2*3 + 3*4 + ... + (-1)^{n+1} * n * (n+1)$
B. $E(n) = 1*2 - 2*3 + 3*4 + ... + (-1)^{n} * n * (n+1)$
C. $E(n) = 1*2 + 2*3 + 3*4 + ... + (-1)^{n+1} * n * (n+1)$
D. $E(n) = 1*2 + 2*3 + 3*4 + ... + (-1)^{n} * n * (n+1)$

**A.8. Expresie logică (6 puncte)**

Se consideră următoarea expresie logică: `(NOT Y OR Z) OR (X AND Y)`. Alegeți valorile pentru ***X***, ***Y***, ***Z*** astfel încât rezultatul evaluării expresiei să fie *adevărat*:

A. *X ← FALSE; Y ← FALSE; Z ← FALSE;*
B. *X ← FALSE; Y ← FALSE; Z ← TRUE;*
C. *X ← FALSE; Y ← TRUE; Z ← FALSE;*
D. *X ← TRUE; Y ← FALSE; Z ← TRUE;*

**A.9. Valori ale parametrului de intrare (6 puncte)**

Se consideră următorul subalgoritm:

```
Subalgoritm SA9(a):
    Dacă a < 50 atunci
        Dacă a MOD 3 = 0 atunci
            returnează SA9(2 * a - 3)
        altfel
            returnează SA9(2 * a - 1)
        SfDacă
    altfel
        returnează a
    SfDacă
SfSubalgoritm
```

Pentru care dintre valorile parametrului de intrare ***a*** subalgoritmul va returna valoarea 61?

A. 16
B. 61
C. 4
D. 31

**A.10. Instrucțiuni lipsă (6 puncte)**

Se dă următorul subalgoritm:

```
1:  Subalgoritm cautare(x, n, val):
2:      Dacă n = 1 atunci
3:          returnează (x[1] = val)
4:      altfel
5:          returnează cautare(x, n - 1, val)
6:      SfDacă
7:  SfSubalgoritm
```

Ce instrucțiune sau instrucțiuni trebuie adăugate și unde astfel încât în urma apelului, subalgoritmul `cautare(x, n, val)` să determine dacă elementul ***val*** face sau nu parte din șirul ***x*** cu ***n*** elemente (***n*** număr natural strict mai mare ca zero)?

A. Linia 5 trebuie modificată în: `returnează ((x[n] = val) și cautare(x - 1, n, val))`
B. Linia 5 trebuie modificată în: `returnează ((x[n] = val) sau cautare(x, n - 1, val))`
C. Linia 5 trebuie modificată în: `dacă (x[n] = val) atunci returnează true altfel returnează cautare(x, n - 1, val)`
D. nu trebuie modificată nici o instrucțiune

## Partea B (30 puncte)

**1. Prefix (15 puncte)**

*Cifra de control* a unui număr natural se determină calculând suma cifrelor numărului, apoi suma cifrelor sumei și așa mai departe până când suma obținută reprezintă un număr cu o singură cifră. De exemplu, *cifra de control* a numărului 182 este 2 (1 + 8 + 2 = 11, 1 + 1 = 2).

Un număr ***p*** format din exact ***k*** cifre este *prefix* al unui număr ***q*** cu cel puțin ***k*** cifre dacă numărul format din primele ***k*** cifre ale numărului ***q*** (parcurse de la stânga la dreapta) este egal cu ***p***. De exemplu, 17 este prefix al lui 174, iar 1713 este prefix al lui 1713242.

Se consideră un număr ***nr*** natural ($0 < nr \leq 30000$) și o matrice (un tablou bidimensional) ***A*** cu ***m*** linii și ***n*** coloane ($0 < m \leq 100$, $0 < n \leq 100$), având ca elemente numere naturale mai mici decât 30 000. Se consideră și subalgoritmul `cifrăControl(x)` pentru determinarea cifrei de control asociată numărului ***x***:

```
Subalgoritm cifraControl(x):
    s ← 0
    CâtTimp x > 0 execută
        s ← s + x MOD 10
        x ← x DIV 10
        Dacă x = 0 atunci
            Dacă s < 10 atunci
                Returnează s
            altfel
                x ← s
                s ← 0
            SfDacă
        SfDacă
    SfCâtTimp
    returnează s
SfSubalgoritm
```

**Cerințe:**

a. Scrieți o variantă *recursivă* (fără structuri repetitive) a subalgoritmului `cifrăControl(x)` care are același antet și același efect cu acesta. **(5 puncte)**
b. Scrieți modelul matematic al variantei recursive a subalgoritmului `cifrăControl(x)` (dezvoltat la punctul a). **(3 puncte)**
c. Scrieți un subalgoritm care, folosind subalgoritmul `cifrăControl(x)`, determină cel mai lung prefix (notat ***prefix***) al numărului ***nr*** care se poate construi folosind cifrele de control corespunzătoare elementelor din matricea dată. O astfel de cifră de control poate fi folosită de oricâte ori în construirea prefixului. Dacă nu se poate construi un prefix, ***prefix*** va fi -1. Parametrii de intrare ai subalgoritmului sunt numerele ***nr***, ***m***, ***n***, matricea ***A***, iar parametrul de ieșire este ***prefix***. **(7 puncte)**

***Exemplu***: dacă avem ***nr*** = 12319, ***m*** = 3 și ***n*** = 4 și matricea $A = \begin{pmatrix} 182 & 12 & 274 & 22 \\ 22 & 1 & 98 & 56 \\ 5 & 301 & 51 & 94 \end{pmatrix}$,