

# Sesiunea 14

---

## Agenda

- Rezolvare Varianta de rezerva pentru Iunie 2025
- Pentru acasa

## Rezolvare Varianta de rezerva pentru Iunie 2025

- Link:  
[https://www.pbinfo.ro/resurse/9dc152/examene/2025/E\\_d\\_Informatica\\_2025\\_sp\\_MI\\_C\\_var\\_07\\_LRO.pdf](https://www.pbinfo.ro/resurse/9dc152/examene/2025/E_d_Informatica_2025_sp_MI_C_var_07_LRO.pdf)

## Subiectul I

### 1. Rezolvare:

- Expresia alaturata are valoarea:
  - $7.7 + 7 = 14.7$
  - Atentie la faptul ca pentru prima operatie avem un operand de tip real (float) si unul de tip intreg (int) rezultatul va fi de tipul real (float) iar pentru a doua operatie, ambii operanzi sunt intregi.
- $a \Rightarrow$  Fals
- $b \Rightarrow$  Adevarat
- $c \Rightarrow$  Fals
- $d \Rightarrow$  Fals
- Raspuns corect: **b**

### 2. Rezolvare:

- $a \Rightarrow$  sintaxa gresita. Numele atributului este **expira**, **data** este tipul atributului.
- $b \Rightarrow$  sintaxa gresita, variabila prin care accesam membrii este **m**
- $c \Rightarrow$  corect dpdv sintactic si respecta si enuntul
- $d \Rightarrow$  sintaxa gresita. Nu exista nici o proprietate in **medicament** care sa se numeasca **an** sau **luna**
- Raspuns corect: **c**

### 3. Rezolvare:

Notam multimea astfel:

```
{ 0      1      2      3      4      5  }  
{balerini, botine, cizme, ghete, sandale, tenisi }
```

Stim ca nu avem voie in aceeasi comanda:  
(ghete, cizme) sau (balerini, sandale)  
adica  
(**3,2**) sau (**0,4**)

Primele 4 solutii sunt:  
 (balerini, botine, cizme),  
 (balerini, botine,ghete),  
 (balerini, botine, teniși),  
 (balerini, cizme, teniși)








Adica:  
 (0,1,2),  
 (0,1,3),  
 (0,1,5),  
 (0,2,5)

Inainte de solutia  
 (cizme, sandale, tenisi)  
 (2,4,5)



Avem:  
 (1,4,5)  
 (botine, sandale, teniși)

- Raspuns corect: d

#### 4. Rezolvare:

- Mai jos avem graful obtinut din muchiile prezentate in enunt  Graf initial
- Acum o sa desenam pentru fiecare nod, cum arata arborele daca nodul ales ar fi radacina:
- Daca avem nodul 1 ca si radacina avem arborele de mai jos:  Nod 1 radacina
- Daca avem nodul 2 ca si radacina avem arborele de mai jos:  Nod 2 radacina
- Daca avem nodul 3 ca si radacina avem arborele de mai jos:  Nod 3 radacina
- Daca avem nodul 4 ca si radacina avem arborele de mai jos:  Nod 4 radacina
- Daca avem nodul 5 ca si radacina avem arborele de mai jos:  Nod 5 radacina
- Daca avem nodul 6 ca si radacina avem arborele de mai jos:  Nod 6 radacina
- Observam ca doar in cazurile cand nodurile 3,4,5,6 sunt radacini avem exact 3 frunze.
- Raspuns corect: d

#### 5. Rezolvare:

- Din enunt avem graful de mai jos:  Graf cf enunt
- Componenta tare conexa: Graful se numește tare conex dacă între oricare două noduri distincte există cel puțin un drum.
- Daca adaugam muchia:
  - [3,2]
  - Obținem graful de mai jos cu 2 componente tare conexe
  - Nota: o componenta formata dintr-un nod poate fi vazuta ca o componenta tare conexa
-  Solutie
- Raspuns corect: -> a

## Subiectul II

1.    ◦   a =&gt;

```

n = 252
d = 2, x = 1, y = 1
cat timp 252 > 1 executa
    daca 252 % 2 == 0
        x = d => x = 2, y = 2
        cat timp 252 % 2 == 0
            n = 252 / 2 = 126
        cat timp 126 % 2 == 0
            n = 126 / 2 = 63
        cat timp 63 % 2 == 0 => fals
    d = 3
cat timp 63 > 1
    daca 63 % 3 == 0
        x = 3; y = 6
        cat timp 63 % 3 == 0
            n = 63/3 = 21
        cat timp 21 % 3 == 0
            n = 21 / 3 = 7
        cat timp 7 % 3 == 0 fals
    d = 4
cat timp 7 > 1
    daca 7% 4 == 0 fals
    d = 5
cat timp 7 > 1
    daca 7 % 5 == 0 fals
    d = 6
cat timp 7 > 1
    daca 7% 6 == 0 fals
    d = 7
cat timp 7 > 1
    daca 7 % 7 == 0
        x = 7; y = 6 * 7 = 42
        cat timp 7%7 == 0
            n = 7 / 7 = 1
cat timp 1 > 1 fals

scrie 7 ' ' 42

```

◦   b =&gt;

Programul de mai sus afiseaza prin x ultimul factor prim descoperit  
si prin y produsul factorilor primi distincti aia lui n  
Numere din intervalul 10-100 care are un singur factor prim:

16  
32

o c =>

```
#include <iostream>
using namespace std;

int estePrim(int n);

int main() {
    int n;
    cin >> n;
    int d = 2, x = 1, y = 1;
    while (n > 1) {
        if (n%d == 0) {
            x = d;
            y = y * d;
            while (n % d == 0) {
                n = n / d;
            }
        }
        d++;
    }
    cout << x << " " << y;
    return 0;
}
```

o d =>

```
citește n
(număr natural,  $n \geq 2$ )
d<-2; x<-1; y<-1
┌cât timp n>1 execută
│┌dacă n%d=0 atunci
││ x<-d; y<-y*d
││┌daca n%d = 0 atunci
│││┌execută
││││ n<-[n/d]
│││└cât timp n%d=0
│││■
││└■
│└d<-d+1
└■
scrie x, ' ', y
```

2. Rezolvare:

```
f(17)
x = 17
```

```

    if (x<=4) false
    else return 17 - f(15)
        x = 15
        if(x<= 4) false
        else return 15 - f(13)
            x = 13
            if (x <= 4) false
            else return 13 - f(11)
                x = 11
                if (x <= 4) false
                else return 11 - f (9)
                    x = 9
                    if (x <= 4) false
                    else return 9 - f(7)
                        x = 7
                        if (x <= 4) false
                        else return 7 - f(5)
                            x = 5
                            if (x <= 4) false
                            else return 5 - f(3)
                                x = 3
                                if (x <= 4)

return 3
17-15-13-11-9-7-5-4 doar ca facem de la capat adica
17 - (15 - (13 - (11 - (9 - (7 - (5 - 3)))))) =
17 - (15 - (13 - (11 - (9 - (7 - 2)))))) =
17 - (15 - (13 - (11 - (9 - 5)))) =
17 - (15 - (13 - (11 - 4))) =
17 - (15 - (13 - 7))
17 - (15 - 6) =
17 - 9 = 8

```

Pentru f(2) este simplu avem direct 2  
 Raspuns corect: 2 si 8

### 3. Rezolvare:

```

strcpy(s1,"parcare"); // s1="parcare"
strcpy(s2,strstr(s1,"car")); //s2=carea
n=strlen(s2); //n = 5
strcpy(s1+n-2, s2+n-2); // asta devine strcpy(s1+3, s2+3)=> s1=parea
//n = 5, s1=parea

```

## Subiectul III

### 1. Rezolvare:

```
#include <iostream>
```

```

using namespace std;

void consecutiv(int n, int &f);

int main() {
    int n = 19;
    int f;
    consecutiv(n, f);
    cout << f;
    return 0;
}

void consecutiv(int n, int &f) {
    int rezultat = 1;
    while (rezultat <=n) {
        if ((rezultat -1) * rezultat < n && n<= rezultat * (rezultat+1))
        {
            f = rezultat;
            break;
        }
        rezultat++;
    }
}

```

## 2. Rezolvare:

- Aici avem doua variante.

1. Varianta in care atunci cand parcurgem coloanele, mai facem un for care merge de la j-ul curent inca 3 pozitii in fata

```

#include <iostream>

using namespace std;

int main() {
    int nr,np;
    cin>>nr>>np;
    int parcare[nr][np];
    for (int i =0; i<nr; i++) {
        for (int j=0; j<np; j++) {
            cin >> parcare[i][j];
        }
    }
    int contorTriplete = 0;
    for (int i = 0; i < nr; i++) {
        if (i == 0 || i == nr-1) {
            for (int j =0; j < np; j++) {
                int existaTripleta = 1;
                for (int k = j; k < j+3 && k < np ; k++) {
                    if (parcare[i][k] == 1) {
                        existaTripleta = 0;
                        break;
                    }
                }
            }
        }
    }
}

```

```

    }
    }
    if (existaTripleta == 1) {
        contorTriplete++;
    }
}
}
cout << contorTriplete << endl;
return 0;
}

```

2. Varianta in care nu mai facem un extra `for` si hardcodam valorile

```

#include <iostream>
using namespace std;

int main() {
    int nr, np;
    cin >> nr >> np;
    int parcare[51][51]; // dimensiuni maxime conform
    enunțului

    for (int i = 0; i < nr; i++) {
        for (int j = 0; j < np; j++) {
            cin >> parcare[i][j];
        }
    }

    int contorTriplete = 0;

    // verificăm doar primul rând (i=0) și ultimul rând (i=nr-
    1)
    for (int i = 0; i < nr; i++) {
        if (i == 0 || i == nr-1) {
            for (int j = 0; j <= np - 3; j++) { // mergem doar
                până la np-3
                if (parcare[i][j] == 0 && parcare[i][j+1] == 0
                && parcare[i][j+2] == 0) {
                    contorTriplete++;
                }
            }
        }
    }

    cout << contorTriplete << endl;
    return 0;
}

```

- Alege varianta care ti se pare tie mai clara, sau propria ta varianta.

### 3. Rezolvare:

◦ a

Observam ca enuntul problemei cere sa gasim 2 minime si cel mai mare numar din fisier.

Inainte de a trece la rezolvarea propriu zisa a problemei, observam ca avem nevoie de a transforma fiecare numar din fisier in inversul sau. De aceea o sa scriu un subprogram care primeste un numar si intoarce inversul sau, pentru a ne usura logica ce urmeaza.

Continuam cu declararea variabilele corespunzatoare pentru cele doua minime (minim1, minim2) si pentru valoarea maxima pe care o sa o gasim (maxim). minim1 si minim2 vor fi initializate cu valoarea 10.000 deoarece stim ca numerele din fisier au maximum valoarea 9999, pe aceleasi principiu, initializam variabila maxim cu 0 deoarece stim ca numerele noastre sunt mai mari sau egale cu 1001.

Urmatorul pas este sa parcurgem fisierul, numar cu numar si aplicand logica urmatoare pentru fiecare numar:

- Daca inversul numarului citit este mai mic decat minim1 atunci salvam in minim2 ce am avut in minim1 si in minim1 punem inversul.

- Altfel daca inversul nmarului citit este mai mic decat minim2 atunci salvam in minim2, inversul numarului citit.

- Daca inversul numarului citit nu este nici mai mic decat minim1, nici mai mic decat minim2, verificam daca este mai mare decat maxim, caz in care salvam in maxim valoarea inversului.

Dupa ce am parcurs fisierul, mai facem o verificare, anume daca exista cel putin una dintre cele 3 variabile (minim1, minim2, si maxim) ale carei valori nu a fost schimbata, atunci afisam "nu exista" altfel afisam numerele conform cerintei adica minim1, urmat de un spatiu, urmat de maxim, inca un spatiu si ultima oara, minim2.

Algoritmul este eficient din punct de vedere al timpului de executie deoarece rezultatul este calculat dintr-o singura parcurgere a fisierului plus atunci cand intalnim valori pe care le avem

deja salvate intr-una din cele 3 variabile, sarim peste iteratia respectiva. Totodata, algoritmul este eficient din punct de vedere al memoriei deoarece din totalul de  $10^6$  numere cate pot fi in fisier, noi tinem in memorie doar unul si nu folosim alte structuri de date pentru a stoca numerele.

◦ b

```
#include <iostream>
#include <fstream>

using namespace std;

int construiesteInvers(int n);
```



```

int main() {
    ifstream fin("bac.in");
    int minim1 = 10000,minim2=10000,maxim=0;
    int numar;
    while (fin >> numar) {
        int invers = construiesteInvers(numar);
        if (invers == minim1 || invers == minim2 || invers ==
maxim) continue;
        if (invers < minim1) {
            minim2 = minim1;
            minim1 = invers;
        } else if (invers < minim2) {
            minim2 = invers;
        } else if (invers > maxim) {
            maxim = invers;
        }
    }

    if (minim1 == 10000 || minim2 == 10000 || maxim == 0) {
        cout << "nu exista";
    } else {
        cout << minim1 << " " << maxim << " " << minim2 <<
endl;
    }
    fin.close();
    return 0;
}

int construiesteInvers(int n) {
    int rezultat=0;
    while (n > 0) {
        int cifra = n % 10;
        rezultat = rezultat * 10 + cifra;
        n= n / 10;
    }

    return rezultat;
}

```

## Pentru acasa

1. Reia singur varianta de rezerva si daca sunt probleme compara cu ce am facut noi si/sau intreaba-ma oricand
2. Varianta August 2025 -> link mai jos

- Link:

[https://www.pbinfo.ro/resurse/9dc152/examene/2025/E\\_D\\_Informatica\\_2025\\_sp\\_MI\\_C\\_var\\_04\\_LRO.pdf](https://www.pbinfo.ro/resurse/9dc152/examene/2025/E_D_Informatica_2025_sp_MI_C_var_04_LRO.pdf)