

Sesiunea 6

Agenda

- Recapitulare sesiunea anterioara
- Rezolvare tema
- Introducere in vectori de frecventa
- Introducere in Matrici/Vectori 2D
- Exercitii tema

Recapitulare sesiunea anterioara

- Ce ti-a atras atentia cel mai mult din ce am facut?
- Care parte ti s-a parut cea mai complicata?
- A fost ceva care inainte parea greu si acum e mai usor de inteles?
- Din ce am facut pana acum, de care iti este cel mai frica sa iti pice la BAC?
 - De ce iti este frica, nu scapi, asa ca mai bine ne focusam pe chestiile cele mai neclare.

Rezolvare tema

- Nota: exercitiul 1 il ai deja rezolvat in sesiunile anterioare. Exercitiul 2 se rezolva ducandu-te pe site-ul pbinfo, acolo ai direct toate transformarile.
3. Enuntul il gasesti in fisierul de sesiunea trecuta.

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

// Nota: doar ce este intre linii trebuie sa scrii pe foaia de la bac
//-----
struct bijuterie {
    int greutate;
    struct {char denumire[31]; int pret; } metal;
}b;
//-----
int main() {

    // Nota: partea de mai jos nu trebuie sa o scrii la bac. Eu o fac
    doar ca sa testez ca totul merge bine.
    strcpy(b.metal.denumire, "aur");
    b.metal.pret = 100;
    b.greutate = 121;

    cout << b.greutate * b.metal.pret<<endl;

    return 0;
}
```

4. Enuntul il gasesti in fisierul de sesiunea trecuta.

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

// Nota: doar ce este intre linii trebuie sa scrii pe foaia de la bac
//-----
struct prajitura {
    int cod;
    float pret;
    int informatii[3];
}p;
//-----

int main() {

    // Nota: partea de mai jos nu trebuie sa o scrii la bac. Eu o fac
    // doar ca sa testez ca totul merge bine.
    p.cod = 20;
    p.pret = 21.34;
    p.informatii[0] = 1;
    p.informatii[1] = 2;
    p.informatii[2] = 3;

    cout << "Prajitura p are codul " << p.cod << "; pretul: " << p.pret
    << "; si urmatoarele informatii: "<<endl;
    cout << "Tipul glazurii: " << p.informatii[0] << endl;
    cout << "Tipul cremei principale: " << p.informatii[1] << endl;
    cout << "Numar de blaturi: " << p.informatii[2] << endl;
    return 0;
}
```

5. Enuntul il gasesti in fisierul de sesiunea trecuta

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

struct punct {
    int x,y;
};
struct figura {
    punct A, B;
} d;

int main() {
    //Deoarece avem coordonatele la punctul din stanga sus A(x1,y1) si
    // dreapta jos B(x2,y2) pentru a demonstra
    // Pentru a demonstra ca este un patrat trebuie sa aratam ca
    // inaltimea este egala cu lungimea
```

```

        // Citim coordonatele punctelor
        // Nota: pentru bac ai nevoie sa scrii pe foaie doar ce este intre
liniile hasurate.
        cin >> d.A.x >> d.A.y >> d.B.x >> d.B.y;
        //-----
        int inaltime, lungime;
        if (d.A.y > d.B.y ) {
            inaltime = d.A.y - d.B.y;
        } else {
            inaltime = d.B.y - d.A.y;
        }

        if (d.A.x > d.B.x ) {
            lungime = d.A.x - d.B.x;
        } else {
            lungime = d.B.x - d.A.x;
        }

        if (lungime == inaltime) {
            cout << "DA";
        } else {
            cout << "NU";
        }
        //-----
        return 0;
    }

```

6. Enuntul il gasesti in fisierul de sesiunea trecuta

```

#include <iostream>
#include <cstring>
using namespace std;
//-----
struct specialist {
    int anAngajare;
    struct {char CNP[13]; int anNastere;} personal;
}s[30];
//-----

// Nota: pentru bac ai nevoie doar de codul ce se gaseste intre liniile
hasurate:
// Mai jos este doar pentru a ne verifica
int main() {
    int anNastereInitial = 1970;
    int anAngajareInitial = 1990;
    for (int i = 0; i < 30; i++) {
        s[i].anAngajare = anAngajareInitial+i;
        s[i].personal.anNastere = anNastereInitial+i;
        strcpy(s[i].personal.CNP, "19211111111");
    }
}

```

```
cout << s[5].personal.CNP<<endl;  
cout << s[5].personal.anNastere<<endl;  
cout << s[5].anAngajare<<endl;;  
return 0;  
}
```

Introducere in vectori de frecventa

Un **vector de frecvență** (sau vector de apariții) este ca o **listă în care ținem socoteala de câte ori apare fiecare element** dintr-un set de valori.

Gândește-te la el ca la o **listă de bile colorate**:

- ai un borcan cu bile roșii, albastre și verzi;
- vrei să știi câte bile sunt din fiecare culoare;
- faci o listă unde fiecare culoare are numărul de bile → asta este vectorul de frecvență.
- Un exemplu simplu cu litere: Avem cuvântul "banana"
- **b** apare **1 dată**
- **a** apare **3 ori**
- **n** apare **2 ori**

Vectorul de frecvență pentru literele acestui cuvânt:

Litera	Frecvență
a	3
b	1
n	2

☒ Asta ne spune rapid câte apariții are fiecare literă fără să mai căutăm în șir de fiecare dată.

3. Cum îl facem în C++

- Pas 1: Alegem "universul" de valori - Ex: toate literele mici **a-z**.
- Pas 2: Creăm vectorul de frecvență

```
#include <iostream>  
using namespace std;  
  
int main() {  
    char text[7] = "banana";  
    int frecventa[26] = {0}; // 26 litere, toate inițial 0  
}
```

```

        for(int i =0; i < 7; i++) {
            // incrementăm poziția corespunzătoare literei. Poziția o
            calculăm ca fiind distanța de la aceea
            // litera față de litera 'a'
            frecventa[text[i] - 'a']++;
        }

        // afișăm rezultatul
        for(int i = 0; i < 26; i++) {
            if(frecventa[i] > 0) {
                cout << char('a'+i) << " apare de " << frecventa[i] << "
ori\n";
            }
        }

        return 0;
    }

```

- Acum să vedem un exemplu simplu cu numere.
 - Pentru vectorul: [2,5,2,3,5,2]
 - Numărăm câte apariții are fiecare număr între 1 și 5: | Număr | Frecvență | |-----|-----|

-| 1 | 0 | 2 | 3 | 3 | 1 | 4 | 0 | 5 | 2 |
- În c++ scriem:

```

#include <iostream>
using namespace std;

int main() {
    int numere[6] = {2,5,2,3,5,2};
    int frecventa[10] = {0};

    for (int i = 0; i < 6; i++) {
        frecventa[numere[i]]++;
    }

    for (int i = 0; i < 6; i++) {
        cout << i << " apare de " << frecventa[i] << " ori." << endl;
    }
    return 0;
}

```

Când putem folosi vectorii de frecvență:

- Când vrem să numărăm aparițiile fiecărui element, dar cât mai rapid. Exemple:
 - Verificarea anagramelor – două cuvinte sunt anagrame dacă au același vector de frecvență.
 - Ex: "listen" și "silent" → vectori identici.

- Găsirea numărului majoritar într-un șir de numere.
- Statistică simplă – câte note 10 sunt într-o clasă.
- Exemplu popular la bac: dacă ni se da un șir de numere, care este cel mai mare/mic număr pe care îl putem forma cu cifrele tuturor numerelor
- Etc

IMPORTANT - Când declarăm vectorul de frecvență, trebuie să fim atenți la mărimea lui. Mereu mărimea lui o să fie destul de mare cât să poată conține cel mai mare număr care poate să apară în mulțimea careia îi calculăm frecvența. Mai jos câteva exemple:

- Pentru mulțimea: [1,2,3,5,8,9]
 - Vectorul de frecvență arată: `int frecventa[10]` -> deoarece avem nevoie să putem scrie în `frecventa[9]`
- Pentru mulțimea: [2,5,99,123,124]
 - Vectorul de frecvență arată: `int frecventa[125]` -> deoarece avem nevoie să putem scrie în `frecventa[124]`
- Pentru mulțimea: [2,5,99,4,3,2,8,12,34]
 - Vectorul de frecvență arată: `int frecventa[100]` -> deoarece avem nevoie să putem scrie în `frecventa[100]`
- Exemplu: Se citește un fișier care conține până la 100000 de numere. Să se afișeze cel mai mare număr ce poate fi creat din cifrele tuturor numerelor:

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

using namespace std;

int main() {
    int frecventa[10] = {0};
    ifstream fin("numere.txt");

    int numar;
    // citim număr cu număr din fișier
    while (fin >> numar) {
        //citim cifră cu cifră din numărul citit;
        while (numar > 0) {
            int ultimaCifra = numar % 10;
            frecventa[ultimaCifra]++;
            numar = numar / 10;
        }
    }

    for (int i = 9; i >= 0; i--) {
        if (frecventa[i] > 0) {
            for (int j = 0; j < frecventa[i]; j++) {
                cout << i;
            }
        }
    }
}
```

```
    fin.close();  
    return 0;  
}
```

Introducere in Matrici/Vectori 2D

- Cand am vorbit despre vectori, am spus ca este o structura de date, unde punem date de acelasi tip
 - De exemplu o lista de numere, de caractere, de structuri, etc
- O matrice (sau vector 2D) poate fi vazuta ca un vector in care fiecare element este un vector (practic un vector de vectori)
- Pentru a ne usura viata atunci cand lucram cu matricele, ne putem imagina matricea ca un plan 2d (XoY) unde pe Y avem liniile si pe x coloanele.
- Exista doua tipuri de matrici:
 1. Oarecare => adica numarul de linii difera de numarul de coloane
 2. Patratice => adica numarul de linii este egal cu cel al coloanelor
- Mai jos avem un program care citeste de la tastatura o matrice oarecare:

```
#include <iostream>  
  
using namespace std;  
  
int main() {  
  
    int nrLinii, nrColoane;  
    cin >> nrLinii >> nrColoane;  
    int matrice[nrLinii][nrColoane];  
  
    for (int i = 0; i < nrLinii; i++) {  
        for (int j = 0; j < nrColoane; j++) {  
            cin >> matrice[i][j];  
        }  
    }  
  
    // Afisam matricea parcurgand linie cu linie  
    for (int i = 0; i < nrLinii; i++) {  
        //apoi afisand toate elementele de pe o linie, separate cu un  
        spatiu  
        for (int j = 0; j < nrColoane; j++) {  
            cout << matrice[i][j] << " ";  
        }  
        // dupa ce am afisat o linie, pentru a fi totul formatat mai  
        frumos, ne mutam pe linia urmatoare  
        cout << endl;  
    }  
}
```

```

    }
    return 0;
}

```

- Acum sa vedem cum citim o matrice patratica. O sa vezi ca e foarte similar si chiar mai simplu:

```

#include <iostream>

using namespace std;

int main() {

    int n;
    cin >> n;
    int matrice[n][n];

    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            cin >> matrice[i][j];
        }
    }

    // Afisam matricea parcurgand linie cu linie
    for (int i = 0; i < n; i++) {
        //apoi afisand toate elementele de pe o linie, separate cu un
spatiu
        for (int j = 0; j < n; j++) {
            cout << matrice[i][j] << " ";
        }
        // dupa ce am afisat o linie, pentru a fi totul formatat mai
frumos, ne mutam pe linia urmatoare
        cout << endl;
    }
    return 0;
}

```

- Acum o sa rezolvam un exercitiu de pe pbinfo care ne cere sa determinam suma valorilor pare dintr-o matrice:

```

#include <iostream>
using namespace std;

int main() {
    int n, m;
    cin >> n >> m; // n => linii, m => coloane
    int matrice[n][m];
    for(int i = 0; i < n; i++) {
        for(int j = 0; j < m; j++) {
            cin >> matrice[i][j];
        }
    }
}

```



```

    }
}

int suma = 0;
for(int i = 0; i < n; i++) {
    for(int j = 0; j < m; j++) {
        if (matrice[i][j] % 2 == 0) {
            suma += matrice[i][j];
        }
    }
}

cout << suma;
return 0;
}

```

- Urmatoarea problema, tot de pe pbinfo ne cere sa afisam suma elementelor de pe fiecare linie.

```

#include <iostream>
using namespace std;

int main() {
    int n, m;
    cin >> n >> m; // n => linii, m => coloane
    int matrice[n][m];
    int suma = 0;

    for(int i = 0; i < n; i++) {
        for(int j = 0; j < m; j++) {
            cin >> matrice[i][j];
        }
    }

    for(int i = 0; i < n; i++){
        int sumaLinie = 0;
        for(int j = 0; j < m; j++) {
            sumaLinie += matrice[i][j];
        }
        cout << sumaLinie << " ";
    }

    return 0;
}

```

Exercitii tema

Exercitii vector de frecventa

Exercițiile sunt aranjate astfel: **8 exerciții cu numere** și **2 exerciții cu litere**, cu dificultate progresivă.

1. Numărul de apariții al unui număr

Citește un vector de n numere între 1 și 10 și un număr x . Afișează de câte ori apare x .

2. Vector de frecvență simplu

Citește un vector de n numere între 1 și 10. Construiește vectorul de frecvență și afișează câte apariții are fiecare număr de la 1 la 10.

3. Numărul elementelor unice

Citește un vector de n numere întregi (între 1 și 20) și află câte numere apar o singură dată.

4. Element majoritar

Citește un vector de n numere (între 1 și 50) și afișează elementul care apare mai mult de jumătate din n , dacă există. Dacă nu există, afișează un mesaj corespunzător.

5. Cifrele unui număr

Citește un număr întreg pozitiv și construiește vectorul de frecvență pentru cifrele sale (0–9). Afișează câte apariții are fiecare cifră.

6. Reordonare după frecvență

Citește un vector de n numere mici (1–10). Construiește vectorul de frecvență și afișează numerele în ordinea descrescătoare a frecvenței lor. Dacă două numere au aceeași frecvență, afișează-le în ordine crescătoare.

7. Numărul cel mai rar

Citește un vector de n numere între 1 și 20. Afișează numărul care apare cel mai puțin. Dacă sunt mai multe, afișează-le pe toate.

8. Diferența dintre cel mai frecvent și cel mai rar

Citește un vector de n numere între 1 și 50. Determină diferența dintre numărul care apare cel mai des și numărul care apare cel mai rar.

9. Numără aparițiile unui caracter

Citește un șir de litere mici și un caracter c . Afișează de câte ori apare c în șir.

10. Cea mai frecventă literă

Citește un șir de litere mici și afișează litera care apare de cele mai multe ori. Dacă există mai multe litere cu aceeași frecvență maximă, afișează-le pe toate.

Exercitii Vectori 2D oarecare

Nota: mai jos ai doar link-ul catre probleme. Ele sunt de pe pbinfo.ro. Cand ai rezolvat o problema, trebuie sa o verifici pe site-ul lor sa vezi daca iti merge sau nu. La fel, daca te blochezi, oricand, scrie-mi, nu astepta pana la sesiunea urmatoare.

1. <https://www.pbinfo.ro/probleme/666/nrprime>
2. <https://www.pbinfo.ro/probleme/4690/varfuri9>
3. <https://www.pbinfo.ro/probleme/804/colegale>