MINISTERUL EDUCAȚIEI ȘI CERCETĂRII AL REPUBLICII MOLDOVA

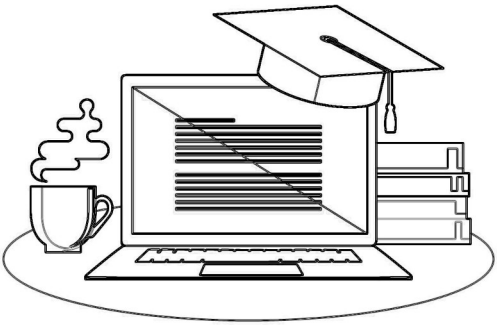
IP CENTRUL DE EXCELENȚĂ ÎN INFORMATICĂ ȘI TEHNOLOGII INFORMAȚIONALE

CATEDRA INFORMATICA II

ANDRIAN DASCAL

PROGRAMARE PROCEDURALĂ ÎN C++

Îndrumar pentru activităţile practice,
destinat cadrelor didactice şi elevilor din IP CEITI



CHIȘINĂU, 2022

Elaborat conform Curriculumului modular „Programarea procedurală”, ediția 2020. Discutat și examinat la ședința catedrei „Informatica II” din 5.11.2021, Proces Verbal Nr. 3, șef catedră, Luminița Gribineț

Autor:

Andrian Dascal, magistru în Informatică și Tehnologii Informaționale, grad didactic II, IP CEITI.

Recenzie:

Galina Luncașu, prof. la discipline de informatică, grad didactic superior, IP CEITI.

Redactare științifică:

Ioan Jeleascov, „architect advisor” în cadrul unității "StoneHard".

DRAGI PRIETENI,

„Un limbaj de programare care nu afectează modul în care gândești despre programare nu merită să fie cunoscut și dacă nu eșuezi cel puțin 90% din timp, nu ții suficient de sus.”
Alan Perlis (creatorul ALGOL)

Programarea procedurală este o paradigmă de programare, derivată din programarea imperativă, bazată pe conceptul apelului de procedură. Procedurile (un tip de rutină sau subrutină) conțin pur și simplu o serie de pași de calcul care trebuie efectuați. Orice procedură dată poate fi apelată în orice moment al executării unui program, inclusiv prin alte proceduri sau de la sine. Primele limbaje majore de programare procedurală au apărut în jurul anilor 1957–1964, inclusiv Fortran, ALGOL, COBOL, PL/I și BASIC. Pascal și C au fost publicate în jurul anilor 1970–1972.

Modularitatea este în general de dorit, în special în programele mari și complicate. Intrările sunt de obicei specificate sintactic sub formă de argumente și ieșirile livrate ca valori returnate.

Scoping este o altă tehnică care ajută la menținerea procedurilor modulare. Împiedică procedura să acceseze variabilele altor proceduri (și invers), inclusiv instanțe anterioare, fără autorizație explicită.

Procedurile mai puțin modulare, utilizate adesea în programe mici sau scrise rapid, tind să interacționeze cu un număr mare de variabile din mediul de execuție, pe care alte proceduri le-ar putea modifica.

Datorită capacității de a specifica o interfață simplă, de a fi autonomă și de a fi reutilizată, procedurile sunt un vehicul convenabil pentru realizarea unor bucăți de cod scrise de diferite persoane sau grupuri diferite, inclusiv prin intermediul bibliotecilor de programare.

Prezenta lucrare reprezintă un suport praxiologic având drept scop însușirea de către elevi a cunoștințelor necesare gândirii algoritmice și formării culturii informaționale prin prisma rezolvării problemelor, incluzând peste 100 probleme model rezolvate, însoțite de comentarii corespunzătoare.

Acest îndrumar este destinat cadrelor didactice debutante și elevilor, dar și fiecărui om cult care, va activa într-un mediu bazat pe cele mai moderne tehnologii informaţionale. Nu uitați: „Cel mai bun mod de a prezice viitorul este să-l implementăm.”, afirmația lui Dijkstra.

Autorul

CUPRINSUL

INTRODUCERE

Când sunteți nou în programare, paradigmele de programare nu au prea multă importanță. Dar pe măsură ce urcați scările și începeți să creați programe și software complexe, este vital să înțelegeți ce paradigmă de programare este cea mai potrivită pentru proiectul dvs. Înainte de a începe, este important să știm ce este exact o paradigmă. Conform multor definiții citate, o paradigmă este „un set de presupuneri, concepte, valori și practici care constituie un mod de a vedea realitatea pentru comunitatea care le împărtășește, în special într-o disciplină intelectuală".

Această definiție este la fața locului, deoarece ceea ce diferențiază paradigma este modul diferit de a vedea realitatea pentru comunitate. Paradigmele contează pe măsură ce călătoresc adesea împreună cu o cultură specifică de a scrie programe și de a se gândi la ele. În continuare, vom aborda principalele paradigme de programare, cu un accent special pe paradigma de programare procedurală.

1. Ce este programarea procedurală?

Programarea procedurală poate fi prima paradigmă de programare pe care o va învăța un nou dezvoltator. În esență, codul procedural este cel care instruiește direct un dispozitiv despre cum să finalizeze o sarcină în pași logici. Această paradigmă utilizează o abordare liniară de sus în jos și tratează datele și procedurile ca două entități diferite.

Bazat pe conceptul de apel de procedură, Programarea procedurală împarte programul în proceduri, care sunt, de asemenea, cunoscute sub numele de rutine sau funcții, conținând pur și simplu o serie de pași care trebuie efectuați. Programarea procedurală implică scrierea unei liste de instrucțiuni pentru a spune computerului ce ar trebui să facă pas cu pas pentru a finaliza sarcina la îndemână.

2. Care sunt caracteristicile cheie ale programării procedurale?

***Funcții predefinite:** o funcție predefinită este de obicei o instrucțiune identificată printr-un nume. De obicei, funcțiile predefinite sunt încorporate în limbaje de programare de nivel superior, dar sunt derivate din bibliotecă sau*

registru, mai degrabă decât din program. Un exemplu de funcție predefinită este „charAt ()", care caută o poziție de caracter într-un șir.

***Variabilă locală:** o variabilă locală este o variabilă care este declarată în structura principală a unei metode și este limitată la domeniul de aplicare local pe care i se dă. Variabila locală poate fi utilizată numai în metoda în care este definită și, dacă ar fi folosită în afara metodei definite, codul va înceta să funcționeze.*

***Variabilă globală:** o variabilă globală este o variabilă care este declarată în afara oricărei alte funcții definite în cod. Datorită acestui fapt, variabilele globale pot fi utilizate în toate funcțiile, spre deosebire de o variabilă locală.*

***Modularitate:** Modularitatea este atunci când două sisteme diferite au două sarcini diferite la îndemână, dar sunt grupate împreună pentru a încheia mai întâi o sarcină mai mare. Fiecare grup de sisteme ar avea apoi propriile sarcini terminate unul după altul până când toate sarcinile vor fi finalizate.*

***Trecerea parametrilor:** Trecerea parametrilor este un mecanism utilizat pentru a transmite parametrii către funcții, subrutine sau proceduri. Trecerea parametrilor se poate face prin „trece prin valoare", „trece prin referință", „trece prin rezultat", „trece prin valoare-rezultat" și „trece prin nume".*

3. Care sunt aspectele pozitive și cele negative ale programării procedurale?

Avantaje:

1. *Programarea procedurală este excelentă pentru programarea generală;*
2. *Simplitatea codificată împreună cu ușurința implementării compilatorilor și interpretilor;*
3. *O mare varietate de cărți și materiale de curs online disponibile pe algoritmi testați, facilitând învățarea pe parcurs;*
4. *Codul sursă este portabil, prin urmare, poate fi utilizat și pentru a viza un alt procesor;*
5. *Codul poate fi reutilizat în diferite părți ale programului, fără a fi nevoie să îl copiați;*
6. *Prin tehnica de programare procedurală, cerința de memorie, de asemenea, reduce;*
7. *Fluxul programului poate fi urmărit cu ușurință.*

Dezavantaje:

1. *Codul programului este mai greu de scris când se folosește programarea procedurală;*
2. *Codul procedural nu este adesea reutilizabil, ceea ce poate crea nevoia de a recrea codul dacă este necesar pentru a fi utilizat într-o altă aplicație;*
3. *Greu de relaționat cu obiecte din lumea reală;*
4. *Importanța este dată mai degrabă operațiunii decât datelor, ceea ce ar putea crea probleme în unele cazuri sensibile la date;*
5. *Datele sunt expuse întregului program, ceea ce îl face să nu fie atât de prietenos cu securitatea.*

Așa cum am analizat aceste informații, Programarea procedurală este mai mult ceea ce faceți decât modul în care o faceți. Este abordarea standard utilizată în multe limbaje de calculator, cum ar fi C, Pascal și BASIC. Deși nu există o paradigmă de programare perfectă, este important să înțelegem că paradigma corectă va depinde întotdeauna de tipul de limbaj pe care îl utilizați și de programul pe care doriți să îl creați.

Se recomandă ca, pentru rezultate maxime și un portofoliu puternic, să se cunoască toate cele patru paradigme majore de programare: imperativă, logică, funcțională și orientată pe obiecte. Cel mai bun mod de a încerca să vă îmbunătățiți paradigmele de programare este să încercați să le studiați.

DE REȚINUT ACESTE REPREZENTĂRI

Remarcă	*Important*	*Rezolvă individual*
		
Informații suplimentare pe care elevii ar trebui să le cunoască sau la care are trebui să atragă atenția.	*Informații importante, cu scopul de a dirija activitatea practică a elevilor.*	*Sarcini ce trebuie realizate în mod individual, cu sau fără suportul profesorului.*

Capitolul 1

APLICAȚII CU POINTERI

1.1 Aplicaţii fundamentale

- ✔ *Unele probleme rezolvate vor fi însoțite de comentarii succinte.*
- ✔ *În acest compartiment vom aplica în mod implicit tipul de date pointer, alocarea statică și dinamică a memoriei.*
- ✔ *Verificați secvențele de cod C/C++ ale problemelor rezolvate, utilizați IDE-ul Code::Blocks.*

Problema 1.1 - 01

Elaborați un program C/C++ care va arăta cum se poate utiliza operatorul de adresă pentru a afișa adresele unor variabile de tipuri diferite.

Implementarea C/C++

```
#include<iostream>
using namespace std;
int i=1; //datele unui indice (sau a unei pozitii)
float sE=200.00; //datele unui salariu in euro (EUR)
long sL=500000000; //datele unui isalariu in lei (MDA)
int main(){
     cout<<"\nAdresa variabilei i este:\t"<<&i;
     cout<<"\nValoarea variabilei i este:\t"<<i<<endl;
     cout<<"\nAdresa variabilei sE este:\t"<<&sE;
     cout<<"\nValoarea variabilei sE este:\t"<<sE<<endl;
     cout<<"\nAdresa variabilei sL este:\t"<<&sL;
     cout<<"\nValoarea variabilei sL este:\t"<<sL<<endl;
}
```

Problema 1.1 - 02

Elaborați un program C/C++ care atribuie operatorul de adresă unei variabile de tip pointer.

Implementarea C/C++

```
#include<iostream>
#include<iomanip>
using namespace std;
int i=1,*p; //datele unui indice (sau a unei pozitii)
float sE=200.00, *q; // sE - datele unui salariu in euro (EUR)
int main(){
     // Pentru tipul de date integer
     p=&i; cout<<"\nAdresa pointerului p este:\t"<<p;
     cout<<"\nAdresa variabilei i este:\t"<<&i<<endl;
     *p=i; cout<<"\nValoarea pointerului p este:\t"<<*p;
     cout<<"\nValoarea variabilei i este:\t"<<i<<endl;

     // Pentru tipul de date float
     q=&sE; cout<<"\nAdresa pointerului q este:\t"<<q;
     cout<<"\nAdresa variabilei sE este:\t"<<&sE<<endl;
     *q=sE; cout<<"\nValoarea pointerului q este:\t";
            cout<<fixed<<setprecision(2)<<*q;
     cout<<"\nValoarea variabilei sE este:\t";
            cout<<fixed<<setprecision(2)<<sE<<endl;

     // Modificarea valorii indicate de catre pointer
     int Var=5,*pointerVar;
     pointerVar = &Var; // Pastrarea adresei variabilei Var
     cout<<"\nVar = "<<Var<<" si *pointerVar = "<<*pointerVar;
     cout<<"\nAdresele pentru: Var = "<<&Var<<" si *pointerVar =
"<<pointerVar;
     Var = 7; // Modificarea valorii pointerului la 7
     cout<<"\n\nVar = "<<Var<<" si *pointerVar = "<<*pointerVar;
     cout<<"\nAdresele pentru: Var = "<<&Var<<" si *pointerVar =
"<<pointerVar;
}
```

Problema 1.1 - 03

Elaborați un program C/C++ care va afișa adresa fiecărui element al unui array unidimensional (1D).

Implementarea C/C++

```
#include <iostream>
using namespace std;
int main(){
    float a[3];
    float *pointer; // declaram variabila de tip pointer
    cout<<"Afisarea adresei folosind un array 1D: "<<endl;
    // Utilizam o bucla pentru a imprima adresele tuturor ele-
mentelor array-ului 1D
    for (int i=0;i<3;++i){
        cout<<"&a["<<i<<"] = "<<&a[i]<<endl;
    }
    pointer = a;  // pointer = &a[0]
    cout<<"\nAfisarea adresei folosind un pointer: "<<endl;
    // Utilizam o bucla pentru a imprima adresele tuturor ele-
mentelor array-ului 1D. Aici folosim notatiile pointerului.
    for (int i=0;i<3;++i) {
        cout<<"pointer + "<<i<<" = "<<pointer+i<<endl;
    }
}
```

Problema 1.1 - 04

Elaborați un program C/C++ care va insera și va afișa datele unui array unidimensional (1D) introduse utilizând notația pointerului.

Implementarea C/C++

```
#include <iostream>
using namespace std;
int main() {
    float a[5];
    // Inserarea datelor utilizand notatia pointerului
    cout<<"Introdu 5 numere reale: ";
    for (int i=0;i<5;++i) {
        cin>>*(a+i); // salvam datele in a[i]
    }
    // Afisarea datelor utilizand notatia pointerului
    cout<<"Afisarea datelor introduse: "<<endl;
    for (int i=0;i<5;++i) {
        cout<<*(a+i)<<" "; // afisam valorile pentru a[i]
    }
}
```

Problema 1.1 - 05

Elaborați un program C/C++ care va prezenta un exemplu de alocare dinamică a memoriei.

Implementarea C/C++

```
#include <iostream>
using namespace std;
int main() {
    int* pInt; // declaram un pointer de tip int
    float* pFloat;  // declaram un pointer de tip float
    // alocarea dinamica a memoriei
    pInt = new int; pFloat = new float;
    // inserarea valorilor in memorie
    *pInt = 45; *pFloat = 45.99f;
    // Afisarea valorilor pointerilor
    cout<<"Valoare pointerului pInt: \t"<<*pInt<<endl;
    cout<<"Valoare pointerului pFloat: \t"<<*pFloat<<endl;
    // Afisarea adreselor pointerilor
    cout<<"Adresa pointerului pInt: \t"<<pInt<<endl;
    cout<<"Adresa pointerului pFloat: \t"<<pFloat<<endl;
    // Dealocarea dinamica a memoriei
    delete pInt; delete pFloat;
}
```

Problema 1.1 - 06

*Elaborați un program C/C++ care va prezenta un exemplu de implementare a operatorilor **new**, **delete** și **null**.*

Implementarea C/C++

```
#include <iostream>
using namespace std;
int main() {
    int n; float *p=NULL,s=0; // declararea pointerului p NULL
    cout<<"Valoarea pointerului null p: "<<p<<endl;
    cout<<"Introdu numarul de studenti: "; cin>>n;
    p = new float[n]; // memoria alocata pentru variabila pointer
de tip float
    cout<<"Introdu mediile celor "<<n<<" studenti."<<endl;
    for (int i=0;i<n;++i) {
        cout<<"Student "<<i+1<<": "; cin>>*(p+i);
    }
```

```
    cout<<"\nAfisarea mediilor fiecarui student."<<endl;
    for (int i=0;i<n;++i) {
        cout<<"Studentul "<<i+1<<": "<<*(p+i)<<endl;
        s+= *(p+i);
    }
    cout<<"\nMedia generala a celor "<<n<<" studenti: "<<s/n<<endl;
    delete[] p; // eliberarea memoriei pentru pointer
}
```

Problema 1.1 - 07

Elaborați un program C/C++ care va implementa interschimbarea valorilor a două variabile de tip pointer.

Implementarea C/C++

```
#include <iostream>
using namespace std;
int main() {
   int n1, n2; // Declaram variabilele
   // Citim datele de intrare
   cout<<"Introduceti valoarea 1: "; cin>>n1;
   cout<<"introduceti valoarea 2: "; cin>>n2;
   //Afisam valorile datelor inainte de interschimbare
   cout<<"Variabila 1 = "<<n1<<" & Variabila 2 = "<<n2<<"\n";
   // Secventa de cod pentru interschimbarea valorilor pointerilor
   int temp,*val1=&n1,*val2=&n2;
   temp = *val1; *val1 = *val2; *val2 = temp;
   //Afisam valorile datelor dupa interschimbare
   cout<<"Variabila 1 = "<<*val1<<" & Variabila 2 = "<<*val2<<"\n";
}
```

Problema 1.1 - 08

Elaborați un program C/C++ care va implementa operațiile fundamentale cu valorile a două variabile de tip pointer (inclusiv incrementarea și decrementarea).

Implementarea C/C++

```
#include <iostream>
using namespace std;
int main() {
   int *p1, *p2, n1, n2;
```

```
    cout<<"\nIntroduceti doua valori intregi: "; cin>>n1>>n2;
    p1 = &n1; p2 = &n2;
    // Prezentarea rezultatelor operatiilor fundamentale
    cout<<"\nSuma: \t\t"<<n1<<" + "<<n2<<" = "<<*p1 + *p2;
    cout<<"\nDiferenta: \t"<<n1<<" - "<<n2<<" = "<<*p1 - *p2;
    cout<<"\nProdusul: \t"<<n1<<" * "<<n2<<" = "<<*p1 * *p2;
    cout<<"\nCatul: \t\t"<<n1<<" / "<<n2<<" = "<<*p1 / *p2;
    cout<<"\nRestul: \t"<<n1<<" % "<<n2<<" = "<<*p1 % *p2;
    // Incrementarea si decrementarea pointerilor
    (*p1)++; cout<<"\n\nPost incrementarea pointerului p1:\t"<<n1;
    ++(*p1); cout<<"\nPre incrementarea pointerului p1:\t"<<n1;
    (*p2)--; cout<<"\nPost decrementarea pointerului p2:\t"<<n2;
    --(*p2); cout<<"\nPre decrementarea pointerului p2:\t"<<n2;
}
```

Problema 1.1 - 09

Elaborați un program C/C++ care va arăta legătura dintre pointeri și tablouri. Se foloseşte în paralel referirea elementelor unui tablou unidimensional de numere întregi, prin indecşi şi, respectiv, folosind operații cu pointeri.

Implementarea C/C++

```
#include<iostream>
using namespace std;
int main(){
    int vect[]={1, 2, 3, 4, 5, 6, 7, 8, 9}, *p; int i;
    p=vect; cout<<"\n";
    //se afiseaza elementele tabloului vect - referirea elementelor se face prin index
    for(i=0 ; i<9 ; i++) cout<<"\t"<<vect[i]; cout<<"\n";
    //se afiseaza elementele tabloului vect - referirea elementelor se face prin adrese
    for(i=0 ; i<9 ; i++) cout<<"\t"<<*(vect+i); cout<<"\n";
    //se afiseaza elementele tabloului vect - referirea elementelor se face prin adrese, folosind//pointerul p (p nu isi modifica valoarea)
    for(i=0 ; i<9 ; i++)cout<<"\t"<<*(p+i); cout<<"\n";
    //se afiseaza elementele tabloului vect - folosind pointerul p, referirea se face prin index//(p nu isi modifica valoarea)
    for(i=0 ; i<9 ; i++) cout<<"\t"<<p[i]; cout<<"\n";
    //se afiseaza elementele tabloului vect - folosind pointerul p (p isi modifica valoarea prin//incrementare)
```

```
    for(i=0 ; i<9 ; i++) cout<<"\t"<<*p++;
}
```

Problema 1.1 - 10

Elaborați un program C/C++ care va transforma un număr întreg (sau real) dintr-o bază oarecare b în baza 10.

Implementarea C/C++

```
#include<iostream>
#include<stdlib.h>
using namespace std;
int b; char numar[30], *p; long int nr;
int main(){
    cout<<"\n Introduceti baza: "; cin>>b; // baza sistemului
    cout<<" Introduceti numarul in baza: "<<b<<" = ";
    cin>>numar; // citim numarul in baza b
    nr=strtol(numar,&p,b);
    if(p) cout<<" Numarul "<<numar<<" in baza "<<b<<" este: "<<nr;
    else cout<<" Eroare la caracterul"<<*p;
}
```

Problema 1.1 - 11

Elaborați un program C/C++ care va arăta legătura dintre pointeri și tablouri. Se foloseşte în paralel referirea elementelor unui tablou unidimensional de caractere (șir de caractere), folosind operații cu pointeri.

Implementarea C/C++

```
#include<iostream>
#include<cstring>
#define n 18
using namespace std;
int main(){
    char sir[] = "siruri si pointeri"; int i;
    // Se afiseaza toate caracterele in format char si int.
    cout<<"Afisarea 1: "<<endl;
    for(i=0;i<n;i++){
        cout<<"\t"<<*(sir+i)<<" - "<<(int)*(sir+i);
    }
    cout<<"\nAfisarea 2: "<<endl;
```

```
    // Se afiseaza toate caracterele in format char si int.
    for(i=0;*(sir+i)!= 0;i++) {
        cout<<"\t"<<*(sir+i)<<" - "<<(int)*(sir+i);
        *sir = 'S'; *(sir+7) = 'S'; *(sir+10) = 'P';
    }
    cout<<"\n\nSirul este: "<<sir;
}
```

Problema 1.1 - 12

Elaborați un program C/C++ care va afișa un șir de caractere cu majuscule, utilizând tipul de date pointer. Determinați numărul de vocale și numărul de spații din mesaj.

Implementarea C/C++

```
#include <iostream>
using namespace std;
int main() {
   char mesaj[20], *p; // Declaram variabilele
   int i=0, vocale=0, spatiu=0;
   cout<<"Introduceti un mesaj de maxim 20 caractere: ";
   cin.get(mesaj,20);
   p = mesaj; // Atribuim mesajul variabilei de tip pointer
   cout<<"Afisam mesajul citit de la tastatura cu majuscule: \t";
   while (*p != '\0') {
      *p=putchar (toupper(*p)); // Afisam mesajul cu majuscule
      if (*p=='A' || *p=='E' || *p=='I' || *p=='O' || *p=='U')
      vocale++;
      if (*p==' ') spatiu++; i++; p++;
   }
   cout<<"\n\nLungimea mesajului introdus: \t"<<i;
   cout<<"\nNumarul de vocale din mesaj: \t"<<vocale;
   cout<<"\nNumarul de spatii din mesaj: \t"<<spatiu;
}
```

Problema 1.1 - 13

Elaborați un program C/C++ care va arăta legătura dintre pointeri și tablouri. Se foloseşte în paralel referirea elementelor unui tablou unidimensional de numere întregi, folosind operații cu pointeri. Determinați suma și produsul elementelor dintr-un array unidimensional de N componente întregi, declarat implicit.

Implementarea C/C++

```
#include <iostream>
using namespace std;
#define N 5
int main() {
        int var[]={1,2,3,4,5}, i=0, s=0, p1=1;
        int*p;
        //& - ia adresa variabilei, acum p == & var, deci *p==var
        p = &var[0];
        while (i<N) {
         s+=*p; // Calculam suma elementelor folosind pointeri
        p1*=*p; // Calculam produsul elementelor folosind pointeri
        i++; p++; // p++ este incrementarea adresei
   }
   cout<<"Suma elementelor: "<<s;
   cout<<"\nProdusul elementelor: "<<p1;
}
```

Problema 1.1 - 14

Elaborați un program C/C++ care va arăta legătura dintre pointeri și tablouri. Se foloseşte în paralel referirea elementelor unui tablou bidimensional de numere întregi, folosind operații cu pointeri. Determinați elementul maxim de pe fiecare linie a unui array 2D. Datele se for citi din fișierul de intrare ***matrice.txt****, iar rezultatele se vor afișa la ecran.*

Implementarea C/C++

```
#include <iostream>
#include <fstream>
#include <stdlib.h> // utilizam system("CLS");
using namespace std;
int M[10][20],dim1,dim2,i,j,*maxim[10];
ifstream fin("matrice.txt");
int main() {
    /* Citim dimensiunile matricii si verificam datele! */
    do {
       cout<<"Citim numarul de linii si de coloane din fisier!\n";
       fin>>dim1>>dim2;
       if ((dim1<1) || (dim1>10) || (dim2<1) || (dim2>20))
       cout<<"Numere invalide. Matricea trebuie sa contina maxim 10
linii si 20 de coloane.\n";
```

```
    }
    while ((dim1<1) || (dim1>10) || (dim2<1) || (dim2>20));
    /* Citim elementele matricii! */
    cout<<"Citim valorile matricii!\n";
    for (i=0; i<dim1; i++){
         for (j=0; j<dim2; j++) {
             fin>>M[i][j];
         }
    }
    cout<<"Apasa o tasta pentru a curati ecranul!"; getchar();
    system("CLS"); // functia de curatare a ecranului
    /* Afisare elemente matrice */
    cout<<"\nAfisarea valorilor matricii la ecran:\n";
    for (i=0; i<dim1; i++) {
        for (j=0; j<dim2; j++) {
             cout<<" "<<M[i][j];
        }
        cout<<endl;
    }
    cout<<endl;
    /* Determinarea elemenului maxim de pe fiecare linie */
    for (i=0; i<dim1; i++){
        maxim[i]=&M[i][0];
        for (j=0; j<dim2; j++) {
            if (M[i][j] > *maxim[i]) *maxim[i] = M[i][j];
        }
        cout<<"Elementul maxim de pe linia "<<i+1<<" este: ";
        cout<<*maxim[i]<<endl;
    }
}
```

Problema 1.1 - 15

Elaborați un program C/C++ care citește de la tastatură două șiruri de caractere de maxim 100 litere mici și verifică dacă cele două șiruri sunt anagrame (conțin aceleași litere, dar în ordine diferită). Se cere afișarea mesajului „Anagrame” în caz afirmativ și a mesajului „Nu sunt anagrame” - în caz contrar.

Implementarea C/C++

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

```
int main(){
    char s[101], t[101],c,*p;  int k,x,ok=1;
    cout<<"Introduceti primul sir: "; cin.get(s,101); cin.get();
    cout<<"Introduceti al doilea sir: "; cin.get(t,101);
    if(strlen(s)!=strlen(t)) ok=0; //au lungimi diferite
    else {
        for(c='a';c<='z' && ok==1;c++){
            x=k=0; p=strchr(s,c);
            while(p!=0){
                  x++; p=strchr(p+1,c);
            }
            p=strchr(t,c);
            while(p!=0){
                  k++; p=strchr(p+1,c);
            }
            if(x!=k) ok=0;
        }
    }
    if(ok==1) cout<<"\nSirurile sunt anagrame!";
    else cout<<"\nSirurile nu sunt anagrame!";
}
```

Problema 1.1 - 16

Elaborați un program C/C++ care citește de la tastatură un text format din cuvinte separate între ele prin câte un singur spațiu. Fiecare cuvânt are cel mult 20 de caractere, doar litere mici ale alfabetului englez. Textul are cel mult 200 de caractere. Afișați la ecran, pe linii separate, doar cuvintele din textul citit care conțin cel mult trei vocale. Se consideră vocale: a, e, i, o, u.

Implementarea C/C++

```
#include <iostream>
#include <cstring>
using namespace std;
int main(){
        char s[201], *p, *q, v[]="aeiou"; int i,k;
        cout<<"Introduceti textul:"; cin.get(s,201);
        p=strtok(s," "); //primul cuvant
        while(p!=0){
                k=0; //se numara vocalele din p
                for(i=0; p[i]!=0; i++)
```

```
                    if(strchr(v,p[i])!=0) k++;
                if(k<=3) cout<<p<<endl;
                p=strtok(NULL," "); //urmatorul cuvant din text
        }
}
```

Problema 1.1 - 17

Elaborați un program C/C++ care citește de la tastatură un cuvânt de cel mult 20 de litere mici ale alfabetului englez și care să afișeze la ecran, pe linii diferite, cuvintele obținute prin ștergerea succesivă a vocalelor în ordinea alfabetică a lor (a, e, i, o, u). La fiecare pas se vor șterge toate aparițiile din cuvânt ale unei vocale.

Implementarea C/C++

```
#include <iostream>
#include <cstring>
using namespace std;
int main(){
    char s[21], v[]="aeiou",*p; int i=0;
    cout<<"Introduceti cuvantul: "; cin.get(s,21);
    for(i=0;v[i]!=0;i++){
        p=strchr(s,v[i]); //vocale v[i] apare in text
        if(p!=0){
           while(p!=0){
                strcpy(p,p+1); p=strchr(s,v[i]);
           }
           cout<<s<<endl; //se afiseaza sirul obtinut
        }
    }
}
```

1.2 Probleme propuse

1. *Scrieţi o funcţie care calculează produsul scalar a doi vectori X și Y, având câte N componente fiecare. Se vor aplica pointerii.*
2. *Scrieţi o funcţie care afişează elementele distincte dintr-un șir S cu N elemente întregi, unde N<1000.Se vor aplica pointerii.*
3. *Scrieţi o funcţie care determină media aritmetică a elementelor pozitive situate între primul element pozitiv şi ultimul element negativ al şirului S cu N elemente întregi, unde N<1000, exceptând aceste elemente. Cazurile speciale vor fi clarificate prin mesaje corespunzătoare. Se vor aplica pointerii.*
4. *Scrieţi cele trei funcţii care vor afișa: reuniunea, intersecția și scăderea elementelor a doi vectori X și Y ce conțin N elemente numere întregi, unde N<1000. Se vor aplica pointerii.*
5. *Scrieţi o funcţie care va afișa produsul cartezian a doi vectori X și Y ce conțin N elemente numere întregi, unde N<1000. Se vor aplica pointerii.*

6. *Elaborați un program C/C++ care va permite calcularea şi afişarea polinoamelor cât şi rest ale împărţirii celor a două polinoame. Aceste polinoame sunt date prin gradele lor şi tablourile coeficienţilor după puterile descrescătoare ale lui X.*
7. *Elaborați un program C/C++ care va permite citirea de la tastatură a unei valori întregi N şi N valori reale cu care se crează un vector X. Afișați la ecran valoarea medie și abaterea medie pătratică a elementelor vectorului.*
8. *Elaborați un program C/C++ care va permite afişarea elementelor unui tablou unidimensional citit de la intrarea standard, câte 10 pe un rând, se vor aplica pointerii.*
9. *Elaborați un program C/C++ care va permite citirea de la tastatură a unui vector cu 10 elemente numere întregi. Să se afişeze pe ecran adresele de memorie ale elementelor vectorului şi apoi să se determine suma elementelor acestuia. Operațiile se vor realiza prin intermediul pointerilor.*
10. *Elaborați un program C/C++ care va permite construirea a două tablouri, primul conţinând factorii primi ai unui număr întreg N, iar cel de-al doilea – primele N numere ale secvenței Fibonacci.*

11. *Elaborați un program C/C++ care va permite citirea de la tastatură a unei valori întregi N şi N valori reale cu care se crează un vector X. Să se creeze un vector Y cu componentele din X mai mari decât valoarea medie şi să se afişeze câte 5 elemente pe o linie.*

12. *Elaborați un program C/C++ care va permite citirea de la tastatură a unei matrici de numere întregi, de dimensiuni 10 x 20. Citirea matricei se va realiza cu ajutorul unei funcții. Să se afişeze pe ecran minimul şi maximul valorilor de pe fiecare linie a matricei. Calculul minimului și maximului valorilor unei linii a unei matrice se va realiza cu o funcție.*
13. *Elaborați un program C/C++ care va permite citirea de la tastatură şi afişarea a două matrici de numere întregi, de dimensiuni introduse de la tastatură. Programul va calcula şi afişa matricea diferență a celor două matrici, calculată element cu element. Alocarea memoriei pentru matrice se va realiza dinamic. Pentru soluționarea problemei se vor folosi pointeri.*
14. *Elaborați un program C/C++ care va permite interschimbarea a două linii ale unei matrici pătratice de elemente reale. Se vor defini şi utiliza funcţii diferite pentru citirea, afişarea şi pentru interschimbarea liniilor unei matrici. Operaţiile se vor realiza folosind pointeri.*

15. *Elaborați un program C/C++ care va permite conversia unui şir de caractere reprezentând un număr scris cu cifre romane în corespondentul său cu cifre arabe, se vor aplica pointerii. Exemplu: numărul roman MCMXCVIII are ca valoare pe 1998.*
16. *Elaborați un program C/C++ care va permite citirea de la tastatură a două şiruri de caractere şi afinşează pe ecran şirul concatenat. Operaţiile se vor realiza folosind pointeri.*
17. *Elaborați un program C/C++ care va permite determinarea elementului maxim dintr-un şir cu n elemente şi poziţia pe care el apare. În cazul unui maxim cu mai multe apariţii se va nota prima sa apariţie.*
18. *Elaborați un program C/C++ care va permite citirea a două şiruri X şi Y ordonate strict crescător, având M şi respectiv N elemente. Să se construiască un şir Z ordonat strict crescător conţinând elementele şirurilor X şi Y- ("interclasare de şiruri").*

Capitolul 2

APLICAȚII CU FIȘIERE

2.1 Aplicaţii fundamentale

- ✔ *Unele probleme rezolvate vor fi însoțite de comentarii succinte.*
- ✔ *În acest compartiment vom aplica în mod implicit tipul de date fișier împreună cu funcțiile sale specifice.*
- ✔ *Închiderea fișierelor este recomandată la fine de program sau conform necesității, la unele compilatoare dacă un fișier nu va fi închis poate prezenta o eroare.*
- ✔ *Verificați secvențele de cod C/C++ ale problemelor rezolvate, utilizați IDE-ul Code::Blocks.*

Problema 2.1 - 01

Elaborați un program C/C++ care va permite citirea a două numere întregi dintr-un fișier și afișarea la ecran a sumei și mediei aritmetice a acestor numere citite.

Implementarea C++

```
#include <iostream>
#include <fstream> // biblioteca pentru fisiere
using namespace std;
ifstream fin("date.txt"); // variabila fin va reprezenta fisierul
int a,b,s=0;
int main(){
        cout<<"Citim datele din fisier!"<<endl;
        fin>>a>>b;// citim din fisierul de intrare
        cout<<"Afisam datele din fisier: "<<a<<" "<<b<<endl<<endl;
        s=a+b; cout<<"Suma: "<<s<<endl;
        cout<<"Media aritmetica: "<<s/2.0<<endl;
        fin.close(); // inchidem fișierul de intrare
}
```

Problema 2.1 - 02

Elaborați un program C/C++ care va permite citirea a trei numere reale dintr-un fișier și afișarea în alt fișier a produsului și mediei geometrice a acestor numere citite.

Implementarea C++

```
#include <iostream>
#include <iomanip>
#include <cmath> // pentru a aplica functia cbrt()
#include <fstream>
using namespace std;
ifstream fin("date.txt"); // fisierul de intrare
ofstream fout("solutie.txt"); // fisierul de iesire
float a,b,c,p=1;
int main(){
    cout<<"Citim datele din fisier!"<<endl; fin>>a>>b>>c;
    cout<<"Afisam datele din fisier: "<<a<<" "<<b<<" "<<c<<endl;
    p=a*b*c;
    fout<<"Produsul: "<<p<<endl; // scriem in fisier
    //cbrt() = radical de ordinul 3
    fout<<"Media geometica: "; // scriem in fisierul solutie.txt
    fout<<fixed<<setprecision(3)<<cbrt(p)<<endl;
    fin.close(); fout.close(); // inchidem fișierele
}
```

Problema 2.1 - 03

Elaborați un program C/C++ care va permite citirea a trei numere intregi dintr-un fișier și va afișa în același fișier suma și produsul acestor numere fără a fi rescrise datele inițile ale fișierului.

Implementarea C++

```
#include <iostream>
#include <fstream>
using namespace std;
int a,b,c;
int main(){
        //Prelucram datele din fișierul "data.txt", citim
        ifstream fin; fin.open ("data.txt");
        fin>>a>>b>>c;// citim datele din fisier
        //Prelucram datele din fișierul "data.txt", scriem
```

```
        ofstream fout;
        fout.open("data.txt",ios::app);
        fout<<endl<<"Suma numerelor: "<<a+b+c;
        fout<<endl<<"Produsul numerelor: "<<a*b*c;
        fin.close(); gout.close(); // inchidem fisierele
}
```

Problema 2.1 - 04

Elaborați un program C/C++ care va permite citirea a două numere întregi dintr-un fișier și afișarea în alt fișier a ultimelor două cifre ale produsului acestor numere citite.

Implementarea C++

```
#include <iostream>
#include <fstream>
using namespace std;
int a,b;
ifstream fin("numere.in");
ofstream gout("numere.out");
int main(){
    fin>>a>>b; // citim din fisier
    cout<<"Avem numerele: a= "<<a<<", b= "<<b<<endl;
    //Afisarea datelor in fisierul de iesire
    cout<<"Produsul dintre "<<a<<" si "<<b<<" este: "<<a*b<<endl;
    gout<<"Produsul intre "<<a<<" si "<<b<<" este: \t"<<a*b<<endl;
    gout<<"Ultima cifra a produsului este: \t"<<(a*b)%10<<endl;
    gout<<"Penultima cifra a produsului este: \t";
    gout<<((a*b)/10)%10<<endl;
    fin.close(); gout.close(); // inchidem fisierele
}
```

Problema 2.1 - 05

*Elaborați un program C/C++ care să verifice dacă 3 numere a, b, c - citite din fișierul **pitagora.in** - sunt pitagorice (pătratul unuia poate fi scris ca suma pătratelor celorlalte două). Numerele, respectiv rezultatul testului vor fi scrise în fișierul **pitagora.out**.*

Implementarea C++

```
#include <iostream>
```

```
#include <fstream>
using namespace std;
int a,b,c;
ifstream fin("pitagora.in"); // fisierul de intrare
ofstream gout("pitagora.out"); // fisierul de iesire
int main(){
    fin>>a>>b>>c; // citim din fisier
    cout<<"Avem numerele: a= "<<a<<", b= "<<b<<", c= "<<c<<endl;
    if ((a*a+b*b==c*c) || (a*a+c*c==b*b) || (b*b+c*c==a*a))
         gout<<"Cele 3 numere: "<<a<<", "<<b<<", "<<c;
         gout<<" sunt pitagorice!"<<endl; // afisam rezultatele
    else gout<<"Cele 3 numere: "<<a<<", "<<b<<", "<<c;
         gout<<" nu sunt pitagorice!"<<endl; // afisam rezultatele
    fin.close(); gout.close(); // inchidem fisierele
}
```

Problema 2.1 - 06

*Fișierul **segmente.txt** conține - pe câte un rând - perechi de numere reprezentând coordonatele extremităților a două segmente. Elaborați un program C/C++ care să afișeze lungimea segmentului mai mare și să se scrie în fișierul **rezultat.txt**.*

Exemplu:

Explicații	segmente.txt	rezultat.txt
Segmentul AB va avea coordonatele lui A: A(2,2) și B: B(5,6). Segmentul CD va avea coordonatele lui C: C(3,2) și D: D(9,10).	2 2 5 6 3 2 9 10	10

Implementarea C++

```
#include <iostream>
#include <fstream>
#include <cmath>
using namespace std;
float l1,l2;
ifstream fin("data.txt"); //segmente.txt
ofstream gout("rezultat.txt");
int main(){
```

```
    int x1,y1,x2,y2,x3,y3,x4,y4;
    fin>>x1>>y1>>x2>>y2;
    cout<<"Segmentului 1: ["<<x1<<","<<y1<<"], ["<<x2<<","<<y2<<"]";
    fin>>x3>>y3>>x4>>y4; cout<<endl;
    cout<<"Segmentului 2: ["<<x3<<","<<y3<<"], ["<<x4<<","<<y4<<"]";
    // lungimea segmentului 1
    l1=sqrt(pow((x1-x2),2)+pow((y1-y2),2));
    // lungimea segmentului 2
    l2=sqrt(pow((x3-x4),2)+pow((y3-y4),2));
    if (l1==l2)
    gout<<"Lungimile celor doua segmente sunt egale!"<<endl;
    else if (l1>l2)
    gout<<"Lungimea mai mare este l1="<<l1<<endl;
    else gout<<"Lungimea mai mare este l2="<<l2<<endl;
    fin.close(); gout.close(); // inchidem fisierele
}
```

Problema 2.1 - 07

Elaborați un program C/C++ care va permite citirea unui număr de la tastatură, iar alte două numere din fișier. Numerele din fișier vor reprezenta un anumit interval. Preluați datele din fișier și determinați apartenența la acest interval a numărului introdus de la tastură.

Implementarea C++

```
#include <iostream>
#include <fstream>
using namespace std;
int n,a,b;
ifstream fin("numere.in");
ofstream gout("numere.out");
int main(){
    cout<<"Introduceti numarul n: "; cin>>n;
    fin>>a>>b; // citim coordonatele intervalului inchis
    cout<<"Intervalul este: ["<<a<<", "<<b<<"]";
    if (n>=a && n<=b)
        gout<<"Numarul "<<n<<" apartine ["<<a<<", "<<b<<"]\n";
    else
        gout<<"Numarul "<<n<<" nu apartine ["<<a<<", "<<b<<"]\n";
    fin.close(); gout.close();
}
```

Problema 2.1 - 08

Elaborați un program C/C++ care va permite scrierea într-un fișier pe două coloane primele n cupluri de forma: n n!.

Exemplu:

Explicații	date.txt	Factorial.txt
Din fișierul date.txt se va citi numărul maxim de date pentru a fi calculate. Fie un exemplu particular: n=4, atunci n! = 1*2*3*4 = 24.	4	1! = 1 2! = 2 3! = 6 4! = 24

Implementarea C++

```
#include <iostream>
#include <fstream>
using namespace std;
int n,i,j; long long p;
ifstream fin("data.txt");
ofstream gout("Factorial.txt");
int main(){
    fin>>n; // citim numărul maxim pentru calcule
    for (i=1;i<=n;i++){
       p=1;
       for (j=1;j<=i;j++) p=p*j;
       gout<<i<<" "<<p<<endl;
    }
    fin.close(); gout.close();
}
```

Problema 2.1 - 09

Elaborați un program C/C++ care va permite citirea unui șir de numere întregi dintr-un fișier separate prin spațiu. Afișați în același fișier minimul și maximul acestor numere citite, fără a păstra datele inițiale.

Implementarea C++

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

```
#include <climits>
using namespace std;
ifstream fin("date.txt"); // fin reprezinta fisierul de intrare
ofstream fout; // fout reprezinta fisierul de iesire
int x,maxim=INT_MIN,minim=INT_MAX;
int main(){
    if(!fin){  //sau fin=NULL
       cout<<"Eroare la deschiderea fisierului!"<<endl;
    }
    while(!fin.eof()){
        fin>>x;
        if (maxim<x) maxim=x;
        else if (minim>x) minim=x;
    }
    fin.close(); // inchidem fișierul de intrare
    // stabilim fostul fisier de intrare ca fisier de ieșire
    fout.open ("date.txt");
    fout<<"Maximul este: "<<maxim<<endl<<"Minimul este: "<<minim;
    fout.close(); // inchidem fișierul de iesire
}
```

Problema 2.1 - 10

Elaborați un program C/C++ care va permite citirea unui șir de cifre din fișier până la întâlnirea caracterului "$". Preluați datele din fișier și contorizați fiecare cifră pară.

Implementarea C++

```
#include<iostream>
#include<fstream>
char c; int zero=0, doi=0, patru=0, sase=0, opt=0;
using namespace std;
int main(){
    ifstream fin("date.txt");
    cout<<"Citim datele din fisier!"<<endl;
    while(c!='$'){
        fin>>c;
        // efectuam contorizarea cifrelor pare
        switch(c){
            case '2': doi++; break; case '4': patru++; break;
            case '6': sase++; break; case '8': opt++; break;
            case '0': zero++; break;
        }
```

```
    }
    // afisam contorizarea cifrelor pare in forma tabelara
    cout<<"\tCifre \t 0 \t 2 \t 4 \t 6 \t 8 "<<endl;
    cout<<"\tContor \t "<<zero<<"\t "<<doi<<"\t "<<patru;
    cout<<"\t "<<sase<<"\t "<<opt<<endl;
    cout<<"\nContorizare a cifrelorpare  a reusit!";
    fin.close();
}
```

Problema 2.1 - 11

Elaborați un program C/C++ care va permite citirea unui șir de la tastatură până la întâlnirea caracterului "$". După citire datele se scriu într-un fișier text. Preluați datele din fișier și faceți contorizarea fiecărei vocale.

Implementarea C++

```
#include<iostream>
#include<fstream>
using namespace std;
char c; int a=0,e=0,i=0,o=0,u=0;
ofstream fout("date.txt");
int main(){
    if(!fout){
        cout<<"Eroare la deschidere."<<endl;
    }
    cout<<"Introduceti un sir de caractere mici, iar la sfarsit in-
cludeti simbolul $: "<<endl;
    do{
        c=getchar();
        switch(c){
                case 'a': a++; break; case 'e': e++; break;
                case 'i': i++; break; case 'o': o++; break;
                case 'u': u++;
        }
        // scrierea caracterelor citie de la tastatura in fisier
        if(c!='$') fout.put(c); //ar fi mers si fout<<c;
     }
     while(c!='$');
     fout.close(); // inchidem fișierul de intrare
     ofstream gout; gout.open("solutie.txt");
     if(!gout){
        cout<<"Eroare la deschidere fisierului de citire."<<endl;
```

```
    }
    // afisarea contorizarii vocalelor in fisierul solutie
    gout<<"Vocala a se gaseste de : "<<a<<" ori.\n";
    gout<<"Vocala e se gaseste de : "<<e<<" ori.\n";
    gout<<"Vocala i se gaseste de : "<<i<<" ori.\n";
    gout<<"Vocala o se gaseste de : "<<o<<" ori.\n";
    gout<<"Vocala u se gaseste de : "<<u<<" ori.\n";
    gout.close(); // inchidem fișierul de iesire
    cout<<"Opratia de contorizare a vocalelor a reusit!"<<endl;
}
```

Problema 2.1 - 12

Elaborați un program C/C++ care va citi dintr-un fișier un număr întreg N de minim 9 cifre. Să se scrie, pe linii separate, în fișierul de ieșire datele:

- *câte cifre are numărul;*
- *suma și media cifrelor numărului;*
- *cea mai mică și cea mai mare cifră a numărului;*
- *de câte ori se găsește o cifră (citită de la tastatură) în numărul.*

Implementarea C++

```
#include <iostream>
#include <fstream>
#include <limits>
using namespace std;
long long n,m;
int i=0,j=0, r, s=0, cifra, maxim=INT_MIN, minim=INT_MAX;
ifstream fin("data.txt");
ofstream gout("solutii.txt");
int main(){
    fin>>n; cout<<"n= "<<n<<endl;
    m=n; // atribuire
    cout<<"Introduceti cifra de cautat in numarul "<<n<<": ";
    cin>>cifra; // cifra care o contorizam
    while (n!=0){
        r=n%10;
        if (r>maxim) maxim=r; // cifra maxima obtinuta
        else minim=r; // cifra minima obtinuta
        if (r==cifra) j++;
        n=n/10; s=s+r; i++;
   }
```

```
    // Afisam rezultatele în fișierul de iesire
    gout<<"Numarul "<<m<<" are "<<i<<" cifre!"<<endl;
    gout<<"Suma cifrelor numarului "<<m<<" este: "<<s<<endl;
    gout<<"Media cifrelor numarului "<<m<<" este: ";
    gout<<(float) s/i<<endl;
    gout<<"Cea mai mare cifra a numarului "<<m<<" este: "
    gout<<maxim<<endl;
    gout<<"Cea mai mica cifra a numarului "<<m<<" este: "
    gout<<minim<<endl;
    gout<<"Cifra "<<cifra<<" se gaseste in numarul "<<m<<" de ";
    gout<<j<<" ori."<<endl;
    fin.close(); gout.close(); // inchidem ambele fisiere
}
```

2.2 Aplicaţii ale fişierelor la array-uri şi string-uri

- ✔ *Unele probleme rezolvate vor fi însoțite de comentarii succinte.*
- ✔ *În acest compartiment vom aplica în mod implicit tipul de date fișier împreună cu funcțiile sale specifice asupra array-urilor și a string-urilor.*
- ✔ *Verificați secvențele de cod C/C++ ale problemelor rezolvate, utilizați IDE-ul Code::Blocks.*

Problema 2.2 - 01

Elaborați un cod C/C++ care va permite introducerea din fișier a componentelor unui vector și le va afișa la ecran în ordinea introducerii acestora.

Implementarea C++

```
#include <iostream>
#include <fstream>
using namespace std;
int main() {
    int v[100],n,i;
    ifstream fin("array1D.txt");
    fin>>n; // citim dimensiunea vectorului
    cout<<"Citim componentele vectorului din fisier!"<<endl;
    for(i=0;i<n;i++){
        fin>>v[i]; // citim elementele vectorului din fisier
    }
    cout<<"Afisam componentele vectorului la ecran: \n\t";
    for(i=0;i<n;i++){
        cout<<v[i]<<" ";
    }
    fin.close(); // inchidem fisierul de intrare
}
```

Problema 2.2 - 02

*Elaborați un cod C/C++ în care componente unui vector vor fi citite din fișierul **array1D.txt** și va afișa în fișierul **rezultat.txt** suma și media aritmetică a componentelor sale.*

Implementarea C++

```
#include <iostream>
#include <fstream>
using namespace std;
int main() {
    int v[100],n,i,s=0;
    ifstream fin("array1D.txt"); // fișierul de intrare
    ofstream gout("rezultat.txt"); // fișierul de iesire
    fin>>n; // citim dimensiunea vectorului
    cout<<"Citim componentele vectorului din fisier!"<<endl;
    for(i=0;i<n;i++){
        fin>>v[i]; // citim elementele vectorului din fisier
        s+=v[i]; // calculam suma elementelor
    }
    cout<<"Afisam componentele vectorului la ecran: \n\t";
    for(i=0;i<n;i++){
        cout<<v[i]<<" ";
    }
    gout<<"Suma elementelor este: "<<s<<endl;
    gout<<"Media elementelor este: "<<(float)s/n<<endl;
    fin.close(); gout.close(); // inchidem fisierele
}
```

Problema 2.2 - 03

Elaborați un program C/C++ care va permite citirea unui șir de cifre de la tastatură până la întâlnirea caracterului "$". După citire datele se scriu într-un fișier text și se preiau din fișierul respectiv, apoi se calculează suma valorilor numerice.

Implementarea C++

```
#include <iostream>
#include <cstring>
#include <stdlib.h>
#include <fstream>
using namespace std;
int main(){
    char a[255], separator[]=" ,", cifre[]="0123456789.+-", *p;
    double s=0;
    ifstream fin("data.txt"); // fisierul de intrare
    if(!fin) {
        cout<<"Eoare la deschiderea fisierului."<<endl;
```

```
    }
    while(!fin.eof()){
        fin.getline(a,255,'\n');
        //se separa sirul folosind ca separator spatiul sau virgula
        p=strtok(a,separator);
        while (p){
             if (strspn(p,cifre)==strlen(p)) s+=atof(p);
             cout<<p<<"\n";
             p=strtok(NULL,separator);
        }
     }
     cout<<"Suma este: "<<s<<endl;
     fin.close(); // inchidem fisierul de intrare
}
```

Problema 2.2 - 04

Elaborați un cod C/C++ în care componentele a două matrici vor fi citite dintr-un fișier. Afișați produsul acestor două matrici în fișierul ***produs.txt****.*

Implementarea C++

```
#include <iostream>
#include <fstream>
using namespace std;
int i,j,k,n,m,M[20][20],N[20][20],C[20][20];
int main() {
    ifstream fin("array2D.txt");
    ofstream gout("produs.txt");
    fin>>m>>n; // citim dimensiunea matricilor
    // citim matricea M cu componentele sale
    for(i=0;i<n;i++){
        for(j=0;j<m;j++){
            fin>>M[i][j];
        }
    }
    // citim matricea N cu componentele sale
    for(i=0;i<n;i++){
        for(j=0;j<m;j++){
            fin>>N[i][j];
        }
    }
    gout<<"Afisam componentele matricii M: \n";
    for(i=0;i<n;i++){
```

```
        for(j=0;j<m;j++){
            gout<<"  "<<M[i][j];
        }
        gout<<endl;
    }
    gout<<"Afisam componentele matricii N: \n";
    for(i=0;i<n;i++){
        for(j=0;j<m;j++){
            gout<<"  "<<N[i][j];
        }
        gout<<endl;
    }
    // calculam produsul M*N
    for(i=0;i<n;i++){
        for (j=0;j<m;j++){
            C[i][j]=0;
            for(k=0;k<m;k++){
                C[i][j]+=M[i][k]*N[k][j];
            }
        }
    }
    gout<<"Afisam componentele matricii M*N: \n";
    for(i=0;i<n;i++){
        for(j=0;j<m;j++){
            gout<<"  "<<C[i][j];
        }
        gout<<endl;
    }
    fin.close(); gout.close(); // inchidem fisierele
}
```

Problema 2.2 – 05

Elaborați un cod C/C++ în care va copia elementele array-ului bidimensional, începând cu prima linie, într-un array unidimensional, rezultatul se va scrie în fișierul ***array1D.txt****.*

Implementarea C++

```
#include <iostream>
#include <fstream>
using namespace std;
int i,j,k=0,m,n,M[10][10],V[100];
ifstream fin("array2D.txt");
```

```
ofstream gout("array1D.txt");
int main() {
    fin>>m>>n; // citim dimensiunea matricilor
    // citim matricea M cu componentele sale
    for(i=0;i<n;i++){
        for(j=0;j<m;j++){
            fin>>M[i][j];
        }
    }
    gout<<"Afisam componentele matricii M: \n";
    for(i=0;i<n;i++){
        for(j=0;j<m;j++){
            gout<<"  "<<M[i][j];
        }
        gout<<endl;
    }
    // transformam matricea in vector
    for(i=0;i<n*m;i++){
        for(j=0;j<m;j++){
            V[k++]=M[i][j];
        }
    }
    gout<<"Afisam componentele vectorului V: \n";
    for(i=0;i<n*m;i++){
        gout<<"  "<<V[i];
    }
    fin.close(); gout.close(); // inchidem fisierele
}
```

Problema 2.2 - 06

*Elaborați un cod C/C++ care va citi din fișierul **mesaj.txt** un text mare (paragraf dintr-o carte). Afișați numărul total de vocale (mici și mari) și numărul total de cifre. Șirul de caractere va fi prezentat în stilul limbajului C.*

Implementarea C++

```
#include <iostream>
#include <cstring>
#include <fstream>
using namespace std;
char a[2000],vocale[]="aeiouAEIOU",cifre[]="0123456789";
int i,nrV=0,nrC=0;
int main(){
```

```
    ifstream fin("mesaj.txt");
    cout<<"Citim mesajul din fisier!\n";
    fin.get(a,2000); // citim datele din fisier
    for (i=0; i<strlen(a); i++){
        if (strchr(vocale,a[i]))
           nrV++; // contorizam vocalele
        if (strchr(cifre,a[i]))
           nrC++; // contorizam cifrele
    }
    cout<<"Numarul de vocale din textul citit este: \t"<<nrV<<'\n';
    cout<<"Numarul de cifre din textul citit este: \t"<<nrC<<'\n';
    fin.close(); // inchidem fișierul de intrare
}
```

Problema 2.2 - 07

*Elaborați un cod C/C++ care va citi un cuvânt din fișierul **string.txt** și va afișa fiecare caracter individual separat prin spațiu în ordine inversă. Șirul de caractere va fi prezentat în stilul limbajului C.*

Implementarea C++

```
#include <iostream>
#include <cstring>
#include <fstream>
using namespace std;
char c[25]; int i,n;
ifstream fin("string.txt");
int main(){
    cout<<"Citim cuvantul din fisier!";
    fin>>c; // citim datele din fisier
    cout<<"\nCaracterele cuvantului in ordinea inversa: \n\t";
    i=strlen(c);
    // inversam caracterele cuvantului citit din fisier
    while(i>=0){
       cout<<c[i]<<" "; i--;
    }
    fin.close(); // inchidem fișierul de intrare
}
```

Problema 2.2 - 08

*Elaborați un cod C/C++ care va citi un mesaj din fișierul **string.txt** și va afișa numărul de cuvinte ale acestuia în fișierul **cuvinte.txt**. Șirul de caractere va fi prezentat în stilul limbajului C.*

Implementarea C++

```
#include <iostream>
#include <cstring>
#include <fstream>
using namespace std;
char mesaj[100]; int i=0,cuvinte=1;
ifstream fin("string.txt");
ofstream gout("cuvinte.txt");
int main(){
    cout<<"Introducem mesajul din fisier!"; fin.get(mesaj,100);
    gout<<"Mesajul citit din fisier este: ";
    for(i=0;i<strlen(mesaj);i++){
        gout<<mesaj[i];
    }
    while(mesaj[i]!='\0'){
    /*verificam daca caracterul curent este spatiu, linie noua sau
caracter tab*/
        if(mesaj[i]==' ' || mesaj[i]=='\n' || mesaj[i]=='\t'){
            cuvinte++;
        }
        i++;
    }
    gout<<"\nNumarul total de cuvinte ale mesajului: "<<cuvinte;
     fin.close(); gout.close(); // inchidem fisierele
}
```

- ✔ *Acest îndrumar conține unele probleme rezolvate nestandard, adică se aplică Standard Template Library (STL). STL este o bibliotecă de software concepută inițial de Alexander Stepanov pentru limbajul de programare C++ care a influențat multe părți ale Bibliotecii standard C++. Acesta oferă patru componente denumite: algoritmi, containere, funcții și iteratori.*
- ✔ *Toate problemele rezolvate în titlul cărora veți întâlni simbolica „*** ”, sunt rezolvate prin STL.*

Problema 2.2 – 09 ***

*Elaborați un cod C/C++ care va citi din fișierul **mesaj.txt** două enunțuri. Afișați șirul obținut prin concatenarea enunțului al doilea la primul.*

Implementarea C++

```
#include <iostream>
#include <string>
#include <fstream>
using namespace std;
string enunt1,enunt2;
ifstream fin("mesaj.txt");
int main() {
    // Citim doua enunturi de la tastatura
    cout<<"Introducem propozitia 1 din fisier!"<<endl;
    getline(fin,enunt1);
    cout<<"Introducem propozitia 2 din fisier!"<<endl;
    getline(fin,enunt2);
    // METODA 1
    cout<<"\nMetoda 1:\n\tConcatenarea enuntului 2 la primul va fi:
"<<endl;
    cout<<"\t"<<enunt1+' '+enunt2<<endl;
    // METODA 2
    cout<<"\nMetoda 2:\n\tConcatenarea enuntului 2 la primul va fi:
"<<endl;
    cout<<"\t"<<enunt1.append(' '+enunt2)<<endl;
    fin.close(); // inchidem fișierul de intrare
}
```

Problema 2.2 – 10 ***

*Elaborați un cod C/C++ care va citi din fișierul **numar.txt** două numere. Afișați șirul obținut prin concatenarea numărului al doilea la primul în același fișier, fără a pierde datele inițiale din fișierul respectiv.*

Implementarea C++

```
#include <iostream>
#include <string>
#include <fstream>
using namespace std;
string num1,num2;
ifstream fin("numar.txt");
int main() {
```

```
    // Citim doua numere din fisierul numar.txt
    cout<<"Citim numarul 1 din fisier!\n"; getline(fin,num1);
    cout<<"Citim numarul 2 din fisier!\n"; getline(fin,num2);
    fin.close(); // inchidem fisierul de intrare
    //deschidem fisierul pentru scrierea rezultatelor, fara a pier-
de datele initiale
    ofstream fout; fout.open("numar.txt",ios::app);
    // METODA 1
    fout<<"\nMetoda 1:\n\tConcatenarea numarului 2 la primul va fi:
"<<endl;
    fout<<"\t"<<num1+num2<<endl;
    // METODA 2
    fout<<"\nMetoda 2:\n\tConcatenarea numarului 2 la primul va fi:
"<<endl;
    fout<<"\t"<<num1.append(num2)<<endl;
    fout.close(); // inchidem fisierul de iesire
}
```

2.3 Probleme propuse

- ✔ *Problemele propuse în compartimentul 1.2 din „Rezolvarea problemelor cu pointeri” pot fi de asemenea realizate cu tipul de date fișier.*
- ✔ *Tipul de date fișier poate fi aplicat și pentru toate tipurile de probleme propuse în îndrumarul „Programare structurată în C++”, ediția 2022.*

1. *Elaborați un cod C/C++ care va afișa într-un fișier extern temperatura T ^{0}C (Celsius) exprimată în ^{0}F (Fahrenheit), ^{0}R (Rankin), ^{0}Re (Reaumur) și ^{0}K (Kelvin), dacă se cunoaște că* $T \in [-1000000, 1000000]$.
   Exemplu:

Date de intrare	***Date de ieșire***

`100`	`212`
	`671.67`
	`80`
	`315.15`

2. *Elaborați un cod C/C++ care va afișa într-un fișier extern timpul T (ani) exprimat în secunde, minute, ore și zile, dacă se cunoaște că* $T \in [0, 1000000]$.
   Exemplu:

Date de intrare	***Date de ieșire***

`12`	`378683424`
	`6311390.4`
	`105189.84`
	`4382.91`

3. *Elaborați un cod C/C++ care va afișa într-un fișier extern lungimea laturii unui poligon regulat de N laturi înscris (l1) și circumscris (l2) unui cerc de rază R1, respectiv R2. Precizați lungimile laturilor a cel puțin 5 poligoane regulate.*

Exemplu:

Date de intrare	***Date de ieșire***
45 15	3 77.94228 51.96152 4 63.63961 30 5 52.90067 21.79627 6 45 17.32050 7 39.04953 14.44723

4. *Elaborați un cod C/C++ care va afișa într-un fișier extern răspusul dacă trei numere întregi a, b și c pot satisface condițiile pitagorice.*
   Exemplu:

Date de intrare	***Date de ieșire***
3 4 5	9 16 25 - Satisface

5. *Elaborați un cod C/C++ care va afișa într-un fișier extern descompunerea în factori primi a unui număr întreg.*
   Exemplu:

Date de intrare	***Date de ieșire***
2020	2^2·5^1·101^1

6. *Elaborați un program în C/C++ care citește patru numere a, b, c și d. Considerând un sistem de axe ortogonale xOy, determinți poziția dreptelor $y_1=ax+b$ și $y_2=cx+d$. În caz că dreptele se intersectează, afișați și coordonatele punctului de intersecție.*
   Exemplu:

Date de intrare	***Date de ieșire***
-2 4 & -1 3	Se intersectează în: (1, 2)
-2 4 & -2 3	Dreptele sunt paralele
-2 4 & -2 4	Dreptele coincid

7. *Elaborați un program în C/C++ care citește coordonatele a 3 puncte dintr-un sistem de axe ortogonale xOy și determinți poziția unui punct P(x,y) față de triunghiul care poate fi construit.*
   Exemplu:

Date de intrare	*Date de ieșire*
-2 4 2 1 -1 3 1 1	Punctul P(1, 1) se afla in interiorul triunghiului.

8. *Elaborați un program C/C++ care va permite citirea de la tastatură a unui număr natural P, care este perimetrul unui triunghi isoscel. Afișați toate tripletele de numere naturale ce pot reprezenta lungimile laturilor acestui triunghi.*
   Exemplu:

Date de intrare	*Date de ieșire*
12	T1: 4 4 4 T2: 5 5 2 Total solutii: 2

9. *Elaborați un program C/C++ care va permite afișarea inversului unui număr întreg pozitiv introdus de la tastatură.*
   Exemplu:

Date de intrare	*Date de ieșire*
24689	Pentru 24689 avem 98642.

10. *Elaborați un cod C/C++ care va prelucra operațiuni cu elementele din fiecare zonă necolorată a unui array bidimensional de forma NxN, ce-conține doar numere întregi:*
    a) afișați elementul maxim divizibil cu 5;
    b) afișați elementul minim, pătrat perfect;
    c) afișați suma numerelor pare;
    d) afișați toate numerele prime din matrice separate prin spațiu;
    e) afișați toate numerele Fibonacci din matrice separate prin spațiu;
    f) afișați elementul minim și coordonatele acestuia.

Exemplu:

Date de intrare	***Date de ieșire***
4 4 12 13 14 15 16 11 10 17 18 19 20 21 23 24 22 25 26 27 28 29	25 16 190 13 11 17 19 23 29 13 21 10 cu coordonatele (1, 1)

11. *Elaborați un cod C/C++ care va calcula suma și produsul elementelor din seria Fibonacci a N numere naturale care se citesc dintr-un fișier extern numere.txt, iar rezultatele să se scrie în fisierul Solutie.txt.*

Exemplu:

Date de intrare	***Date de ieșire***
7 15 34 23 55 96 28 89	34+55+89 = 178 34·55·89 = 166430

12. *Elaborați un cod C/C++ care va determina valoarea unui polinom cu N monoame într-un punct x_0. Coeficienţii polinomului sunt memoraţi într-un tablou de numere reale, iar aceste date se citesc dintr-un fișier extern.*

Exemplu:

Date de intrare	***Date de ieșire***
3 0 2 -3 -5	Valoarea polinomului este y=-5

13. *Elaborați un cod C/C++ care va determina persoana din grup care are cea mai mare influență. Pentru un grup de N persoane, se definesc N perechi de forma (i,j) cu semnificaţia că persoana i este influenţată de persoana j.*

Exemplu:

Date de intrare	***Date de ieșire***
6 1 2 4 2	Persoana care are cea mai mare influiență este reprezentată prin numărul 2, ea

5 3 3 1 4 6	poate avea influiență asupra persoanelor 1, 3 și 4.

14. *Elaborați un cod C/C++ care va lista valorile tuturor punctelor şa şi poziţia lor dintr-o matrice dată M. Un punct M[i,j] este considerat punct şa dacă este poziționat minim pe linia i şi maxim pe coloana j sau invers.*
Exemplu:

Date de intrare	***Date de ieșire***
2 6 5 2 8 4 9 3 7 1 6 3 8 5	Puncte sa: M[1,2]=2 M[2,5]=8

15. *Elaborați un cod C/C++ care va verifica dacă matricea este pătrat magic sau nu (matricea este pătratică). O matrice pătratică este pătrat magic dacă sumele de pe fiecare linie, coloană și de pe cele două diagonale sunt egale.*
Exemplu:

Date de intrare	***Date de ieșire***
4 3 8 9 5 1 2 7 6	Da, matricea patratica reprezinta un patrat magic cu suma de 15.

16. *Elaborați un cod C/C++ care va verifica dacă matricea este pătrat prim sau nu (matricea este pătratică). O matrice pătratică este pătrat prim dacă sumele de pe fiecare linie și coloană sunt un număr prim.*
Exemplu:

Date de intrare	***Date de ieșire***
41 37 103 97 13 71 43 131 7	Da, matricea patratica reprezinta un patrat prim cu suma de 181.

Capitolul 3

APLICAŢII CU SORTĂRI DE DATE

3.1 Aplicaţii fundamentale

- ✔ *Unele probleme rezolvate vor fi însoțite de comentarii succinte.*
- ✔ *În acest compartiment vom aplica în mod implicit tipurile de sortări clasice asupra elementelor array-urilor.*
- ✔ *Verificați secvențele de cod C/C++ ale problemelor rezolvate, utilizați IDE-ul Code::Blocks.*

Problema 3.1 - 01

Elaborați un program C/C++ care va permite citirea datelor de la tastatură și aranjarea în ordine ascendentă a elementelor dintr-un array 1D de minim 10 elemente de două cifre fiecare. Pentru aranjare se va aplica metoda SelectionSort.

Implementarea C++

```
#include<iostream>
int v[25],n,i,j,aux;
using namespace std;
int main(){
     // Introducerea datelor problemei
     cout<<"Introduceti numarul de elemente ale tabloului unidimen-
sional: "; cin>>n;
     cout<<"Introduceti elementele tabloului unidimensional: \n";
     for(i=0;i<n;i++){
          cout<<"\tv["<<i<<"]= "; cin>>v[i];
     }
     cout<<"Tabloul unidimensional are urmatoarele elemente: ";
     for(i=0;i<n;i++)
     cout<<"\t"<<v[i]<<" ";
     // Implementarea algoritmului metodei SelectionSort
```

```
     for(i=0;i<n-1;i++)
         for(j=i+1;j<n;j++)
            if(v[i]>v[j]){
               aux=v[j];  v[j]=v[i]; v[i]=aux;
            }
     cout<<endl;
     // Afisarea rezultatelor problemei
     cout<<"Tabloul unidimensional sortat ascendent: \t";
     for(i=0;i<n;i++) cout<<"\t"<<v[i]<<" ";
}
```

Problema 3.1 - 02

Elaborați un program C/C++ care va permite citirea datelor de la tastatură și aranjarea în ordine ascendentă a elementelor dintr-un array 1D de minim 10 elemente de două cifre fiecare. Pentru aranjare se va aplica metoda InsertionSort.

Implementarea C++

```
#include<iostream>
int v[25],n,i,j,aux;
using namespace std;
int main(){
     // Introducerea datelor problemei
     cout<<"Introduceti numarul de elemente ale tabloului unidimen-
sional: "; cin>>n;
     cout<<"Introduceti elementele tabloului unidimensional: \n";
     for(i=0;i<n;i++){
          cout<<"\tv["<<i<<"]= "; cin>>v[i];
     }
     cout<<"Tabloul unidimensional are urmatoarele elemente: ";
     for(i=0;i<n;i++) cout<<"\t"<<v[i]<<" ";
     // Implementarea algoritmului metodei InsertionSort
     for(i=1;i<n;i++)
        if (v[i]<v[i-1]){
            aux=v[i];j=i-1;
            while (j>=0 && v[j]>aux){
                 v[j+1]=v[j]; j--;
            }
            v[j+1]=aux;
         }
     cout<<endl;
     // Afisarea rezultatelor problemei
```

```
     cout<<"Tabloul unidimensional sortat ascendent: \t";
     for(i=0;i<n;i++) cout<<"\t"<<v[i]<<" ";
}
```

Problema 3.1 - 03

Elaborați un program C/C++ care va permite citirea datelor de la tastatură și aranjarea în ordine ascendentă a elementelor dintr-un array 1D de minim 10 elemente de două cifre fiecare. Pentru aranjare se va aplica metoda BubbleSort.

Implementarea C++

```
#include<iostream>
int v[25],n,i,ok,aux;
using namespace std;
int main(){
     // Introducerea datelor problemei
     cout<<"Introduceti numarul de elemente ale tabloului unidimen-
sional: "; cin>>n;
     cout<<"Introduceti elementele tabloului unidimensional: \n";
     for(i=0;i<n;i++){
          cout<<"\tv["<<i<<"]= "; cin>>v[i];
     }
     cout<<"Tabloul unidimensional are urmatoarele elemente: ";
     for(i=0;i<n;i++) cout<<"\t"<<v[i]<<" ";
     // Implementarea algoritmului metodei BubbleSort
     while(ok!=1){
          ok=1;
          for(i=0;i<n-1;i++)
             if(v[i]>v[i+1]) {
                 aux=v[i]; v[i]=v[i+1]; v[i+1]=aux; ok=0;
             }
     }
     cout<<endl;
     // Afisarea rezultatelor problemei
     cout<<"Tabloul unidimensional sortat ascendent: \t";
     for(i=0;i<n;i++) cout<<"\t"<<v[i]<<" ";
}
```

Problema 3.1 - 04

Elaborați un program C/C++ care va permite citirea datelor de la tastatură și aranjarea în ordine ascendentă a elementelor dintr-un array 1D de minim 10 elemente de două cifre fiecare. Pentru aranjare se va aplica metoda CountingSort.

Implementarea C++

```
#include<iostream>
using namespace std;
int v[25],x[25],y[25],n,i,j;
int main(){
    // Introducerea datelor problemei
     cout<<"Introduceti numarul de elemente ale tabloului unidimen-
sional: "; cin>>n;
     cout<<"Introduceti elementele tabloului unidimensional: \n";
     for(i=0;i<n;i++){
          cout<<"\tv["<<i<<"]= ";  cin>>v[i];
     }
     cout<<"Tabloul unidimensional are urmatoarele elemente: \t\t";
     for(i=0;i<n;i++) cout<<"  "<<v[i];
     // Implementarea algoritmului metodei CuntingSort
     for(i=0;i<n;i++) x[i]=v[i];
     for(i=0;i<n;i++)
        for(j=i+1;j<n;j++)
            if (v[i]<v[j]) y[j]++;
            else y[i]++;
        for(i=0;i<n;i++) v[y[i]]=x[i];
     // Afisarea rezultatelor problemei
     cout<<"\nPozitiile elementelor care trebuie sa fie in vectorul
sortat: \t";
     for(i=0;i<n;i++) cout<<"   "<<y[i];
     cout<<"\nTabloul unidimensional sortat ascendent: \t\t\t";
     for(i=0;i<n;i++) cout<<"  "<<v[i];
}
```

Problema 3.1 - 05

Elaborați un program C/C++ care va permite citirea datelor de la tastatură a unui array 2D de numere întregi și va aranja în ordine ascendentă toate elemente unei coloane cu indicele citit de la tastatură. Pentru aranjare se va aplica metoda SelectionSort.

Implementarea C++

```
#include <iostream>
#include <cstdlib>
using namespace std;
int main (){
    int  a[100][100],n,k,i,j,aux;
    // n – dimensiunea matricii patratice si k - indicele coloanei
     cout<<"Introduceti dimeniunea matricii patratice N si indicele
coloanei K: "; cin>>n>>k;
    cout<<"Introduceti elementele matricii patratice de dimensiunea
"<<n<<"!"<<endl;
    for(i=0;i<n;i++){
        for(j=0;j<n;j++) cin>>a[i][j];
    }
    getchar(); system("CLS"); // curatim consola
    cout<<"Matricea patratica de dimensiunea "<<n<<" este: "<<endl;
    for(i=0;i<n;i++){
        for(j=0;j<n;j++) cout<<a[i][j]<<" ";
        cout<<endl;
    }
    ///Sortam coloana cu indicele k conform metodei SelectionSort
    for(i=0;i<n-1;i++)
        for(j=i+1;j<n;j++)
            if(a[i][k]>a[j][k]){
                aux=a[j][k]; a[j][k]=a[i][k]; a[i][k]=aux;
            }
    // Afisarea rezultatelor problemei
    cout<<"Afisam matricea in care coloana cu indicele "<<k<<" este
sortata ascendent!"<<endl;
    for(i=0;i<n;i++){
        for(j=0;j<n;j++) cout<<a[i][j]<<" ";
        cout<<endl;
    }
}
```

Problema 3.1 - 06

Elaborați un program C/C++ care va permite citirea datelor de la tastatură a unui array 2D de numere întregi și va aranja în ordine ascendentă (sau descendentă) toate elemente unei linii cu indicele citit de la tastatură. Se va alege una dintre metodele de sortare studiate: SelectionSort / InsertionSort.

Implementarea C++

```
#include <iostream>
#include <cstdlib>
using namespace std;
int main (){
    int  a[100][100],n,k,i,j,aux;
    // n – dimensiunea matricii patratice si k - indicele liniei
     cout<<"Introduceti dimeniunea matricii patratice N si indicele
liniei K: "; cin>>n>>k;
    cout<<"Introduceti elementele matricii patratice de dimensiunea
"<<n<<"!"<<endl;
    for(i=0;i<n;i++){
        for(j=0;j<n;j++) cin>>a[i][j];
    }
    getchar(); system("CLS"); // curatim consola
    cout<<"Matricea patratica de dimensiunea "<<n<<" este: "<<endl;
    for(i=0;i<n;i++){
        for(j=0;j<n;j++) cout<<a[i][j]<<" ";
        cout<<endl;
    }
    ///Sortam linia cu indicele k conform metodei SelectionSort
    for(i=0;i<n-1;i++)
        for(j=i+1;j<n;j++)
            if(a[k][i]>a[k][j]){
                aux=a[k][i]; a[k][i]=a[k][j]; a[k][j]=aux;
            }
     // Afisarea rezultatelor problemei
     cout<<"Afisam matricea in care linia cu indicele "<<k<<" este
sortata ascendent!"<<endl;
    for(i=0;i<n;i++){
        for(j=0;j<n;j++) cout<<a[i][j]<<" ";
        cout<<endl;
    }
}
```

Problema 3.1 - 07

Elaborați un program C/C++ care va permite citirea datelor de la tastatură a unui array 2D de numere întregi și va aranja în ordine ascendentă toate liniile conform sumei elementelor. Se va alege una dintre metodele de sortare studiate: SelectionSort / BubbleSort.

Implementarea C++

```
#include <iostream>
using namespace std;
int main(){
    int a[101][101],s[101],m,n;
    cout<<"Introduceti dimeniunile matricii: "; cin>>n>>m;
    cout<<"Introduceti elementele matricii: "<<endl;
    for(int i=0;i<n;i++)
        for(int j=0;j<m;j++) cin>>a[i][j];
    // Calculam suma elementelro de pe fiecare rand
    for(int i=0;i<n;i++){
        int sum=0;
        for(int j=0;j<m;j++) sum=sum+a[i][j];
        s[i]=sum;
    }
    // Interschimbam fiecare rand in dependenta de suma obtinuta
    for(int i=0;i<n-1;i++)
        for(int j=i;j<n;j++)
            if(s[i]>s[j]){
                int aux=s[i]; s[i]=s[j]; s[j]=aux;
                for(int k=0;k<m;k++){
                    aux=a[i][k]; a[i][k]=a[j][k]; a[j][k]=aux;
                }
            }
      // Afisarea rezultatelor problemei
     cout<<"Afisam elementele matricii dupa interschimbarea lini-
ilor: "<<endl;
    for(int i=0;i<n;i++){
        for(int j=0;j<m;j++) cout<<a[i][j]<<" ";
        cout<<" cu suma elementelor: "<<s[i]<<endl;
    }
}
```

Problema 3.1 - 08

Elaborați un program C/C++ care va permite citirea unui șir de cuvinte de la tastatură și aranjarea în ordine alfabetică (ascendentă) a elementelor. Se va aplica o variabilă de tip pointer și BubbleSort.

Implementarea C++

```
#include <iostream>
#include <cstring>
```

```
using namespace std;
int main(){
    char cuvinte[10][25],*aux; int i,n,gasit;
    cout<<"Introduceti numarul de cuvinte: "; cin>>n;
    for(i=0;i<n;i++){
        cout<<"cuvant["<<i<<"]= "; cin>>cuvinte[i];
    }
    ///Sortam cuvintele conform metodei BubbleSort
    while (gasit){
        gasit=0;
        for(i=0;i<n-1;i++){
            if (strcmp(cuvinte[i],cuvinte[i+1])>0){
                strcpy(aux,cuvinte[i]);
                strcpy(cuvinte[i],cuvinte[i+1]);
                strcpy(cuvinte[i+1],aux); gasit=1;
            }
        }
     }
     // Afisarea rezultatelor problemei
     cout<<"Cuvintele sortate alfabetic (ascendent) sunt: \n";
     for(i=0;i<n;i++){
        cout<<i<<" "<<cuvinte[i]<<endl;
     }
}
```

Problema 3.1 – 09 ***

Elaborați un program C/C++ care va permite citirea unui șir de caractere (cuvânt) implicit și aranjarea în ordine alfabetică (ascendentă și descendentă) a elementelor (literelor sale).

Implementarea C++

```
#include <iostream>
#include <cstring>
#include <algorithm> // aplicam functia sort();
using namespace std;
string s = "PROGRAMARE"; // cuvantul declarat implicit
int main(){
    cout<<"Datele initiale pentru sortare: \t"<<s<<endl;
    // sortarea ascendenta
    cout<<"Sortarea literelor ascendent (A-Z): \t";
    sort(s.begin(), s.end());
    cout<<s<<endl;
```

```
    // sortarea descendenta
    cout<<"Sortarea literelor descendent (Z-A): \t";
    sort(s.begin(), s.end(),greater<char>());
    cout<<s<<endl;
    return 0;
}
```

- ✔ *Sortarea este una dintre cele mai de bază funcții aplicate datelor. Înseamnă aranjarea datelor într-un mod anume, care poate fi în creștere sau în descreștere. Există o funcție încorporată în C++ STL cu numele sort ().*
- ✔ *Această funcție utilizează intern IntroSort. În mai multe detalii, este implementat folosind hibridul QuickSort, HeapSort și InsertionSort. În mod implicit, folosește QuickSort, dar dacă QuickSort face partiționare neloială și necesită mai mult de N*logN timp, acesta trece la HeapSort și când dimensiunea tabloului devine cu adevărat mică, trece la InsertionSort.*

Problema 3.1 – 10 ***

Elaborați un program C/C++ care va permite citirea unui șir de caractere (cuvânt) implicit și aranjarea în ordine alfabetică (ascendentă sau descendentă) a elementelor (literelor sale). Optimizați complexitatea temporală a algoritmului. Șirul de caractere va fi prezentat ca un obiect.

Implementarea C++

```
#include <iostream>
#include <cstring>
const int MAX_CHAR = 26;
using namespace std;
string s = "PROGRAMARE"; // cuvantul declarat implicit
int main(){
    cout<<"Datele initiale pentru sortare: \t"<<s<<endl;
    // sortarea ascendenta
    cout<<"Sortarea literelor ascendent (A-Z): \t";
    //Numarul initial al tuturor cartelor este inițializat la zero.
```

```
    int charNumar[MAX_CHAR]={0};
    // Traversam string-ul si incrementam numarul de caractere
    for (int i=0; i<s.length(); i++)
        charNumar[s[i]-'A']++;
    for (int i=0;i<MAX_CHAR;i++)
        for (int j=0;j<charNumar[i];j++)
            cout<<(char)('A'+i);
    return 0;
}
```

- ✔ *Deși nu s-a studiat în detaliu noțiunile de complexitate a algoritmilor, problema 3.1-09 este soluționată cu o complexitate temorală de: O(n log n), iar problema 3.1-10 este soluționată cu o complexitate temorală de: O(n), unde n este lungimea șirului de intrare.*
- ✔ *O abordare eficientă va fi aceea de a observa mai întâi că pot exista în total doar 26 de caractere unice. Deci, putem stoca numărul de apariții al tuturor caracterelor de la „A” la „Z” într-un array hash. Primul index al array-ului hash va reprezenta caracterul „A”, al doilea va reprezenta „B” ș.a.m.d. În cele din urmă, vom parcurge pur și simplu array-ul hash și vom imprima caracterele de la „A” la „Z” de câte ori au apărut în șirul de intrare.*
- ✔ *Deci, algoritmul de la problema 3.1-10 este mai eficient față de algoritmul problemei 3.1-09.*

3.2 Aplicaţii ale sortărilor cu pointeri şi fişiere

- ✔ *Unele probleme rezolvate vor fi însoțite de comentarii succinte.*
- ✔ *În acest compartiment vom aplica diverse tipuri de sortări cu implementarea pointerilor și a fișierelor.*
- ✔ *Verificați secvențele de cod C/C++ ale problemelor rezolvate, utilizați IDE-ul Code::Blocks.*

Problema 3.2 - 01

Elaborați un program C/C++ care va include un array 1D de numere întregi, declarat implicit în program. Sortați ascendent și descendent elementele array-ului 1D folosind metoda selecției cu pointeri.

Implementarea C/C++

```
#include <iostream>
using namespace std;
int n=5, a[]={10, 23, 14, 21, 91};
int i, j, t, *ptr;
int main(){
    cout<<"\nVectorul initial:\n";
    for(i=0; i<n; i++)
        cout<<a[i]<<" ";
    // Metoda selectie de sortare ascendentă
    for(i=0; i<n; i++) {
        ptr=&a[0];
        for(j=i+1; j<n; j++)
           if(*(ptr+j)<*(ptr+i)) {
              t=*(ptr+i); *(ptr+i)=*(ptr+j); *(ptr+j)=t;
        }
    }
    cout<<"\n\nVectorul sortat ascendent:\n";
    for(i=0; i<n; i++)
        cout<<*(ptr+i)<<" ";
    // Metoda selectie de sortare descendentă
    for(i=0; i<n; i++) {
        ptr=&a[0];
        for(j=i+1; j<n; j++)
```

```
            if(*(ptr+j)>*(ptr+i)) {
                t=*(ptr+i); *(ptr+i)=*(ptr+j); *(ptr+j)=t;
        }
    }
    cout<<"\n\nVectorul sortat descendent:\n";
    for(i=0; i<n; i++)
        cout<<*(ptr+i)<<" ";
}
```

Problema 3.2 - 02

Elaborați un program C/C++ care va include un array 1D de numere întregi, citit dintr-un fișier extern ***array1D.txt****. Sortați ascendent și descendent elementele array-ului 1D folosind metoda selecției cu pointeri. Soluția se va păstra în fișierul* ***selection.txt****.*

Implementarea C/C++

```
#include <iostream>
#include <fstream>
using namespace std;
int n,a[20],i,j,t,*ptr;
ifstream fin("array1D.txt"); // fisierul de intrare
ofstream fout("selection.txt"); // fisierul de iesiere
int main(){
    //Citim datele din fisierul de intrare
    fin>>n; //dimensiunea vectorului
    for(i=0; i<n; i++)
        fin>>a[i];
    //Afisarea datelor din fisier la consola
    cout<<"\nVectorul initial:\n";
    for(i=0; i<n; i++)
        cout<<a[i]<<" ";
    // Metoda selectie de sortare ascendentă
    for(i=0; i<n; i++) {
        ptr=&a[0];
        for(j=i+1; j<n; j++)
            if(*(ptr+j)<*(ptr+i)) {
                t=*(ptr+i); *(ptr+i)=*(ptr+j); *(ptr+j)=t;
        }
    }
    fout<<"Vectorul sortat ascendent:\n";
    for(i=0; i<n; i++)
        fout<<*(ptr+i)<<" ";
```

```
    // Metoda selectie de sortare descendentă
    for(i=0; i<n; i++) {
        ptr=&a[0];
        for(j=i+1; j<n; j++)
           if(*(ptr+j)>*(ptr+i)) {
              t=*(ptr+i); *(ptr+i)=*(ptr+j); *(ptr+j)=t;
        }
    }
    fout<<"\n\nVectorul sortat descendent:\n";
    for(i=0; i<n; i++)
        fout<<*(ptr+i)<<" ";
    cout<<"\nRezultatele sunt pastrate in fisierul selection.txt!";
    cout<<endl;
    fin.close(); fout.close(); // inchidem fisierele
}
```

Problema 3.2 - 03

Elaborați un program C/C++ care va include un array 1D de numere reale, citit dintr-un fișier extern ***array1D.txt****. Sortați ascendent și descendent elementele array-ului 1D folosind metoda inserției fără pointeri. Soluția se va păstra în fișierul* ***insertion.txt****.*

Implementarea C/C++

```
#include<iostream>
#include <fstream>
float v[25],aux; int n,i,j;
using namespace std;
ifstream fin("array1D.txt"); // fisierul de intrare
ofstream fout("insertion.txt"); // fisierul de iesiere
int main(){
    //Citim datele din fisierul de intrare
    fin>>n; //dimensiunea vectorului
    for(i=0; i<n; i++)
        fin>>v[i];
    cout<<"Tabloul unidimensional are urmatoarele elemente: \t";
    for(i=0;i<n;i++) cout<<"  "<<v[i]<<" ";
    // Implementarea algoritmului metodei InsertionSort ascendent
    for(i=1;i<n;i++)
       if (v[i]<v[i-1]){
           aux=v[i];j=i-1;
           while (j>=0 && v[j]>aux){
                v[j+1]=v[j]; j--;
```

```
            }
            v[j+1]=aux;
         }
    cout<<endl;
    // Afisarea rezultatelor problemei
    fout<<"Tabloul unidimensional sortat ascendent: \t\t";
    for(i=0;i<n;i++) fout<<"  "<<v[i]<<" ";
    // Implementarea algoritmului metodei InsertionSort descendent
    for(i=1;i<n;i++)
        if (v[i]>v[i-1]){
            aux=v[i];j=i-1;
            while (j>=0 && v[j]<aux){
                 v[j+1]=v[j]; j--;
            }
            v[j+1]=aux;
         }
    cout<<endl;
    // Afisarea rezultatelor problemei
    fout<<"\nTabloul unidimensional sortat descendent: \t\t";
    for(i=0;i<n;i++) fout<<"  "<<v[i]<<" ";
    cout<<"\nRezultatele sunt pastrate in fisierul insertion.txt!";
    cout<<endl;
    fin.close(); fout.close(); // inchidem fisierele
}
```

Problema 3.2 - 04

Elaborați un program C/C++ care va include un array 1D de numere reale, citit dintr-un fișier extern ***array1D.txt****. Sortați ascendent și descendent elementele array-ului 1D folosind metoda bulelor cu pointeri. Soluția se va păstra în fișierul* ***bubble.txt****.*

Implementarea C/C++

```
#include<iostream>
#include <fstream>
float v[25],aux; int n,i,j,ok;
using namespace std;
ifstream fin("array1D.txt"); // fisierul de intrare
ofstream fout("buble.txt"); // fisierul de iesiere
int main(){
     //Citim datele din fisierul de intrare
     fin>>n; //dimensiunea vectorului
     for(i=0; i<n; i++)
```

```
        fin>>v[i];
     cout<<"Tabloul unidimensional are urmatoarele elemente: ";
     for(i=0;i<n;i++) cout<<"\t"<<v[i]<<" ";
     // Implementarea algoritmului metodei BubbleSort ascendent
     while(ok!=1){
          ok=1; aux=*v;
          for(i=0;i<n-1;i++)
             if(v[i]>v[i+1]) {
                 aux=v[i]; v[i]=v[i+1]; v[i+1]=aux; ok=0;
             }
     }
     cout<<endl;
     // Afisarea rezultatelor problemei
     fout<<"Tabloul unidimensional sortat ascendent: \t";
     for(i=0;i<n;i++) fout<<"\t"<<v[i]<<" ";
     // Implementarea algoritmului metodei BubbleSort descendent
     aux=*v;
     for (i = 0; i < n; ++i){
        for (j = i + 1; j < n; ++j){
            if (v[i] < v[j]){
                aux = v[i]; v[i] = v[j]; v[j] = aux;
            }
        }
    }
    cout<<endl;
    // Afisarea rezultatelor problemei
    fout<<"\nTabloul unidimensional sortat descendent: \t";
    for(i=0;i<n;i++) fout<<"\t"<<v[i]<<" ";
    cout<<"\nRezultatele sunt pastrate in fisierul bubble.txt!";
    cout<<endl;
    fin.close(); fout.close(); // inchidem fisierele
}
```

Problema 3.2 - 05

Elaborați un program C/C++ care va include un array 1D de litere mici, declarat implicit. Sortați ascendent și descendent elementele array-ului 1D folosind metoda bulelor cu pointeri. Soluția se va păstra în fișierul ***bubblechar.txt***.

Implementarea C/C++
`#include <iostream>` `#include <fstream>`

```
#define n 6
using namespace std;
int i,j;
char v[20] = {'A','N','D','R','E','W'},*p[20],*temp,aux;
ofstream fout("bublechar.txt"); // fisierul de iesiere
int main(){
    // Afisam datele initiale ale vectorului
    cout<<"Tabloul unidimensional are urmatoarele elemente: \t";
    for(i=0;i<n;i++) cout<<"  "<<v[i];
    // Implementarea algoritmului metodei BubbleSort ascendent
    for (i=0;i<n;i++){
        p[i] = (char *)(v) + i;
    }
    for(i=0;i<n;i++){
        for(j=i+1;j<n;j++){
            if(*p[j-1]>*p[j]){
                temp = p[j]; p[j] = p[j-1]; p[j-1] = temp;
            }
        }
    }
    // Afisarea rezultatelor problemei
    fout<<"\nTabloul unidimensional sortat ascendent: \t\t";
    for(i=0;i<n;i++) fout<<"  "<<*p[i];
    // Implementarea algoritmului metodei BubbleSort descendent
    aux=*v;
     for (i = 0; i < n; ++i){
        for (j = i + 1; j < n; ++j){
            if (v[i] < v[j]){
                aux = v[i]; v[i] = v[j]; v[j] = aux;
            }
        }
    }
    // Afisarea rezultatelor problemei
    fout<<"\nTabloul unidimensional sortat descendent: \t\t";
    for(i=0;i<n;i++) fout<<"  "<<v[i];
    cout<<"\nRezultatele sunt pastrate in fisierul bublechar.txt!";
    cout<<endl;
    fout.close(); // inchidem fisierul de iesire
}
```

Problema 3.2 - 06

Elaborați un program C/C++ care va include un array 2D de numere întregi, declarat implicit. Sortați ascendent elementele pare și

descendent elementele impare ale array-ului 2D sub formă de vector folosind metoda selecției fără pointeri. Soluția se va păstra în fișierul ***solmatrix.txt***.

Implementarea C/C++

```
#include <iostream>
#include <fstream>
using namespace std;
int n=3,m=4,i,j;
ofstream fout("solmatrix.txt");
int main(){
    int a[n][m] = {  {11,55,78,99},
                     {76,33,51,60},
                     {37,18,59,40}
                  };
    // Afisarea elementelor matricii initiale
    cout<<"Elementele matricii initiale sunt: "<<endl;
    for (int i=0;i<n;i++) {
        for (int j=0;j<m;j++) {
            cout<<a[i][j]<<" ";
        }
        cout<<endl;
    }
    // sortare ascendenta aplicand SelectionSort
    for (i=0;i<n;i++) {
        for (j=0;j<m;j++) {
            int minimum = a[i][j]; // elementul minim
            int z=i,q=j;
            // Verificam daca exista un element mai mic in matrice
            int w = j;
            for (int k=i;k<n;k++) {
                for (;w<m;w++) {
                    // actualizam elementul minim
                    if (a[k][w] < minimum && a[i][j]%2==0) {
                        minimum = a[k][w];
                        // actualizam indicele elementului minim
                        z=k;q=w;
                    }
                }
                w=0;
            }
           // Schimbam elementul curent cu cel minim
           int temp=a[i][j]; a[i][j]=a[z][q]; a[z][q]=temp;
```

```
        }
    }
    // Afisarea elementelor matricii in ordine ascendenta
    fout<<"Elementele matricii sortate ascendent sunt: \n";
    for (i=0;i<n;i++) {
        for (j=0;j<m;j++) {
            fout<<a[i][j]<<" ";
        }
        fout<<endl;
    }
    // sortare descendenta aplicand SelectionSort
    for (i=0;i<n;i++) {
        for (j=0;j<m;j++) {
            int minimum=a[i][j]; // elementul minim
            int z=i,q=j;
            // Verificam daca exista un element mai mic in matrice
            int w=j;
            for (int k=i;k<n;k++) {
                for (;w<m;w++) {
                    // actualizam elementul minim
                    if (a[k][w] > minimum && a[i][j]%2!=0) {
                        minimum = a[k][w];
                        // actualizam indicele elementului minim
                        z=k;q=w;
                    }
                }
                w=0;
            }
           // Schimbam elementul curent cu cel minim
           int temp=a[i][j]; a[i][j]=a[z][q]; a[z][q]=temp;
        }
    }
    // Afisarea elementelor matricii in ordine descendenta
    fout<<"Elementele matricii sortate descendent sunt: \n";
    for (i=0;i<n;i++) {
        for (j=0;j<m;j++) {
            fout<<a[i][j]<<" ";
        }
        fout<<endl;
    }
    cout<<"\nRezultatele sunt pastrate in fisierul solmatrix.txt!";
    cout<<endl;
    fout.close(); // inchidem fisierul de iesire
}
```

Problema 3.2 - 07

Elaborați un program C/C++ care va include un array 2D de litere, declarat implicit. Sortați ascendent și descendent elementele array-ului 2D sub formă de vector folosind metoda selecției fără pointeri. Soluția se va păstra în fișierul ***solchar.txt****.*

Implementarea C/C++

```
#include <iostream>
#include <fstream>
using namespace std;
int n=3,m=10,i,j;
ofstream fout("solchar.txt");
int main(){
    char a[n][m] = { {'Z','X','C','V','B','N','M','A','S','D'},
                     {'F','G','H','J','K','L','Q','W','E','R'},
                     {'T','Y','U','I','O','P','0','1','2','3'}
                   };
    // Afisarea elementelor matricii initiale
    cout<<"Elementele matricii initiale sunt: "<<endl;
    for (i=0;i<n;i++) {
        for (j=0;j<m;j++) {
            cout<<a[i][j]<<" ";
        }
        cout<<endl;
    }
    // sortare ascendenta aplicand SelectionSort
    for (i=0;i<n;i++) {
        for (j=0;j<m;j++) {
            int minimum = a[i][j]; // elementul minim
            int z=i,q=j;
            // Verificam daca exista un element mai mic in matrice
            int w = j;
            for (int k=i;k<n;k++) {
                for (;w<m;w++) {
                    // actualizam elementul minim
                    if (a[k][w] < minimum) {
                        minimum = a[k][w];
                        // actualizam indicele elementului minim
                        z=k;q=w;
                    }
                }
                w=0;
            }
```

```
            // Schimbam elementul curent cu cel minim
            int temp=a[i][j]; a[i][j]=a[z][q]; a[z][q]=temp;
        }
    }
    // Afisarea elementelor matricii in ordine ascendenta
    fout<<"Elementele matricii sortate ascendent sunt: \n";
    for (i=0;i<n;i++) {
        for (j=0;j<m;j++) {
            fout<<a[i][j]<<" ";
        }
        fout<<endl;
    }
    // sortare descendenta aplicand SelectionSort
    for (i=0;i<n;i++) {
        for (j=0;j<m;j++) {
            int minimum=a[i][j]; // elementul minim
            int z=i,q=j;
            // Verificam daca exista un element mai mic in matrice
            int w=j;
            for (int k=i;k<n;k++) {
                for (;w<m;w++) {
                    // actualizam elementul minim
                    if (a[k][w] > minimum) {
                        minimum = a[k][w];
                        // actualizam indicele elementului minim
                        z=k;q=w;
                    }
                }
                w=0;
            }
            // Schimbam elementul curent cu cel minim
            int temp=a[i][j]; a[i][j]=a[z][q]; a[z][q]=temp;
        }
    }
    // Afisarea elementelor matricii in ordine descendenta
    fout<<"Elementele matricii sortate descendent sunt: \n";
    for (i=0;i<n;i++) {
        for (j=0;j<m;j++) {
            fout<<a[i][j]<<" ";
        }
        fout<<endl;
    }
    cout<<"\nRezultatele sunt pastrate in fisierul solchar.txt!";
    cout<<endl; fout.close(); // inchidem fisierul de iesire
}
```

Problema 3.2 - 08

Elaborați un program C/C++ care va include un array 2D de numere, declarat implicit. Sortați ascendent și descendent elementele array-ului 2D sub formă de vector folosind metoda selecției fără pointeri. Soluția se va păstra în fișierul ***solnum.txt****.*

Implementarea C/C++

```
#include <iostream>
#include <fstream>
using namespace std;
int n=3,m=5,i,j;
ofstream fout("solnum.txt");
int main(){
    int a[n][m] = { {17,18,19,20,22},
                    {15,16,14,21,23},
                    {13,11,12,25,24}
                  };
    // Afisarea elementelor matricii initiale
    cout<<"Elementele matricii initiale sunt: "<<endl;
    for (int i=0;i<n;i++) {
        for (int j=0;j<m;j++) {
            cout<<a[i][j]<<" ";
        }
        cout<<endl;
    }
    // sortare ascendenta aplicand SelectionSort
    for (int i=0;i<n;i++) {
        for (int j=0;j<m;j++) {
            int minimum = a[i][j]; // elementul minim
            int z=i,q=j;
            // Verificam daca exista un element mai mic in matrice
            int w = j;
            for (int k=i;k<n;k++) {
                for (;w<m;w++) {
                    // actualizam elementul minim
                    if (a[k][w] < minimum) {
                        minimum = a[k][w];
                        // actualizam indicele elementului minim
                        z=k;q=w;
                    }
                }
                w=0;
            }
```

```
            // Schimbam elementul curent cu cel minim
            int temp=a[i][j]; a[i][j]=a[z][q]; a[z][q]=temp;
        }
    }
    // Afisarea elementelor matricii in ordine ascendenta
    fout<<"Elementele matricii sortate ascendent sunt: \n";
    for (int i=0;i<n;i++) {
        for (int j=0;j<m;j++) {
            fout<<a[i][j]<<" ";
        }
        fout<<endl;
    }
    // sortare descendenta aplicand SelectionSort
    for (int i=0;i<n;i++) {
        for (int j=0;j<m;j++) {
            int minimum=a[i][j]; // elementul minim
            int z=i,q=j;
            // Verificam daca exista un element mai mic in matrice
            int w=j;
            for (int k=i;k<n;k++) {
                for (;w<m;w++) {
                    // actualizam elementul minim
                    if (a[k][w] > minimum) {
                        minimum = a[k][w];
                        // actualizam indicele elementului minim
                        z=k;q=w;
                    }
                }
                w=0;
            }
            // Schimbam elementul curent cu cel minim
            int temp=a[i][j]; a[i][j]=a[z][q]; a[z][q]=temp;
        }
    }
    // Afisarea elementelor matricii in ordine descendenta
    fout<<"Elementele matricii sortate descendent sunt: \n";
    for (int i=0;i<n;i++) {
        for (int j=0;j<m;j++) {
            fout<<a[i][j]<<" ";
        }
        fout<<endl;
    }
    cout<<"\nRezultatele sunt pastrate in fisierul solnum.txt!";
    cout<<endl; fout.close(); // inchidem fisierul de iesire
}
```

- ✔ *Cuvântul cheie **auto** specifică faptul că tipul variabilei care este declarată va fi dedus automat din inițializator.*
- ✔ *În cazul funcțiilor, dacă tipul lor de returnare este automat, atunci acesta va fi evaluat prin expresia tipului de returnare la runtime.*
- ✔ *Variabila declarată cu cuvânt cheie **auto** ar trebui inițializată doar în momentul declarării sale, altfel va exista o eroare în timpul compilării.*

Problema 3.2 – 09 ***

*Elaborați un program C/C++ care va include un șir de cuvinte, declarat implicit în program. Sortați ascendent și descendent cuvintele din șir, aplicați tipul de date pointer. Soluția se va păstra în fișierul **elevi.txt**.*

Implementarea C/C++

```
#include <iostream>
#include <string>
#include <algorithm>
#include <sstream>
#include <vector>
#include <fstream>
using namespace std;
ofstream fout("elevi.txt");
int main(){
    vector<string>v;
    string s="Eva Victor Alex Rafael Andy Oleg Onisim Iov Xerx",ss;
    istringstream t(s);
    while(t>>ss){
        v.push_back(ss);
    }
    // Afisam datele initiale ale sirului de cuvinte
    cout<<"Lista initiala a elevilor din grupa: \n";
    for (auto i=v.begin(); i!=v.end(); ++i){
        cout<<"\t"<<*i<<endl;
    }
    // sortare ascendenta aplicand functia standarda sort()
    sort(v.begin(),v.end());
```

```
    fout<<"Sortarea ascendenta a elevilor: \n";
    //auto specifică faptul că tipul variabilei care este declarată
va fi dedus automat din inițializator
    for (auto i=v.begin(); i!=v.end(); ++i){
        fout<<"\t"<<*i<<endl;
    }
    // sortare descendenta aplicand functia standarda sort()
    sort(v.begin(),v.end(),greater<string>());
    fout<<"Sortarea descendenta a elevilor: \n";
    for (auto i=v.begin(); i!=v.end(); ++i){
        fout<<"\t"<<*i<<endl;
    }
    cout<<"\nRezultatele sunt pastrate in fisierul elevi.txt!";
    cout<<endl;
    fout.close(); // inchidem fisierul de iesire
}
```

Problema 3.2 – 10 ***

Elaborați un program C/C++ care va include un array 1D de cuvinte, declarat implicit în program. Sortați ascendent și descendent elementele array-ului 1D în conformitate cu lungimea cuvintelor. Se va aplica algoritmul InsertionSort, iar soluția se va păstra în fișierul ***wordlength.txt****.*

Exemple:

Date de intrare	*Date de ieşire*
A={ “MARCEL”, “EU”, “SUNT” }	A={ “EU”, “SUNT”, “MARCEL”}
B={ “NOI”, “STUDENTI”, “SUNTEM”, “DISCIPLINATI” }	B={ “NOI”, “SUNTEM”, “STUDENTI”, “DISCIPLINATI” }

Implementarea C/C++

```
#include <iostream>
#include <fstream>
using namespace std;
string a[] = {"MARCEL", "EVA", "JOHN", "JACOB"};
int n = sizeof(a)/sizeof(a[0]),i,j;
ofstream fout("wordlength.txt");
int main(){
    // Afisarea datelor initiale
```

```
    fout<<"Elementele sirului de cuvinte initiale sunt: \t\t";
    for (int i=0; i<n; i++)
        fout<<a[i]<<" "; fout<<endl;
    // Secventa de cod care aplica metoda InsertionSort ascendent
    for (i=1;i<n;i++){
        string temp=a[i];
        // Inseram a[j] in pozitia lui corespunzatoare
        j=i-1;
        while (j>=0 && temp.length()<a[j].length()){
            a[j+1]=a[j]; j--;
        }
        a[j+1]=temp;
    }
    // Afisam rezultatul sortarii ascendente
    fout<<"\nElementele sirului de cuvinte sortat ascendent sunt: \
t";
    for (int i=0; i<n; i++)
        fout<<a[i]<<" "; fout<<endl;
    // Secventa de cod care aplica metoda InsertionSort descendent
    for (i=1;i<n;i++){
        string temp=a[i];
        // Inseram a[j] in pozitia lui corespunzatoare
        j=i-1;
        while (j>=0 && temp.length()>a[j].length()){
            a[j+1]=a[j]; j--;
        }
        a[j+1]=temp;
    }
    // Afisam rezultatul sortarii descendente
       fout<<"\nElementele sirului de cuvinte sortat descendent
sunt: \t";
    for (int i=0; i<n; i++)
        fout<<a[i]<<" "; fout<<endl;
    cout<<"Rezultatele sunt pastrate in fisierul wordlength.txt!";
    cout<<endl;
    fout.close(); // inchidem fisierul de iesire
}
```

3.3 Probleme propuse

Implementaţi tipul pointer pentru metodele de sortare *Selection Sort* şi *Insertion Sort* pentru problemele 1 - 10 propuse mai jos:

1. *Elaboraţi un program C/C++ care va permite citirea unui vector X cu N componente, ordonat strict crescător şi o valoare Y. Să se insereze această valoare în vectorul X, astfel încât el să rămână ordonat strict crescător. Se va face o căutare rapidă a lui Y în şir (căutare binară). Şirul iniţial şi cel rezultat se vor tipări cu câte 3 elemente pe linie.*
   Exemplu:

Date de intrare	***Date de ieşire***

`5 9`
`3 6 2 7 9 4 7 0 3`
	`0 2 3`
	`3 4 5`
	`6 7 7`
	`9`

2. *Elaboraţi un program C/C++ care va permite citirea unei valori X şi un vector A cu N elemente, să se separe acest vector în două partiţii, astfel încât elementele din prima partiţie să fie mai mici sau egale cu X, iar cele din a doua partiţie să fie strict mai mari decât X.*
   Exemplu:

Date de intrare	***Date de ieşire***
`6 10` `3 6 2 7 9 4 7 9 0 3`	`0 2 3 3 4 6` `7 7 9 9`

3. *Elaboraţi un program C/C++ care va permite citirea unui vector A cu numere întregi cu scopul de a sorta elementele acestuia după frecvenţa indecşilor. Dacă două elemente au frecvenţe diferite, atunci elementul care va avea cea mai mare frecvenţă a apariţiei va fi poziţionat înainte (cazul descrescător), în caz contrar, cel care va avea cea mai mică frecvenţă a apariţiei sale va fi poziţionat înainte (cazul crescător).*

Exemplu:

Date de intrare	***Date de ieșire***
11 3 3 1 1 1 8 3 6 8 7 8	3 3 3 1 1 1 8 8 8 6 7

4. *Elaborați un program C/C++ care va permite citirea a doi vectori A și B cu numere întregi cu scopul de a ordona elementele din vectorul A conform ordinii elementelor definite de vectorul B. Elementele care nu sunt prezente în vectorul B, dar sunt prezente în vectorul A vor fi plasate la sfârșitul vectorului sortat. Vectorul B poate conține elemente ce nu sunt prezente în vectorul A.*
Exemplu:

Date de intrare	***Date de ieșire***
15 4 5 8 9 3 5 7 1 3 4 9 3 5 1 8 4 3 5 7 2	3 3 3 5 5 5 7 1 1 4 4 8 8 9 9

5. *Elaborați un program C/C++ care va permite citirea unui vector A cu numere întregi cu scopul de a sorta elementele acestora (crescător sau descrescător), apoi va determina perechile de numere din vector, dacă acestea există ce pot forma în sumă un număr întreg X și va stabili câte perechi posibile pot exista.*
Exemplu:

Date de intrare	***Date de ieșire***
7 10 1 9 8 3 7 2 5	1 2 3 5 7 8 9 Numărul de perechi este 3: (9,1) (8,2) (7,3)

6. *Elaborați un program C/C++ care va permite citirea unui vector A cu numere întregi pozitive cu scopul de a găsi cel mai mare număr posibil de format dintr-o mulțime de numere propuse.*
Exemplu:

Date de intrare	***Date de ieșire***
6 12 21 75 10 68 7	77568211210

7. *Elaborați un program C/C++ care va permite citirea unui vector A cu numere întregi pozitive din intervalul [0, 9], cu scopul de a găsi cele două numere cu suma maximă folosind toate elementele vectorului A. Diferența dintre numărul de cifre ale numerelor formate poate fi ±1.*
Exemplu:

Date de intrare	***Date de ieșire***
6 9 2 5 6 0 4	952 640

8. *Elaborați un program C/C++ care va permite citirea unei matrici M formată din 2 linii și N coloane cu numere întregi pozitive, reprezentând perechile de activități, indicând astfel ora de început și ora de sfârșit. Scopul este de a găsi numărul maxim de activități ce pot fi realizate de o persoană cu condiția că poate realiza doar o singură activitate într-un anumit interval de timp.*
Exemplu:

Date de intrare	***Date de ieșire***
2 11 1 3 0 5 3 5 6 8 8 2 12 4 5 6 7 8 9 10 11 12 13 14	Vor fi realizate maximum 4: 1 5 8 12 4 7 11 14

9. *Elaborați un program C/C++ care va permite citirea unei matrici M formată din N linii și 3 coloane cu numere întregi pozitive, reprezentând tripletul unei activități, indicând astfel numărul activității, timp maxim acordar și profitul obținut după realizare. Scopul este de a găsi activități ce pot fi realizate de o persoană cu condiția că poate realiza doar o singură activitate într-un anumit interval de timp, obținând la final un profit maxim posibil.*
Exemplu:

Date de intrare	***Date de ieșire***
10 3 1 9 15 2 2 2 3 5 18	Activitatile prevazute sunt: 7 6 9 5 3 4 8 1 Profitul total obtinut: 109

4 7 1 5 4 25 6 2 20 7 5 8 8 7 10 9 4 12 10 3 5	

10. *Elaborați un program C/C++ care va permite citirea unei matrici M formată din N linii și 3 coloane cu numere întregi pozitive. Având în vedere un set de cutii 3D, dreptunghiulare (Lungime, Lățime și Înnălțime). Creați un teanc de cutii cât mai înalt posibil. O cutie poate fi plasat deasupra unei alte cutii numai dacă dimensiunile bazei 2D este strict mai mare decât cele din baza 2D a cutiei superioare. Multiple cazuri pot fi utilizate ale aceleiași cutii, astfel încât o cutie să poată fi rotită pentru a utiliza oricare dintre aceste laturile ca bază.*
Exemplu:

Date de intrare	***Date de ieșire***
4 3 4 2 5 3 1 6 3 2 1 6 3 8	Inaltimea maxima posibila a turnului construit din aceste cutii este 22: 3 1 6 4 2 5 6 3 8 8 6 3

Explicație:

Metode de rotație a cutiilor sunt:

- *{4, 2, 5}, {5, 4, 2}, {5, 2, 4};*
- *{3, 1, 6}, {6, 3, 1}, {6, 1, 3};*
- *{3, 2, 1}, {3, 1, 2}, {2, 1, 3};*
- *{6, 3, 8}, {8, 6, 3}, {8, 3, 6};*

11. *Elaborați un program C/C++ care va sorta ascendent și descendent elementele unui array bidimensional M[N][N] sub formă șerpuită aplicând tipul de date pointer. Implementați metodele studiate.*

Cazul 1A

Date de intrare			Date de ieșire		
4	6	5	1	2	3
8	2	1	6	5	4
7	3	9	7	8	9

Date de intrare			Date de ieșire		
4	6	5	9	8	7
8	2	1	4	5	6
7	3	9	3	2	1

Cazul 1B

Date de intrare			Date de ieșire		
8	2	1	1	6	7
4	6	5	2	5	8
7	3	9	3	4	9

Date de intrare			Date de ieșire		
8	2	1	9	4	3
4	6	5	8	5	2
7	3	9	7	6	1

Cazul 2A

Date de intrare			Date de ieșire		
7	3	9	7	8	9
4	6	5	6	5	4
1	2	8	1	2	3

Date de intrare			Date de ieșire		
7	3	9	3	2	1
4	6	5	4	5	6
1	2	8	9	8	7

Cazul 2B

Date de intrare			Date de ieșire		
1	2	9	3	4	9
5	6	7	2	5	8
8	3	4	1	6	7

Date de intrare			Date de ieșire		
1	2	9	7	6	1
5	6	7	8	5	2
8	3	4	9	4	3

*** Suplimentar implementaţi cazurile 3A, 3B, 4A şi 4B. Mult succes!**

12. *Elaborați un program C/C++ care va sorta ascendent și descendent elementele unui array bidimensional M[K][K] sub formă de spirală aplicând tipul de date pointer. Implementați metodele studiate.*

Cazul 1A

Date de intrare			Date de ieșire		
4	6	5	1	2	3
8	2	1	8	9	4
7	3	9	7	6	5

Date de intrare			Date de ieșire		
4	6	5	9	8	7
8	2	1	2	1	6
7	3	9	3	4	5

Cazul 1B

Date de intrare			Date de ieșire		
8	2	1	1	8	7
4	6	5	2	9	6
7	3	9	3	4	5

Date de intrare			Date de ieșire		
8	2	1	9	2	3
4	6	5	8	1	4
7	3	9	7	6	5

Cazul 2A

Date de intrare			Date de ieșire		
7	3	9	7	6	5
4	6	5	8	9	4
1	2	8	1	2	3

Date de intrare			Date de ieșire		
7	3	9	3	4	5
4	6	5	2	1	6
1	2	8	9	8	7

Cazul 2B

Date de intrare			Date de ieșire		
1	2	9	3	4	5
5	6	7	2	9	6
8	3	4	1	8	7

Date de intrare			Date de ieșire		
1	2	9	7	6	5
5	6	7	8	1	4
8	3	4	9	2	3

*** Suplimentar implementaţi cazurile 3A, 3B, 4A şi 4B. Mult succes!**

13. *Elaborați un program C/C++ care va sorta ascendent și descendent elementele unui array bidimensional M[X][X] sub formă diagonală aplicând tipul de date pointer. Implementați metodele studiate.*

Cazul 1A

Date de intrare			Date de ieșire		
4	6	5	1	2	6
8	2	1	3	5	7
7	3	9	4	8	9

Date de intrare			Date de ieșire		
4	6	5	9	8	4
8	2	1	7	5	3
7	3	9	6	2	1

Cazul 1B

Date de intrare			Date de ieșire		
8	2	1	1	3	4
4	6	5	2	5	8
7	3	9	6	7	9

Date de intrare			Date de ieșire		
8	2	1	9	7	6
4	6	5	8	5	2
7	3	9	4	3	1

Cazul 2A

Date de intrare			Date de ieșire		
7	3	9	4	8	9
4	6	5	3	5	7
1	2	8	1	2	6

Date de intrare			Date de ieșire		
7	3	9	6	2	1
4	6	5	7	5	3
1	2	8	9	8	4

Cazul 2B

Date de intrare			Date de ieșire		
1	2	9	6	7	9
5	6	7	2	5	8
8	3	4	1	3	4

Date de intrare			Date de ieșire		
1	2	9	4	3	1
5	6	7	8	5	2
8	3	4	9	7	6

*** Suplimentar implementaţi cazurile 3A, 3B, 4A şi 4B. Mult succes!**

Implementați tipul pointer și fișier pentru metodele de sortare *Bubble Sort* și *Counting Sort* pentru problemele 14 - 18 propuse:

14. *Un arhitector din Chișinău a reușit să rezolve o problemă de la primărie. El a stabilit că în oraș nu există clădiri cu înălțimea mai mare decât H. Primarul cere de la arhitector afișarea clădirilor în ordine ascendentă (sau descendentă), precum și o verificare: pentru fiecare clădire din lista ordonată, dacă înălțimea ei este egală cu media aritmetică a înălțimilor celor două clădiri vecine. Arhitectorul cere ajutorul ca să-și termine proiectul care a devenit puțin complicat de realizat, deoarece pentru 40% din teste n ≤ 50 000, pentru 80% din teste n ≤ 500 000 și se consideră că înaintea primei clădiri și după ultima clădire se află câte o pseudoclădire de înălțime 0 – care va interveni în verificarea cerută.*
Exemplu:

Date de intrare	Date de ieșire
10 5 10 10 9 5 2 5 8 5 2	10 10 9 8 5 5 5 5 2 2 0 0 1 0 0 1 1 0 0 0

Explicație:
- *Șirul devine 10 10 9 8 5 5 5 5 2 2*
- *Doar 9 si cei doi de 5 din mijloc respectă condiția.*

15. *Un profesor de algebră a găsit un șir cu N numere naturale, numerotate de la 1 la N și un număr K. Dorind să cerceteze o ipoteză, profesorul vă cere să-l ajutați efectuând următoarea sarcină: deteminați câți divizori are numărul din șir aflat pe poziția p și afișați în ordine descendentă (sau ascendentă) numerele din șir care au același număr de divizori ca cel aflat pe poziția p.*
Exemplu:

Date de intrare	Date de ieșire
10 3 1 5 95 23 16 39 77 74 97 57	4 divizori 95 77 74 57 39

16. *Un profesor de geometrie își propune să verifice o cerință a unui elev. Se consideră N intervale de numere întregi* $[A_i, B_i]$, *unde* $i \in [1, N]$. *Determinați numărul maxim de intervale care se suprapun (au cel puțin*

o valoare comună). Se cunoaște că: $n \in [1,100000]$, *că* $A_i < B_i$ *și că valorile intervalelor* $A_i, B_i \in [-1000000,1000000]$. *Aranjați descendent aceste intervale conform lungimii lor.*
Exemplu:

Date de intrare	Date de ieșire
5 1 4 0 3 8 12 6 9 5 11	3 intervale 5 11 8 12 6 9

17. *Elaborați un program C/C++ care va sorta descendent (sau ascendent) un șir de cuvinte dintr-o anumită categorie conform numărului total de vocale ale cuvântului. Dacă există cuvinte ce au același număr de vocale, acestea se vor aranja în ordine alfabetică.*
Exemplu:

Date de intrare	Date de ieșire
9 ghiocei branduse viorele primula narcise zambile lalele frezie anemone	viorele ghiocei anemone zambile primula narcise lalele frezie branduse

18. *Elaborați un program C/C++ care va citi un șir de cuvinte dintr-o anumită categorie și va afișa pentru fiecare cuvânt de câte ori se conține diftongul ascendent „ea”, aceste date se vor salva într-un array unidimensional. Determinați numărul cel mai mic și cel mai mare ce poate fi format din elementele array-ului obținut, cu condiția că un număr nu poate începe cu cifra 0 (zero).*
Exemplu:

Date de intrare	Date de ieșire
7 gandirea memoria imaginatia atentia invatarea creativitatea motivatia	1 0 0 0 1 2 0 maxim: 2110000 minim: 1000012

Capitolul 4

APLICAŢII CU SUBPROGRAME

4.1 Aplicaţii fundamentale şi recursive

- ✔ *Unele probleme rezolvate vor fi însoțite de comentarii succinte.*
- ✔ *În acest compartiment vom aplica în mod implicit crearea subprogramelor elementare și recursive;*
- ✔ *Verificați secvențele de cod C/C++ ale problemelor rezolvate, utilizați IDE-ul Code::Blocks.*

Problema 4.1 - 01

*Elaborați un program C/C++ care va aplica subprograme. Vom efectua operațiile standard utilizând asupra a două numere întregi citite dintr-un fișier extern (**Varianta 1**).*

Implementarea în C/C++

```
#include <iostream>
#include <fstream>
using namespace std;
ifstream fin("date.txt");
// subprogramele create
int suma(int a,int b){ return a+b; }
int diferenta(int a,int b){ return a-b; }
int catul(int a,int b){ return a/b; }
int produsul(int a,int b){ return a*b; }
int restul(int a,int b){ return a%b; }
// programul principal unde se apeleaza subprogramele
int main(){
    int a,b;
    fin>>a>>b; // citim datele din fisier
    cout<<a<<" + "<<b<<" = "<<suma(a,b)<<endl;
```

```
    cout<<a<<" - "<<b<<" = "<<diferenta(a,b)<<endl;
    cout<<a<<" / "<<b<<" = "<<catul(a,b)<<endl;
    cout<<a<<" * "<<b<<" = "<<produsul(a,b)<<endl;
    cout<<a<<" % "<<b<<" = "<<restul(a,b)<<endl;
}
```

Problema 4.1 - 02

*Elaborați un program C/C++ care va aplica subprograme. Vom efectua operațiile standard utilizând asupra a două numere întregi citite dintr-un fișier extern (**Varianta 2**).*

Implementarea în C/C++

```
#include <iostream>
#include <fstream>
using namespace std;
ifstream fin("date.txt");
int a,b;
// subprogramele create
int suma(){ return a+b; }
int diferenta(){ return a-b; }
int catul(){ return a/b; }
int produsul(){ return a*b; }
int restul(){ return a%b; }
// programul principal unde se apeleaza subprogramele
int main(){
    fin>>a>>b; // citim datele din fisier
    cout<<a<<" + "<<b<<" = "<<suma()<<endl;
    cout<<a<<" - "<<b<<" = "<<diferenta()<<endl;
    cout<<a<<" / "<<b<<" = "<<catul()<<endl;
    cout<<a<<" * "<<b<<" = "<<produsul()<<endl;
    cout<<a<<" % "<<b<<" = "<<restul()<<endl;
}
```

Problema 4.1 - 03

*Elaborați un program C/C++ care va aplica subprograme. Vom efectua operațiile standard utilizând asupra a două numere întregi citite dintr-un fișier extern (**Varianta 3**).*

Implementarea în C/C++

```
#include <iostream>
#include <fstream>
using namespace std;
ifstream fin("date.txt");
int a,b;
// subprogramele create
void suma(int a,int b){ cout<<a+b; }
void diferenta(int a,int b){ cout<<a-b; }
void catul(int a,int b){ cout<<a/b; }
void produsul(int a,int b){ cout<<a*b; }
void restul(int a,int b){ cout<<a%b; }
// programul principal unde se apeleaza subprogramele
int main(){
    fin>>a>>b; // citim datele din fisier
    cout<<a<<" + "<<b<<" = "; suma(a,b); cout<<endl;
    cout<<a<<" - "<<b<<" = "; diferenta(a,b); cout<<endl;
    cout<<a<<" / "<<b<<" = "; catul(a,b); cout<<endl;
    cout<<a<<" * "<<b<<" = "; produsul(a,b); cout<<endl;
    cout<<a<<" % "<<b<<" = "; restul(a,b); cout<<endl;
}
```

Problema 4.1 - 04

Elaborați un program C/C++ care va aplica subprograme. Vom efectua următoarea sumă aplicând funcții recursive în baza relației de recurență.

Suma	Relaţia de recurenţă
$S_n=\frac{1}{1}+\frac{1}{2}+\frac{1}{3}+\frac{1}{4}+...+\frac{1}{n}.$	$S_n=S_{n-1}+\frac{1}{n}$ și $S_0=0.$

Implementarea în C/C++

```
#include <iostream>
#include <fstream>
using namespace std;
// declaram forma termenului sumei
double a(int n){
    return 1.0/n;
```

```
}
double sumaIterativa(int n){
    double suma=0;
    for(int i=1;i<=n;i++) suma+=a(i);
    return suma;
}
double sumaRecursiva(int n){
    if (n==0) return 0;
    return sumaRecursiva(n-1) + a(n);
}
// programul principal unde se apeleaza subprogramele
int main(){
    int n=4; cout.precision(10);
    cout<<"Numarul de termeni ai sumei: "<<n<<endl;
    cout<<"\tSuma iterativa = "<<sumaIterativa(n)<<endl;
    cout<<"\tSuma recursiva = "<<sumaRecursiva(n)<<endl;
}
```

Problema 4.1 - 05

Elaborați un program C/C++ care va aplica subprograme. Vom efectua următorul produs aplicând funcții recursive în baza relației de recurență.

Suma	Relaţia de recurenţă
$P_n=(\frac{1}{1})*(\frac{1}{2})*(\frac{1}{3})*(\frac{1}{4})*...*(\frac{1}{n}).$	$P_n=P_{n-1}*(\frac{1}{n})$ şi $P_0=1.$

Implementarea în C/C++

```
#include <iostream>
#include <fstream>
using namespace std;
// declaram forma factorului produsului
double a(int n){
    return 1.0/n;
}
double produsIterativ(int n){
    double produs=1;
    for(int i=1;i<=n;i++) produs*=a(i);
    return produs;
```

```
}
double produsRecursiv(int n){
    if (n==0) return 1;
    return produsRecursiv(n-1) * a(n);
}
// programul principal unde se apeleaza subprogramele
int main(){
    int n=4; cout.precision(10);
    cout<<"Numarul de termeni ai produsului: "<<n<<endl;
    cout<<"\tProdus iterativ = "<<produsIterativ(n)<<endl;
    cout<<"\tProdus recursiv = "<<produsRecursiv(n)<<endl;
}
```

Problema 4.1 - 06

Elaborați un program C/C++ care va aplica subprograme. Vom efectua suma și produsul primelor N numere naturale, aplicând funcții recursive în baza relației de recurență.

Operaţia asupra şirului	Relaţia de recurenţă
$S_n=1+2+3+4+...+n.$	$S_n=S_{n-1}+n$ şi $S_0=0.$
$P_n=1*2*3*4*...*n.$	$P_n=P_{n-1}*n$ şi $P_0=1.$

Implementarea în C/C++

```
#include <iostream>
#include <fstream>
using namespace std;
int sumaRecursiv(int n){
    if (n==0) return 0;
    return sumaRecursiv(n-1)+n;
}
int produsRecursiv(int n){
    if (n==0) return 1;
    return produsRecursiv(n-1)*n;
}
// programul principal unde se apeleaza subprogramele
int main(){
    int n=4;
    cout<<"\tSuma recursiva a primilor "<<n<<" termeni: \t";
```

```
    cout<<sumaRecursiv(n)<<endl;
    cout<<"\tProdusul recursiv a primilor "<<n<<" factori: ";
    cout<<produsRecursiv(n)<<endl;
}
```

Problema 4.1 - 07

Elaborați un program C/C++ care va aplica subprograme. Vom afișa primele N sume și primele N produse ale seriei, aplicând funcții recursive în baza relației de recurență.

Operația asupra șirului	Relația de recurență
$S_n=1+2+3+4+...+n.$	$S_n=S_{n-1}+n$ și $S_0=0.$
$P_n=1*2*3*4*...*n.$	$P_n=P_{n-1}*n$ și $P_0=1.$

Implementarea în C/C++

```
#include <iostream>
#include <fstream>
using namespace std;
int sumaRecursiv(int n){
    if (n==0) return 0;
    return sumaRecursiv(n-1)+n;
}
int produsRecursiv(int n){
    if (n==0) return 1;
    return produsRecursiv(n-1)*n;
}
// programul principal unde se apeleaza subprogramele
int main(){
    int n=7;
    cout<<"Primele "<<n<<" sume: \t";
    for(int i=1;i<=n;i++){
        cout<<sumaRecursiv(i)<<" ";
    }
    cout<<"\nPrimele "<<n<<" produse: \t";
    for(int i=1;i<=n;i++){
        cout<<produsRecursiv(i)<<" ";
    }
}
```

4.2 Aplicaţii cu array-uri şi string-uri

- ✔ *Unele probleme rezolvate vor fi însoțite de comentarii succinte.*
- ✔ *În acest compartiment vom aplica în mod implicit crearea subprogramelor pentru array-uri unidimensionale și bidimensionale și string-uri;*
- ✔ *Verificați secvențele de cod C/C++ ale problemelor rezolvate, utilizați IDE-ul Code::Blocks.*

Problema 4.2 - 01

Elaborați un program C/C++ care va aplica subprograme. Vom crea subprograme cu ajutorul cărora vom afișa elementele: pare, impare și prime dintr-un array 1D.

Implementarea în C/C++

```
#include <iostream>
using namespace std;
int i,n,a[100];
// functia de citire a datelor de la tastatura
int intputArray1D(){
    for (i=0;i<n;i++){
        cout<<"a["<<i<<"]= "; cin>>a[i];
    }
}
// functia de afisare a datelor de la tastatura citite
int outputArray1D(){
    for (i=0;i<n;i++){
        cout<<" "<<a[i];
    }
}
// functia de afisare a elementelor pare din array-ul 1D
int pareArray1D(){
    for (i=0;i<n;i++){
        if (a[i]%2==0) cout<<" "<<a[i];
    }
}
// functia de afisare a elementelor impare din array-ul 1D
int impareArray1D(){
```

```
    for (i=0;i<n;i++){
        if (a[i]%2!=0) cout<<" "<<a[i];
    }
}
// functia recursiva de verificare a unui numar prim
bool prim(int div,int n){
    if(n==2 || n==1) return true;
    if(n%div==0 || n==1) return false;
    if(div*div>n) return true;
    if(div==2) return prim(3,n);
    return prim(div+2,n);
}
// functia de afisare a elementelor prime din array-ul 1D
int primeArray1D(){
    for (i=0;i<n;i++){
        if (prim(2,a[i])== true) cout<<" "<<a[i];
    }
}
// programul principal unde se apeleaza subprogramele
int main(){
    cout<<"Introducem dimensiunea lui A: ";
    cin>>n;
    cout<<"Introducem cele "<<n<<" elemente:\n";
    intputArray1D();
    getchar(); system("CLS"); // curatirea ecranului
    cout<<"Afisam la ecran elemente introduse:\n";
    outputArray1D();
    cout<<"\nAfisam la ecran elementele pare:\n";
    pareArray1D();
    cout<<"\nAfisam la ecran elementele impare:\n";
    impareArray1D();
    cout<<"\nAfisam la ecran elementele prime:\n";
    primeArray1D();
}
```

Problema 4.2 - 02

Elaborați un program C/C++ care va aplica subprograme. În acest modul vom crea subprograme cu ajutorul cărora vom afișa elementele: pare pozitive, impare negative și prime dintr-un array 2D de pe fiecare linie a acestuia.

Implementarea în C/C++

```
#include <iostream>
using namespace std;
int i,j,n,m,a[100][200];
// functia de citire a datelor de la tastatura
int intputArray2D(){
    for (i=0;i<n;i++){
        for (j=0;j<m;j++){
            cout<<"a["<<i<<"]["<<j<<"]= "; cin>>a[i][j];
        }
    }
}
// functia de afisare a datelor de la tastatura citite
int outputArray2D(){
    for (i=0;i<n;i++){
        for (j=0;j<m;j++){
            cout<<" "<<a[i][j];
        }
        cout<<endl;
    }
}
// functia de afisare a elementelor pare din array-ul 2D
int parePozArray2D(){
    for (i=0;i<n;i++){
        for (j=0;j<m;j++){
            if (a[i][j]%2==0 && a[i][j]>0) cout<<" "<<a[i][j];
        }
        cout<<endl;
    }
}
// functia de afisare a elementelor impare din array-ul 2D
int impareNegArray2D(){
    for (i=0;i<n;i++){
        for (j=0;j<m;j++){
            if (a[i][j]%2!=0 && a[i][j]<0) cout<<" "<<a[i][j];
        }
        cout<<endl;
    }
}
// functia recursiva de verificare a unui numar prim
bool prim(int div,int n){
```

```
    if(n==2 || n==1) return true;
    if(n%div==0 || n==1) return false;
    if(div*div>n) return true;
    if(div==2) return prim(3,n);
    return prim(div+2,n);
}
// functia de afisare a elementelor prime din array-ul 1D
int primeArray2D(){
    for (i=0;i<n;i++){
        for (j=0;j<m;j++){
            if (prim(2,a[i][j])== true) cout<<" "<<a[i][j];
        }
        cout<<endl;
    }
}
// programul principal unde se apeleaza subprogramele
int main(){
    cout<<"Introducem dimensiunile lui A: ";
    cin>>n>>m;
    cout<<"Introducem cele "<<n<<" elemente:\n";
    intputArray2D();
    getchar(); system("CLS");
    cout<<"Afisam la ecran elemente introduse:\n";
    outputArray2D();
    cout<<"\nAfisam la ecran elementele pare pozitive:\n";
    parePozArray2D();
    cout<<"\nAfisam la ecran elementele impare negative:\n";
    impareNegArray2D();
    cout<<"\nAfisam la ecran elementele prime:\n";
    primeArray2D();
}
```

Problema 4.2 – 03 ***

Elaborați un program C/C++ care va aplica subprograme. În acest modul vom crea subprograme cu ajutorul cărora vom afișa dintr-un enunț: numărul de cifre, numărul de vocale, numărul de consoane și numărul de spații. Mesajul va fi citit dintr-un fișier extern.

Implementarea în C/C++
`#include <iostream>`

```
#include <cstring>
#include <fstream>
using namespace std;
ifstream fin("mesaj.txt");
string mes;
int i,v=0,c=0,d=0,s=0;
// citirea si afisarea datelor din fisierul extern
int inputMesaj(){
    while (!fin.eof()){
        getline(fin,mes);
    }
}
// numarul de vocale, consoane, cifre si spatii
int VCNS(){
    for (i=0;i<mes.length();i++){
        if(mes[i]=='a' || mes[i]=='e' || mes[i]=='i' ||
           mes[i]=='o' || mes[i]=='u' || mes[i]=='A' ||
           mes[i]=='E' || mes[i]=='I' || mes[i]=='O' ||
           mes[i]=='U'){
            v++;
        }
          else if((mes[i]>='a'&& mes[i]<='z') || (mes[i]>='A'&&
mes[i]<='Z')){
            c++;
        }
        else if(mes[i]>='0' && mes[i]<='9'){
            d++;
        }
        else if (mes[i]==' ') {
            s++;
        }
    }
    cout<<"\nNumarul de vocale din mesaj: \t"<<v;
    cout<<"\nNumarul de consoane din mesaj: \t"<<c;
    cout<<"\nNumarul de cifre din mesaj: \t"<<d;
    cout<<"\nNumarul de spatii din mesaj: \t"<<s;
}
// programul principal unde se apeleaza subprogramele
int main(){
    cout<<"Afisam mesajul citit din fisierul extern:\n";
    inputMesaj(); cout<<mes<<endl;
    VCNS();  //afisam rezultatele conform functiei create
```

```
}
```

Problema 4.2 – 04 ***

Elaborați un program C/C++ care va aplica subprograme. Vom crea subprograme cu ajutorul cărora vom converti un mesaj implicit în cod ASCII și de asemenea conversia din cod ASCII.

Implementarea în C/C++

```
#include <iostream>
#include <cstring>
using namespace std;
string Text1="You are the best!";
string Text2="8911111732971141013211610410132981011151633";
// functia de conversie in cod ASCII
void convertToASCII(string s){
    for (int i=0; i<s.length(); i++){
        cout<<(int)s[i];
    }
}
// functia de conversie din cod ASCII
void convertFromASCII(string s){
    int num=0;
    for (int i=0; i<s.length(); i++) {
        num=num*10+(s[i]-'0');
        if (num>=32 && num<=122){
            char ch=(char)num; cout<<ch; num = 0;
        }
    }
}
// programul principal unde se apeleaza subprogramele
int main(){
    cout<<"Mesajul initial: \n\t"; cout<<Text1;
    cout<<"\nMesajul convertit in cod ASCII: \n\t";
    convertToASCII(Text1);
    cout<<"\nMesajul convertit din cod ASCII: \n\t";
    convertFromASCII(Text2);
}
```

4.3 Probleme propuse

1. ***Ecuații de gradul II.*** *Elaborați un program C/C++ care va permite afișarea soluției pentru următoarele funcții în același cod de program. Pentru fiecare funcție de gradul II se va utiliza un subprogram corespunzător.*

1	$f(x)=x^2+14x+24;$	3	$h(x)=2x^2-7x+3;$
2	$g(x)=x^2-13x+40;$	4	$i(x)=-3x^2-x+2;$

2. ***Ecuații exponențiale.*** *Elaborați un program C/C++ care va permite afișarea soluției pentru următoarele expresii în același cod de program. Pentru fiecare expresie se va crea un subprogram corespunzător.*

1	$2^{2x^2-3x+0.5}=\frac{\sqrt{2}}{2};$	3	$(0.6)^{5-x}=\left(\frac{5}{3}\right)^{-2};$
2	$7^{x+2}=\left(\frac{1}{7}\right)^2;$	4	$\sqrt{3}=\left(\frac{1}{3}\right)^{-x+2}.$

3. ***Combinatorică.*** *Elaborați un program C/C++ care va permite afișarea soluției pentru următoarele expresii în același cod de program. Pentru fiecare caz elaborăm un subprogram pentru determinarea valorii lui n.*

1	$C_n^4+C_n^3=10A_{n-1}^2;$	3	$3C_n^1+2C_n^2=8;$
2	$C_n^2+A_n^2=30;$	4	$C_n^0+2C_n^1+2C_n^2=36.$

4. ***Ecuații logaritmice.*** *Elaborați un program C/C++ care va permite afișarea soluției pentru următoarele expresii în același cod de program. Pentru fiecare expresie se va crea un subprogram corespunzător.*

1	$\log_2(x-1)=\log_2(2x+4);$	3	$\log_4(x)+\log_x(4)=2;$
2	$lg(x+1)-lg(9)=1-lg(x);$	4	$\log_x(2)+\log_{\sqrt{x}}(2)=9.$

5. ***Ecuații trigonometrice.*** *Elaborați un program C/C++ care va permite afișarea soluției pentru următoarele expresii în același cod de program. Pentru fiecare expresie se va crea un subprogram corespunzător.*

1	$\cos(2x)+\sin(x)=0;$	3	$\sin(x-\frac{\pi}{2})=\sin(3x+\frac{\pi}{4});$
2	$\cos(2x+\frac{\pi}{2})=\cos(x-\frac{\pi}{2});$	4	$tg(x+\frac{\pi}{3})=tg(\frac{\pi}{2}-x), x\in(0,\pi).$

6. ***Ecuații iraționale.*** *Elaborați un program C/C++ care va permite afișarea soluției pentru următoarele expresii în același cod de program. Pentru fiecare expresie se va crea un subprogram corespunzător.*

1	$3\sqrt{x-4}=x-8;$	3	$4(5\sqrt{x}-9)-7(\sqrt{x}-3)=3\sqrt{x};$
2	$3x-\sqrt{2x-1}=34;$	4	$\frac{\sqrt{3x-5}}{2}-\frac{2\sqrt{3x-5}}{6}=\frac{1}{2}+\frac{\sqrt{3x-5}}{12}.$

7. ***Numere complexe.*** *Elaborați un program C/C++ care va permite afișarea soluției pentru următoarele expresii în același cod de program. Pentru fiecare expresie se va crea un subprogram corespunzător. Se va crea o structură cu denumirea Complex. Știind că:* $z_1=3-4i$ *și* $z_2=5+i$ *și* $z_3=-2-i$. *Calculați valorile:*

1	$z_1+2z_2-3z_3;$	4	$2z_1-z_2+\bar{z}_3;$
2	$3z_2+2\bar{z}_2-\bar{z}_3;$	5	$\bar{z}_2-\bar{z}_1+5z_3;$
3	$\frac{z_1}{\bar{z}_2}-\frac{\bar{z}_1}{z_2};$	6	$\frac{\bar{z}_1}{z_2}+\frac{z_1}{\bar{z}_2}.$

8. *Elaborați un program C/C++ care va permite crearea subprogramelor pentru prelucrarea următoarelor operațiuni:*
   a) *afișarea primelor 100 numere: pare, impare, prime, pătrate perfecte, cuburi perfecte, Fibonacci, Fibonacci pătrate perfecte (dacă există așa numere);*
   b) *suma primelor N numere: pare, impare, prime, pătrate perfecte, cuburi perfecte, Fibonacci, Fibonacci pătrate perfecte (dacă există așa numere);*
   c) *produsul primelor N numere: pare, impare, prime, pătrate perfecte, cuburi perfecte, Fibonacci, Fibonacci pătrate perfecte (dacă există așa numere).*
9. *Elaborați un program C/C++ care va permite crearea subprogramelor pentru prelucrarea următoarelor operațiuni:*
   a) *afișarea dintr-un interval [a,b] a tuturor numerelor: pare negative, impare pozitive, prime, pătrate perfecte, cuburi perfecte, Fibonacci, Fibonacci pătrate perfecte (dacă există așa numere);*
   b) *suma dintr-un interval [a,b] a tuturor numerelor: pare, impare, prime, pătrate perfecte, cuburi perfecte, Fibonacci, Fibonacci pătrate perfecte (dacă există așa numere);*
   c) *produsul dintr-un interval [a,b] a tuturor numerelor: pare, impare, prime, pătrate perfecte, cuburi perfecte, Fibonacci, Fibonacci pătrate perfecte (dacă există așa numere).*
10. *Elaborați un program C/C++ care va permite crearea subprogramelor pentru prelucrarea următoarelor operațiuni:*
   a) *generarea primelor N numere de 4 cifre ce au suma cifrelor un pătrat perfect;*
   b) *generarea primelor N numere de 4 cifre ce au suma cifrelor un cub perfect;*
   c) *generarea primelor N numere de 4 cifre ce au suma cifrelor un număr prim de maxim 3 cifre.*
11. *Elaborați un program C/C++ care va permite crearea subprogramelor pentru prelucrarea următoarelor operațiuni asupra unui array unidimensional de numere întregi:*
   a) *afișarea elementelor pare negative de pe poziții impare;*
   b) *afișarea elementelor impare pozitive de pe poziții pare;*
   c) *afișarea elementelor prime și suma acestora;*
   d) *afișarea elementelor prime din seria Fibonacci.*

12. *Elaborați un program C/C++ care va permite crearea subprogramelor pentru prelucrarea următoarelor operațiuni asupra unui array unidimensional de caractere:*
    a) *afișarea într-un fișier a numărului total de vocale, precum și numărul pentru fiecare vocală în parte;*
    b) *afișarea într-un fișier a numărului total de consoane, precum și numărul pentru fiecare consoană în parte;*
    c) *afișarea pentru vocalele mari a codului său ASCII;*
    d) *afișarea pentru vocalele micii și a cifrelor, caracterul (simbolul) imediat următor conform tabelului ASCII.*
13. *Elaborați un program C/C++ care va permite crearea subprogramelor pentru prelucrarea următoarelor operațiuni asupra unui mesaj textual citit dintr-un fișier extern:*
    a) *afișarea în același fișier a mesajului fără cuvintele ce conțin mai puțin de 3 caractere;*
    b) *afișarea în același fișier a tuturor cuvintelor ce conțin mai mult de 3 caractere și încep cu o consoană;*
    c) *afișarea în același fișier a tuturor cuvintelor ce conțin cuvinte care încep cu o consoană și se termină tot cu o consoană;*
    d) *afișarea în același fișier a tuturor cuvintelor ce conțin exact 3 vocale una după alta.*
14. *Elaborați un program C/C++ care va permite crearea subprogramelor pentru prelucrarea următoarelor operațiuni asupra unui unui array bidimensional de cuvinte citite de la tastatură:*
    a) *afișarea în ordine ascendentă a elementelor array-ului bidimensional pe fiecare rând (aplicăm SelectionSort);*
    b) *afișarea în ordine descendentă a elementelor array-ului bidimensional pe fiecare coloană (aplicăm BubbleSort);*
    c) *afișarea în ordine ascendentă a elementelor array-ului bidimensional sub formă de spirală (aplicăm InsertionSort);*
    d) *afișarea în ordine descendentă a elementelor array-ului bidimensional sub formă de diagonală (alegem metoda);*
    e) *afișarea în ordine ascendentă a elementelor de pe diagonala principală a array-ului bidimensional (alegem metoda);*
    f) *afișarea în ordine ascendentă a elementelor de pe diagonala secundară a array-ului bidimensional (alegem metoda).*

Capitolul 5

APLICAȚII CU STRUCTURI, UNIUNI ŞI ENUMERĂRI

5.1 Aplicaţii ale structurilor standard şi imbricate

- ✔ *Unele probleme rezolvate vor fi însoțite de comentarii succinte.*
- ✔ *În acest compartiment vom aplica în mod implicit tipurile de sortări clasice asupra elementelor array-urilor.*
- ✔ *Verificați secvențele de cod C/C++ ale problemelor rezolvate, utilizați IDE-ul Code::Blocks.*

Problema 5.1 - 01

Elaborați un program C/C++ care va permite descompunerea în factori primi a unui număr întreg pozitiv, memorând rezultatul sub forma unui array de structuri. Fiecare element al structurii va conține două câmpuri:

- *unul care conține divizorul prim;*
- *unul care conține puterea la care apare acest număr în descompunere.*

Implementarea C++

```
#include<iostream>
using namespace std;
// declaram datele structurii factor
struct factor{
    long d;
    int p;
} x[20];
long m; int n, i;
```

```
// programul principal
int main(){
    cout<<"Introduceti numarul: "; cin>>m;
    long a=2;
    while( m != 1){
        //puterea la care apare a in descompunerea numarului m
        int q=0;
        while( m%a == 0){
            m = m/a; q++;
        }
        if( q != 0 ){
            n++; x[n].d = a; x[n].p = q;
        }
        if( a == 2 ) a = 3;
        else a+=2;
    }
    // afisarea rezultatului
    cout<<"Descompunerea in factori primi: \n";
    for(i=1; i<=n; i++)
    cout<<"\t"<<x[i].d<<"^"<<x[i].p<<endl;
}
```

Problema 5.1 - 02

Elaborați un program C/C++ care va permite crearea unei structuri cu denumirea Elev. Se consideră o listă cu elevii care au susținut teza la matematică. Pentru fiecare elev se cunosc: numele, prenumele și nota obținută. Se cere să se ordoneze elevii promovați descrescător după nota obținută, de asemenea să se calculeze media aritmetică a notelor obținute.

Implementarea C++

```
# include <iostream>
using namespace std;
// declaram datele structurii elev
struct elev {
    char nume[10], prenume[20];
    int nota;
} a[20], c;
int n, i, j, S;
// programul principal
```

```
int main(){
    cout<<"Introduceti nr. de elevi: "; cin>>n;
    cout<<"Datele despre elev: "<<endl;
    for(i=1;i<=n;i++){
        cout<<"Elevul "<<i<<": "<<endl;
        cout<<"\tNume: "; cin>>a[i].nume;
        cout<<"\tPrenume: "; cin>>a[i].prenume;
        cout<<"\tNota: "; cin>>a[i].nota;
        // calculam suma totala a notelor elevilor
        S+=a[i].nota;
    }
    /// Sortarea prin metoda selecției
    for(i=1;i<n;i++)
    for(j=i+1;j<=n;j++)
    if(a[i].nota<a[j].nota){
        c=a[i]; // variabila c este de tip struct
        a[i]=a[j]; a[j]=c;
    }
    // afisarea elevilor promovati
    cout<<"Afisarea elevilor promovati: "<<endl;
    for(i=1;i<=n&&a[i].nota>=5;i++)
     cout<<"\t"<<a[i].nume<<" "<<a[i].prenume<<" "<<a[i].nota<<"
"<<endl;
    // afisarea mediei generale a elevilor
    cout<<"Media notelor este: "<<S*1.0/n<<endl;
}
```

Problema 5.1 - 03

Elaborați un program C/C++ care va permite crearea unei structuri imbricate cu denumirea Profesor. Se citesc datele despre profesorii unei instituții, respectiv numele, prenumele şi data naşterii. Să se ordoneze profesorii în ordinea crescătoare după data naşterii şi să se afişeze rezultatul.

Implementarea C++

```
#include <iostream>
using namespace std;
// declaram structura data
struct data{
```

```
    int zi,ln,an;
};
// declaram structura imbricata profesor
struct profesor{
    char nume[20],prenume[20]; data dn;
}e[30],x;
int n,i,j;
// programul principal
int main(){
    cout<<"Nr. de profesori: "; cin>>n;
    for(i=1;i<=n;i++){
        cin.get(); cout<<"Profesorul "<<i<<endl;
        cout<<"\tNume: ";cin.get(e[i].nume,20);cin.get();
        cout<<"\tPrenume: ";cin.get(e[i].prenume,20);cin.get();
        cout<<"\tZi: ";cin>>e[i].dn.zi;
        cout<<"\tLuna: ";cin>>e[i].dn.ln;
        cout<<"\tAn: ";cin>>e[i].dn.an;
    }
    /// Sortarea prin metoda selecției
    for(i=1;i<=n-1;i++)
    for(j=i+1;j<=n;j++)
    if(e[i].dn.an>e[j].dn.an){
        x=e[i];e[i]=e[j];e[j]=x;
    }
    else if(e[i].dn.an==e[j].dn.an)
    if(e[i].dn.ln>e[j].dn.ln){
        x=e[i];e[i]=e[j];e[j]=x;
    }
    else if(e[i].dn.ln==e[j].dn.ln)
    if(e[i].dn.zi>e[j].dn.zi){
        x=e[i];e[i]=e[j];e[j]=x;
    }
    // afisarea in ordine ascendenta a profesorilor
    cout<<"Afisarea ascendenta dupa data nasterii: \n";
    for(i=1;i<=n;i++){
        cout<<"\t"<<e[i].nume<<"\t"<<e[i].prenume;
                    cout<<"\t"<<e[i].dn.zi<<"\t"<<e[i].dn.ln<<"\
t"<<e[i].dn.an;
    }
}
```

Problema 5.1 - 04

Elaborați un program C/C++ care va permite crearea unei uniuni cu denumirea Data. Să se prezinte unele operații asupra datelor de tip uniune.

Implementarea C++

```
#include <iostream>
#include <cstring>
using namespace std;
// declaram uniunea data
union data{
    char ch[10];
    short s;
    long l;
    float f;
    double d;
} a;
// programaul principal
int main(){
    strcpy(a.ch,"INFORMATICA");
    cout<<"\nDimensiunea zonei de memorie rezervata este ;
    cout<<"de "<<sizeof(a)<<" octeti.\n";
    // afisarea rezultatelor conform conditiei
    cout<<"\nCONTINUTUL ZONEI: ";
    cout<<"\n\tSir de caractere: \t\t"<<a.ch;
    cout<<"\n\tNumar intreg de tipul short: \t"<<a.s;
    cout<<"\n\tNumar intreg de tipul long: \t"<<a.l;
    cout<<"\n\tNumar real de tipul float: \t"<<a.f;
    cout<<"\n\tNumar real de tipul double: \t"<<a.d;
}
```

Problema 5.1 - 05

Elaborați un program C/C++ care va permite crearea unei uniuni imbricate cu denumirea Caracter. Să se afişeze reprezentarea internă a unei date de tip caracter folosind reprezentarea pe biți.

Implementarea C++

```
#include <iostream>
```

```
using namespace std;
// declaram structura biti
struct biti{
    unsigned b0:1, b1:1, b2:1, b3:1, b4:1, b5:1, b6:1, b7:1;
};
// declaram uniunea imbricata caracter
union caracter{
    biti b;
    char c;
} octet;
// programul principal
int main(){
    cout<<"Introduceti un caracter: \t"; cin>>octet.c;
    cout<<"Reprezentarea interna: \t\t";
    cout<<octet.b.b7<<octet.b.b6<<octet.b.b5<<octet.b.b4;
    cout<<octet.b.b3<<octet.b.b2<<octet.b.b1<<octet.b.b0;
    cout<<endl;
}
```

Problema 5.1 - 06

Elaborați un program C/C++ care va permite crearea unei enumerări cu titlul Steag. Să presupunem că proiectați un buton pentru aplicația Windows. Puteți seta steaguri pentru text: ITALICS, BOLD și UNDERLINE. În sistemul binar avem următoarele date: ITALIC=00 000 001, BOLD=00 000 010 și UNDERLINE=00 000 100. Spre exemplu: dacă vom utiliza în text BOLD și UNDERLINE, se va afișa valoarea 5. Argumentați rezultatul pentru fiecare caz prezentat.

Implementarea C++

```
#include <iostream>
using namespace std;
// enumerarea steag
enum steag {
    BOLD = 1, ITALICS = 2, UNDERLINE = 4
};
// programul principal
int main(){
    // cazul 1 de design textual
```

```
    int Design1 = BOLD | UNDERLINE;
    cout<<"\nSteagul 1 de stare va afisa valoarea: "<<Design1;
    // cazul 2 de design textual
    int Design2 = BOLD | ITALICS;
    cout<<"\nSteagul 2 de stare va afisa valoarea: "<<Design2;
    return 0;
}
```

Problema 5.1 - 07

Elaborați un program C/C++ care va permite crearea unei enumerări cu denumirea Zile. Să se prezinte unele operații asupra datelor de tip enumerare.

Implementarea C++

```
#include <iostream>
using namespace std;
// declaram enumerarea zile
enum zile{
     zero,unu,doi,trei,patru,cinci,sase
} x,y;
// programul principal
int main(){
    int z,w;
    // partea 1 in baza datelor enumerate
    x=doi; /* x=2 */  y=trei; /*x=3*/
    z=x+y; w=x*y;
    cout<<"\nNumarul z= "<<z<<" si Numarul w= "<<w;
    // partea 2 in baza pozitiei datelor enumerate
    x==2;y==3;
    /* o astfel de atribuire nu este intocmai indicata; ar trebui folosite numele sugestive din cadrul enum-ului*/
    z=x+y; w=x*y;
    cout<<"\nNumarul z= "<<z<<" si Numarul w= "<<w<<endl;
}
```

Problema 5.1 - 08

Elaborați un program C/C++ care va permite crearea unei enumerări cu denumirea Luna, afișând poziția lunilor pare (impare).

Implementarea C++

```
#include <iostream>
using namespace std;
//enumerarea Luna
enum Luna {
    Ianuarie,Februarie,Martie,Aprilie,Mai,Iunie,
    Iulie,August,Septembrie,Octombrie,Noiembrie,Decembrie
};
// programul principal
int main(){
    int x; // pozitia elementelor in enum Luna
    cout<<"\nAfisam pozitia lunilor pare din an: \n\t";
    for (x=Ianuarie; x<=Decembrie; x++){
        if (x%2==0) cout <<x<<" ";
    }
    cout<<"\nAfisam pozitia lunilor impare din an: \n\t";
    for (x=Ianuarie; x<=Decembrie; x++){
        if (x%2!=0) cout <<x<<" ";
    }
}
```

Problema 5.1 - 09

Elaborați un program C/C++ care va permite crearea unei enumerări cu denumirea Profil, se va utiliza un switch pentru a verifica profilul unui elev și va afișa un mesaj cu indicele profilului și denumirea lui corespunzătoare.

Implementarea C++

```
#include <iostream>
using namespace std;
int main(){
    // Definim enumerarea Profil
    enum Profil { U, R, A, S };
    char p; // declaram variabila p de tip caracter
    cout<<"Indicati profilul: "; cin>>p;
    switch (p){
        case 'u':
            cout<<"A fost ales profilul "<<Profil::U;
            cout<<" = Uman "<<endl; break;
```

```
        case 'r':
            cout<<"A fost ales profilul "<<Profil::R;
            cout<<" = Real "<<endl; break;
        case 'a':
            cout<<"A fost ales profilul "<<Profil::A;
            cout<<" = Arte "<<endl; break;
        case 's':
            cout<<"A fost ales profilul "<<Profil::S;
            cout<<" = Sport "<<endl; break;
        default:
            cout<<"Va rugam sa introduceti una din optiuni:\n";
            cout<<"u - Uman, r - Real, a - Arte, s - Sport.\n";
    }
}
```

Problema 5.1 - 10

Elaborați un program C/C++ care va permite crearea unei enumerări cu denumirea Zi, se va afișa dacă o zi este lucrătoare sau este de odihnă.

Implementarea C++

```
#include <iostream>
using namespace std;
// declararea enumerarii
enum{
      Luni, Marti, Miercuri, Joi, Vineri, Sambata, Duminica
} zi;
// programul principal
int main() {
    zi = Joi; // declaram valoarea implicita
    cout<<"A fost introdusa implicit ziua "<<zi<<endl;
    if(zi == Sambata || zi == Duminica){
    cout<<"Este o zi de weekend."<<endl;
    }
    else if(zi == Miercuri){
    cout<<"Este o zi de la mijlocul saptamanii."<<endl;
    }
    else cout<<"Este o zi obisnuita de lucru."<<endl;
}
```

Problema 5.1 – 11

Elaborați un program C/C++ care va aplica subprograme. Vom crea subprograme cu ajutorul cărora vom studia unele operațiuni asupra stucturii profesor. Se propune realizarea subprogramelor pentru a afișa în ordine ascendentă după vârstă și descendentă după salariu. Rezultatele execuției se vor păstra în fișierul extern ***profesori.txt***.

Implementarea în C/C++

```
#include <iostream>
#include <cstring>
#include <fstream>
using namespace std;
int n,i,g;
ofstream fout("profesori.txt");
// structura profesor
struct profesor{
    char n[15],p[15]; // nume si prenume
    int v; // varsta
    float s; // salariul
}a[25],aux;
// functia de citire a datelor de la tastatura
void citire(){
    for(i=1; i<=n; i++){
        cout<<"Profesorul "<<i<<endl;
        cout<<"\tNumele: "; cin>>a[i].n;
        cout<<"\tPreumele: "; cin>>a[i].p;
        cout<<"\tVarsta: "; cin>>a[i].v;
        cout<<"\tSalariul: "; cin>>a[i].s;
    }
};
// functia de afisare a datelor citite de la tastatura
void afisare(){
    for(i=1; i<=n; i++){
    fout<<"\t"<<a[i].n<<" "<<a[i].p<<" "<<a[i].v<<" "<<a[i].s;
    fout<<endl;
    }
};
// functia de ordonare ascendenta dupa varsta
void ordonareVarsta(){
    do{
        g=1;
```

```
        for(i=1; i<n; i++)
                    if((a[i+1].v<a[i].v)||(a[i+1].v==a[i].v   &&
strcmp(a[i+1].n,a[i].n)<0)){
            aux=a[i+1]; a[i+1]=a[i]; a[i]=aux; g=0;
        }
    } while(!g);
};
// functia de ordonare descendenta dupa salariu
void ordonareSalariu(){
    do{
        g=1;
        for(i=1; i<n; i++)
                    if((a[i+1].s>a[i].s)||(a[i+1].s==a[i].s   &&
strcmp(a[i+1].n,a[i].n)<0)){
            aux=a[i+1]; a[i+1]=a[i]; a[i]=aux; g=0;
        }
    } while(!g);
};
// programul principal unde se apeleaza subprogramele
int main(){
   cout<<"Introducem numarul de profesori: "; cin>>n;
   citire(); getchar(); system("CLS");
   // pastram rezultatele in fisierul extern
   fout<<"Afisarea datelor despre profesori: \n";
   afisare();
   fout<<"Profesorii ordonati ascendent dupa varsta: "<<endl;
   ordonareVarsta(); afisare();
   fout<<"Profesorii ordonati descendent dupa salariu: "<<endl;
   ordonareSalariu(); afisare();
}
```

5.2 Aplicaţii cu pointeri şi fişiere

- ✔ *Unele probleme rezolvate vor fi însoțite de comentarii succinte.*
- ✔ *În acest compartiment vom aplica în mod implicit tipurile de sortări clasice asupra elementelor array-urilor.*
- ✔ *Verificați secvențele de cod C/C++ ale problemelor rezolvate, utilizați IDE-ul Code::Blocks.*

Problema 5.2 - 01

Elaborați un program C/C++ care va permite declararea unei înregistrări imbricate despre un angajat al unei companii și afișarea datelor acestuia.

Implementarea C++

```
#include <iostream>
using namespace std;
/* structurile declarate, in principiu vor fi declarate in
afara oricarei functii, pentru a putea fi utilizata in intreg
programul */
struct data{
    int an,luna,zi;
};
struct angajat{
       int cod_angajat;
       char *nume,*prenume;
       data data_angajarii;
       float sal_incadrare;
} x;
// programul principal
int main(){
    // Initializarea datelor unui angajat
    x.cod_angajat=1021;
    x.nume="Popescu"; x.prenume="Ioan";
    x.data_angajarii.zi=10; x.data_angajarii.luna=9;
    x.data_angajarii.an=2002; x.sal_incadrare=1250;
    // afisarea datelor initializate
    cout<<"Angajat: \t\t"<<x.nume<<" "<<x.prenume<<endl;
```

```
   cout<<"Data angajarii: \t"<<x.data_angajarii.zi<<".";
   cout<<x.data_angajarii.luna<<"."<<x.data_angajarii.an<<endl;
   cout<<"Salariu incadrare: \t"<<x.sal_incadrare<<endl;
}
```

Problema 5.2 – 02

Elaborați un program C/C++ care va permite crearea unei structuri imbricate cu denumirea Complex. Se citesc datele despre două numere complexe, efectuați între aceste două numere următoarele operații: adunarea, scăderea, produsul și câtul.

Implementarea C++

```
#include <iostream>
using namespace std;
struct Complex{
 float re, im;
} a,b,c;
int aduna(Complex *a, Complex *b, Complex *c){
 /* transmiterea parametrilor prin pointeri */
    c->re=a->re+b->re; c->im=a->im+b->im;
}
int scade(Complex a, Complex b, Complex *c){
 /* transmiterea parametrilor de intrare prin valoare si a rezul-
tatului prin pointer */
    c->re=a.re-b.re; c->im=a.im-b.im;
}
int produs(Complex *a, Complex *b, Complex *c){
/*transmiterea parametrilor prin pointeri */
    c->re=a->re*b->re-a->im*b->im; c->im=a->im*b->re+a->re*b->im;
}
int impartire(Complex *a, Complex *b, Complex *c){
/*transmiterea parametrilor prin pointeri */
    float x;
    x=b->re*b->re+b->im*b->im;
    if (x==0) {
        cout<<"\nImpartire la zero!\n";
        exit(1);
    }
    else{
        c->re=(a->re*b->re+a->im*b->im)/x;
        c->im=(a->im*b->re-a->re*b->im)/x;
    }
```

```
}
 int main(){
    char ch='D';
    while ((ch=='D')|| (ch=='d')){
       cout<<"\nIntroduceti coeficientii numarului complex C1: \n";
       cin>>a.re>>a.im;
       cout<<"\nIntroduceti coeficientii numarului complex C2: \n";
       cin>>b.re>>b.im;
       // operatii cu numere complexe si rezultatele obtinute
       aduna(&a,&b,&c);
                              cout<<"\n("<<a.re<<"+"<<a.im<<"*i)    +
("<<b.re<<"+"<<b.im<<"*i) = ("<<c.re<<"+"<<c.im<<"*i)";
        scade(a,b,&c);
                              cout<<"\n("<<a.re<<"+"<<a.im<<"*i)    -
("<<b.re<<"+"<<b.im<<"*i) = ("<<c.re<<"+"<<c.im<<"*i)";
        produs(&a,&b,&c);
                              cout<<"\n("<<a.re<<"+"<<a.im<<"*i)    *
("<<b.re<<"+"<<b.im<<"*i) = ("<<c.re<<"+"<<c.im<<"*i)";
        impartire(&a,&b,&c);
                              cout<<"\n("<<a.re<<"+"<<a.im<<"*i)    /
("<<b.re<<"+"<<b.im<<"*i) = ("<<c.re<<"+"<<c.im<<"*i)";
        cout<<"\n\nCONTINUATI?DA=D/d,Nu=alt caracter ";
        cin>>ch;
    }
 }
```

Problema 5.2 - 03

Elaborați un program C/C++ care va permite crearea unei aplicații pentru lucrul cu numere raționale (valori numerice ce pot fi scrise sub formă de fracție) și anume: citirea, afișarea, suma, diferența, câtul și produsul a două numere raționale.

Implementarea C++

```
#include <iostream>
#include <cmath>
using namespace std;
struct rational{
       int a,b;
       // functia pentru determinarea CMMDC a doua numere
       int cmmdc(int x,int y)       {
              if(x==y) return x;
```

```
                else if(x>y) return cmmdc(x-y,y);
                        else return cmmdc(x,y-x);
        }
        // functia care ne va simplifica fractia
        void ireductibil()    {
                if(b<0){
                        a=-a; b=-b;
                }
                if(!a) b=1;
                else if(abs(a)!=1 && abs(b)!=1){
                    int d=cmmdc(abs(a),abs(b));
                    a=a/d; b=b/d;
                 }
        }
};
// functia de citire a numerelor rationale
void citire(rational* x, char c){
        cout<<"Introduceti numarul rational: "<<c<<endl;
        cout<<"\tNumaratorul: "; cin>>(x->a);
        int n;
        do{
                cout<<"\tNumitorul: "; cin>>n;
        }
        while(n==0);
        x->b=n; x->ireductibil();
}
// functia de afisare a datelor
void afisare(rational x, char c[]){
        cout<<"\t"<<c<<" = "<<x.a<<"/"<<x.b<<endl;
}
// functia de adunare a 2 numere rationale
rational adunare(rational x, rational y){
        rational r; r.a=x.a*y.b+x.b*y.a;
        r.b=x.b*y.b; r.ireductibil();
        return r;
}
// functia de scadere a 2 numere rationale
rational scadere(rational x, rational y){
        rational r; r.a=x.a*y.b-x.b*y.a;
        r.b=x.b*y.b; r.ireductibil();
        return r;
}
```

```
// functia de inmultire a 2 numere rationale
rational inmultire(rational x, rational y){
	rational r; r.a=x.a*y.a;
	r.b=x.b*y.b; r.ireductibil();
	return r;
}
// functia de impartire a 2 numere rationale
rational impartire(rational x, rational y){
	rational r; r.a=x.a*y.b;
	r.b=x.b*y.a; r.ireductibil();
	return r;
}
// programul principal
int main() {
	rational x,y,sum,dif,p,cat;
	citire(&x,'x'); citire(&y,'y');
	// afisam numerele rationale si operatiile
	cout<<"Numerele rationale sunt: "<<endl;
	afisare(x,"x"); afisare(y,"y");
	cout<<"Operatii cu numere rationale: "<<endl;
	sum=adunare(x,y); afisare(sum,"\tx+y");
	dif=scadere(x,y); afisare(dif,"\tx-y");
	p=inmultire(x,y); afisare(p,"\tx*y");
	cat=impartire(x,y); afisare(cat,"\tx/y");
}
```

Problema 5.2 - 04

Elaborați un program C/C++ care va permite administrarea unui parc de automobile. Informațiile relative la un automobil sunt: numărul de locuri, puterea (în cai putere), marca, culoarea, anul fabricației automobilului. Se vor păstra datele finale într-un fișier.

a) *se vor citi informatiile relative la cele n automobile și se vor afișa doar acele automobilele care au 5 locuri;*
b) *se vor ordona crescător după putere automobilele;*
c) *se va scrie o funcție care afișează toate automobilele fabricate într-un anumit an dat ca parametru.*

Implementarea C++
`#include <iostream>`

```
#include <fstream>
using namespace std;
// structura automobil
struct automobil{
        char marca[20],model[20],culoare[20];
        int an;
        struct date{
                char nl;
                int put;
        }d;
}a[20];
ofstream fout("automobile.txt"); // fișierul extern
// functia de citire
void citire(automobil a[], int *n){
        int i;
        cout<<"Inroduceti numarul de automobile: ";
        cin>>*n;
        for(i=0; i<*n; i++){
                cout<<"Introduceti   datele   despre   automobilul
"<<i+1<<endl;
                cout<<"\tMarca: "; cin>>a[i].marca;
                cout<<"\tModelul: "; cin>>a[i].model;
                cout<<"\tCuloarea: "; cin>>a[i].culoare;
                cout<<"\tAnul: "; cin>>a[i].an;
                cout<<"\tNumarul de locuri: "; cin>>&a[i].d.nl;
                cout<<"\tPuterea      (cai      putere):      ";
cin>>a[i].d.put;
        }
}
// functia de afisare
void afisare(automobil a[],int n){
    int i;
    for (i=0;i<n;i++){
                fout<<"Automobilul ["<<i+1<<"]:\n";
                fout<<"\tMarca: "<<a[i].marca<<endl;
                fout<<"\tModelul: "<<a[i].model<<endl;
                fout<<"\tCuloarea: "<<a[i].culoare<<endl;
                fout<<"\tAn fabr.: "<<a[i].an<<endl;
                fout<<"\tNr locuri: "<<a[i].d.nl<<endl;
                fout<<"\tPuterea: "<<a[i].d.put<<endl<<endl;
         }
}
```

```
// functia de verificare a parametrului anul
void an(automobil a[],int n,int p){
        int i;
        for(i=0;i<n-1;i++){
         if(a[i].an==p){
                        fout<<"Automobilul ["<<i+1<<"]:\n";
                }
        }
}
// programul principal
int main(){
        int n,aux,i,j,p;
        citire(a,&n);
        afisare(a,n);
        fout<<"\nSolutia conform conditiei a 1-a:"<<endl<<endl;
        for(i=0;i<n;i++){
            if(a[i].d.nl==5){
               fout<<"Automobilul ["<<i+1<<"]:\n";
            }
        }
        fout<<"\nSolutia conform conditiei a 2-a:"<<endl<<endl;
        for(i=0;i<n-1;i++)
        for(j=i+1;j<n;j++){
            if(a[i].d.put>a[j].d.put){
               aux=i; i=j; j=aux;
            }
         }
        fout<<"Automobilul ["<<i+1<<"]:\n";
        fout<<"\nSolutia conform conditiei a 3-a:"<<endl<<endl;
        cout<<"\n\tIntroduceti anul dupa care vor fi afisate
masinile: ";
        cin>>p; an(a,n,p);
}
```

Problema 5.2 - 05

Elaborați un program C/C++ care va permite crearea unei structuri cu denumirea Complex. Se citesc datele despre un număr complex, reprezentați acest număr sub forma sa algebrică și trigonometrică. Se vor aplica formulele matematice corespunzătoare.

Implementarea C++

```
#include <iostream>
#include <cmath>
#include <fstream>
using namespace std;
ofstream fout("rezultat.txt"); // fisierul extern
struct Complex{
    double x,y;
};
//reprezentarea algebrica
Complex initAlg(double x, double y){
    Complex z={x,y};
    return z;
}
//reprezentarea trigonometrica
Complex initTrig(double ro, double fi){
    return initAlg(ro*cos(fi),ro*sin(fi));
}
void scrie(Complex z){
    if(z.y==0) { fout<<z.x; return;}
    if(z.x==0 && z.y!=0){ fout<<z.y<<"i"; return;}
    if(z.x!=0 && z.y>0) { fout<<z.x<<"+"<<z.y<<"i"; return;}
    if(z.x!=0 && z.y<0) { fout<<z.x<<z.y<<"i"; return;}
    return;
}
Complex conj(Complex u){
    return initAlg(u.x,-u.y);
}
Complex realToComplex(double x){
    return initAlg(x,0);
}
const double PI = 3.141592653589; // constanta pi
int main(void){
    Complex z1,z2,z3;
    cout<<"Verificati rezultatul in fisierul extern!"<<endl;
    z1=initAlg(2,-3);
    fout<<"\nForma algebrica: \tz1 = "; scrie(z1);
    z2=initTrig(1,PI/3);
    fout<<"\nForma trigonometrica: \tz2 = "; scrie(z2);
    return 0;
}
```

Problema 5.2 - 06

Elaborați un program C/C++ care va permite crearea unei uniuni Student. Prezentați un exemplu pentru a înțelege diferența dintre accesarea valorilor unor membri de tip uniune. Rezultatele se vor păstra într-un fișier extern.

Implementarea C++

```
#include <iostream>
#include <cstring>
#include <fstream>
using namespace std;
// structura uniunii student
union student{
    char nume[20];
    char curs[20];
    float randament;
};
ofstream fout("Student.txt"); // fisierul de iesire
//programul principal
int main(){
    union student date1;
    union student date2;
   // atribuirea valorilor pentru variabila de tip union, date1
       strcpy(date1.nume, "Alexei");
       strcpy(date1.curs, "Matematica");
       date1.randament = 86.50;
       fout<<"\nPrezentarea datelor studentului 1:\n";
       fout<<" Nume       : "<<date1.nume<<endl;
       fout<<" Curs       : "<<date1.curs<<endl;
       fout<<" Randament  : "<<date1.randament<<endl;
   // atribuirea valorilor pentru variabila de tip union, date2
       fout<<"\nPrezentarea datelor studentului 2:\n";
       strcpy(date2.nume, "Malia");
       fout<<" Nume       : "<<date2.nume<<endl;
       strcpy(date2.curs, "Informatica");
       fout<<" Curs       : "<<date2.curs<<endl;
       date2.randament = 99.50;
       fout<<" Prandament : "<<date2.randament<<endl;
}
```

5.3 Probleme propuse

1. *Elaborați un program C/C++ care va permite aplicarea tipului structură pentru o dată curentă: an, lună, zi. Scrieți un program pentru a afișa a câta zi din an este ziua respectivă și câte zile au mai rămas până la sfârșitul anului.*
2. *Elaborați un program C/C++ care va permite aplicarea tipului structură pentru data voastră de naștere. Știind că în anul curent vă aniversați ziua de naștere în ziua de x [luni, marți, ..., duminică], scrieți un program pentru va afișa ziua (din săptămână) în care v-ați născut.*
3. *Elaborați un program C/C++ care va permite introducerea tipului "rațional" ca o structură formată din numărător și numitor. Scrieți funcții de simplificare, adunare, scădere, înmulțire, împărțire, ridicare la putere pentru implementarea structurii.*
4. *Elaborați un program C/C++ care va permite aplicarea tipului uniune, ce conține câmpurile necesare pentru a putea reprezenta un cerc, un dreptunghi, un pătrat, un triunghi echilateral. Scrieți doar o singură funcție pentru a calcula aria figurilor respective.*
5. *Elaborați un program C/C++ care va permite citirea unui șir de caractere format din litere și cifre. Să se indice frecvența de apariție a caracterelor întâlnite în șir folosind un tablou de elemente structură (câmpurile structurii sunt: caracterul și frecvența de apariție).*
6. *Elaborați un program C/C++ care va permite determina o posibilă încărcare a unui camion cu anumite cantități din materiale astfel încât valoarea totală să fie maximă. Un camion poate fi încărcat cu maxim m kilograme. Numele materialelor disponibile, cantitățile exprimate în kilograme și prețul per kilogram sunt cunoscute.*
7. *Elaborați un program C/C++ care va memoriza eficient un polinom rar de grad m și funcții pentru afișarea lui și găsirea valorii lui într-un anumit punct. Un polinom rar este un polinom în care majoritatea monoamelor sunt zero.*
8. *Elaborați un program C/C++ care va memoriza eficient o matrice rară și funcții pentru calculul sumei, diferenței și produsului de perechi de matrici rare. O matrice rară este un array bidimensional în care majoritatea elementelor sunt zero.*

*** Pentru problemele 1-4 aplicaţi: subprograme, pointeri, fişiere şi sortări.**

1. Studenții dintr-o facultate vor să facă un top al celor mai bune 10 piese muzicale pentru fiecare săptămană a lunii mai. Ajutați-i să realizeze un program prin care se introduc N date. Afișați primele 10 titluri, în ordinea descrescătoare a punctelor săptămânale.

```
struct piese{
    char titlul[20];
    char interpret[20];
    int scor_zilnic[7];
    int scor_saptamanal;
    long buget;
} a[100];
```

2. Un dealer auto are N automobile propuse spre vânzare pe parcursul primului trimestru al anului. Ajutaţi-l să realizeze un program prin care se introduc N date. Afișați primele 10 modele de automobile care care s-au vândut cu un preț mai mare de 10.000.000 de euro, în ordinea crescătoare a anului fabricației.

```
struct automobil{
    char marca[20];
    char model[20];
    int cap_motor;
    int an_fabricatie;
    double km_parcursi;
    float pret_auto;
} b[100];
```

3. Un profesor de chimie are la dispoziție N informații despre elemente chimice din categoria metalelor grele neferoase. Ajutați-i să realizeze un program prin care se introduc N date. Afișați primele 3 elemente chimice ce au masa atomică relativă mai mare decât 200 unități și sunt sortați ascendent după grupa din sistemul periodic.

```
struct metale{
    char denimire[20];
    char culoare[20];
    int grupa, perioada;
    int densitatea;
    float masa_atomica;
    float raza_atomica;
} c[100];
```

4. Un profesor de biologie are la dispoziție N informații despre viețuitoarele Terrei. Ajutați-l să realizeze un program prin care se introduc N date. Afișați primele 5 cele mai rapide viețuitoase în ordinea alfabetică a denumirii acestora.

```
struct vietuitoare{
   char denumirea[20];
   char familia[20];
   char alimentatie[20];
   double viteza;
   float greutate;
} d[100];
```

5. *Rețeaua de hotele „Jolli Alon" din mun. Chișinău dispune de mai multe restaurante de elită, iar informațiile cu privire la clienți și toate comenzile din restaurante sunt stocate într-o bază de date ce cuprinde 2 fișiere textuale:*

Denumire fișier	**Câmpurile fișierului**
Restaurante.txt	CodRest, Responsabil, Denumire, Reducere, Loc_disponibil.
Client.txt	CodRest, IDNP, Nume, Prenume, Avans, Achitat

Restaurante.txt

CodRest	**Responsabil**	**Denumire**	**Reducere (%)**	**Loc_Disponibil**
0001	Alexei	Evrica	10	120
0002	Eugenia	Pegasus	15	85

Client.txt

CodRest	**IDNP**	**Nume**	**Prenume**	**Avans**	**Achitat**
0001	2008052002527	Crețu	Chiril	300	500
0002	2009012002528	Burduja	Elena	250	600

Elaborați subprograme în limbajul C++ care vor realiza la solicitarea utilizatorului a următoarelor informații:

a) *înregistrează un nou restaurant împreună cu caracteristicile sale corespunzătoare fișierului Restaurante.txt;*
b) *înregistrează un nou client împreună cu caracteristicile sale corespunzătoare fișierului Client.txt;*
c) *afișarea atributelor tuturor restaurandelor unde este reducere de 5% și locuri disponibile sunt mai mult de un număr K introdus de la tastatură;*
d) *afișarea atributelor tuturor clienților care vizitează cel mai renumit restaurant cu condiția ca prețul achitat pentru o comandă să fie mai mic decât un număr S citit de la tastatură.*

6. *Rețeaua de automobile „SmartCars" din mun. Chișinău dispune de mai multe linii de producție a automobilelor inteligente, iar informațiile cu privire la clienți și toate comenzile sunt stocate într-o bază de date ce cuprinde 2 fișiere textuale:*

Denumire fișier	**Câmpurile fișierului**
Car_line.txt	CodLine, Denumire, Model, Reducere, CP, Preț
Client.txt	CodLine, Nume, Prenume, Sex, Credit, RataLunară

Car_line.txt

CodLine	**Denumire**	**Model**	**Reducere (%)**	**CP**	**Preț ($)**
0001	Mclauren	720S	20	721	390 000
0002	Ferrari	SF90	35	1000	700 000

Client.txt

CodLine	**Nume**	**Prenume**	**Sex**	**Credit**	**RataLunară**
0001	Cerbu	Aureliu	M	Da	500
0002	Zohan	Eleonora	F	Nu	0

Elaborați subprograme în limbajul C++ care vor realiza la solicitarea utilizatorului a următoarelor informații:

a) *înregistrează un nou automobil împreună cu caracteristicile sale corespunzătoare fișierului Car_Line.txt;*
b) *înregistrează un nou client împreună cu caracteristicile sale corespunzătoare fișierului Client.txt;*
c) *afișarea atributelor tuturor automobilelor în ordine ascendentă a denumirii sale cu condiția că acestea au o reducere de peste 35 %, ai clienților de sex feminin și nu sunt cumpărate în credit;*
d) *afișarea atributelor tuturor clienților care procură cel mai ieftin automobil cu o rată lunară mai mică decât 5000 și determinați în câte luni ar putea închide creditul.*

Capitolul 6

APLICAȚII CU MODULE

6.1 Aplicații fundamentale

- ✔ *Unele probleme rezolvate vor fi însoțite de comentarii succinte.*
- ✔ *În acest compartiment vom aplica în mod implicit crearea modulelor elementare și modulelor recursive;*
- ✔ *Verificați secvențele de cod C/C++ ale problemelor rezolvate, utilizați IDE-ul Code::Blocks.*

Problema 6.1 - 01

Elaborați un program C/C++ care va aplica un modul cu titlul Operatii. În acest modul vom efectua operațiile standard utilizând asupra a două numere întregi citite dintr-un fișier extern (Varianta 1).

Modul în C++

```
#ifndef OPERATII_H_INCLUDED
#define OPERATII_H_INCLUDED
int suma(int a,int b){ return a+b; }
int diferenta(int a,int b){ return a-b; }
int catul(int a,int b){ return a/b; }
int produsul(int a,int b){ return a*b; }
int restul(int a,int b){ return a%b; }
#endif // OPERATII_H_INCLUDED
```

Implementarea modulului creat în C++

```
#include <iostream>
#include <fstream>
//includem fisierul antet creat
#include "operatii.h"
```

```
using namespace std;
ifstream fin("date.txt");
int main(){
    int a,b;
    fin>>a>>b; // citim datele din fisier
    cout<<a<<" + "<<b<<" = "<<suma(a,b)<<endl;
    cout<<a<<" - "<<b<<" = "<<diferenta(a,b)<<endl;
    cout<<a<<" / "<<b<<" = "<<catul(a,b)<<endl;
    cout<<a<<" * "<<b<<" = "<<produsul(a,b)<<endl;
    cout<<a<<" % "<<b<<" = "<<restul(a,b)<<endl;
}
```

Problema 6.1 - 02

Elaborați un program C/C++ care va aplica un modul cu titlul Operatii. În acest modul vom efectua operațiile standard utilizând asupra a două numere întregi citite dintr-un fișier extern (Varianta 2).

Modul în C++

```
#ifndef OPERATII_H_INCLUDED
#define OPERATII_H_INCLUDED
int a,b;
int suma(){ return a+b; }
int diferenta(){ return a-b; }
int catul(){ return a/b; }
int produsul(){ return a*b; }
int restul(){ return a%b; }
#endif // OPERATII_H_INCLUDED
```

Implementarea modulului creat în C++

```
#include <iostream>
#include <fstream>
//includem fisierul antet creat
#include "operatii.h"
using namespace std;
ifstream fin("date.txt");
int main(){
    fin>>a>>b; // citim datele din fisier
    cout<<a<<" + "<<b<<" = "<<suma()<<endl;
    cout<<a<<" - "<<b<<" = "<<diferenta()<<endl;
```

```
    cout<<a<<" / "<<b<<" = "<<catul()<<endl;
    cout<<a<<" * "<<b<<" = "<<produsul()<<endl;
    cout<<a<<" % "<<b<<" = "<<restul()<<endl;
}
```

Problema 6.1 - 03

Elaborați un program C/C++ care va aplica un modul cu titlul Operatii. În acest modul vom efectua operațiile standard utilizând asupra a două numere întregi citite dintr-un fișier extern (Varianta 3).

Modul în C++

```
#ifndef OPERATII_H_INCLUDED
#define OPERATII_H_INCLUDED
using namespace std;
void suma(int a,int b){ cout<<a+b; }
void diferenta(int a,int b){ cout<<a-b; }
void catul(int a,int b){ cout<<a/b; }
void produsul(int a,int b){ cout<<a*b; }
void restul(int a,int b){ cout<<a%b; }
#endif // OPERATII_H_INCLUDED
```

Implementarea modulului creat în C++

```
#include <iostream>
#include <fstream>
//includem fisierul antet creat
#include "operatii.h"
ifstream fin("date.txt");
int main(){
    int a,b;
    fin>>a>>b; // citim datele din fisier
    cout<<a<<" + "<<b<<" = "; suma(a,b); cout<<endl;
    cout<<a<<" - "<<b<<" = "; diferenta(a,b); cout<<endl;
    cout<<a<<" / "<<b<<" = "; catul(a,b); cout<<endl;
    cout<<a<<" * "<<b<<" = "; produsul(a,b); cout<<endl;
    cout<<a<<" % "<<b<<" = "; restul(a,b); cout<<endl;
}
```

Problema 6.1 - 04

Elaborați un program C/C++ care va aplica un modul cu titlul Serie. În acest modul vom efectua următoarea sumă aplicând funcții recursive în baza relației de recurență.

Suma	Relaţia de recurenţă
$S_n=\frac{1}{1}+\frac{1}{2}+\frac{1}{3}+\frac{1}{4}+...+\frac{1}{n}.$	$S_n=S_{n-1}+\frac{1}{n}$ şi $S_0=0.$

Modul în C++

```
#ifndef SERIE_H_INCLUDED
#define SERIE_H_INCLUDED
// declaram forma termenului sumei
double a(int n){
    return 1.0/n;
}
double sumaIterativa(int n){
    double suma=0;
    for(int i=1;i<=n;i++) suma+=a(i);
    return suma;
}
double sumaRecursiva(int n){
    if (n==0) return 0;
    return sumaRecursiva(n-1) + a(n);
}
#endif // SERIE_H_INCLUDED
```

Implementarea modulului creat în C++

```
#include <iostream>
#include "serie.h"
using namespace std;
int main(){
    int n=4; cout.precision(10);
    cout<<"Numarul de termeni ai sumei: "<<n<<endl;
    cout<<"\tSuma iterativa = "<<sumaIterativa(n)<<endl;
    cout<<"\tSuma recursiva = "<<sumaRecursiva(n)<<endl;
}
```

Problema 6.1 - 05

Elaborați un program C/C++ care va aplica un modul cu titlul Serie. În acest modul vom efectua următorul produs aplicând funcții recursive în baza relației de recurență.

Produsul	Relaţia de recurenţă
$P_n=(\frac{1}{1})*(\frac{1}{2})*(\frac{1}{3})*(\frac{1}{4})*...*(\frac{1}{n}).$	$P_n=P_{n-1}*(\frac{1}{n})$ şi $P_0=1.$

Modul în C++

```
#ifndef SERIE_H_INCLUDED
#define SERIE_H_INCLUDED
// declaram forma factorului produsului
double a(int n){
    return 1.0/n;
}
double produsIterativ(int n){
    double produs=1;
    for(int i=1;i<=n;i++) produs*=a(i);
    return produs;
}
double produsRecursiv(int n){
    if (n==0) return 0;
    return produsRecursiv(n-1) * a(n);
}
#endif // SERIE_H_INCLUDED
```

Implementarea modulului creat în C++

```
#include <iostream>
#include "serie.h"
using namespace std;
int main(){
    int n=4; cout.precision(10);
    cout<<"Numarul de termeni ai produsului: "<<n<<endl;
    cout<<"\tProdus iterativ = "<<produsIterativ(n)<<endl;
    cout<<"\tProdus recursiv = "<<produsRecursiv(n)<<endl;
}
```

Problema 6.1 - 06

Elaborați un program C/C++ care va aplica un modul cu titlul Serie. În acest modul vom efectua suma și produsul primelor N numere naturale, aplicând funcții recursive în baza relației de recurență.

Operaţia asupra şirului	Relaţia de recurenţă
$S_n=1+2+3+4+...+n.$	$S_n=S_{n-1}+n$ şi $S_0=0.$
$P_n=1*2*3*4*...*n.$	$P_n=P_{n-1}*n$ şi $P_0=1.$

Modul în C++

```
#ifndef SERIE_H_INCLUDED
#define SERIE_H_INCLUDED
int sumaRecursiv(int n){
    if (n==0) return 0;
    return sumaRecursiv(n-1)+n;
}
int produsRecursiv(int n){
    if (n==0) return 1;
    return produsRecursiv(n-1)*n;
}
#endif // SERIE_H_INCLUDED
```

Implementarea modulului creat în C++

```
#include <iostream>
#include "serie.h"
using namespace std;
int main(){
    int n=4;
    cout<<"\tSuma recursiva a primilor "<<n<<" termeni: \t";
    cout<<sumaRecursiv(n)<<endl;
    cout<<"\tProdusul recursiv a primilor "<<n<<" factori: ";
    cout<<produsRecursiv(n)<<endl;
}
```

Problema 6.1 - 07

Elaborați un program C/C++ care va aplica un modul cu titlul Serie. În acest modul vom afișa primele N sume și primele N produse ale seriei, aplicând funcții recursive în baza relației de recurență.

Operaţia asupra şirului	Relaţia de recurenţă
$S_n=1+2+3+4+...+n.$	$S_n=S_{n-1}+n$ şi $S_0=0.$
$P_n=1*2*3*4*...*n.$	$P_n=P_{n-1}*n$ şi $P_0=1.$

Modul în C++

```
#ifndef SERIE_H_INCLUDED
#define SERIE_H_INCLUDED
int sumaRecursiv(int n){
    if (n==0) return 0;
    return sumaRecursiv(n-1)+n;
}
int produsRecursiv(int n){
    if (n==0) return 1;
    return produsRecursiv(n-1)*n;
}
#endif // SERIE_H_INCLUDED
```

Implementarea modulului creat în C++

```
#include <iostream>
#include "serie.h"
using namespace std;
int main(){
    int n=7;
    cout<<"Primele "<<n<<" sume: \t";
    for(int i=1;i<=n;i++){
        cout<<sumaRecursiv(i)<<" ";
    }
    cout<<"\nPrimele "<<n<<" produse: \t";
    for(int i=1;i<=n;i++){
        cout<<produsRecursiv(i)<<" ";
    }
}
```

Problema 6.1 - 08

Elaborați un program C/C++ care va aplica un modul cu titlul Array1D. În acest modul vom crea subprograme cu ajutorul cărora vom afișa elementele: pare, impare și prime dintr-un array 1D.

Modul în C++

```
#ifndef ARRAY1D_H_INCLUDED
#define ARRAY1D_H_INCLUDED
#include <iostream>
using namespace std;
int i,n,a[100];
// functia de citire a datelor de la tastatura
int intputArray1D(){
    for (i=0;i<n;i++){
        cout<<"a["<<i<<"]= "; cin>>a[i];
    }
}
// functia de afisare a datelor de la tastatura citite
int outputArray1D(){
    for (i=0;i<n;i++){
        cout<<" "<<a[i];
    }
}
// functia de afisare a elementelor pare din array-ul 1D
int pareArray1D(){
    for (i=0;i<n;i++){
        if (a[i]%2==0) cout<<" "<<a[i];
    }
}
// functia de afisare a elementelor impare din array-ul 1D
int impareArray1D(){
    for (i=0;i<n;i++){
        if (a[i]%2!=0) cout<<" "<<a[i];
    }
}
// functia recursiva de verificare a unui numar prim
bool prim(int div,int n){
    if(n==2 || n==1) return true;
    if(n%div==0 || n==1) return false;
    if(div*div>n) return true;
    if(div==2) return prim(3,n);
```

```
    return prim(div+2,n);
}
// functia de afisare a elementelor prime din array-ul 1D
int primeArray1D(){
    for (i=0;i<n;i++){
        if (prim(2,a[i])== true) cout<<" "<<a[i];
    }
}
#endif // ARRAY1D_H_INCLUDED
```

Implementarea modulului creat în C++

```
#include <iostream>
#include "Array1D.h"
int main(){
    cout<<"Introducem dimensiunea lui A: ";
    cin>>n;
    cout<<"Introducem cele "<<n<<" elemente:\n";
    intputArray1D();
    getchar(); system("CLS"); // curatirea ecranului
    cout<<"Afisam la ecran elemente introduse:\n";
    outputArray1D();
    cout<<"\nAfisam la ecran elementele pare:\n";
    pareArray1D();
    cout<<"\nAfisam la ecran elementele impare:\n";
    impareArray1D();
    cout<<"\nAfisam la ecran elementele prime:\n";
    primeArray1D();
}
```

Problema 6.1 - 09

Elaborați un program C/C++ care va aplica un modul cu titlul Array2D. În acest modul vom crea subprograme cu ajutorul cărora vom afișa elementele: pare pozitive, impare negative și prime dintr-un array 2D de pe fiecare linie a acestuia.

Modul în C++

```
#ifndef ARRAY2D_H_INCLUDED
#define ARRAY2D_H_INCLUDED
#include <iostream>
```

```
using namespace std;
int i,j,n,m,a[100][200];
// functia de citire a datelor de la tastatura
int intputArray2D(){
    for (i=0;i<n;i++){
        for (j=0;j<m;j++){
            cout<<"a["<<i<<"]["<<j<<"]= "; cin>>a[i][j];
        }
    }
}
// functia de afisare a datelor de la tastatura citite
int outputArray2D(){
    for (i=0;i<n;i++){
        for (j=0;j<m;j++){
            cout<<" "<<a[i][j];
        }
        cout<<endl;
    }
}
// functia de afisare a elementelor pare din array-ul 2D
int parePozArray2D(){
    for (i=0;i<n;i++){
        for (j=0;j<m;j++){
            if (a[i][j]%2==0 && a[i][j]>0) cout<<" "<<a[i][j];
        }
        cout<<endl;
    }
}
// functia de afisare a elementelor impare din array-ul 2D
int impareNegArray2D(){
    for (i=0;i<n;i++){
        for (j=0;j<m;j++){
            if (a[i][j]%2!=0 && a[i][j]<0) cout<<" "<<a[i][j];
        }
        cout<<endl;
    }
}
// functia recursiva de verificare a unui numar prim
bool prim(int div,int n){
    if(n==2 || n==1) return true;
    if(n%div==0 || n==1) return false;
    if(div*div>n) return true;
```

```
    if(div==2) return prim(3,n);
    return prim(div+2,n);
}
// functia de afisare a elementelor prime din array-ul 1D
int primeArray2D(){
    for (i=0;i<n;i++){
        for (j=0;j<m;j++){
            if (prim(2,a[i][j])== true) cout<<" "<<a[i][j];
        }
        cout<<endl;
    }
}
#endif // ARRAY2D_H_INCLUDED
```

Implementarea modulului creat în C++

```
#include <iostream>
#include "Array2D.h"
int main(){
    cout<<"Introducem dimensiunile lui A: ";
    cin>>n>>m;
    cout<<"Introducem cele "<<n<<" elemente:\n";
    intputArray2D();
    getchar(); system("CLS");
    cout<<"Afisam la ecran elemente introduse:\n";
    outputArray2D();
    cout<<"\nAfisam la ecran elementele pare pozitive:\n";
    parePozArray2D();
    cout<<"\nAfisam la ecran elementele impare negative:\n";
    impareNegArray2D();
    cout<<"\nAfisam la ecran elementele prime:\n";
    primeArray2D();
}
```

Problema 6.1 - 10

Elaborați un program C/C++ care va aplica un modul cu titlul Profesor. Vom crea subprograme cu ajutorul cărora vom studia unele operațiuni asupra stucturii profesor. Se propune realizarea subprogramelor pentru a afișa în ordine ascendentă după vârstă și descendentă după salariu. Rezultatele execuției se vor păstra în fișierul extern ***profesori.txt****.*

Modul în C++

```
#ifndef PROFESOR_H_INCLUDED
#define PROFESOR_H_INCLUDED
#include <cstring>
#include <fstream>
using namespace std;
int k,i,g;
ofstream fout("profesori.txt");
// structura profesor
struct profesor{
    char n[15],p[15]; // nume si prenume
    int v; // varsta
    float s; // salariul
}a[25],aux;
// functia de citire a datelor de la tastatura
void citire(){
    for(i=1; i<=k; i++){
        cout<<"Profesorul "<<i<<endl;
        cout<<"\tNumele: "; cin>>a[i].n;
        cout<<"\tPreumele: "; cin>>a[i].p;
        cout<<"\tVarsta: "; cin>>a[i].v;
        cout<<"\tSalariul: "; cin>>a[i].s;
    }
};
// functia de afisare a datelor citite de la tastatura
void afisare(){
    for(i=1; i<=k; i++){
    fout<<"\t"<<a[i].n<<" "<<a[i].p<<" "<<a[i].v<<" "<<a[i].s;
    fout<<endl;
    }
};
// functia de ordonare ascendenta dupa varsta
void ordonareVarsta(){
    do{
        g=1;
        for(i=1; i<k; i++)
                    if((a[i+1].v<a[i].v)||(a[i+1].v==a[i].v   &&
strcmp(a[i+1].n,a[i].n)<0)){
            aux=a[i+1]; a[i+1]=a[i]; a[i]=aux; g=0;
        }
    } while(!g);
```

```
};
// functia de ordonare descendenta dupa salariu
void ordonareSalariu(){
    do{
        g=1;
        for(i=1; i<k; i++)
                    if((a[i+1].s>a[i].s)||(a[i+1].s==a[i].s   &&
strcmp(a[i+1].n,a[i].n)<0)){
            aux=a[i+1]; a[i+1]=a[i]; a[i]=aux; g=0;
        }
    } while(!g);
};
#endif // PROFESOR_H_INCLUDED
```

Implementarea modulului creat în C++

```
#include <iostream>
#include "profesor.h"
int main(){
   cout<<"Introducem numarul de profesori: "; cin>>k;
   citire();
   // curatarea ecranului executiei
   getchar(); system("CLS");
   // afisarea rezultatelor in fisierul extern
   fout<<"Afisarea datelor despre profesori: \n";
   afisare();
   fout<<"Profesorii ordonati ascendent dupa varsta: "<<endl;
   ordonareVarsta(); afisare();
   fout<<"Profesorii ordonati descendent dupa salariu: "<<endl;
   ordonareSalariu(); afisare();
}
```

Problema 6.1 – 11 ***

Elaborați un program C/C++ care va aplica un modul cu titlul Mesaj. În acest modul vom crea subprograme cu ajutorul cărora vom afișa dintr-un enunț: numărul de cifre, numărul de vocale, numărul de consoane și numărul de spații. Mesajul va fi citit dintr-un fișier extern.

Modul în C++

```
#ifndef MESAJ_H_INCLUDED
#define MESAJ_H_INCLUDED
#include <cstring>
#include <fstream>
using namespace std;
ifstream fin("mesaj.txt");
string mes;
int i,v=0,c=0,d=0,s=0;
// citirea si afisarea datelor din fisierul extern
int inputMesaj(){
    while (!fin.eof()){
        getline(fin,mes);
    }
}
// numarul de vocale, consoane, cifre si spatii
int VCNS(){
    for (i=0;i<mes.length();i++){
        if(mes[i]=='a' || mes[i]=='e' || mes[i]=='i' ||
           mes[i]=='o' || mes[i]=='u' || mes[i]=='A' ||
           mes[i]=='E' || mes[i]=='I' || mes[i]=='O' ||
           mes[i]=='U'){
            v++;
        }
          else if((mes[i]>='a'&& mes[i]<='z') || (mes[i]>='A'&&
mes[i]<='Z')){
            c++;
        }
        else if(mes[i]>='0' && mes[i]<='9'){
            d++;
        }
        else if (mes[i]==' ') {
            s++;
        }
    }
    cout<<"\nNumarul de vocale din mesaj: \t"<<v;
    cout<<"\nNumarul de consoane din mesaj: \t"<<c;
    cout<<"\nNumarul de cifre din mesaj: \t"<<d;
    cout<<"\nNumarul de spatii din mesaj: \t"<<s;
}
#endif // MESAJ_H_INCLUDED
```

Implementarea modulului creat în C++

```
#include <iostream>
#include "mesaj.h"
// programul principal
int main(){
    cout<<"Afisam mesajul citit din fisierul extern:\n";
    inputMesaj(); cout<<mes<<endl;
    VCNS();  //afisam rezultatele conform functiei create
}
```

Problema 6.1 – 12 ***

Elaborați un program C/C++ care va aplica un modul cu titlul Mesaj. Vom crea subprograme cu ajutorul cărora vom converti un mesaj implicit în cod ASCII și conversia din cod ASCII a mesajului.

Modul în C++

```
#ifndef MESAJ_H_INCLUDED
#define MESAJ_H_INCLUDED
#include <cstring>
using namespace std;
// functia de conversie in cod ASCII
void convertToASCII(string s){
    for (int i=0; i<s.length(); i++){
        cout<<(int)s[i];
    }
}
// functia de conversie din cod ASCII
void convertFromASCII(string s){
    int num=0;
    for (int i=0; i<s.length(); i++) {
        num=num*10+(s[i]-'0');
        if (num>=32 && num<=122){
            char ch=(char)num; cout<<ch; num = 0;
        }
    }
}
#endif // MESAJ_H_INCLUDED
```

Implementarea modulului creat în C++

```
#include <iostream>
#include "mesaj.h"
// programul principal
int main(){
    // mesajele text de intrare
    string Text1 = "Matematica";
    string Text2 = "779711610110997116105997";
    cout<<"Mesajul initial: \n\t"; cout<<Text1;
    cout<<"\nMesajul convertit in cod ASCII: \n\t";
    convertToASCII(Text1);
    cout<<"\nMesajul convertit din cod ASCII: \n\t";
    convertFromASCII(Text2);
    cout<<"Mesajul initial: \n\t"; cout<<Text1;
    cout<<"\nMesajul convertit in cod ASCII: \n\t";
    convertToASCII(Text1);
    cout<<"\nMesajul convertit din cod ASCII: \n\t";
    convertFromASCII(Text2);
}
```

6.2 Probleme propuse

1. *Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi create subprograme pentru a efectua unele operații cu două fracții: adunarea, scăderea, înmulțirea și împărțirea acestora. Condiția este ca rezultatul să fie o fracție ireductibilă.*
2. *Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi create subprograme pentru a determina numărul maxim și cel minim ce poate fi obținut prin eliminarea unei cifre dintr-un număr n ce conține cel mult 7 cifre.*
3. *Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi prezentate cel puțin 5 metode (subprograme) pentru determinarea elementului maxim (sau minim) dintre două numere citite dintr-un fișier extern. Aceste numere nu pot fi comparate, nu se permite aplicarea funcțiilor implicite ale limbajului C++ cum ar fi Min (a,b) sau Max(a,b), de asemenea instrucțiunile de decizie nu pot fi aplicate. Se permite doar utilizarea formulelor și a expresiilor logico-matematice.*
4. *Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi prezentate cel puțin 5 metode (subprograme) pentru rezolvarea ecuației: 5x+3y=n, unde n,x,y sunt numere naturale, iar n>7. De exemplu: pentru n=8, soluția va fi perechea de numere (1,1).*
5. *Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi create subprograme pentru a determina apartenența unui punct de coordonate carteziene (x,y) la un poligon de n laturi. Se cunoaște că punctul poate fi situat: în interior, în exterior sau pe una din laturile poligonului. Este important de specificat că un poligon poate avea minim 3 laturi, iar pentru condiția noastră vom aplica o limită de n=10.*
6. *Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi create subprograme pentru a determina coordonatele carteziene ale puctului lipsă ale unui poligon regulat de 3-10 laturi (se vor crea subprograme pentru fiecare caz în dependență de numărul de laturi). De exemplu: pentru un*

poligon neregulat, dreptunghi, cu coordonatele punctelor: (5,1), (1,1) și (1,7), atunci se va afișa (5,7).

7. *Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi create subprograme pentru a determina numerele prime mai mici sau egale cu n prin următoarea metodă: se substituie cu 0 numere divizibile cu 2, apoi cele divizibile cu 3, apoi cele divizibile cu 5 etc. Se cunoaște că pentru început avem un număr natural n.*
8. *Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi create subprograme pentru a determina perechile de numere gemene, dacă se cunoaște că acestea trebuie să fie prime, iar diferența dintre ele să fie 2. De exemplu: perechile (3,5) și (5,7) sunt numere gemene.*
9. *Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi create subprograme pentru a determina toate tripletele ascendente și descendente aplicând un array unidimensional de numere naturale.*
10. *Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi create subprograme pentru a determina reuniunea, intersecția și diferența a două mulțimi A și B, reprezentate sub forma unui array unidimensional de numere întregi.*
11. *Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi create subprograme pentru a determina distanța minimă și cea maximă într-o mulțime de puncte din sistemul cartezian de coordonale. Afișați aceste distanțe și determinți coordonatele acestor puncte ce determină distanțele respective (aplicați această problemă și pentru cazul în care o mulțime de puncte este situată în spațiun 3D în loc de 2D).*
12. *Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi create subprograme pentru a determina detreminantul unei matrici pătratice prin cel puțin 3 metode diferite.*
13. *Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi create subprograme pentru a scrie un număr natural cu cifre romane (I,V,X,L,C,D,M) și apoi dintr-un număr cu cifre romane să se scrie numărul cu cifre arabe. De exemplu: 1609 va fi MDCIX și MCMLXXI va fi 1971.*

14. *Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi create subprograme pentru a modela jocul „Vrei să fii milionar?” pentru minim 5 domenii se vor crea 15 întrebări diferite. Întrebările pot conține doar un singur răspuns corect și cel puțin 4 varinate de răspuns.*
15. *Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi create subprograme pentru a verifica dacă un mesaj citit dintr-un fișier exter reprezintă o expresi ematematică. De exemplu: pentru „a*b+c-x ” se va afișa „Corect”, iar pentru „a*b+c-+x ” se va afișa „Incorect”.*
16. *Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi create subprograme pentru a efectua operații cu două polinoame cu o singură necunoscută: adunarea, scăderea, înmulțirea și împărțirea (se vor afișa separat partea întreagă și restul de la împărțire). Se va aplica o structuriă de date corespunzătoare.*
17. *Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi create subprograme pentru a sorta ascendent (sau descendent) elementele unui array unidimensional de elemente întregi citite dintr-un fișier extern Array.txt. În fișierul extern Sortare.txt se vor păstra rezultatele obținute în urma implementării algoritmilor BubbleSort, SelectionSort și InsertionSort pentru datele citite din fișier. Se va converti fiecare număr în codul său ASCII și se va rescrie în fișierul inițial de unde s-a citit, elementele negative se vor reprezenta ca modul. De exemplu: numărul 65 se va converti în A, iar –65 tot va fi A.*
18. *Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi create subprograme pentru a arhiva și a dezarhiva fișiere text. Fișierul arhivat va trebui să ocupe mai puțină memorie decât fișierele nearhivate.*
19. *Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi create subprograme pentru a reprezenta un număr par, mai mare decât 6 ca sumă de două, trei, patru și cinci numere prime. Se va afișa cel puțin o modalitate.*
20. *Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi create subprograme recursive pentru a determina CMMDC și CMMMC a doua, trei și patru numere naturale citite dintr-un fișier extern.*

Capitolul 7

STUDIUL INDIVIDUAL GHIDAT DE PROFESOR

7.1 Lucrarea nr.1 – Pointeri. Fişiere. Sortări.

- ✔ *Fiecare elev va realiza sarcinile strict conform cerinței. Rezultatele se vor prezenta profesorului sub formă de raport.*
- ✔ *Fiecare elev va citi atent condițiile problemei individuale și va schița schema bloc a problemei.*
- ✔ *Elevii vor fi repartizați conform listelor de la lecțiile practice.*
- ✔ *Pentru elaborarea secvenței de cod C/C++ se va aplica mediul de programare Code::Blocks.*

Problema 1

Elaborați un program care va genera un array unidimensional de date, apoi ordonați acest array conform metodei de sortare stabilită de profesor și aplicând tipul de date pointer. Toate rezultatele se vor păstra într-un fișier text ***Nume_Prenume_Pr1.txt***.

Nr.	Array-ul de date	Metoda	Tip
1	86, 31, 41, 78, 43, 34, 23, 66, 75, 74	SelectionSort	Ascendent
2	N, G, I, B, S, H, V, T, P, D, C, A, J, Z. R	InsertionSort	Descendent
3	79, 82, 68, 90, 51, 56, 33, 86, 94, 25	BubbleSort	Ascendent
4	J, I, Z, M, E, B, F, Q, L, K, H, G, T, C, D	CountingSort	Descendent
5	21, 97, 77, 68, 19, 39, 67, 81, 26, 57	SelectionSort	Ascendent
6	B, N, W, U, E, D, Z, F, I, K, Y, A, V, H, R	InsertionSort	Descendent

Nr.	Array-ul de date	Metoda	Tip
7	69, 28, 11, 60, 48, 35, 23, 32, 57, 98	BubbleSort	Ascendent
8	T, O, K, Q, G, W, C, S, V, Y, H, D, A, Z, L	CountingSort	Descendent
9	39, 77, 40, 93, 75, 33, 90, 57, 78, 34	CountingSort	Ascendent
10	E, O, Q, L, J, H, N, G, V, D, Y, S, T, B, F	BubbleSort	Descendent
11	37, 63, 75, 79, 38, 39, 51, 53, 87, 68	InsertionSort	Ascendent
12	H, Q, T, A, Z, M, U, W, L, G, X, O, J, I, B	SelectionSort	Descendent
13	23, 71, 44, 56, 16, 75, 25, 32, 43, 63	CountingSort	Ascendent
14	R, T, P, Z, B, I, G, J, O, X, M, A, V, K, C	BubbleSort	Descendent
15	69, 78, 62, 59, 19, 90, 37, 41, 67, 64	InsertionSort	Ascendent
16	Z, O, J, N, K, M, P, V, C, E, S, I, D, H, W	SelectionSort	Descendent

Problema 2

Elaborați un program care va genera un array bidimensional (3x3) de date dintr-un array unidimensional, apoi ordonați acest array conform metodei de sortare stabilită de profesor. Toate rezultatele se vor păstra într-un fișier text ***Nume_Prenume_Pr2.txt***.

Cele 8 modalități de parcurgere sub formă de spirală:

1. NV-oriz. : 1 2 3 6 9 8 7 4 5
2. NE-oriz. : 3 2 1 4 7 8 9 6 5
3. SV-oriz. : 7 8 9 6 3 2 1 4 5
4. SE-oriz. : 9 8 7 4 1 2 3 6 5
5. NV-vert. : 1 4 7 8 9 6 3 2 5
6. NE-vert. : 3 6 9 8 7 4 1 2 5
7. SV-vert. : 7 1 4 2 3 6 9 8 5
8. SE-vert. : 9 6 3 2 1 4 7 8 5

1	2	3
4	5	6
7	8	9

Cele 8 modalități de parcurgere sub formă de șarpe:

1	2	3
4	5	6
7	8	9

1. NV-oriz. : 1 2 3 6 5 4 7 8 9
2. NE-oriz. : 3 2 1 4 5 6 9 8 7
3. SV-oriz. : 7 8 9 6 5 4 1 2 3
4. SE-oriz. : 9 8 7 4 5 6 3 2 1
5. NV-vert. : 1 4 7 8 5 2 3 6 9
6. NE-vert. : 3 6 9 8 5 2 1 4 7
7. SV-vert. : 7 4 1 2 5 8 9 6 3
8. SE-vert. : 9 6 3 2 5 8 7 4 1

Nr.	Array-ul de date	Metoda	Tip	Parcurgere
1	86, 31, 41, 78, 43, 34, 23, 66, 75	SelectionSort	Ascendent	Spirală NV-orizontală
2	N, G, I, J, B, S, H, Z, V, T, P,Q	InsertionSort	Descendent	Spirală NE-orizontală
3	79, 82, 68, 90, 51, 56, 33, 86, 94	BubbleSort	Ascendent	Spirală SV-orizontală
4	J, I, Z, A, M, E, B, C, F, Q, L, K	CountingSort	Descendent	Spirală SE-orizontală
5	21, 97, 77, 68, 19, 39, 67, 81, 26	SelectionSort	Ascendent	Spirală NV-verticală
6	B, N, W, X, U, E, D, C, Z, F, I, M	InsertionSort	Descendent	Spirală NE-verticală
7	69, 28, 11, 60, 48, 35, 23, 32, 57	BubbleSort	Ascendent	Spirală SV-verticală
8	T, O, K, L Q, G, W, B, C, S, V, A	CountingSort	Descendent	Spirală SE-verticală
9	39, 77, 40, 93, 75, 33, 90, 57, 78	CountingSort	Ascendent	Șerpuită NV-orizontală
10	E, O, Q, K, L, J, H, S, N, G, V, F	BubbleSort	Descendent	Șerpuită NE-orizontală

11	37, 63, 75, 79, 38, 39, 51, 53, 87	InsertionSort	Ascendent	Şerpuită SV-orizontală
12	H, Q, T, R, A, Z, M, P, U, W, L, Z	SelectionSort	Descendent	Şerpuită SE-orizontală
13	23, 71, 44, 56, 16, 75, 25, 32, 43	CountingSort	Ascendent	Şerpuită NV-verticală
14	R, T, P, W, Z, B, I, C, G, J, O, H	BubbleSort	Descendent	Şerpuită NE-verticală
15	69, 78, 62, 59, 19, 90, 37, 41, 67	InsertionSort	Ascendent	Şerpuită SV-verticală
16	Z, O, J, N, K, M, P, V, C, E, S, I	SelectionSort	Descendent	Şerpuită SE-verticală

7.2 Lucrarea nr.2 – Structuri imbricate şi subprograme.

- ✔ *Fiecare elev va realiza sarcinile strict conform cerinței. Rezultatele se vor prezenta profesorului sub formă de raport.*
- ✔ *Fiecare elev va citi atent condițiile problemei individuale și va schița schema bloc a problemei.*
- ✔ *Elevii vor fi repartizați conform listelor de la lecțiile practice.*
- ✔ *Pentru elaborarea secvenței de cod C/C++ se va aplica mediul de programare Code::Blocks.*

Modelul I - Înmatriculare

În mun. Chișinău în anul 2020 au fost înmatriculate un număr de automobile. Aceste date au fost colectate în baza a două fișiere cu câmpurile:
- Șofer.txt (IDNP, Nume, Prenume, vârstă, sex, data_nașterii, GSM, CodAuto);
- Auto.txt (CodAuto, Marca, Model, MPS, CP, preț, data_inmatriculării).

Nr.	Conținutul problemei individuale
1	*Elaborați o secvență de cod C++ care va:* a) *înregistra un nou șofer și un nou automobil, datele se vor citi de la tastatură și se vor păstra în fișierele corespunzătoare;* b) *afișa la ecran atributele tuturor șoferilor de sex feminin la care numărul GSM începe cu 067 și ultimele 3 cifre sunt 999 în ordine ascendentă prin metoda SelectionSort după câmpul Nume;* c) *afișa atributele tuturor automobilelor care au rămas după eliminarea celor ce au mai puțin de 750 CP și model MPS – automat.*
2	*Elaborați o secvență de cod C++ care va:* a) *înregistra un nou șofer și un nou automobil, datele se vor citi de la tastatură și se vor păstra în fișierele corespunzătoare;* b) *afișa la ecran atributele tuturor șoferilor ce au vârsta mai mică decât 25 ani și sunt născuți în lunile de toamnă în ordine descendentă prin metoda InsertionSort după câmpul Nume;*

Nr.	Conținutul problemei individuale
	c) *afișa atributele tuturor automobilelor care au rămas după eliminarea celor ce au cel mult 450 CP și sunt înmatriculați în ultimele 30 de zile.*

Modelul II - Bacalaureat

În Republica Moldova în anul 2021 au depus ceceri pentru susținerea examenului de BAC mai mulți elevi. Aceste date au fost colectate în baza a două fișiere cu câmpurile:
- Elev.txt (IDNP, Nume, Prenume, sex, data_nașterii, Centrul_Raional, profil);
- Note.txt (IDNP, Disciplina_1, Disciplina_2, Disciplina_3, Disciplina_4, Note_insuficiente, Media).

Profil real	Profil uman	Profil arte	Profil sport
Matematica; Informatica; Fizica; Biologia; L. engleză.	L. română; Geografia; Istoria; Filosofia; L. engleză.	L. română; Istoria artei; Compoziție; Practică; L. engleză.	L. română; Istoria sportului; Cultura fizică; Practică; L. engleză.

Nr.	Conținutul problemei individuale
3	*Elaborați o secvență de cod C++ care va:* a) *înregistra un nou elev și notele acestuia, datele se vor citi de la tastatură și se vor păstra în fișierele corespunzătoare;* b) *afișa la ecran atributele tuturor elevilor de sex masculin din centrul raional Bălți în ordine descendentă prin metoda SelectionSort după câmpul Profil;* c) *afișa atributele tuturor elevilor care au rămas după eliminarea celor ce au media la BAC cel puțin egală cu 8.00 și de profil arte.*
4	*Elaborați o secvență de cod C++ care va:* a) *înregistra un nou elev și notele acestuia, datele se vor citi de la tastatură și se vor păstra în fișierele corespunzătoare;*

Nr.	Conţinutul problemei individuale
	b) *afișa la ecran atributele tuturor elevilor născuți în linile de tiamnă din centrul raional Soroca în ordine ascendentă prin metoda InsertionSort după câmpul IDNP;* c) *afișa atributele tuturor elevilor care au rămas după eliminarea celor ce au cel puțin o notă insuficientă la BAC și de profil real.*

Modelul III – Natalitate

În Republica Moldova în anul 2020 au fost născuți un număr de bebeluși. Aceste date au fost colectate prin două fișiere cu campurile:
- Bebeluș.txt (IDNP, Nume, Prenume, sex, data_nașterii, Greutate, CodR);
- Centru_raional.txt (CodR, An_Atestare, Populație, Suprafață, Altitudine).

Nr.	Conţinutul problemei individuale
5	*Elaborați o secvență de cod C++ care va:* a) *înregistra un nou bebeluș și un nou centru raional, datele se vor citi de la tastatură și se vor păstra în fișierele corespunzătoare;* b) *afișa la ecran atributele tuturor bebelușilor la care Prenumele începe cu litera „P” și au o greutate mai mică de 3.20 kg în ordine descendentă prin metoda BubbleSort după câmpul CodR;* c) *afișa atributele tuturor centrelor raionale ce au o suprafață mai mică decât 1050 km^2 și unde s-au născut cei mai mulți băieți.*
6	*Elaborați o secvență de cod C++ care va:* a) *înregistra un nou bebeluș și un nou centru raional, datele se vor citi de la tastatură și se vor păstra în fișierele corespunzătoare;* b) *afișa la ecran atributele tuturor bebelușilor la sunt născuți iarna și care au ultima cifră de la IDNP un număr par în ordine ascendentă prin metoda CountingSort după câmpul Greutate;* c) *afișa atributele tuturor centrelor raionale unde s-au născut cei mai puțini băieți în lunile de primăvară.*

Modelul IV - Pensionari

În Republica Moldova în anul 2020 au fost născuți un număr de bebeluși. Aceste date au fost colectate prin două fișiere cu campurile:
- Pensionari.txt (IDNP, Nume, Prenume, sex, vârsta, Profesie, Pensie, CodR);
- Centru_raional.txt (CodR, An_Atestare, Populație, Suprafață, Altitudine).

Nr.	Conținutul problemei individuale
7	*Elaborați o secvență de cod C++ care va:* a) *înregistra un nou pensionar și un nou centru raional, datele se vor citi de la tastatură și se vor păstra în fișierele corespunzătoare;* b) *afișa la ecran atributele tuturor pensionarilor ce au o vârstă mai mare decât 65 de ani și care sunt din raionul „Glodeni” în ordine descendentă prin metoda SelectionSort după câmpul Profesie;* c) *afișa atributele tuturor centrelor raionale unde s-au pensionat cei mai mulți bărbați în lunile de vară.*
8	*Elaborați o secvență de cod C++ care va:* a) *înregistra un nou pensionar și un nou centru raional, datele se vor citi de la tastatură și se vor păstra în fișierele corespunzătoare;* b) *afișa la ecran atributele tuturor pensionarilor ce au o pensie mai mică decât 5000 de lei și care au ultima cifră din IDNP impară în ordine descendentă prin metoda SelectionSort după câmpul Pensie;* c) *afișa atributele tuturor centrelor raionale unde s-au pensionat cele mai puține femei de profesie „Pedagog”.*

Modelul V – Transport

În mun. Chișinău în anul 2020 au fost înmatriculate un număr de automobile. Aceste date au fost colectate prin două fișiere cu campurile:
- Șofer.txt (IDNP, Nume, Prenume, vârstă, sex, data_nașterii, Ani_Experiență, GSM, CodAuto);
- Transport.txt (CodAuto, Denumire, preț_călătorie, data_inmatriculării, Prima-Tură, Ultima_Tură).

Nr.	Conţinutul problemei individuale
9	*Elaborați o secvență de cod C++ care va:* a) *înregistra un nou șofer și un nou mijloc de transport, datele se vor citi de la tastatură și se vor păstra în fișierele corespunzătoare;* b) *afișa la ecran atributele tuturor șoferilor de sex feminin ce are minim 5 ani de experiență și la care data de naștere este un număr prim în ordine descendentă prin metoda InsertionSort după câmpul vârsta;* c) *afișa atributele tuturor mijloacelor de transport au rămas după eliminarea celor ce au prețul călătoriei egal cu 5 lei și la care ultima cursă este la ora 23:30.*
10	*Elaborați o secvență de cod C++ care va:* a) *înregistra un nou șofer și un nou mijloc de transport, datele se vor citi de la tastatură și se vor păstra în fișierele corespunzătoare;* b) *afișa la ecran atributele tuturor șoferilor de sex masculin ce au ultime două cifre pare la GSM în ordine ascendentă prin metoda BubbleSort după câmpul Ani_Experianță;* c) *afișa atributele tuturor mijloacelor de transport au rămas după eliminarea celor ce au prețul călătoriei mai mic sau egal cu 3 lei și la care prima cursă este la ora 05:30.*

Modelul VI – Cazare în cămin

La IP CEITI elevii au depus cereri pentru a fi cazați în căminele disponibile instituției din mun. Chișinău pentru anul de studii 2021 - 2022. Aceste date au fost colectate în baza a două fișiere cu câmpurile:
- Elev.txt (IDNP, Nume, Prenume, sex, data_nașterii, GSM, Grupa, Specialitatea, Odaia, CodCamin);
- Camin.txt (CodCamin, Locația, Număr_Nivele, preț, data_cazării).

Nr.	Conţinutul problemei individuale
11	*Elaborați o secvență de cod C++ care va:* a) *înregistra un nou elev și un nou bloc de cămin, datele se vor citi de*

Nr.	Conținutul problemei individuale
	la tastatură și se vor păstra în fișierele corespunzătoare; b) *afișa la ecran atributele tuturor elevilorilor de sex feminin de la specialitatea „Rețele de calculatoare” în ordine ascendentă prin metoda InsertionSort după câmpul Grupa;* c) *afișa atributele căminului în care sunt cei mai mulți și cei mai puțini elevi cazați, dacă se cunoaște că instituția dispune de 5 blocuri de cazare a câte maxim 450 de locuri de cazare.*
12	*Elaborați o secvență de cod C++ care va:* a) *înregistra un nou elev și un nou bloc de cămin, datele se vor citi de la tastatură și se vor păstra în fișierele corespunzătoare;* b) *afișa la ecran atributele tuturor elevilorilor de la specialitatea „Contabilitate” în ordine descendentă prin metoda BubbleSort după câmpul Data_nașterii;* c) *afișa atributele căminului în care sunt cei mai mulți și cei mai puțini elevi cazați din grupa „P-2112”, dacă se cunoaște că elevii au fost repartizați în cele 5 cămine în mod aleatoriu.*

Modelul VII – Schimb valutar

În mun. Chișinău în anul 2020 au existat mai multe unități de schimb valutar. Aceste date au fost colectate prin două fișiere cu câmpurile:
- UnitateSchimb.txt (CodUnitate, Denumire, Sectorul, Telefon, CodCV);
- CursValutar.txt (CodCV, cUSD, vUSD, cEUR, vEUR, cRON, vRON).

Nr.	Conținutul problemei individuale
13	*Elaborați o secvență de cod C++ care va:* a) *înregistra o nouă unitate de schimb valutare și datele specifice cursurilor valutare, datele se vor citi de la tastatură și se vor păstra în fișierele corespunzătoare;* b) *afișa la ecran atributele tuturor unităților bancare din sectorul Botanica ce au ultima cifră de la telefon este un număr prim în*

Nr.	Conţinutul problemei individuale
	ordine descendentă prin metoda CountingSort după câmpul Denumire; c) *afișa atributele de la unitatea de schimb valutar unde este cel mai mic preț de vânzare și cel mai scump la cumpărarea pentru euro.*
14	*Elaborați o secvență de cod C++ care va:* a) *înregistra o nouă unitate de schimb valutare și datele specifice cursurilor valutare, datele se vor citi de la tastatură și se vor păstra în fișierele corespunzătoare;* b) *afișa la ecran atributele tuturor unităților bancare din sectorul Centru ce au penultima cifră de la telefon un număr impar divizibil cu 3 în ordine ascendentă prin metoda CountingSort după câmpul Telefon;* c) *afișa atributele de la toate unitățile de schimb valutar unde este prețul de vânzare și cel de cumpărarea pentru ron mai mic decât media de vânzare și cumpărare.*

Modelul VIII - Vaccinare

În mun. Chișinău în anul 2020 a început procesul de vaccinare a cadrelor medicale împotriva Covid-19. Aceste date au fost colectate în baza a două fișiere cu câmpurile:
- CadreMedicale.txt (IDNP, Nume, Prenume, vârstă, sex, data_nașterii, GSM, Sectorul, Specializare, DataDozei1, DataDozei2, DataDoza3, CodVaccin);
- Vaccine.txt (CodVaccin, Denumire, Producător, Țara, preț, EficacitateDoza1, EficacitateDoza2, EficacitateDoza3, ReacțiiAdverse).

Nr.	Conţinutul problemei individuale
15	*Elaborați o secvență de cod C++ care va:* a) *înregistra un nou cadru medical și datele specifice vaccinului, care se vor citi de la tastatură și se vor păstra în fișierele corespunzătoare;* b) *afișa la ecran atributele tuturor cadrelor medicale din sectorul*

Nr.	Conținutul problemei individuale
	Centru ce au specialitatea cardiolog cu vârsta mai mică de 45 de nii în ordine descendentă prin metoda SelectionSort după câmpul Nume; c) *afișa atributele celui mai ieftin și celui mai scump tip de vaccin dintr-o țară denumirea căreia va fi introdusă de la tastatură.*
16	*Elaborați o secvență de cod C++ care va:* a) *înregistra un nou cadru medical și datele specifice vaccinului administrat, care se vor citi de la tastatură și se vor păstra în fișierele corespunzătoare;* b) *afișa la ecran atributele tuturor cadrelor medicale din sectorul Telecentru ce au specialitatea stomatolog cu vârsta mai mare de 50 de nii în ordine ascendentă prin metoda InsertionSort după câmpul DataDozei1;* c) *afișa atributele celui mai ieftin vaccin și ale celor care au ca reacție adversă Cardiomiopatie.*

- ✔ *Pentru modelele de probleme prezentate mai sus, fiecare cadru didactic poate simplifica sau amplifica cerințele.*
- ✔ *De asemenea sunt binevenite și alte tipuri de probleme care să fie structurate într-o formă aproximativă celei prezentate în modelele de mai sus.*
- ✔ *O recomandare ar fi păstrarea tuturor datelor în fișiere externe și de asemenea este binevenită implementarea pointerilor .*

7.3 Lucrarea nr.3 – Subprograme recursive şi module.

- ✔ *Fiecare elev va realiza sarcinile strict conform cerinței. Rezultatele se vor prezenta profesorului sub formă de raport.*
- ✔ *Fiecare elev va citi atent condițiile problemei individuale și va schița schema bloc a problemei.*
- ✔ *Pentru elaborarea secvenței de cod C/C++ se va aplica mediul de programare Code::Blocks.*

Nr.	Conținutul problemei individuale
1	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru un triunghi echilateral: Perimetrul, Aria, Raza cercului înscris și Raza cercului circumscris.*
2	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru un pătrat: Perimetrul, Aria, Raza cercului înscris și Raza cercului circumscris.*
3	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru un trapez isoscel: Perimetrul, Aria, Raza cercului înscris și Raza cercului circumscris.*
4	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru un paralelogram: Perimetrul, Aria, Raza cercului înscris și Raza cercului circumscris.*
5	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru un pentagon: Perimetrul, Aria, Raza cercului înscris și Raza cercului circumscris.*
6	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a deter-*

Nr.	Conținutul problemei individuale
	mina pentru un hexagon: Perimetrul, Aria, Raza cercului înscris și Raza cercului circumscris.
7	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru un cerc: lungimea, Aria, latura unui triunghi echilateral și latura unui pătrat.*
8	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru un cub: Aria totală, Aria laterală, Volumul, diagonala_bazei și diagonala_mare.*
9	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru un paralelipiped dreptunghic: Aria totală, Aria laterală, Volumul, diagonala_bazei și diagonala_mare.*
10	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru o prismă regulată triunghiulară: Aria totală, Aria laterală, Volumul și diagonala_mare.*
11	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru o prismă regulată patrulateră: Aria totală, Aria laterală, Volumul și diagonala_mare.*
12	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru o prismă regulată pentagonală: Aria totală, Aria laterală, Volumul și diagonala_mare.*
13	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru o prismă regulată hexagonală: Aria totală, Aria laterală, Volumul și diagonala_mare.*
14	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a deter-*

Nr.	Conținutul problemei individuale
	mina pentru o piramidă regulată triunghiulară: Aria totală, Aria laterală, Volumul și Raza maximă a unei sferei înscrise.
15	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru o piramidă regulată patrulateră: Aria totală, Aria laterală, Volumul și Raza maximă a unei sferei înscrise.*
16	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru o piramidă regulată pentagonală: Aria totală, Aria laterală, Volumul și Raza maximă a unei sferei înscrise.*
17	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru o piramidă regulată hexagonală: Aria totală, Aria laterală, Volumul și Raza maximă a unei sferei înscrise.*
18	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru un trunchi de piramidă regulată triunghiulară: Aria totală, Aria laterală, Volumul și Raza a unei sferei circumscrise.*
19	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru un trunchi de piramidă regulată patrulateră: Aria totală, Aria laterală, Volumul și Raza a unei sferei circumscrise.*
20	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru un trunchi de piramidă regulată pentagonală: Aria totală, Aria laterală, Volumul și Raza a unei sferei circumscrise.*
21	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru un trunchi de piramidă regulată hexagonală:*

Nr.	Conținutul problemei individuale
	Aria totală, Aria laterală, Volumul și Raza a unei sferei circumscrise.
22	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru un cilindru: Aria totală, Aria laterală, Volumul și Raza maximă a unei sferei înscrise.*
23	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru un con: Aria totală, Aria laterală, Volumul și Raza maximă a unei sferei înscrise.*
24	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru un trunchi de con: Aria totală, Aria laterală, Volumul și Raza maximă a unei sferei înscrise.*
25	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru un butoi: Aria totală, Aria laterală și Volumul.*
26	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru un butoi: Raza a unei sferei înscrise și raza unei sfere circumscrise.*
27	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru o sferă: Volumul unui tetraedru înscris și circumscris sferei.*
28	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru o sferă: Volumul unui cub înscris și circumscris sferei.*
29	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru o sferă: Volumul unui cilindru înscris și circum-*

Nr.	Conținutul problemei individuale
	scris sferei.
30	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru o sferă: Volumul unui con înscris și circumscris sferei.*
31	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru o sferă: Volumul unei prisme regulate pentagonale înscris și circumscris sferei.*
32	*Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru o sferă: Volumul unei prisme regulate hexagonale înscris și circumscris sferei.*

- ✔ *Pentru modelele de probleme prezentate mai sus, subprogramele ce trebuie de realizat pentru soluționarea sarcinilor propuse pot fi redactate de către profesor sau pot fi compuse altele noi.*
- ✔ *O recomandare ar fi păstrarea tuturor datelor în fișiere externe și de asemenea este binevenită implementarea pointerilor și a subprogramelor recursive.*

7.4 Model de lucrare pentru studiul individual

Studiul individual nr.1

Model de foaie de titlu a lucrării

Studiu individual nr.1 la unitatea de curs
Programare procedurală
cu genericul
Tipul de date pointer și fișier, algoritmi de sortare în C/C++

Profesor NUME PRENUME

Nume Prenume	*Grupa*	*Link-ul de la secvența video*	*Nota*

Unități de conținut propuse:

1. *Tipul de date fișier. Funcții de prelucrare a datelor din fișiere;*
2. *Tipul de date pointer. Alocarea dinamică a memoriei;*
3. *Algoritmi de sortare: bule, selecție, inserție și numerare.*

Modalități de evaluare a lucrării individuale:

1. *Comunicare sincronă și asincronă;*
2. *Demonstrare funcționalității produsului program pe un dispozitiv electronic de calcul: smartPhone, smartTab sau notebook (sau PC);*
3. *Crearea unei secvențe video în care elevul explică pas cu pas realizarea tuturor sarcinilor propuse, deci pentru fiecare studiu individual elevul va trebui să realizeze câte o secvență video corespunzătoare și să aplice în comunicarea sa limbajul de specialitate corespunzător unității de conținut;*
4. *În lucrarea finală fiecare elev va trimite și link-ul cu secvența video realizată conform sarcinii propuse.*

Problema 1.1

Elaborați un program care va genera un array unidimensional de date, apoi ordonați acest array conform metodei de sortare stabilită de profesor și aplicând tipul de date pointer. Toate rezultatele se vor păstra într-un fișier text ***Nume_Prenume_Pr1.txt***.

3	29, 86, 28, 90, 54, 16, 30, 82, 54, 23	BubbleSort	Ascendent

Vom rezolva problema conform condițiillor de mai sus.

Rezolvare:

- Generăm numere din intervalul [0,99]: a[i]=rand()%100;
- Generăm numere din intervalul [10,99]: a[i]=10+rand()%100;

Listingul programului în limbajul C/C++

```
#include <iostream>
#include <fstream>
#define N 10
using namespace std;
ofstream fout("NumePrenumeP1.txt");
int i,a[N],ok,aux;
int main(){
    // Generam elementele array-ului unidimensional
    for(int i=0;i<N;i++){
        a[i]=rand()%100; //Generam numere din intervalul [0,99]
    }
    // Afisam elementele array-ului unidimensional generat
    cout<<"Elementele array-ului unidimensional generat: \n\t";
    fout<<"Elementele array-ului unidimensional generat: \n\t";
    for(i=0;i<N;i++){
        cout<<a[i]<<" ";
        fout<<a[i]<<" ";// scriem datele in fisier
    }
    // Implementarea algoritmului metodei BubbleSort ascendent
    while(ok!=1){
        ok=1; aux=*a;
        for(i=0;i<N-1;i++)
```

```
            if(a[i]>a[i+1]) {
                aux=a[i]; a[i]=a[i+1]; a[i+1]=aux; ok=0;
            }
    }
    // Afisam elementele array-ului unidimensional generat
    cout<<"\nElementele array-ului unidimensional sortat: \n\
t";
    fout<<"\nElementele array-ului unidimensional sortat: \n\
t";
    for(i=0;i<N;i++){
        cout<<a[i]<<" ";
        fout<<a[i]<<" ";// scriem datele in fisier
    }
}
```

Rezultatele execuției programului în mediul de programare CodeBlocks:

```
Elementele array-ului unidimensional generat:
        41 67 34 0 69 24 78 58 62 64
Elementele array-ului unidimensional sortat:
        0 24 34 41 58 62 64 67 69 78
```

Problema 1.2

Elaborați un program care va genera un array bidimensional (3x4) de date dintr-un array unidimensional, apoi ordonați acest array conform metodei de sortare stabilită de profesor. Toate rezultatele se vor păstra într-un fișier text ***Nume_Prenume_Pr2.txt***.

6	1, 2. 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16	InsertionSort	Descendent	Spirală NE-orizontală

Vom rezolva problema conform condițiillor de mai sus.

Rezolvare:

Listingul programului în limbajul C/C++

```
#include <iostream>
#include <fstream>
using namespace std;
int a[5][5],b[25],n,i,j,k,aux;
//Fisierul extern pentru pastrarea datelor
ofstream fout("NumePrenumeP2.txt");
int main() {
       cout<<"Introducem dimensiunea matricii patratice: ";
       cin>>n;
       cout<<"Introducem elementele matricii: "<<endl;
       for(i=1;i<=n;i++){
              for(j = 1; j <= n; ++j){
                     cout<<"a["<<i<<"]["<<j<<"]= ";
                     cin>>a[i][j];
              }
       }
       getchar();system("CLS");
       cout<<"Afisarea elementelor matricii: "<<endl;
       fout<<"Afisarea elementelor matricii: "<<endl;
       for(i=1;i<=n;i++){
              for(j = 1; j <= n; ++j){
                  cout<<"  "<<a[i][j]; fout<<"  "<<a[i][j];
              }
              cout<<endl; fout<<endl;
       }
    //Parcurgerea sub forma de spirala NV-Orizontala
    k=1;
    for(i=1;i<=n/2;i++){
       for(j=i;j<=n-i;j++){
            b[k]=a[i][j]; k++;
       }
       for(j=i;j<=n-i;j++){
            b[k]=a[j][n-i+1]; k++;
       }
       for(j=n-i+1;j>=i+1;j--){
            b[k]=a[n-i+1][j]; k++;
       }
       for(j=n-i+1;j>=i+1;j--){
```

```
            b[k]=a[j][i]; k++;
        }
    }
    if(n%2==1) b[n*n]=a[n/2+1][n/2+1];
    //Afisarea array-ului unidimensional
    cout<<"\nAfisarea sub forma de spirala NV-Orizontala:\n";
    fout<<"\nAfisarea sub forma de spirala NV-Orizontala:\n";
    for(i=1;i<=n*n;i++){
        cout<<" "<<b[i]; fout<<" "<<b[i];
    }
    // Implementarea algoritmului metodei InsertionSort descen-
dent
    for(i=1;i<=n*n;i++)
        if (b[i]>b[i-1]){
            aux=b[i];j=i-1;
            while (j>=0 && b[j]<aux){
                 b[j+1]=b[j]; j--;
            }
            b[j+1]=aux;
         }
    //Afisarea array-ului unidimensional sortat
    cout<<"\nAfisarea array-ului sortat descendent: "<<endl;
    fout<<"\nAfisarea array-ului sortat descendent: "<<endl;
    for(i=0;i<n*n;i++){
        cout<<" "<<b[i]; fout<<" "<<b[i];
    }
}
```

Rezultatele execuției programului în mediul de programare CodeBlocks:

```
Afisarea elementelor matricii:
  1  2  3
  4  5  6
  7  8  9

Afisarea sub forma de spirala NV-Orizontala:
 1 2 3 6 9 8 7 4 5
Afisarea array-ului sortat descendent:
 9 8 7 6 5 4 3 2 1
```

Studiul individual nr.2

Model de foaie de titlu a lucrării

Studiu individual nr.2 la unitatea de curs
Programare procedurală
cu genericul
Structuri imbricate și subprograme în C/C++

Profesor NUME PRENUME

Nume Prenume	*Grupa*	*Link-ul de la secvența video*	*Nota*

Unități de conținut propuse:

1. *Tipul de date structură, uniune și enumerare;*
2. *Array-uri de structuri și uniuni;*
3. *Structuri imbricate și uniuni imbricate.*

Modalități de evaluare a lucrării individuale:

1. *Comunicare sincronă și asincronă;*
2. *Demonstrare funcționalității produsului program pe un dispozitiv electronic de calcul: smartPhone, smartTab sau notebook (sau PC);*
3. *Crearea unei secvențe video în care elevul explică pas cu pas realizarea tuturor sarcinilor propuse, deci pentru fiecare studiu individual elevul va trebui să realizeze câte o secvență video corespunzătoare și să aplice în comunicarea sa limbajul de specialitate corespunzător unității de conținut;*
4. *În lucrarea finală fiecare elev va trimite și link-ul cu secvența video realizată conform sarcinii propuse.*

Problema 2.1

La IP CEITI elevii ce au depus cereri pentru a solicita să susțină examenul de BAC în anul de studii 2020 – 2021, au primit rezultatele. Aceste date au fost colectate în baza unui fișier cu câmpurile:
- Bacalaureat.txt (Nume, Prenume, sex, profil, NotaD1, NotaD2, NotaD3, NotaD4, CentruRaional).

Elaborați subprograme în C++ pentru a:

a) *citi datele din fișierul Bacalaureat.txt și de a calcula media fiecărui elev la cele 4 discipline examinate;*
b) *afișa la ecran datele din fișierul Bacalaureat.txt conform câmpurilor specifice fiecărui elev;*
c) *afișa la ecran media fiecărui elev la cele 4 discipline examinate cu cel mult 3 zecimale;*
d) *afișa la ecran procentul calității grupei de elevi (se determină numărul de elevi ce au mediile mai mari sau egale cu 7.00) la cele 4 discipline examinate cu cel mult 3 zecimale, dar și numele și prenumele elevilor împreună cu mediile acestora (mai mari decât 7.00);*
e) *afișa pentru fiecare disciplină nota cea mai mică și câți elevi din listă au acumulat această nota.*

Rezolvare:

Listingul programului în limbajul C/C++

```
#include <iostream>
#include <iomanip>
#include <fstream>
using namespace std;
// declaram datele structurii elev
struct BAC2021 {
    char n[10],p[10],CR[10]; // nume, prenume, centru raional
    int d1,d2,d3,d4; // cele 4 discipline de examinare
    char sex,profil[10];
    float media;
```

```
} a[20];
// Variabilele globale
int i,j,n,k=0,s[20];
// Fisierul de intrare
ifstream fin("Bacalaureat.txt");
// Citim datele din fisier
int citireFisier(){
    if (!fin){
        cout<<"Fisierul nu poate fi deschis!"<<endl;
        exit(1);
    }
    else while(!fin.eof()){

fin>>a[k].n>>a[k].p>>a[k].sex>>a[k].profil>>a[k].d1>>a[k].d2>>a
[k].d3>>a[k].d4>>a[k].CR;
        s[k]=a[k].d1+a[k].d2+a[k].d3+a[k].d4;
        a[k].media=s[k]/4.0;
        k++;
    }
    fin.close();
}
// Afisam datele din fisier
int afisareFisier(){
    for(j=0;j<k-1;j++){
        cout<<" \t"<<a[j].n<<" \t"<<a[j].p<<"\t\t"<<a[j].sex<<"
\t"<<a[j].profil;
        cout<<" \t"<<a[j].d1<<" "<<a[j].d2<<" "<<a[j].d3<<"
"<<a[j].d4<<" \t"<<a[j].CR;
        cout<<endl;
    }
}
// Afisam medii elevi
int afisareMedia(){
    for(j=0;j<k-1;j++){
        cout<<" \t"<<a[j].n<<" \t"<<a[j].p<<"\t\t";
        cout<<fixed<<setprecision(3)<<a[j].media;
        cout<<endl;
    }
}
// Determinam procentul calitatii
int afisareCalitate(){
    int s=0;
```

```
    for(j=0;j<k-1;j++){
        if (a[j].media >= 7.00){
            cout<<" \t"<<a[j].n<<" \t"<<a[j].p<<"\t\t";
            cout<<fixed<<setprecision(3)<<a[j].media;
            cout<<endl;s++;
        }
    }
    cout<<"Procentul calitatii: "<<s*100.0/k;
}
// Numarul de elevi ce au nota minima la fiecare disciplina
int minimDiscipline(){
    int min1=a[0].d1,m1=0;
    int min2=a[0].d2,m2=0;
    int min3=a[0].d3,m3=0;
    int min4=a[0].d4,m4=0;
    for(j=0;j<k-1;j++){
        // pentru disciplina 1
        if (a[j].d1 < min1) min1=a[j].d1;
        if (a[j].d1 == min1) m1++;
        // pentru disciplina 2
        if (a[j].d2 < min2) min2=a[j].d2;
        if (a[j].d2 == min2) m2++;
        // pentru disciplina 3
        if (a[j].d3 < min3) min3=a[j].d3;
        if (a[j].d3 == min3) m3++;
        // pentru disciplina 4
        if (a[j].d4 < min4) min4=a[j].d4;
        if (a[j].d4 == min4) m4++;
    }
    cout<<"\tNota minima la disciplina D1 este: "<<min1<<" si o
au "<<m1<<" elevi!\n";
    cout<<"\tNota minima la disciplina D2 este: "<<min2<<" si o
au "<<m2<<" elevi!\n";
    cout<<"\tNota minima la disciplina D3 este: "<<min3<<" si o
au "<<m3<<" elevi!\n";
    cout<<"\tNota minima la disciplina D4 este: "<<min4<<" si o
au "<<m4<<" elevi!\n";
}
// Programul principal
int main(){
    cout<<"Citim datele din fisierul de intrare!"<<endl;
    citireFisier();
```

```
    getchar(); system("CLS");
    cout<<"Afisam datele din fisierul de intrare!"<<endl;
    afisareFisier();
    getchar(); system("CLS");
    cout<<"Afisam mediile elevilor la cele 4 examene: "<<endl;
    afisareMedia();
    getchar(); system("CLS");
    cout<<"Afisam elevii cu media >= 7.00: "<<endl;
    afisareCalitate();
    getchar(); system("CLS");
    cout<<"Afisam pe discipline numarul de elevi ce au nota
minima: "<<endl;
    minimDiscipline();
}
```

Rezultatele execuției programului în mediul de programare CodeBlocks:

Deoarece nu ne afișează nici o eroare, înseamnă că datele sau citit cu succes din fișierul de intrare Bacalaureat.txt

```
Citim datele din fisierul de intrare!
```

Afișează datele citite cu succes din fișierul de intrare Bacalaureat.txt conform aceleiași structuri care este în fișier.

```
Afisam datele din fisierul de intrare!
    Cerbu   Olga        F     Real    6 7 8 9     Chisinau
    Dascal  Emanuel     M     Arte    9 8 7 10    Soroca
    Grosu   Mihaela     F     Sport   5 6 7 9     Balti
    Crudu   Valeria     F     Uman    5 6 7 6     Cahul
    Rata    Remus       M     Sport   8 9 8 10    Balti
    Calalb  Marcu       M     Real    7 8 9 6     Soroca
    Bousor  George      M     Arte    7 8 9 10    Taraclia
    Bivol   Larisa      F     Arte    6 7 8 10    Rezina
    Creng   Eugen       M     Sport   5 6 9 10    Rascani
    Rata    Livia       F     Arte    8 9 7 10    Glodeni
```

Afișează mediile fiecărui elev la cele 4 discipline de examinare citite din fișierul Bacalaureat.txt cu cel mult 3 zecimale.

```
Afisam mediile elevilor la cele 4 examene:
        Cerbu   Olga            7.500
        Dascal  Emanuel         8.500
        Grosu   Mihaela         6.750
        Crudu   Valeria         6.000
        Rata    Remus           8.750
        Calalb  Marcu           7.500
        Bousor  George          8.500
        Bivol   Larisa          7.750
        Creng   Eugen           7.500
        Rata    Livia           8.500
```

Afișează mediile care sunt mai mari decât 7.00 la cele 4 discipline de examinare citite din fișierul Bacalaureat.txt și calculează procentul calității grupei de elevi cu cel mult 3 zecimale.

```
Afisam elevii cu media >= 7.00:
        Cerbu   Olga            7.500
        Dascal  Emanuel         8.500
        Rata    Remus           8.750
        Calalb  Marcu           7.500
        Bousor  George          8.500
        Bivol   Larisa          7.750
        Creng   Eugen           7.500
        Rata    Livia           8.500
Procentul calitatii: 72.727
```

Afișează nota minimă la fiecare din cele 4 discipline de examinare citite din fișierul Bacalaureat.txt și numărul de elevi ce au acumulat această notă.

```
Afisam pe discipline numarul de elevi ce au nota minima:
        Nota minima la disciplina D1 este: 5 si o au 4 elevi!
        Nota minima la disciplina D2 este: 6 si o au 4 elevi!
        Nota minima la disciplina D3 este: 7 si o au 5 elevi!
        Nota minima la disciplina D4 este: 6 si o au 4 elevi!
```

Problema 2.2

La IP CEITI elevii ce au depus cereri pentru a solicita să susțină examenul de BAC în anul de studii 2020 – 2021, au primit rezultatele. Aceste date au fost colectate în baza unui fișier cu câmpurile:
- Bacalaureat.txt (Nume, Prenume, sex, profil, NotaD1, NotaD2, NotaD3, NotaD4, CentruRaional).

Elaborați subprograme în C++ pentru a:

a) *afișa în ordine ascendentă conform metodei SelectionSort lista de elevi după câmpul Nume;*
b) *afișa în ordine descendentă conform metodei InsertionSort lista de elevi după câmpul CentruRaional;*
c) *afișa în ordine ascendentă conform metodei BubbleSort lista de elevi după câmpul Profil și media la cele 4 examene mai mare sau egală cu 7.00.*

Rezolvare:

Listingul programului în limbajul C/C++

```
#include <iostream>
#include <fstream>
#include <cstring>
using namespace std;
// declaram datele structurii elev
struct BAC2021{
    char N[20],P[20],Sex,Profil[10],CR[20];
    int D1,D2,D3,D4;
}*p;
ofstream fout("SortareElevi.txt");
int i,j,n = 0;
// Citim datele din fisier
void CitireFisier(BAC2021* &p, int &n){
    ifstream fin("Bacalaureat.txt");
    p = new BAC2021[100];
    if (!fin){
```

```
        cout<<"Fisierul nu poate fi deschis!"<<endl;
        exit(1);
    }
    else while(!fin.eof()){

fin>>p[n].N>>p[n].P>>p[n].Sex>>p[n].Profil>>p[n].D1>>p[n].D2>>p
[n].D3>>p[n].D4>>p[n].CR;
        n++;
    }
    fin.close();
}
// Realizam prima conditie a problemei
void AfisareSelectionSort(BAC2021* &p, int &n){
    for(i=0; i<n; i++)
    for(j=i+1; j<n; j++)
    if( strcmp(p[i].N, p[j].N) > 0){
        BAC2021 aux; aux = p[j];
        p[j] = p[i]; p[i] = aux;
    }
    fout<<"\nLista elevilor dupa sortarea numelelor ascendent:\
n\n";
    for(i=0; i<n; i++){
        fout<<"\t"<<p[i].N<<"\t"<<p[i].P<<"\t\t"<<p[i].Sex<<" \
t"<<p[i].Profil<<"\t"<<p[i].D1<<" "<<p[i].D2<<" "<<p[i].D3<<"
"<<p[i].D4<<" \t"<<p[i].CR<<endl;
    }
}
// Realizam a doua conditie a problemei
void AfisareInsertionSort(BAC2021* &p, int &n){
    int j;
    BAC2021 aux;
    for(int i=1; i<n; i++)
        if (strcmp(p[i].CR, p[i-1].CR) > 0){
            aux = p[i];
            j = i-1;
            while (j >= 0 && p[j].CR > aux.CR){
                    p[j+1] = p[j]; j--;
            }
            p[j + 1] = aux;
        }
    fout<<"\nLista elevilor dupa sortarea centerlor raionale
descendent:\n\n";
```

```
    for(i=0; i<n; i++){
        fout<<"\t"<<p[i].N<<"\t"<<p[i].P<<"\t\t"<<p[i].Sex<<" \
t"<<p[i].Profil<<"\t"<<p[i].D1<<" "<<p[i].D2<<" "<<p[i].D3<<"
"<<p[i].D4<<" \t"<<p[i].CR<<endl;
    }
}
// Realizam a treia conditie a problemei
void AfisareBobbleSort(BAC2021* &p, int &n){
    BAC2021 aux;
    int ok;
    do{
        ok=1;
        for(int i = 0; i < n-1; i++)
            if(strcmp(p[i].Profil,  p[i+1].Profil) > 0){
               aux = p[i]; p[i] = p[i+1];
               p[i+1] = aux; ok=0;
            }
    }
    while(ok!=1);
    fout<<"\nLista elevilor sortata ascendent dupa profil cu
media examenelor peste 7.00:\n\n";
    for(int i = 0; i < n; i++){
        if((p[i].D1 + p[i].D2 + p[i].D3 + p[i].D4)/4.0 >= 7.00)
        fout<<"\t"<<p[i].N<<"\t"<<p[i].P<<"\t\t"<<p[i].Sex<<" \
t"<<p[i].Profil<<"\t"<<p[i].D1<<" "<<p[i].D2<<" "<<p[i].D3<<"
"<<p[i].D4<<" \t"<<p[i].CR<<endl;
    }
}
// Programul principal
int main(){
    cout<<"Citim datele din fisierul de intrare!"<<endl;
    CitireFisier(p, n);
    getchar(); system("CLS");
    cout<<"Toate rezultatele se vor pastra in fisierul exter:\
n";
    cout<<"SortareElevi.txt"; getchar(); system("CLS");
    AfisareSelectionSort(p, n); AfisareInsertionSort(p, n);
    AfisareBobbleSort(p, n);
}
```

Rezultatele execuției programului în mediul de programare CodeBlocks:

Sortarea 1:

```
Lista elevilor dupa sortarea numelelor ascendent:

Bivol   Larisa          F       Arte    6 7 8 10      Rezina
Bousor  George          M       Arte    7 8 9 10      Taraclia
Calalb  Marcu           M       Real    7 8 9 6       Soroca
Cerbu   Olga            F       Real    6 7 8 9       Chisinau
Creng   Eugen           M       Sport   5 6 9 10      Rascani
Crudu   Valeria         F       Uman    5 6 7 6       Cahul
Dascal  Emanuel         M       Arte    9 8 7 10      Soroca
Grosu   Mihaela         F       Sport   5 6 7 9       Balti
Rata    Remus           M       Sport   8 9 8 10      Balti
Rata    Livia           F       Arte    8 9 7 10      Glodeni
```

Sortarea 2:

```
Lista elevilor dupa sortarea centerlor raionale descendent:

Rata    Livia           F       Arte    8 9 7 10      Glodeni
Dascal  Emanuel         M       Arte    9 8 7 10      Soroca
Creng   Eugen           M       Sport   5 6 9 10      Rascani
Calalb  Marcu           M       Real    7 8 9 6       Soroca
Bousor  George          M       Arte    7 8 9 10      Taraclia
Bivol   Larisa          F       Arte    6 7 8 10      Rezina
Cerbu   Olga            F       Real    6 7 8 9       Chisinau
Crudu   Valeria         F       Uman    5 6 7 6       Cahul
Grosu   Mihaela         F       Sport   5 6 7 9       Balti
Rata    Remus           M       Sport   8 9 8 10      Balti
```

Sortarea 3:

```
Lista elevilor sortata ascendent dupa profil cu media
examenelor peste 7.00:

Rata    Livia           F       Arte    8 9 7 10      Glodeni
Dascal  Emanuel         M       Arte    9 8 7 10      Soroca
Bousor  George          M       Arte    7 8 9 10      Taraclia
Bivol   Larisa          F       Arte    6 7 8 10      Rezina
Calalb  Marcu           M       Real    7 8 9 6       Soroca
Cerbu   Olga            F       Real    6 7 8 9       Chisinau
Creng   Eugen           M       Sport   5 6 9 10      Rascani
Rata    Remus           M       Sport   8 9 8 10      Balti
```

Studiul individual nr.3

Model de foaie de titlu a lucrării

Studiu individual nr.3 la unitatea de curs
Programare procedurală
cu genericul
Subprograme recursive și module în C/C++

Profesor NUME PRENUME

Nume Prenume	*Grupa*	*Link-ul de la secvența video*	*Nota*

Unități de conținut propuse:

1. *Structuri și subprograme recursive;*
2. *Programare modulară.*

Modalități de evaluare a lucrării individuale:

1. *Comunicare sincronă și asincronă;*
2. *Demonstrare funcționalității produsului program pe un dispozitiv electronic de calcul: smartPhone, smartTab sau notebook (sau PC);*
3. *Crearea unei secvențe video în care elevul explică pas cu pas realizarea tuturor sarcinilor propuse, deci pentru fiecare studiu individual elevul va trebui să realizeze câte o secvență video corespunzătoare și să aplice în comunicarea sa limbajul de specialitate corespunzător unității de conținut;*
4. *În lucrarea finală fiecare elev va trimite și link-ul cu secvența video realizată conform sarcinii propuse.*

Problema 3.1

Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru o prismă regulată heptagonală (7 laturi):

a) *Numărul total de diagonale ale bazei;*
b) *Aria totală:*
c) *Aria laterală;*
d) *Volumul.*

Rezolvare:

Listingul modulului în limbajul C/C++

```
#ifndef PROFESOR_H_INCLUDED
#define PROFESOR_H_INCLUDED
#include <cmath>
#define pi 3.14159265359
// Diagonale: D=n*(n-3)/2
float DiagonaleBaza(int n){
    return (n*(n-3)/2);
}
// Aria laterala: Al=Pb*H
float Aria_laterala(int n,float a,float h){
    return (n*a*h);
}
// Aria totala: At=Al+2*Ab
float Aria_totala(int n,float a,float h){
    return ((n*a*h)+(2*(0.25*n*pow(a,2))*(1/(tan(pi/n)))));
}
// Volumul: V=Ab*H
float Volumul(int n,float a,float h){
    return ((0.25*n*pow(a,2))*(1/(tan(pi/n)))*h);
}
#endif // PROFESOR_H_INCLUDED
```

Listingul programului în limbajul C/C++

```
#include <iostream>
#include "profesor.h"
```

```
using namespace std;
// programul principal
int main(){
    int n=7;
    float a=10.75, h=14.35;
    // Diagonale: D=n*(n-3)/2
    cout<<"Numarul total de diagonale ale bazei este: \t";
    cout<<DiagonaleBaza(n);
    // Aria laterala: Al=Pb*H
    cout<<"\nAria laterala este: \t\t\t\t";
    cout<<Aria_laterala(n,a,h);
    // Aria totala: At=Al+2*Ab
    cout<<"\nAria totala este: \t\t\t\t";
    cout<<Aria_totala(n,a,h);
    // Volumul: V=Ab*H
    cout<<"\nVolumul este: \t\t\t\t\t"<<Volumul(n,a,h);
    cout<<endl;
}
```

Rezultatele execuției programului în mediul de programare CodeBlocks:

```
Numarul total de diagonale ale bazei este:      14
Aria laterala este:                             1079.84
Aria totala este:                               1919.73
Volumul este:                                   6026.2
```

Problema 3.2

Elaborați un program C/C++ care va permite crearea unui fișier antet (header) în care vor fi suprograme pentru a determina pentru două puncte din planul 2D și 3D distanța. Se vor utiliza structuri imbricate.

Rezolvare:

Listingul modulului în limbajul C/C++

```
#ifndef PROFESOR_H_INCLUDED
```

```
#define PROFESOR_H_INCLUDED
#define n 2
#include <cmath>
using namespace std;
// Coordonate puncte în planul 2D
struct plan{
    float x,y;
};
// Coordonate puncte în planul 3D
struct spatiu{
    plan xOy;
    float z;
} a[5];
// Variabile globale
float d[100]; int i,j,k;
// Introducem datele de la tastatura
void Introducem(){
    for(i=1;i<=n;i++){
        cout<<"Coordonatele spatiale "<<i<<": \n";
        cout<<"\tx= ";cin>>a[i].xOy.x;
        cout<<"\ty= ";cin>>a[i].xOy.y;
        cout<<"\tz= ";cin>>a[i].z;
    }
}
// Afisam datele la ecran
void Afisam(){
    for(i=1;i<=n;i++){
        cout<<"Coordonatele spatiale ale punctului "<<i<<": ";
        cout<<"\
t["<<a[i].xOy.x<<","<<a[i].xOy.y<<","<<a[i].z<<"]\n";
    }
}
// Calculam distanta in planul 2D
void Distanta2D(){
    d[1]=sqrt(pow(a[2].xOy.x-a[1].xOy.x,2)+pow(a[2].xOy.y-
a[1].xOy.y,2));
    cout<<"Distanta dintre doua puncte in plan: \t"<<d[1]<<"\
n";
}
// Calculam distanta in planul 3D
void Distanta3D(){
    d[2]=sqrt(pow(a[2].xOy.x-a[1].xOy.x,2)+pow(a[2].xOy.y-
```

```
a[1].xOy.y,2)+pow(a[2].z-a[1].z,2));
    cout<<"Distanta dintre doua puncte in spatiu: \t"<<d[2]<<"\
n";
}
#endif // PROFESOR_H_INCLUDED
```

Listingul programului în limbajul C/C++

```
#include <iostream>
#include "profesor.h"
int main(){
    // Introducem coordonatele spatiale a 2 puncte
    Introducem();
    getchar();system("CLS");
    // Afisam coordonatele spatiale a 2 puncte
    Afisam(); cout<<endl;
    // Afisam distanța dintre 2 puncte în spațiu și în plan
    Distanta3D(); Distanta2D();
    cout<<"\nPentru coordonatele plane, am extras coordonatele
pentru X si Y din cele spatiale!"<<endl;
}
```

Rezultatele execuției programului în mediul de programare CodeBlocks:

```
Coordonatele spatiale ale punctului 1:  [1.34,3.52,5.75]
Coordonatele spatiale ale punctului 2:  [4.75,2.65,3.86]

Distanta dintre doua puncte in spatiu:  3.99463
Distanta dintre doua puncte in plan:    3.51923

Pentru coordonatele plane, am extras coordonatele pentru X si Y
din cele spatiale!
```

BIBLIOGRAFIE

1 A. Gremalschi, I. Mocanu, I. Spinei, L. Gremalschi, „*Informatică - Manual pentru clasa a 10-a* ”, Editura Știința, 2020, 212 pag.
2 Мариус Бансила, „*Решение задач на современном C++*”, 2019, 304 стр.
3 Артур О'Двайр, „*Осваиваем C++17 STL*”, 2019, 354 стр.
4 Алексей Васильев, “*Программирование на C++ в примерах и задачах*”, 2017, 369 стр.
5 Михаил Абрамян, „*Введение в стандартную библиотеку шаблонов C++. Описание, примеры использования, учебные задачи*”, 2017, 400 стр.
6 Антон Спрол, „*Думай как программист. Креативный подход к созданию кода. C++ версия*”, 2017, 274 стр.
7 Daniel Vișan-Dimitriu, „*Limbajul C++ fără profesor, ediția a II-a revizuită pentru Code::Blocks*”, Editura Else, 2016, 118 pag.
8 Tutorial Point, „*Learn C++ programming language*”, 2014, 54 pag.
9 Runceanu A., Runceanu M., „*Noțiuni de programare în limbajul C++*”, Editura Academica Brâncuşi, 2012, 483 pag.
10 Никита Культин, „*C/C++ в задачах и примерах (2-е издание)*”, 2009, 351 стр.
11 J. Saulié, „*C++ Language Tutorial* ”, 2007, 144 pag.
12 Bogdan Pătruţ, Carmen Muraru, „*Aplicații în C și C++*”, Editura EduSoft, 2006, 218 pag.
13 Cerchez E., Şerban M., „*Programarea in limbajul C/C++ pentru liceu* ”, Editura Polirom, 2005, 296 pag.
14 Istrati S. G., „*Iniţializare în limbajele C şi C++* ”, Editura UTM, 2003, 106 pag.
15 Ulla Kirch-Prinz, P. Prinz, „*A Complete Guide to Programming in C++* ”, 2002, 846 pag.

ANEXE

1. MANEVRAREA EXCEPŢIILOR ÎN C++

Este un fenomen "natural" ca în programe să se strecoare erori, de diverse categorii. Activitatea de programare implică și acțiuni mai puțin plăcute, adică testarea, depanarea și corectarea erorilor. Costurile de îndepărtare a erorilor creste de obicei direct proporțional cu întârzierea din cadrul procesului de dezvoltare când sunt descoperite.

Trebuie însă înțeleasă diferența dintre erori (bug-uri) și excepții. Excepțiile sunt situațiile neașteptate apărute în cadrul sistemului care rulează un program. Programele trebuie să fie pregătite pentru a trata aceste situații excepționale. Există două tipuri de excepții: sincrone și asincrone. În C++ s-a realizat un mecanism facil de tratare a excepțiilor. Astfel, o excepție este:

- un obiect a cărui adresă este trimisă din zona de cod, unde a apărut problema către o zonă de cod care trebuie să o rezolve;
- un răspuns la o circumstanță excepțională care apare atunci când se execută un program, cum ar fi o încercare de divizare la zero.

Excepțiile oferă o modalitate de a transfera controlul dintr-o parte a unui program în alta. Tratarea excepțiilor C++ se bazează pe trei cuvinte cheie: încercați, prindeți și aruncați:

a) *throw* - Un program lansează o excepție atunci când apare o problemă. Acest lucru se face folosind un cuvânt cheie throw.
b) *catch* - Un program prinde o excepție cu un handler de excepții la locul unui program în care doriți să rezolvați problema. Cuvântul cheie catch indică captarea unei excepții.
c) *try* - Un bloc try identifică un bloc de cod pentru care vor fi activate anumite excepții. Este urmat de unul sau mai multe blocuri de captură.

Presupunând că un bloc va genera o excepție, o metodă captează o excepție folosind o combinație de cuvinte cheie *try* și *catch*. Un bloc *try / catch* este plasat în jurul codului care ar putea genera o excepție. Codul dintr-un bloc *try / catch* este denumit cod protejat. Puteți enumera mai multe declarații de captură pentru a prinde diferite tipuri de excepții în cazul în care blocul de încercare (*try*) ridică mai multe excepții în situații diferite.

Aruncarea de excepții (throw)
Excepțiile pot fi aruncate oriunde în cadrul unui bloc de cod folosind instrucțiunea throw. Operandul instrucțiunii throw determină un tip pentru excepție și poate fi orice expresie, iar tipul rezultatului expresiei determină tipul de excepție aruncat.

Prinderea excepțiilor (catch)
Blocul de captură care urmează blocului de încercare prinde orice excepție. Puteți specifica ce tip de excepție doriți să prindeți și acest lucru este determinat de declarația de excepție care apare între paranteze după cuvântul cheie catch.

Exemplul 1:

```
#include <iostream>
using namespace std;
// Subprogramul ce permite impartirea a 2 numere intregi
double impartire(int a, int b) {
   if(b==0) {
      throw "Eroare:\nConditia de impartire la zero!";
   }
   return (a/b);
}
// Programul principal
int main () {
    int x=50,y=0; double z = 0.0;
    // Blocul try / catch
    try {
        z=impartire(x,y); cout<<z<<endl;
    }
    catch (const char* mesaj){
        cerr<<mesaj<<endl;
   }
}
```

Deoarece creștem o excepție de tip const char *, în timp ce captăm această excepție, trebuie să folosim const char * în blocul de captură. Dacă compilăm și rulăm codul de mai sus, obținem următorul rezultat:

```
Eroare:
Conditia de impartire la zero!
```

Exemplul 2:

```
#include<iostream>
#include<vector>
using namespace std;
int main() {
    vector<int> v; // cream o variabila v de tip vector
    v.push_back(0); // adaugam componenta 0 la vectorul v
    v.push_back(1); // adaugam componenta 1 la vectorul v
    // Accesarea elementului al 3-lea care nu exista
    try{
        v.at(2);
    }
     // Mesajul de eroare returnat va fi pastrat in variabila
Eroare
    catch (exception &Eroare){
        cout << "Eroare:\nA aparut o exceptie!" << endl;
    }
}
```

Dacă compilăm și rulăm codul de mai sus, obținem următorul rezultat:

```
Eroare:
A aparut o exceptie!
```

Excepții standard în C++

C++ oferă o listă de excepții standard definite în *<exception>* pe care le putem folosi în programele noastre. Aceste excepții sunt incluse în spațiul de nume **std**. Analizați următorul tabel:

Nr.	Excepția	Descrierea
1	std::exception	O excepție și clasa părinte a tuturor excepțiilor C ++ standard.

2	std::bad_alloc	Acest lucru poate fi aruncat de **new**.
3	std::bad_cast	Acest lucru poate fi aruncat prin **dynamic_cast**.
4	std::bad_exception	Acesta este un dispozitiv util pentru a gestiona excepțiile neașteptate într-un program C++.
5	std::bad_typeid	Acest lucru poate fi aruncat de **typeid**.
6	std::logic_error	O excepție care teoretic poate fi detectată prin citirea codului.
7	std::domain_error	Aceasta este o excepție aruncată atunci când este utilizat un domeniu matematic nevalid.
8	std::invalid_argument	Acest lucru este aruncat din cauza unor argumente nevalide.
9	std::length_error	Acest lucru este aruncat atunci când este creat un **std :: string** prea mare.
10	std::out_of_range	Acest lucru poate fi aruncat prin metoda **at**, de exemplu un **std :: vector** și **std :: bitset <> :: operator [] ()**.
11	std::runtime_error	O excepție care teoretic nu poate fi detectată prin citirea codului.
12	std::overflow_error	Aceasta este aruncată, dacă apare o supraîncărcare (overflow) matematică.
13	std::range_error	Acest lucru se întâmplă atunci când încercați să stocați o valoare care este în afara intervalului.
14	std::underflow_error	Acest lucru este aruncat dacă apare o scurgere (underflow) matematică.

2. MODEL DE TESTARE FINALĂ ASISTATĂ DE CALCULATOR

Competenţe specifice la unitatea de curs *"Programare procedurală"*:

1. Prelucrarea tipurilor de date structurate în cadrul aplicaţiilor de consolă;
2. Organizarea programelor la nivel de subprograme şi module;
3. Elaborarea programelor pentru problemele întâlnite în activitatea profesională.

Unităţi de competenţă la unitatea de curs *"Programare procedurală"*:

1. Utilizarea fişierelor în cadrul aplicaţiilor de consolă.
2. Sortarea datelor structurate în cadrul aplicaţiilor de consolă.
3. Prelucrarea tipurilor de date de tip structură în cadrul aplicaţiilor de consolă.
4. Utilizarea pointerilor în cadrul aplicaţiilor de consolă.
5. Utilizarea subprogramelor în cadrul aplicaţiilor de consolă.
6. Utilizarea subprogramelor recursive în cadrul aplicaţiilor de consolă.
7. Organizarea aplicaţiilor de consolă la nivel de module.

Model de examen realizat pe platforma Moodle

Testarea finală (examenul nu va depăşi 180 min, profesorul stabileşte timpul probei teoretice şi al celei practice) va fi asistat de calculator, acesta va include două probe: *Proba teoretică (25 min)* şi *Proba practică (155 min)*.

Proba teoretică va include un test grilă din minim 15 itemi dintr-o bancă de itemi de minim 60 de itemi propuşi spre realizare (din această bancă se itemi se vor selecta cei minim 15 itemi corespunzători fiecărei unităţi de competenţă în mod aleator).

Proba practică va include un test grilă (itemi de tip eseu) din minim 5 itemi (probleme de rezolvat în limbajul de programare C++) dintr-o bancă de itemi de minim 20 de itemi propuşi spre realizare (itemi se vor selecta în acelaşi mod ca la proba teoretică).

Nota finală la această probă de examinare se va calcula astfel: NT = 0.4 * nota obţinută la proba teoretică, NP = 0.6 * nota obţinută la proba practică, astfel nota finală va fi: N = NT + NP.

Exemplu: nota la proba teoretică a fost 8 şi la proba practică - 5, atunci nota finală la examen va fi: 0.4*8 + 0.6*5 = 3.2+3.0 = 6.2 (adică va avea 6). Este important ca fiecare elev la ambele probe să obţină nota minimă de promovare, adică nota 5.

Banca de itemi (model orientativ)[1]

C = Cunoaștere; **A** = Aplicare (înțelegere funcțională); **I** = Integrare (soluționarea problemelor).

	Unităţi de competenţă	*Numărul de itemi după nivele cognitive*			Total itemi
		C (30%)	A (45%)	I (25%)	
1	*Utilizarea fișierelor în cadrul aplicațiilor de consolă.*	7 (1)	5 (1)	8 (1)	56 (11)
2	*Sortarea datelor structurate în cadrul aplicațiilor de consolă.*	7 (1)	5 (2)		
3	*Prelucrarea tipurilor de date de tip structură în cadrul aplicațiilor de consolă.*	7 (1)	5 (2)		
4	*Utilizarea pointerilor în cadrul aplicațiilor de consolă.*	7 (1)	5 (1)		
5	*Utilizarea subprogramelor în cadrul aplicațiilor de consolă.*	7 (1)	10 (3)	8 (1)	49 (9)
6	*Utilizarea subprogramelor recursive în cadrul aplicațiilor de consolă.*	7 (1)	5 (1)		
7	*Organizarea aplicațiilor de consolă la nivel de module.*	7 (1)	5 (1)		
	Total itemi: **Total itemi în test:**	**49** **(7)**	**40** **(11)**	**16** **(2)**	**105** **(20)**

Barem de notare pentru proba teoretică[2]

Proba teoretică include cei 18 itemi teoretici conform nivelurilor cognitive notate cu **C** și **I**. Fiecare item din nivelul **C** va fi apreciat cu 1p (punct), iar cele

1 Guțu V., doctor habilitat în pedagogie, profesor universitar, USM, „Evaluarea rezultatelor academice”, link de acces: http://shorturl.at/DJOW6

2 Anexa 5 din „Ghidul metodologic de elaborare a procedurii de evaluare a calificativelor”, link de acces: http://shorturl.at/dmqZ7

din nivelul **I** - cu 2p (puncte). În total vor fi acumulate: 7+22=29 puncte. Nota poate fi stabilita după *baremul în procente* prezentat mai jos.

ITEMI DE CUNOAŞTERE *(model propus)*

Unitatea de competenţă 1 *(itemi cu răspuns simplu, o variantă corectă)*

1. Clasificarea fişierelor din punct de vedere al accesului:
   A) fişier text, fişier cu tip, fişier fără tip;
   B) fişier cu acces secvenţial, fişier cu acces direct;
   C) fişier de intrare, fişier de ieşire, fişier de intrare/ieşire;
   D) nu este prezentat nici un răspuns corect.
2. Clasificarea fişierelor din punct de vedere al conţinutului:
   A) fişier text, fişier cu tip, fişier fără tip;
   B) fişier cu acces secvenţial, fişier cu acces direct;
   C) fişier de intrare, fişier de ieşire, fişier de intrare/ieşire;
   D) nu este prezentat nici un răspuns corect.
3. Pentru a putea accesa lucrul cu fişiere într-un program este nevoie sa avem declarata directiva...
   A) iostream; C) fstream;
   B) ffstream; D) ifstream.

Unitatea de competenţă 2 *(itemi cu răspuns simplu, A sau F)*

1. Metoda bulelor se bazează pe următoare idee: fie un vector X[] cu n elemente; parcurgem vectorul şi pentru oricare două elemente învecinate care nu sunt în ordinea dorită, le interschimbăm valorile; după o singură parcurgere, vectorul nu se va sorta, dar putem repeta parcurgerea; dacă la o parcurgere nu se face nicio interschimbare, vectorul este sortat.
   A) Adevărat B) Fals
2. Sortarea prin inserţie (Insertion Sort) se bazează pe utilizarea unui vector de frecvenţă. Acest algoritm de sortare poate fi extrem de eficient în anumite situaţii, în funcţie de intervalul de valori al elementelor vectorului. Vom considera că vrem să sortăm un vector v, de lungime n, indexat de la 1 şi cu elementele cuprinse între 0 şi MAX-1.
   A) Adevărat B) Fals

Unitatea de competenţă 3 *(itemi cu răspuns simplu, A sau F)*

1. Nu este posibilă definirea unei structuri ale cărei componente sunt de tipul structură, definite anterior. Câmpurile de tip structură se numesc structuri imbricate (incluse).
   A) Adevărat B) Fals
2. O structură este compusă dintr-un număr de componente de anumite tipuri. Componentele structurii se numesc câmpuri.
   A) Adevărat B) Fals
3. Pentru accesul la câmpurile unei variabile de tip struct se foloseşte operatorul de selecţie directă, notat cu '#', operator cu prioritate maximă.
   A) Adevărat B) Fals

Unitatea de competenţă 4 *(itemi cu răspuns simplu, A sau F)*

1. Asupra pointerilor nu se pot aplica următoarele operaţii: incrementarea şi decrementarea, adunarea cu un număr întreg, diferenţa a doi pointeri.
   A) Adevărat B) Fals
2. La declararea unei variabile de tip pointer, aceasta este iniţializată; valoarea ei este 0 sau o adresă la întâmplare.
   A) Adevărat B) Fals
3. Un pointer este o dată a cărei valoare este o adresă de memorie. O variabilă de tip pointer este o variabilă a cărei valoare este adresa altor variabile.
   A) Adevărat B) Fals

Unitatea de competenţă 5 *(itemi cu răspuns simplu, A sau F)*

1. Corpul funcţiei este un bloc, care implementează algoritmul de calcul folosit de către funcţie.
   A) Adevărat B) Fals
2. Funcţiile comunică prin argumente: ele primesc ca parametri datele de intrare, efectuează prelucrările descrise în corpul funcţiei asupra acestora şi pot returna o valoare (rezultatul, datele de ieşire).
   A) Adevărat B) Fals
3. Funcţiile pot fi descrise doar în cadrul aceluiaşi fişier, asamblarea lor realizându-se cu ajutorul linkeditorului de legături.
   A) Adevărat B) Fals

Unitatea de competență 6 *(itemi cu răspuns simplu, o variantă corectă)*

1. In informatică numim recursivitate indirectă, proprietatea funcțiilor de a se autoapela.
   A) Adevărat B) Fals
2. La apel, se atribuie parametrilor formali valorile parametrilor efectivi, dupp care se execută instrucțiunile din corpul funcției.
   A) Adevărat B) Fals
3. Parametrii folosiți la apelul unei funcții sunt parametri reali, efectivi, concreți, actuali, iar valorile lor vor fi atribuite parametrilor formali, la execuție.
   A) Adevărat B) Fals

Unitatea de competență 7 *(itemi cu răspuns simplu, o variantă corectă)*

1. Programul sursa in limbajul C++ este practic un fisier text care contine implementarea unui algoritm in limbajul C++. Un program in C++ poate contine unul sau mai multe fisiere sursa.
   A) Adevărat B) Fals
2. Prototipurile functiilor nestandard precum si alte declaratii de tipuri si constante simbolice necesare la utilizarea functiilor se gasesc in fisiere antet.
   A) Adevărat B) Fals
3. Un program C++ este alcatuit din una sau mai multe functii, din care una este radacina sau functie principala - adica nu poate lipsi si executia incepe automat cu ea. Aceasta se numeste main.
   A) Adevărat B) Fals

Răspunsuri	1	2	3
Unitatea de competență 1	B	A	C
Unitatea de competență 2	Adevărat	Fals	-
Unitatea de competență 3	Fals	Adevărat	Fals
Unitatea de competență 4	Fals	Adevărat	Adevărat
Unitatea de competență 5	Adevărat	Adevărat	Fals
Unitatea de competență 6	Fals	Adevărat	Adevărat
Unitatea de competență 7	Adevărat	Fals	Adevărat

ITEMI DE APLICARE *(model propus)*

Unitatea de competenţă 1 *(itemi cu răspuns simplu, o variantă corectă)*

Ce va afişa următoarea secvenţă de program?
A) 1
B) 2
C) 3
D) 4

```
#include <iostream>
#include <fstream>
using namespace std;
int main(){
    int a=7324,b=-6;
    ofstream g("Rezultat.txt");
    if (a>b) g<<(a-b)%6;
    else g<<(a*b)%3; g.close();
}
```

Ce va afişa următoarea secvenţă de program?
A) 1
B) 2
C) 3
D) 4

```
#include <iostream>
#include <fstream>
using namespace std;
int main(){
    int n=5,i;
    ofstream g("Rezultat.txt");
    for (i=1;i<=n;i++)
    if (i%3==0) g<<i<<" "; g.close();
}
```

Unitatea de competenţă 2 *(itemi cu răspuns simplu, o variantă corectă)*

Fie vectorul A={47, 23, 12, 17, 30}. Care va fi rezultatul obţinut după a 3-a inserare, în urma aplicării algoritmului metodei inserţiei (InsertionSort) de ordonare ascendentă ?

A) 12 47 23 17 30
B) 23 12 17 30 47
C) 12 17 23 30 47
D) 12 17 23 47 30

Fie vectorul A={47, 23, 12, 17, 30}. Care va fi rezultatul obţinut după prima iteraţie (iteraţia 0) de aplicare a algoritmului metodei selecţiei (SelectionSort) de ordonare ascendentă ?

A) 12 47 23 17 30
B) 12 17 47 23 30
C) 12 17 23 47 30
D) 12 17 23 47 30

Fie vectorul A={47, 23, 12, 17, 30}. Care va fi rezultatul obţinut după prima iteraţie (iteraţia 0) de aplicare a algoritmului metodei bulelor (BubbleSort) de ordonare ascendentă ?

A) 12 47 23 17 30
B) 23 12 17 30 47
C) 12 17 23 30 47
D) 12 17 23 47 30

Unitatea de competență 3 *(itemi cu răspuns simplu, o variantă corectă)*	

Ce va afișa următoarea secvență de program?
A) 7.1
B) 6.2
C) 7.3
D) 6.4
	`#include <iostream>`
	`using namespace std;`
	`struct elev{float m;} e[10],c;`
	`int n=3,i; float z;`
	`int main(){`
	`  e[1].m=7.2; e[2].m=6.3; e[3].m=7.8;`
	`  for(i=1;i<=n;i++){z+=e[i].m;}`
	`  cout<<z*1.0/n<<endl;`
	`}`

Ce va afișa următoarea secvență de program?
A) 16
B) 25
C) 36
D) 49
	`#include <iostream>`
	`#include <cmath>`
	`using namespace std;`
	`struct procente{float y;} p[10],c;`
	`int n=3,j; double z;`
	`int main(){`
	`  p[1].y=8.5; p[2].y=5.7; p[3].y=6.8;`
	`  for(j=1;j<=n;j++){z+=p[j].y;}`
	`  cout<<pow(z*1.0/n,2)<<endl;`
	`}`

Unitatea de competență 4 *(itemi cu răspuns simplu, o variantă corectă)*	

Ce va afișa următoarea secvență de program?
A) 0 & 0
B) 0x407 030 & 1
C) 1 & 0x403 010
D) 0x403 010 & 0x407 030
	`#include <iostream>`
	`using namespace std;`
	`int a=1,*b;`
	`int main(){`
	`  b=&a; a=*b;`
	`  cout<<&a<<" & "<<&b<<endl;`
	`}`

Ce va afișa următoarea secvență de program?
A) 16
B) 25
C) 36
D) 49
	`#include <iostream>`
	`using namespace std;`
	`int v[7]={2,3,7,4,9},d=5,i,h=0;`
	`int main(){`
	`  for (i=0;i<d;i++)`
	`  h+=*(v+i);`
	`  cout<<h<<endl;`
	`}`

Unitatea de competenţă 5 *(itemi cu răspuns simplu, o variantă corectă)*

Ce problemă rezolvă următoarea secvenţăde program?

```
int s (int t[]){
    int i=0;
    while (cin>>t[i]!=EOF)
    i++;
    return i;
}
```

A) citirea datelor unui tablou unidimensional de elemente reale;
B) citirea datelor unui tablou unidimensional de elemente întregi;
C) citirea datelor unui tablou bidimensional de elemente reale;
D) citirea datelor unui tablou bidimensional de elemente întregi.

Ce problemă rezolvă următoarea secvenţăde program?

```
int s (float x[],int k){
    int i,im=0;
    for (i=1;i<k;k++)
    if (x[i]>x[im]) im=i;
    return im;
}
```

A) elementul minim dintr-un tablou unidimensional;
B) elementul maxim dintr-un tablou unidimensional;
C) poziţia elementului maxim dintr-un tablou unidimensional;
D) poziţia elementului minim dintr-un tablou unidimensional.

Unitatea de competenţă 6 *(itemi cu răspuns simplu, o variantă corectă)*

Ce problemă rezolvă următoarea secvenţăde program?

```
int s (long k){
   if (k!=0)
   return (k+s(k-1));
}
```

A) suma numerelor naturale de la 1 la k;
B) suma numerelor întregi negative;
C) suma numerelor pare de la 1 la n;
D) suma numerelor impare de la 1 la n.

Ce problemă rezolvă următoarea secvenţăde program?

```
int s (int k){
   if (k==0) return 0;
   else
   return (s(k/10)+k%10);
}
```

A) produsul cifrelor unui număr;
B) suma cifrelor unui număr;
C) diferenţa cifrelor unui număr;
D) factorialul unui număr.

Unitatea de competență 7 *(itemi cu răspuns simplu, o variantă corectă)*	

Ce problemă rezolvă următoarea secvență de program?
 	
`#ifndef PROFESOR_H_INCLUDED`
`#define PROFESOR_H_INCLUDED`
`int s(int a[],int n){`
`  int m=a[0];`
`  for (int i=0;i<n;i++){`
`  if((m<a[i])&&(a[i]%2!=0))`
`  m=a[i];`
`  }`
`  return m;`
`}`
`#endif`
	A) afișarea elementului minim din vector;
	B) afișarea elementului minim par din vector;
	C) afișarea elementului maxim din vector;
	D) afișarea elementului maxim impar din vector.

Ce problemă rezolvă următoarea secvență de program?
 	
`#ifndef PROFESOR_H_INCLUDED`
`#define PROFESOR_H_INCLUDED`
`int s(int b[],int n){`
`  int mo=b[0];`
`  for (int i=0;i<n;i++){`
`  if (mo>b[i]) mo=b[i];`
`  }`
`  return mo;`
`}`
`#endif`
	A) afișarea elementului minim din vector;
	B) afișarea elementului minim par din vector;
	C) afișarea elementului maxim din vector;
	D) afișarea elementului maxim impar din vector.

- ✔ *Itemii propuși ca model pentru aceste două categorii pot fi modificați de fiecare cadru didactic individual.*
- ✔ *Fiecare cadru didactic poate elabora un număr individual de itemi compatibili pentru categoriile de itemi menționate recent.*
- ✔ *Banca de itemi trebuie să fie cât mai voluminoasă!*

ITEMI DE INTEGRARE *(model propus)*

CATEGORIA 1

1	*Se consideră fişierul text* ***Meteo.txt*** *în care sunt înregistrate pe fiecare linie informaţia despre condiţiile meteorologice de câteva zile, separate de un spaţiu: data, temperatura, umiditatea, direcţia vântului. Să se scrie un program care va realiza următoarele subsarcini:* a) *citeşte informaţia din fişier şi o memorează într-o structură;* b) *sortează toate zilele în ordine descrescătoare după temperatură folosind metoda inserţiei;* c) *crează fişierul text* ***Sort.txt,*** *şi transcrie în ordine pe linii informaţia sortată anterior;* d) *pentru crearea fişierului extern şi prelucrarea datelor conform condiţiei, se vor aplica pointerii.*
2	*Se consideră fişierul text* ***Angajati.in*** *constituit din mai multe linii, fiecare linie conţinând informaţia despre angajaţi: numele; numărul de zile lucrate în această lună; plata pe o zi lucrată. Să se scrie un program care va realiza următoarele subsarcini:* a) *citeşte informaţia din fişier şi o memorează într-o structură;* b) *determina salariul lunar al fiecărui angajat;* c) *crează fişierul text* ***Angajati.out,*** *şi transcrie în ordine pe linii numele angajaţilor în ordine descrescătoare salariului lunar;* d) *pentru crearea fişierului extern şi prelucrarea datelor conform condiţiei, se vor aplica pointerii.*
3	*Se consideră fişierul text* ***Abonament.in*** *constituit din mai multe linii, fiecare linie conţinând informaţia despre telefoanele abonaţilor: numele de familie al abonatului, anul instalării telefonului, numărul telefonului. Să se scrie un program care va realiza următoarele subsarcini:* a) *citeşte informaţia din fişier şi o memorează într-o structură;* b) *determina numărul de telefoane instalate după anul X;* c) *crează fişierul text* ***Abonament.out,*** *şi transcrie în ordine pe linii numele abonaţilor în ordinea instalării telefonului, prin metoda selecţiei;* d) *pentru crearea fişierului extern şi prelucrarea datelor conform condiţiei, se vor aplica pointerii.*

CATEGORIA 2	
1	***a)*** *Să se creeze o unitate de program care va conţine câte un subprogram pentru realizarea următoarelor prelucrări:* ➢ *alocarea dinamică a unui vector* ***A*** *cu* ***n*** *componente de tip char (cuvinte), citite de la tastatură;* ➢ *determinarea poziţiei elementului maximal (cel mai lung cuvânt) din vectorul A, alocat dinamic;* ➢ *calculul iterrativ al numărului componentelor ce conţin cel mult 3 vocale ale vectorului A, alocat dinamic;* ***b)*** *Să se scrie un program care va utiliza (apela) subprogramele unităţii create în punctul* ***a)****.*
2	***a)*** *Să se creeze o unitate de program care va conţine câte un subprogram pentru realizarea următoarelor prelucrări:* ➢ *alocarea dinamică* ***a unei matrici pătratică M cu NxN*** *componente de tip caracter, citite de la tastatură;* ➢ *determinarea numărului de elemente (cuvinte) din fiecare coloană ce conţin minim 3 caractere din matricea M, alocată dinamic;* ➢ *calculul iterativ al cuvintelor ce conţin un număr par de caractere din matricea M, alocată dinamic;* ***b)*** *Să se scrie un program care va utiliza (apela) subprogramele unităţii create în punctul* ***a)****.*
3	***a)*** *Să se creeze o unitate de program care va conţine câte un subprogram pentru realizarea următoarelor prelucrări:* ➢ *alocarea dinamică* ***a unei matrici pătratică M cu NxN*** *componente numere reale, citite de la tastatură;* ➢ *determinarea numărului de elemente mai mici decât 21 din fiecare rând al matricii M, alocată dinamic;* ➢ *calculul recursiv al sumei componentelor nenegative ale fiecărei coloane a matricii M, alocată dinamic;* ***b)*** *Să se scrie un program care va utiliza (apela) subprogramele unităţii create în punctul* ***a)****.*

Barem de corectare şi notare pentru proba practică

Nr.	Motivarea punctajului acordat	Puncte	Total
1	***1.1. Scrierea programului principal (main):*** *Structură corectă a funcţiei principale.* *Descriere corectă a antetului funcţiei.* *Declarare corectă a variabilelor locale.* *Apeluri corecte a subprogramelor.*	 1 1 1 1	4p
	1.2. Citirea datelor din fişierul de intrare (item1.in): *Aplicare corectă a tipului articol.* *Organizare adecvată a tabloului de alocare a datelor de citit.* *Citire corectă a datelor din fişierul de intrare.*	 1 1 1	3p
	1.3. Sortarea elementelor vectorului specificat: *Algoritm recursiv/iterativ corect.* *Transmiteri adecvate de parametri dintre subprograme.* *Algoritm conformat cerinţelor specificate.* *Returnare corectă a rezultatului calculat.*	 2 2 1 1	6p
	1.4. Transcrierea rezultatelor în fişierul de ieşire (item1.out): *Valori corecte pentru parametrii de ieşire;* *Formate corecte de scriere a valorilor de ieşire.*	 1 1	2p
2	***2.1. Alocarea dinamică a vectorului citit de la tastatură:*** *Descriere adecvată a antetului funcţiei de citire (Input);* *Declarare corectă a variabilelor locale funcţiei de citire;* *Alocare adecvată a necesarului de memorie dinamică;* *Alocarea dinamică corectă a vectorului preconizat.*	 1 1 1 1	4p
	2.2. Calcule iterative a datelor alocate dinamic: *Declarare adecvată a entităţilor locale subprogramului;* *Algotitm corect de calcul;* *Interfaţă corectă a funcţiei cu exteriorul.*	 1 2 1	4p
	2.3. Calcule recursive a datelor alocate dinamic: *Declarare adecvată a entităţilor locale subprogramului;* *Algotitm corect de calcul;*	 1 2	4p

Nr.	Motivarea punctajului acordat	Puncte	Total
	Interfaţă corectă a funcţiei cu exteriorul.	1	
	2.4. Scrierea fişierului antet al utilizatorului: *Structura generală corectă a unităţii de program;* *Alocarea corectă a textelor subprogramelor în cadrul unităţii.*	 1 1	**2p**
	2.5. Programul principal: *Structura generală corectă a programului principal;* *Includerea adecvată a fişirelor antet (headerelor);* *Descrierea adecvată a variabilelor locale;* *Apeluri corecte ale funcţiilor din biblioteca utilizatorului.*	 1 1 1 3	**5p**
	Punctaj total acumulat:	**35**	

Nota	10	9	8	7	6	5	4	3	2	1
Punctaj	35-33	32-31	30-27	26-22	21-17	16-12	11-7	6-4	3-2	1-0

NOTIŢE