**Ce sunt funcțiile în C++?**

Funcțiile sunt *instrumente* ale limbajului C++/C (dar nu numai) care permit împărțirea unui cod în mai multe secvențe unitare. Funcțiile pot avea două roluri importante:

* Să împartă codul în unități ușor de urmărit pentru om (util în special în cazul programelor complexe, unde lucrează mai multe persoane);
* Să evite repetarea unei secvențe de cod în mai multe locuri.

**Exemplu**

Să luăm următorul program, care citește un număr natural și care calculează numărul de divizori al oglinditului său.

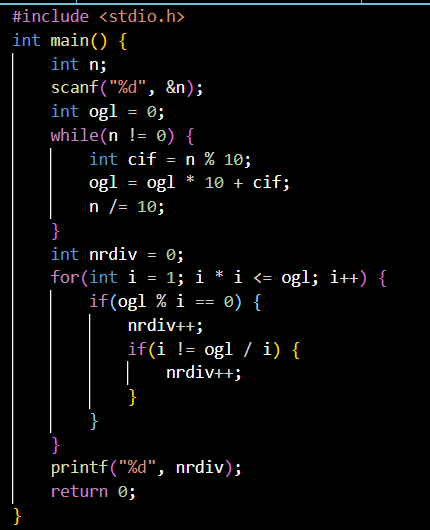


Figura 1. Exemplu fără funcții

Programul funcționează corect, însă codul poate fi cam greu de înțeles. Trebuie să analizăm cu atenție codul pentru a înțelege exact de unde până unde se calculează oglinditul și de unde până unde se calculează numărul de divizori. Aceste două calcule s-ar putea îngloba în două funcții separate, transformând codul într-unul mai logic:

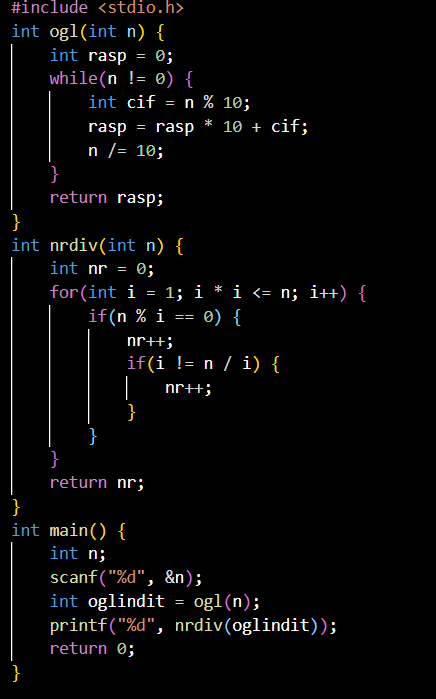


Figura 2. Privire de ansamblu asupra implementării cu funcții

În acest caz, în main, codul este mult mai intuitiv: calculăm oglinditul într-o variabilă oglinidit, și afișăm pe ecran numărul său de divizori.

**Cum se construiește o funcție**

O funcție este alcătuită din mai multe părți.

**Antetul funcției**

Toate funcțiile au un antet, care arată cam așa:



Figura 3. Declarare

Și care ne oferă următoarele informații:

* ce tip de funcție este (int, bool, char, …, sau chiar void);
* cum se numește funcția (asemănător cu un nume de variabilă, care ne ajută să apelăm funcția mai târziu în programul nostru);
* care sunt parametrii funcției:
* parametrii funcției precizează care sunt valorile care se prelucrează în funcție;
* vom vorbi mai încolo despre importanța acestora.

Spre exemplu, următorul antet ar fi unul reprezentativ pentru calcularea sumei a două numere:



Figura 4. Exemplu

Mai exact, funcția se numește suma, are doi parametri numere întregi (a și b — iar pentru fiecare în parte trebuie să specificăm tipul lor, deoarece acestea nu trebuie să fie ambele de același tip). De asemenea, funcția ar trebui să ne furnizeze suma numerelor, reprezentată tot printr-un int.

**Corpul funcției**

Corpul funcției reprezintă un set de instrucțiuni, delimitate de acolade ({}), care precizează pașii care trebuie executați.

Orice variabilă declarată în funcție — inclusiv variabilele din parametri — sunt variabile locale și temporare, adică chiar dacă le vom modifica valorile între timp, acestea nu vor afecta valorile inițiale ale variabilelor din main.

Rezultatul funcției pe care noi vrem să îl furnizăm se poate face cu instrucțiunea return rezultat;, unde rezultat reprezintă valoarea ca rezultat. Tipul de date al variabilei rezultat trebuie să fie tot unul cu tipul funcției.

**Atenție:** După ce se apelează return într-o funcție, toate celelalte operații de după nu se mai execută.

Spre exemplu, pentru o funcție care calculează suma a două numere, vom compune astfel funcția:

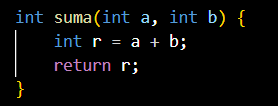


Figura 5.Exemplu returnare în funcție

Sau, direct:

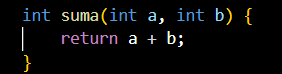


Figura 6.Exemplu returnare în funcție

**Apelarea funcției**

Pentru a apela o funcție, trebuie să menționăm numele funcției, iar între paranteze, să completăm parametrii dați. Spre exemplu, în int main, am putea scrie:



Figura 7.Exemplu apelare funcție

Observăm că putem să furnizăm ca parametri atât variabile (n), cât și valori efectiv (4). Un alt aspect important este faptul că dacă am fi modificat în funcție valoarea primului parametru (care a luat valoarea lui n), n-ul nu și-ar fi modificat valoarea.

**Funcții cu parametri vectori (tablouri unidimensionale) în C++/C**

Pe lângă variabile, putem să oferim ca parametru tablouri (vectori sau matrici). Iată un cod în C++ /C care calculează suma numerelor unui vector:

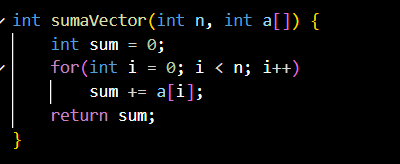


Figura 8.Exemplu sumă vector

**Atenție**: spre deosebire de variabile, dacă vectorul din parametru se modifică în timpul funcției, atunci și vectorul inițial furnizat se modifică. Acest lucru se datorează faptului că un vector este defapt un pointer.

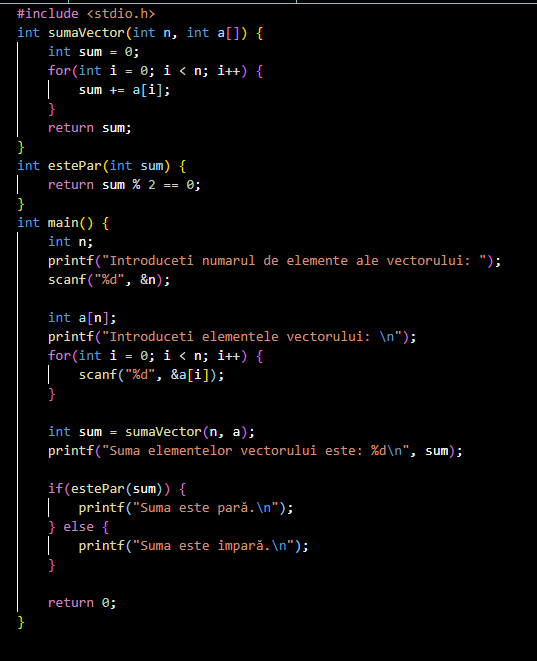


Figura 9 .Exemplu utilizare sumă vector

Putem extinde ideea vectorilor și în 2D, oferind ca parametru matrici.

**Valori prestabilite pentru parametri**

Dacă avem o funcție cu o listă de parametri, putem să specificăm o valoare implicită pentru acestea, în cazul în care nu se furnizează niciuna. Spre exemplu, următoarea funcție calculează suma numerelor a și b, dar dacă b nu este furnizat, se face suma dintre a și 0 (adică se returnează a):

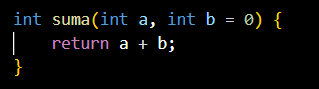


Figura 10.Specificare valori

**Atenție:** Este greșit să furnizăm o valoare prestabilită pentru un parametru, iar pentru parametrii de după să nu dăm nicio valoare. Următorul antet al unei funcții din C++ va da o eroare:

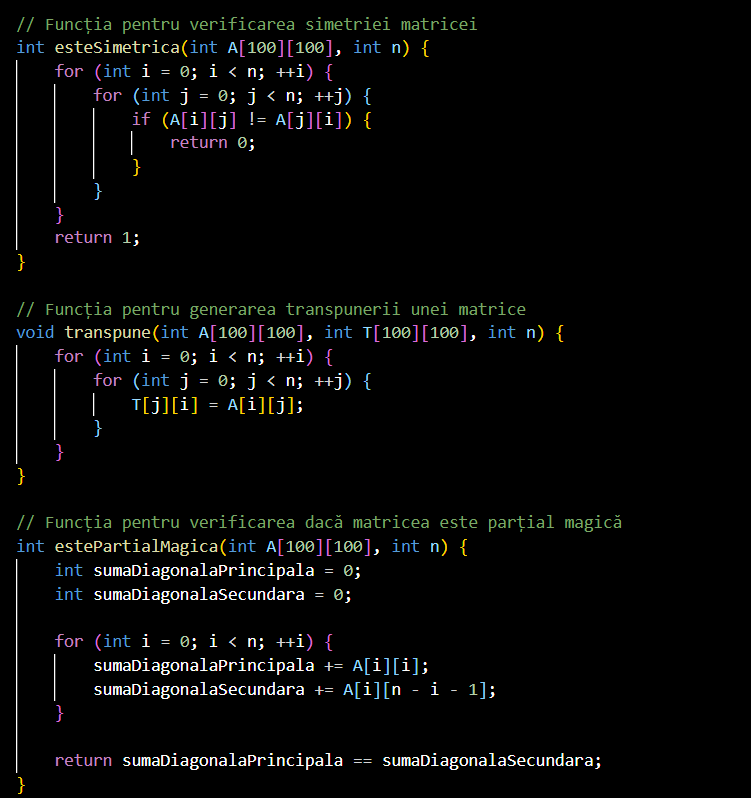


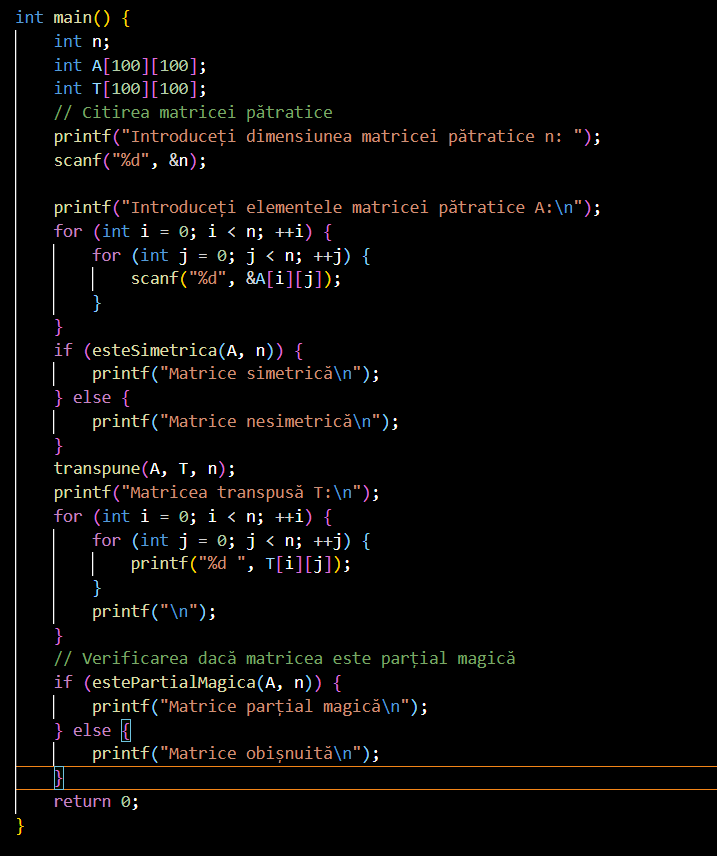
Figura 11. Exemplu greșit

1.Să se realizeze un program în C care să conțină trei funcții, fiecare responsabilă pentru o operație specifică asupra unei singure matrice pătratice, și să le apeleze în funcția main. Programul va efectua următoarele:

1. **Verificarea simetriei unei matrice pătratice:**
   * Programul va verifica dacă matricea este simetrică față de diagonala principală, adică dacă A[i][j] == A[j][i] pentru toate perechile de indici i și j.
   * Se va afișa un mesaj: "Matrice simetrică" dacă matricea este simetrică, altfel "Matrice nesimetrică".
2. **Transpunerea unei matrice:**
   * Programul va genera transpunerea acestei matrice într-o altă matrice T, astfel încât T[j][i] = A[i][j].
   * Se va afișa matricea transpusă T, care va avea dimensiunea n × n.
3. **Verificarea unei matrice pătratice pentru proprietatea de a fi parțial magică:**
   * Programul va calcula suma elementelor de pe diagonala principală și suma elementelor de pe diagonala secundară.
   * Se va verifica dacă cele două sume sunt egale.
   * Se va afișa "Matrice parțial magică" dacă sumele sunt egale, altfel "Matrice obișnuită".

Programul va utiliza funcții pentru fiecare operație, iar verificările vor returna int pentru a indica adevărat (1) sau fals (0).

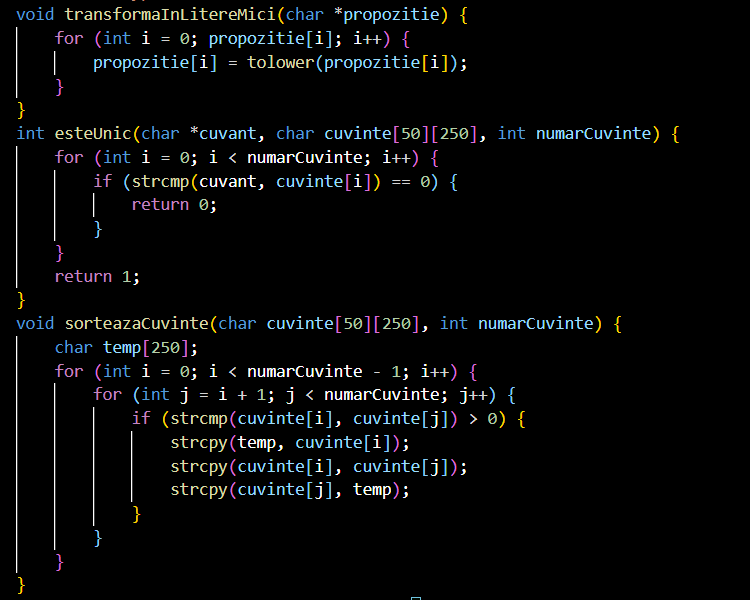


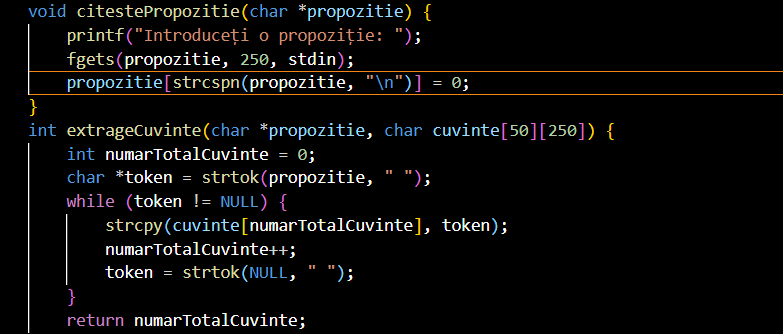


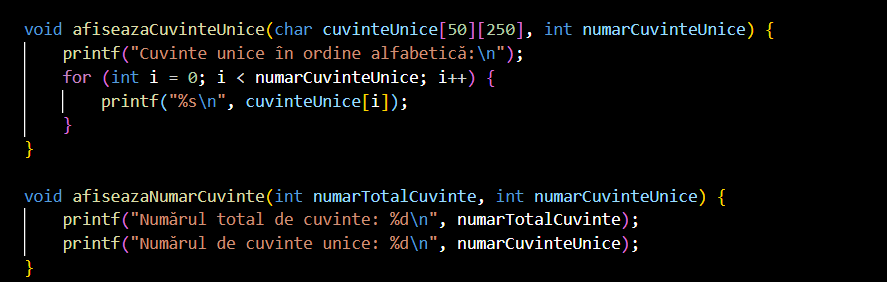
2.Să se scrie un program în C care să conțină funcții pentru fiecare dintre următoarele operații:

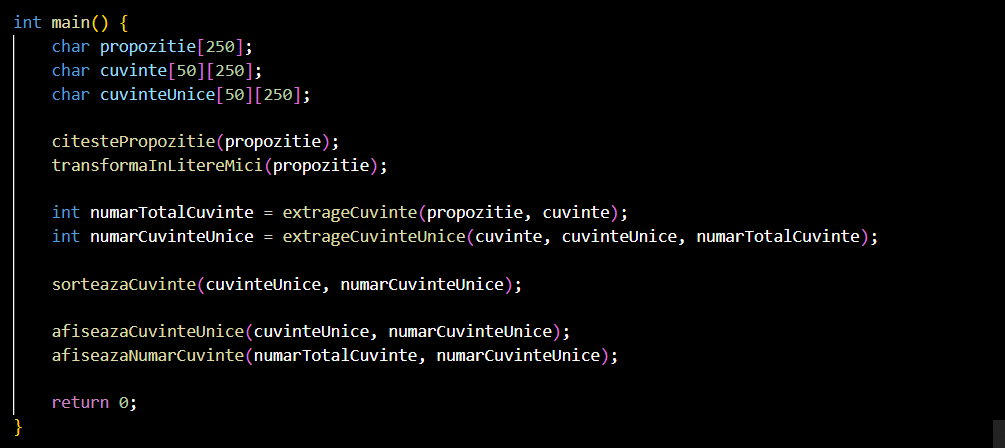
1. **Citirea și transformarea propoziției:**
   * Funcția va citi o propoziție de la tastatură, formată din cuvinte despărțite prin spații (maxim 250 de caractere).
   * Va transforma toate literele în litere mici.
2. **Extracția și stocarea cuvintelor:**
   * Funcția va extrage cuvintele din propoziție folosind strtok și le va salva într-un vector de șiruri.
3. **Afișarea cuvintelor unice:**
   * Funcția va afișa toate cuvintele unice (fără dubluri) în ordine alfabetică (lexicografică).
4. **Calculul și afișarea numărului de cuvinte:**
   * Funcția va calcula și afișa numărul total de cuvinte și numărul de cuvinte unice.

Programul va apela aceste funcții în main pentru a realiza operațiile cerute.









TEMA - De făcut prin definirea unor funcții separate pentru fiecare operație - ÎN C ȘI C++

1. **Vectori:**
   * **Problema:** Implementarea unui algoritm de sortare și căutare într-un vector.
     + Funcția sortareVector(int v[], int n) care sortează elementele unui vector folosind algoritmul quicksort.
     + Funcția cautareBinara(int v[], int n, int x) care caută un element într-un vector sortat folosind căutarea binară.
     + În main(), citește un vector, îl sortează și caută un element dat.
2. **Matrici:**
   * **Problema:** Calculul determinantului unei matrice și verificarea inversabilității.
     + Funcția determinant(int m[][N], int n) care calculează determinantul unei matrice folosind metoda de eliminare Gauss.
     + Funcția esteInversabila(int m[][N], int n) care verifică dacă o matrice este inversabilă (determinantul este diferit de zero).
     + În main(), citește o matrice, calculează determinantul și verifică inversabilitatea.
3. **Șiruri de caractere:**
   * **Problema:** Implementarea unui algoritm de criptare și decriptare a unui șir de caractere.
     + Funcția cripteaza(char s[], int key) care criptează un șir de caractere folosind o cheie dată (de exemplu, cifrul lui Caesar).
     + Funcția decripteaza(char s[], int key) care decriptează un șir de caractere folosind aceeași cheie.
     + În main(), citește un șir de caractere, îl criptează și apoi îl decriptează pentru a verifica corectitudinea.