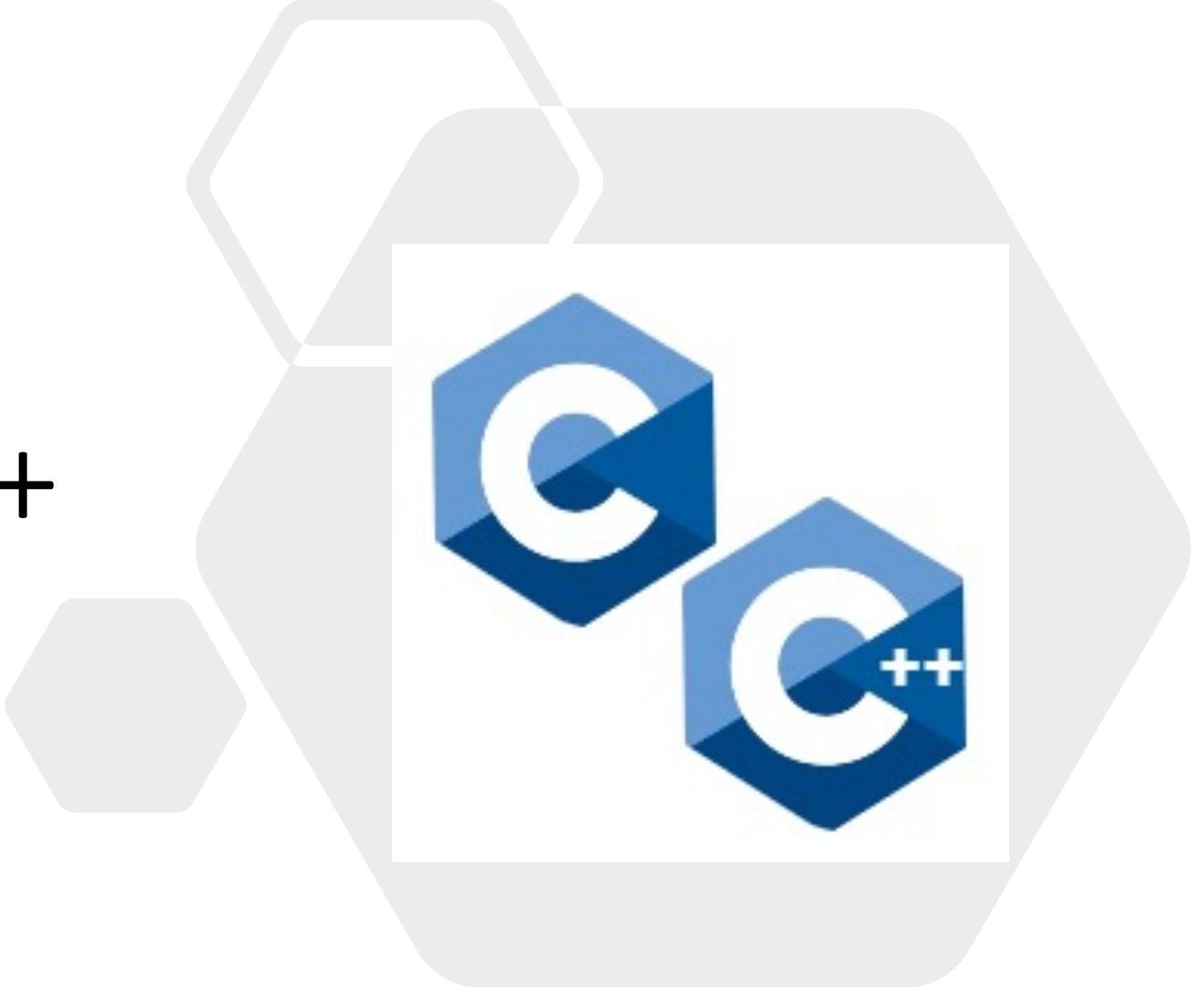


Programare II

Limbajul C/C++

CURS 2



Curs anterior

- ❑ Paradigme de programare
- ❑ Particularități limbaj
 - ❑ Tipuri de date & variabile
 - ❑ Expresii & operatori
 - ❑ Instrucțiuni
 - ❑ Condiționale
 - ❑ Repetitive
 - ❑ Funcții de IO standard
 - ❑ Citirea de la tastatură
 - ❑ Scrierea pe ecran
- ❑ Bibliografie
 - ❑ Kernighan B. and D. Ritchie - The C Programming Language - capitolele 2 si 3

Curs curent

❑ Funcții

- ❑ Prototip
- ❑ Definire
- ❑ Convenția de apel

❑ Macro-definiții

❑ Bibliografie

- ❑ Kernighan B. and D. Ritchie - The C Programming Language - capitolul 4

Functii

❑ Ce este o funcție?

❑ Un **grup de instrucțiuni** care pot fi **identificate** printr-un nume.

❑ De ce să folosim funcții într-un program?

❑ **Refolosirea** unor bucăți de cod

❑ **Simplifică** înțelegerea programelor

❑ Folosirea unui **design top-down**

❑ Descompunerea unei probleme în probleme mai mici până când acestea pot fi rezolvate ușor

❑ Din ce este format un program C?

❑ O multime de funcții definire de utilizator, fiecare

❑ Rezolvând o problemă

❑ Apelată/apelează alte funcții

Tipuri de funcții

❑ De bibliotecă

- ❑ Ex: printf(), scanf(), getch(), ...
- ❑ Funcții matematice, IO, siruri de caractere ...
- ❑ Prototipurile/interfața disponibile în fișiere header .h
 - ❑ Conțin doar decalații de funcții nu și corpul funcției

❑ Definite de utilizator

- ❑ Ex: main()

Terminologie

Descrie modul în care poate fi apelată funcția, trebuie specificat tipul parametrilor, numele lor nu este obligatoriu

- void sau orice tip de date implicit sau definit de utilizator
- Dacă tipul de return lipsește din declarația funcție atunci este considerat implicit int

Terminologie	Sintaxă	Exemplu
Prototip funcție	tipDeReturn numeFunctie ([listaDeParametriFormali]);	int max (int, int);
Definire funcție	tipDeReturn numeFunctie ([listaDeParametriFormali]) { //Bloc de instrucțiuni }	int max (int a, int b){ return a>b?a:b; }
Apel/invocare funcție	numeFunctie(listaDeParametriActuali);	max(4,6);

Coresponde cu numărul și tipul parametrilor din declarație

Prototipul funcție

- ❑ Spune (informează) compilatorului
 - ❑ Numărul și tipul parametrilor funcției
 - ❑ Tipul valorii care va fi returnată
- ❑ Când/unde este utilizat?
 - ❑ Fișiere header .h
 - ❑ Utilizatorul nu are nevoie să știe detaliile de implementare
 - ❑ Apelarea unei funcții înainte de a fi definite
 - ❑ **Modul standard** de specificare a funcțiilor

```
#include <stdio.h>
#include <stdlib.h>
int main(int, int);
void random_nr( int k);
int main(void)
{
    ...
    random_nr(max (4, 6));
}
int max(int a, int b)
{
    ...
}
void random_nr(int k)
{
    ...
}
```

Cuvintele cheie void & return

- ❑ void
 - ❑ Specifica explicit că o funcție nu primește argumente
 - ❑ Specifica că o funcție nu returnează o valoare (nu are valoare de return)
- ❑ return
 - ❑ Când se execută instrucțiunea **controlul** programului este transferat **funcției apelante**
 - ❑ Dacă este urmat de o expresie se returnează valoarea expresiei
- ❑ return vs. exit ()
 - ❑ exit (expr) – Întreruperea execuției programului **controlul** este transferat **sistemului de operare**

Apelul funcțiilor

- Se apelează prin nume și o listă de parametri care corespunde cu tipul și numărul argumentelor din prototip

- Tipuri de apel

- Prin valoare

- Se realizează o copie a variabilelor specificate în apel -> nu se pot modifica valorile variabilelor din funcția apelantă în funcția apelată

- Prin referință

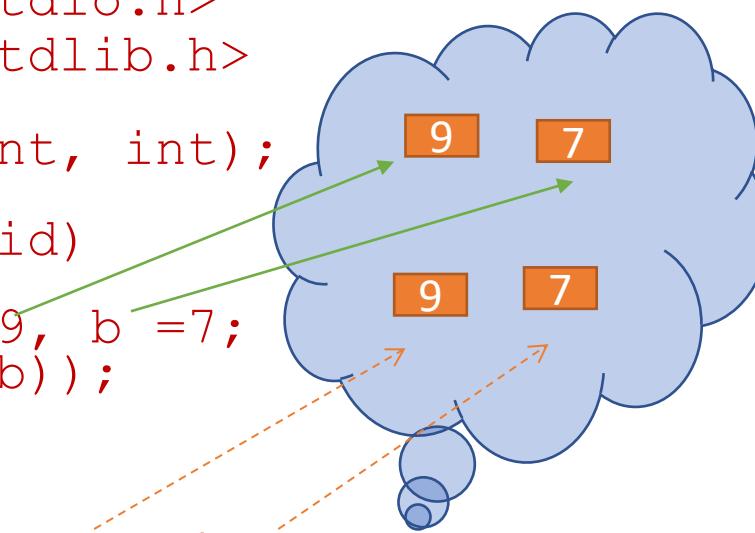
- *Se va discuta la cursul următor, după introducerea noțiunilor despre pointeri*

```
#include <stdio.h>
#include <stdlib.h>

void swap(int, int);

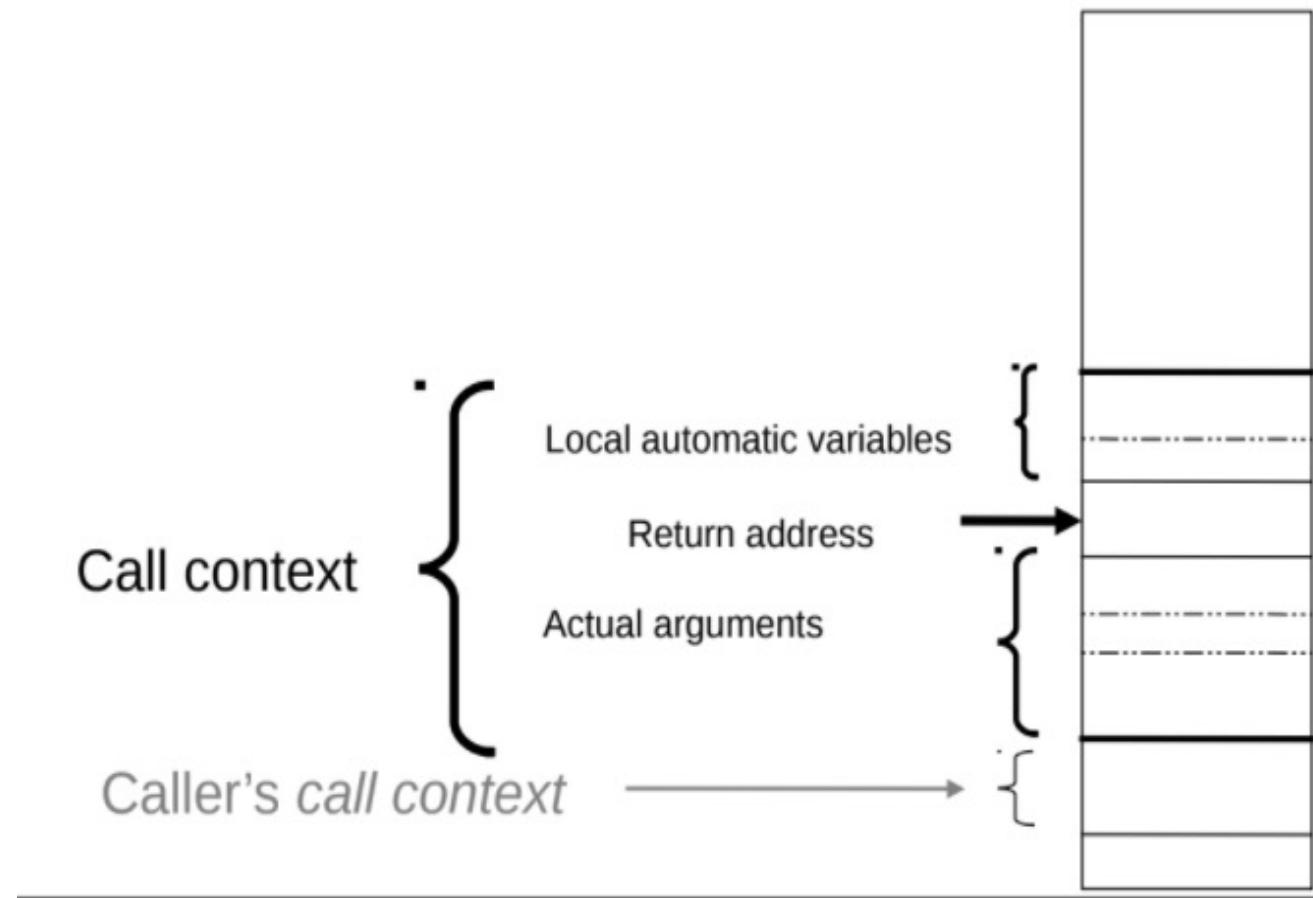
int main(void)
{
    int a = 9, b = 7;
    max (a, b));
}

void swap(int a, int b)
{
    int aux = a;
    a = b;
    b=aux;
}
```

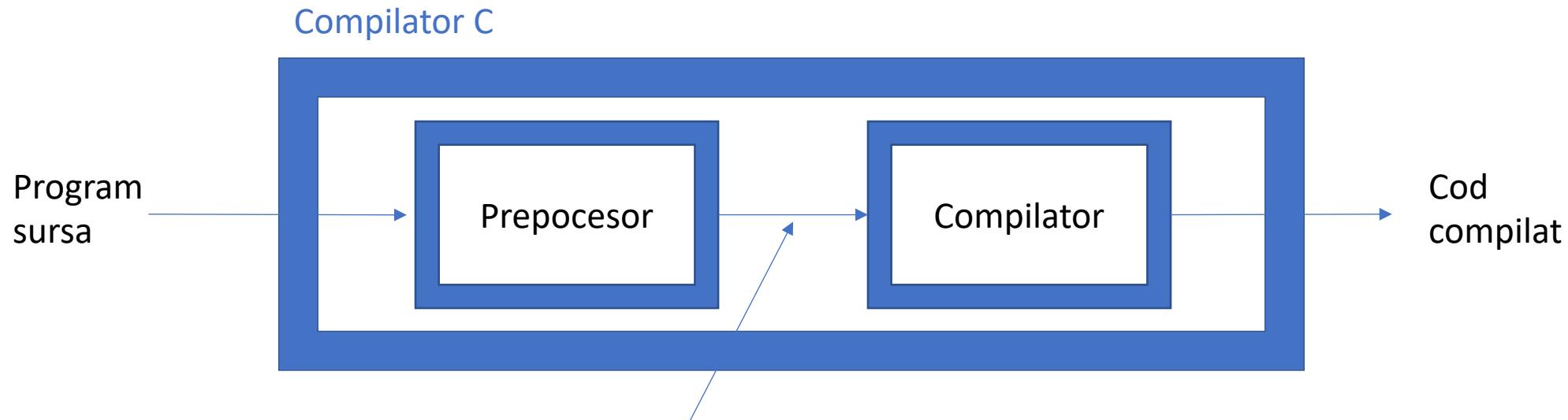


Context de apel

- ❑ Este o zonă de stocare pe stivă care conține
 - ❑ Valorile argumentelor actuale care sunt copiate (*în ordine inversă*)
 - ❑ Adresa de return a funcției
 - ❑ Variabile automate ale funcției apelate



Preprocesorul C



- ❑ Expandare cod
 - ❑ Expandează cod
 - ❑ Elimină comentariile înlocuindu-le cu spațiu
 - ❑ Directive de procesare
 - ❑ Includerea de fișiere #include
 - ❑ Macrodefiniții #define
 - ❑ Compilare condiționată #if, #ifdef, ...

#include

- ❑ Specifică preprocesorului că ar trebui să citească un anume fișier
- ❑ Includere recursivă
 - ❑ Fișierele incluse în fișierul inclus sunt incluse și ele
- ❑ Tipuri
 - ❑ #include <numeFisier>
 - ❑ Caută fișierul într-o listă predefinită de directoare
 - ❑ Lista poate fi extinsă
 - ❑ #include "numeFisier"
 - ❑ Caută fișierul relativ la calea curentă
- ❑ Recomandare
 - ❑ Nu includeți fișiere sursă (ex. cu extensia .c)

Macrodefiniții

❑ Este un simbol recunoscut de procesor, **înlocuit cu corpul macrodefiniției**

❑ Sintaxă

❑ `#define identificator corp_macro`

❑ Exemple

```
#define MAX_INT 32000;  
#define Pi 3.1216;  
#define MAX (x, y) x>y?x:y;
```

Macrodefiniții

□ Observații

- Mai rapide decât apelul de funcții
 - Codul rezultat mai mare -> poate duce la pierderea vitezei

□ Nu validează tipul argumentelor

- Ex: MAX (3, 4) ; MAX ("AZI", "PLOUA")

□ Atenție la ordinea operațiilor

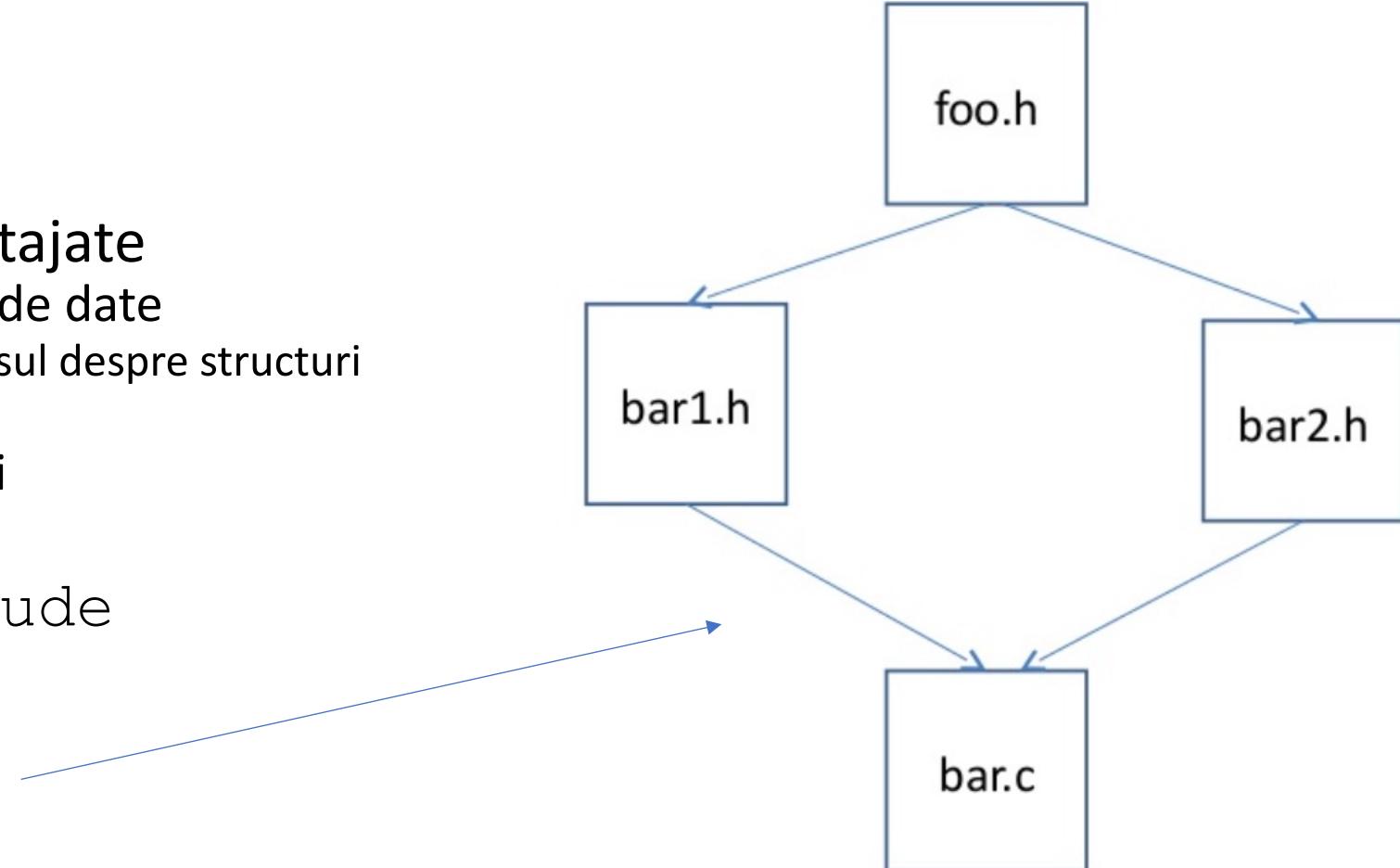
```
#define PATRAT(x) x*x  
A=10; B=9; PATRAT(A+B);
```

□ Pot **evalua de mai multe ori argumentele**

```
#define DUBLU(x) x+x  
A=10; DUBLU(A++);
```

Fișiere header

- ❑ Au extensia .h
- ❑ Conțin declarații partajate
 - ❑ `typedef` – definiții de date
 - ❑ Se va discuta în cursul despre structuri
 - ❑ Macrodefiniții
 - ❑ Prototipuri de funcții
- ❑ Referite prin `#include`
- ❑ Probleme
 - ❑ Includerea multiplă



Compilarea condiționată

- ❑ Conținutul va fi inclus dacă identificatorul a fost definit

```
#ifdef identificator  
...  
#endif
```

- ❑ Conținutul va fi inclus dacă identificatorul nu a fost definit

```
#ifndef identificator  
...  
#endif
```

- ❑ Exemplu utilizare - debug

Includere multiplă fișiere header

```
#ifndef identificator  
#define identificator  
...  
#endif
```

Despre ce vom discuta?

- ❑ Pointeri constanți
 - ❑ Tablouri
- ❑ Alocarea dinamică a memoriei
- ❑ Pointeri la funcții
- ❑ Siruri de caractere
- ❑ Bibliografie
 - ❑ Kernighan B. and D. Ritchie - The C Programming Language - capitolul 5



ÎNTREBĂRI