INFORMATICĂ PENTRU ADUCERE LA NIVEL

FMI

Sem. I, anul I

Cursul 1 / 08 octombrie 2024

CUPRINSUL CURSULUI 1

1. Prezentarea cursului
2. Primul curs

UTILITATEA CURSULUI DE PC (PROGRAMAREA CALCULATOARELOR)

PP (Programare procedurală):

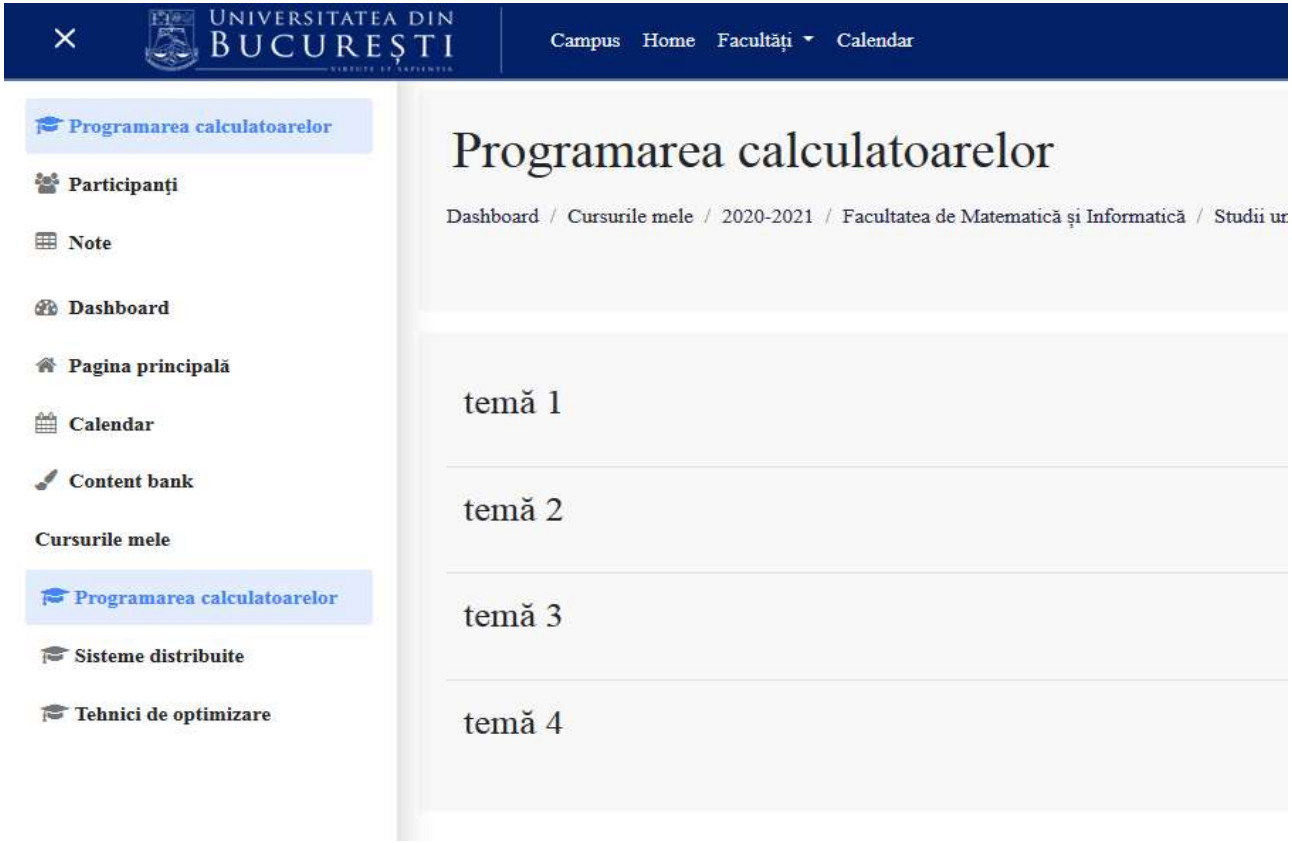
- paradigma de programare bazată pe conceptul de apel de procedură/funcție/rutină/subrutină;
- Un program este privit ca o mulțime ierarhică de funcții care manipulează datele.

Vom studia **limbajul C**:

- limbaj fundamental de programare (1970), exponent al programării procedurale;
- Alte limbaje (C++, Java, PHP, Python) împrumută multe din caracteristicile limbajului C.

MATERIALE

MSTeams si/sau moodle.unibuc.ro



OBIECTIVELE CURSULUI

1. Formarea deprinderilor de programare structurată (modularizare) în limbaje de programare clasice si moderne (descompunerea unei probleme complexe în subprobleme relativ simple și independente);
2. Însușirea caracteristicilor limbajului C:
   - alocarea memoriei,
   - lucrul cu pointerii,
   - lucrul cu fișierele,
   - programarea generică.

(să codați în C, să vă dați seama ce face un cod scris de altcineva, să depanați cod în C)

3. Dezvoltarea unei gândiri algoritmice + abilitate de programare - foarte utile în rezolvarea diverselor probleme cu care vă veți întâlni în facultate sau în viața reală.

PROGRAMA CURSULUI

❑Introducere

- Algoritmi
- Limbaje de programare.
- Introducere în limbajul C. Structura unui program C.

❑Fundamentele limbajului C

- Etapele realizării unui program C.
- Tipuri de date fundamentale. Variabile. Constante. Operatori. Expresii. Conversii.
- Tipuri derivate de date: tablouri, șiruri de caractere, structuri, uniuni, câmpuri de biți, enumerări, pointeri
- Instrucțiuni de control
- Directive de preprocesare. Macrodefiniții.
- Funcții de citire/scriere.

❑Fișiere text

- Funcții specifice de manipulare.

❑Funcții (1)

- Declarare și definire. Apel. Metode de trasmitere a paramerilor. Pointeri la funcții.

❑ Tablouri și pointeri

- Legătura dintre tablouri și pointeri
- Aritmetica pointerilor
- Alocarea dinamică a memoriei
- Clase de memorare

❑ Șiruri de caractere

- Funcții specifice de manipulare.

❑ Fișiere binare

- Funcții specifice de manipulare.

❑ Structuri de date complexe și autoreferite

- Definire și utilizare

❑ Funcții (2)

- Funcții cu număr variabil de argumente.
- Preluarea argumentelor funcției main din linia de comandă.
- Programare generică.

BIBLIOGRAFIE

1. Kernighan & Ritchie: The C programming language
http://zanasi.chem.unisa.it/download/C.pdf

2. Kernighan & Ritchie: Limbajul C
Editura Teora, 2003

3. Liviu Negrescu: Limbajele C si C++ pentru începători, volumul 1, partea I si II (Limbajul C), Editura Albastra, 2001

4. Herbert Schildt: C, manual complet. Editura Teora, 2000?

5. Stephan Prata: C primer plus, 6th Edition
https://vk.com/doc190970339_430409589?hash=2d2b4245bd65b25e27&dl=cd4e96f98aeddd5c1e

REGULAMENT DE EVALUARE ȘI NOTARE

Test la sfârșitul semestrului

Nota minimă 5

CUPRINSUL CURSULUI DE AZI

1. Prezentarea cursului

2. Primul curs

CURSUL 1:

1. Algoritmi
2. Limbaje de programare
3. Introducere în limbajul C. Structura unui program C.

ALGORITMI

Rezolvarea oricărei probleme implică mai multe etape:

1. Analiza problemei
2. Găsirea soluției [optime]
3. Elaborarea algoritmului
4. Implementarea algoritmului într-un limbaj de programare
5. Verificarea corectitudinii algoritmului propus
6. Analiza complexității

ALGORITMI

Algoritm = *o succesiune finită, ordonată și bine definită (exprimată clar și precis) de operații executabile (instrucțiuni, pași) care constituie o metodă corectă de rezolvare a unei probleme pornind dintr-o stare inițială, folosind datele disponibile și ajungând în starea finală dorită*.

REPREZENTAREA ALGORITMILOR

1. Pseudocod/ limbaj natural
2. Schemă logică
3. Program într-un limbaj de programare

1. PSEUDOCOD

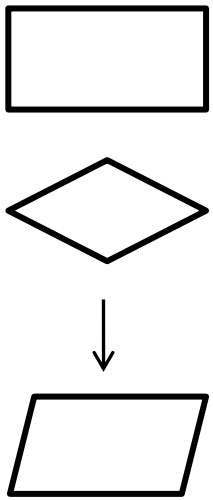
- ✓ limbaj natural structurat exprimat formal
- ✓ fiecare pas al algoritmului este reprezentat de o linie separată, ca o propoziție
- ✓ acțiuni (verbe) aplicate unor date (substantive)
- ✓ indentarea poate reda ierarhia instrucțiunilor

Algoritmul lui Euclid prin scăderi repetate

```
cât timp B != A execută
   dacă A > B atunci
         A = A – B;
    altfel
         B = B – A;
   sfârșit dacă
sfârșit cât timp
afișează A
```

2. SCHEMĂ LOGICĂ

- alăturare de simboluri vizuale care desemnează fluxul logic al pașilor



Bloc de instrucțiuni

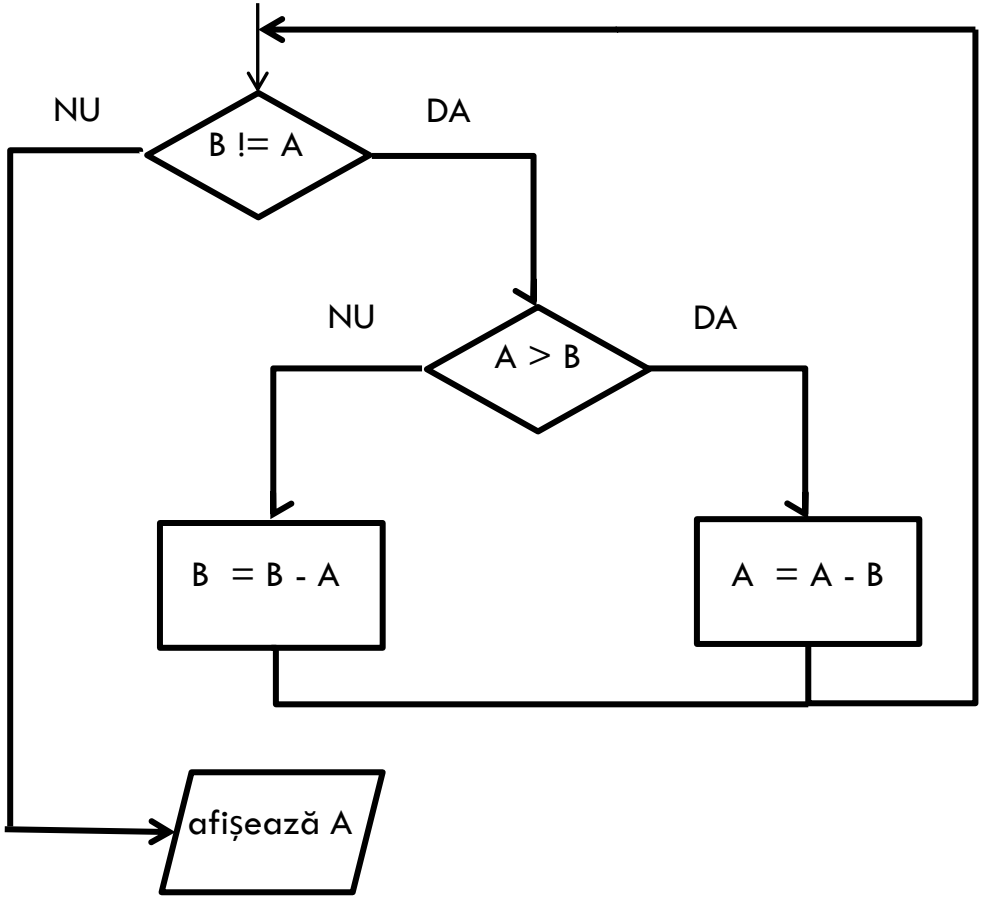
Structura alternativă

Direcția fluxului

Operația de intrare/ieșire

SCHEMĂ LOGICĂ

-alăturare de simboluri vizuale care desemnează fluxul logic al pașilor



Algoritmul lui Euclid
prin scăderi repetate

cât timp B != A execută
  dacă A > B atunci
    A = A – B;
  altfel
    B = B – A;
  sfârșit dacă
sfârșit cât timp
afișează A

CURSUL 1:

1. Algoritmi
2. Limbaje de programare
3. Introducere în limbajul C. Structura unui program C.

LIMBAJE DE PROGRAMARE

Rezolvarea oricărei probleme implică mai multe etape:

1. Analiza problemei

2. Găsirea soluției [optime]

3. Elaborarea algoritmului

4. Implementarea algoritmului într-un limbaj de programare

5. Verificarea corectitudinii algoritmului propus

6. Analiza complexității

LIMBAJ DE PROGRAMARE

- limbaj artificial cu sintaxă și semantică bine definite
- pune la dispoziția programatorilor construcții sintactice prin care sunt specificate succesiunea de operații/instrucțiuni elementare (pe care un calculator le poate executa) asociate algoritmului de rezolvare a unei probleme
- ste necesară cunoaşterea setului de operații/instrucțiuni elementare al calculatorului la care ne referim
- este independent de procesor (pentru asigurarea portabilității codului) și de sistemul de operare
- codul sursă se convertește în cod mașină folosind **compilatoare** sau **interpretoare**

CURSUL 1:

1. Algoritmi
2. Limbaje de programare
3. Introducere în limbajul C. Structura unui program C.

CARACTERISTICI ALE LIMBAJULUI C

- *limbaj procedural*, *structurat*, *compilat*, *de nivel de mijloc*, *scurt*
- limbaj procedural, structurat
  - instrucțiuni specificate sub forma unor comenzi grupate într-o ierarhie de subprograme (denumite funcții) și care pot forma module
- limbaj compilat
  - compilatorul transformă instrucțiunile limbajului C în limbaj mașină
- limbaj de nivel de mijloc
  - permite accesul la date și comenzi aflate aproape de nivelul fizic folosind o sintaxă specifică limbajelor de nivel înalt
- limbaj scurt
  - număr redus de cuvinte cheie
  - multe funcționalități nu sunt incluse în limbajul de bază ci necesită includerea unor biblioteci standard

CARACTERISTICI ALE LIMBAJULUI C

- ***limbaj eficient*, *portabil*, *permisiv*, *poate fi* *dificil de înțeles***
- limbaj eficient
  - viteză mare de execuție a programelor, destinat și aplicațiilor implementate în limbaj de asamblare
  - reutilizarea ulterioară a subprogramelor
- limbaj portabil
  - limbaj independent de hardware
- limbaj permisiv
  - impune puține constrângeri, dă credit programatorului
  - permite introducerea unor erori care sunt foarte greu de depistat
- limbaj dificil de înțeles
  - un stil de programare adecvat este foarte important
  - obfuscated C code contest: www.ioccc.org

CUVINTE CHEIE

C89 = ANSI C : 32 de cuvinte cheie

auto	double	int	struct
break	else	long	switch
case	enum	register	typedef
char	extern	return	union
const	float	short	unsigned
continue	for	signed	void
default	goto	sizeof	volatile
do	if	static	while

C99: ANSI C + alte 5 cuvinte cheie

```
_Bool  _Complex  _Imaginary  inline  restrict
```

STRUCTURA GENERALĂ A UNUI PROGRAM C

- modul principal (funcția main)
- zero, unul sau mai multe module (funcții/proceduri) care comunică între ele și/sau cu modulul principal prin intermediul parametrilor și/sau a unor variabile globale
- unitatea de program cea mai mică și care conține cod este funcția/procedura si conține:
  - partea de declarații/definiții
  - partea imperativă (comenzile care se vor executa)

PRIMUL PROGRAM C

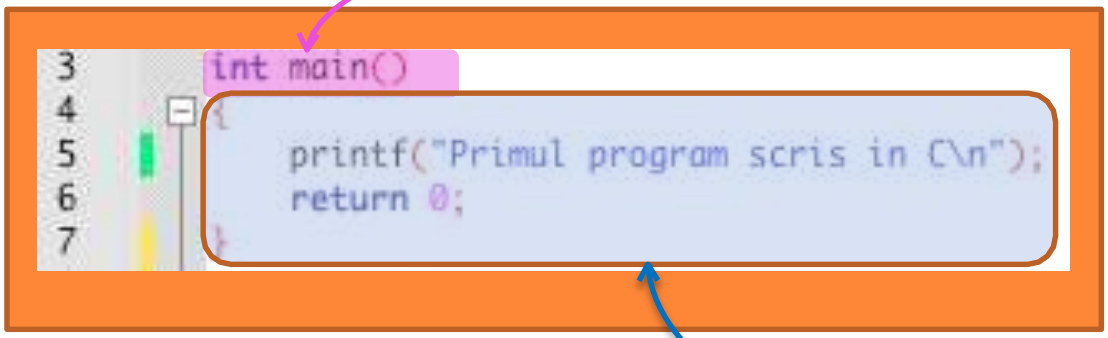
```c
#include <stdio.h>

int main()
{
    printf("Primul program scris in C\n");
    return 0;
}
```

PRIMUL PROGRAM C EXPLICAT

Directivă de preprocesare pentru includerea bibliotecii standard de i/o

Antetul funcției principale



Funcția principală

Corpul funcției

Observații:

- main nu este cuvânt cheie în limbajul C, îl utilizăm pentru numirea funcției principale;
- printf nu este cuvânt cheie, este funcție de bibliotecă (print (afișare) +f (format));
- Limbajul C este case sensitive, se face diferență între litere mici și mari;
- toate cuvintele cheie se scriu cu litere mici;
- instrucțiunile se termină cu caracterul ; (punct și virgulă);
- mai multe instrucțiuni pot fi scrise pe aceeași linie;
- spațiile ajută la organizarea codului.

STRUCTURA UNUI PROGRAM C SIMPLU

directive de preprocesare

```
int main()
{
        instrucțiuni
}
```

```
1  #include <stdio.h>
2
3  int main()
4  {
5      printf("Primul program scris in C\n");
6      return 0;
7  }
```

- Directive de preprocesare
  - directive de definiție: #define N 10
  - directive de includere a bibliotecilor: #include <stdio.h>
  - directive de compilare condiționată: #if, #ifdef, ...
  - alte directive (vorbim în cursurile următoare)
- Funcții
  - grupări de instrucțiuni sub un nume;
  - returnează o valoare sau se rezumă la efectul produs;
  - funcții scrise de programator vs. funcții furnizate de biblioteci;
  - programul poate conține mai multe funcții;
    - main este obligatoriu;
  - antetul și corpul funcției.

STRUCTURA UNUI PROGRAM C SIMPLU

directive de preprocesare

```
int main()
{
        instrucțiuni
}
```

```
1  #include <stdio.h>
2
3  int main()
4  {
5      printf("Primul program scris in C\n");
6      return 0;
7  }
```

- Instrucțiuni
  - formează corpul funcțiilor
    - exprimate sub formă de comenzi
  - **5 tipuri de instructiuni**:
    - instrucțiunea declarație;
    - instrucțiunea atribuire;
    - instrucțiunea apel de funcție;
    - instrucțiuni de control;
    - instrucțiunea vidă;
  - toate instrucțiunile (cu excepția celor compuse) se termină cu caracterul ”;”
    - caracterul ; nu are doar rol de separator de instrucțiuni ci instrucțiunile incorporează caracterul ; ca ultim caracter
    - omiterea caracterului ; reprezintă eroare de sintaxă

```
#include <stdio.h>

int main()

{

    int a, b;

    printf("dati doua numere intregi: ");

    scanf("%d %d", &a, &b);

    printf("cele doua numere sunt x= %d si y=%d\n", a, b);

    // int aux=a; a=b; b=aux;

    // printf("cele doua numere dupa interschimbare sunt: x=%d si y=%d\n", a, b);

    while (b != a)

    {       if(a>b)          a=a-b;       else       b=b-a;    }

    printf("cmmdc = %d", b);

     return 0;

}
```

STRUCTURA UNUI PROGRAM C COMPLEX

```
comentarii
directive de preprocesare
declarații și definiții globale

int main()
{
        declarații și definiții locale
        instrucțiuni
}
```

STRUCTURA UNUI PROGRAM C

Comentariile:

- formă de documentare a codului sursă, sunt ignorate de compilator

- 2 tipuri de comentariu:
  - începe cu /* și se termină cu */: se pot extinde pe mai multe linii, nu se pot imbrica, sunt utile pentru inserarea unor explicații mai lungi

  - începe cu // și se termină la sfârșitul liniei: utile pentu comentariile inserate pe marginea codului (apare în C99, nu este în C89)

CURS 1:

Descrierea cursului: Obiective, Continuturi/ Programa, Bibliografie, Notare

1. Algoritmi. Metode de reprezentare
2. Limbaje de programare
3. Limbajul C:
   a) Introducere
   b) Caracteristici ale limbajului C
   c) Structura unui program C

În Curs 2: Fundamentele limbajului C:

- Etapele realizării unui program C.
- Tipuri de date fundamentale. Variabile. Constante. Operatori. Expresii. Conversii.