# Programarea Calculatoarelor
# Curs 1

Traian Rebedea
traian.rebedea@cs.pub.ro / trebedea@gmail.com

# About Me

- Traian Rebedea, Ph.D.
- The Academic Part:
  - Education
    - B.Sc., "Politehnica" University of Bucharest, CS Dept., Romania
    - M.Sc., "Politehnica" University of Bucharest, CS Dept., Romania
    - Ph.D., Natural Language Processing & Technology-Enhanced Learning, "Politehnica" University of Bucharest, CS Dept., Romania
  - Over 25 articles published at world-wide top conferences:
    - http://www.informatik.uni-trier.de/~ley/db/indices/a-tree/r/Rebedea:Traian.html
  - 4 book chapters on NLP & TEL
  - Jobs:
    - Lecturer, "Politehnica" University of Bucharest, CS Dept., Romania
    - Teaching Assistant, "Politehnica" University of Bucharest, CS Dept., Romania

# About Me

- The Industrial Part
  - Jobs
    - PeopleGraph, Bucharest, Romania – Researcher, Natural Language Processing, Machine Learning & Information Retrieval
    - TeamNet, Bucharest, Romania – Research Consultant, Opinion Mining & Natural Language Processing
    - Create IT, Bucharest, Romania – Founder & Web Developer
    - ProSoft Solutions, Bucharest, Romania – Java Developer
- Other
  - Tutor for the Erasmus-Mundus DMKM Information Retrieval course (taught by Ricard Gavalda from UPC)
- Contact
  - traian.rebedea@cs.pub.ro / trebedea@gmail.com

# Multumiri


- Lui Vlad Posea (programare, 1CB) pentru elaborarea slide-urilor pentru curs si echipei de programare pentru pregatirea laboratoarelor


# Obiective si continut curs

- Ciclul de dezvoltare a programului;
- Compilarea, legarea şi execuţia programelor C;
- Elemente fundamentale ale limbajului C: identificatori, cuvinte cheie, literali, comentarii, constante;
- Structura programelor C;
- Tipuri de date. Operatori. Expresii
- Instrucţiuni: expresie, compusă, void, atribuirea, decizia (if), selecţia (switch), cicluri (while, do while, for), continue, salturi disciplinate (break, return) şi nedisciplinate (goto).
- Funcţii: apel, definire, transfer prin valoare, vizibilitate
- Tablouri: declarare, tablouri uni si multidimensionale, sortare, cautare
- **Pointeri: declarare si inițializare, adresare și dereferențiere, aritmetica pointerilor, funcții care returnează pointeri**

# Obiective și conținut curs

- Şiruri de caractere
- Alocarea dinamică a memoriei: Gestiunea memoriei libere (heap), malloc(), calloc(), realloc(), free(). Pointeri la pointeri. Tablouri de pointeri. Alocare dinamică pentru tablouri multidimensionale.
- Structuri: Declarare şi initializare; Accesul la membrii; pointeri la structuri. Tablouri de structuri. Atribuirea structurilor. Structuri şi funcţii. Uniuni, Câmpuri de biţi.
- Funcţii: transfer prin referinţă, Pointeri la funcţii.
- Fişiere: I/O, Fişiere text şi fişiere binare. Funcţii specifice lucrului cu fişiere
- Parametrii liniei de comandă
- Realizarea de programe complexe: Crearea de librarii statice; Vizibilitatea variabilelor.
- Funcţii cu număr variabil de parametri

# Bibliografie


- ***The C Programming Language –*** Brian Kernighan and **Dennis Ritchie** (http://zanasi.chem.unisa.it/download/C.pdf)


# Linkuri utile

- http://cs.curs.pub.ro – site-ul de cursuri al facultății (forum, alte resurse, teme)
- http://ocw.cs.pub.ro/courses/programare - laboratoare + regulament + alte info utile

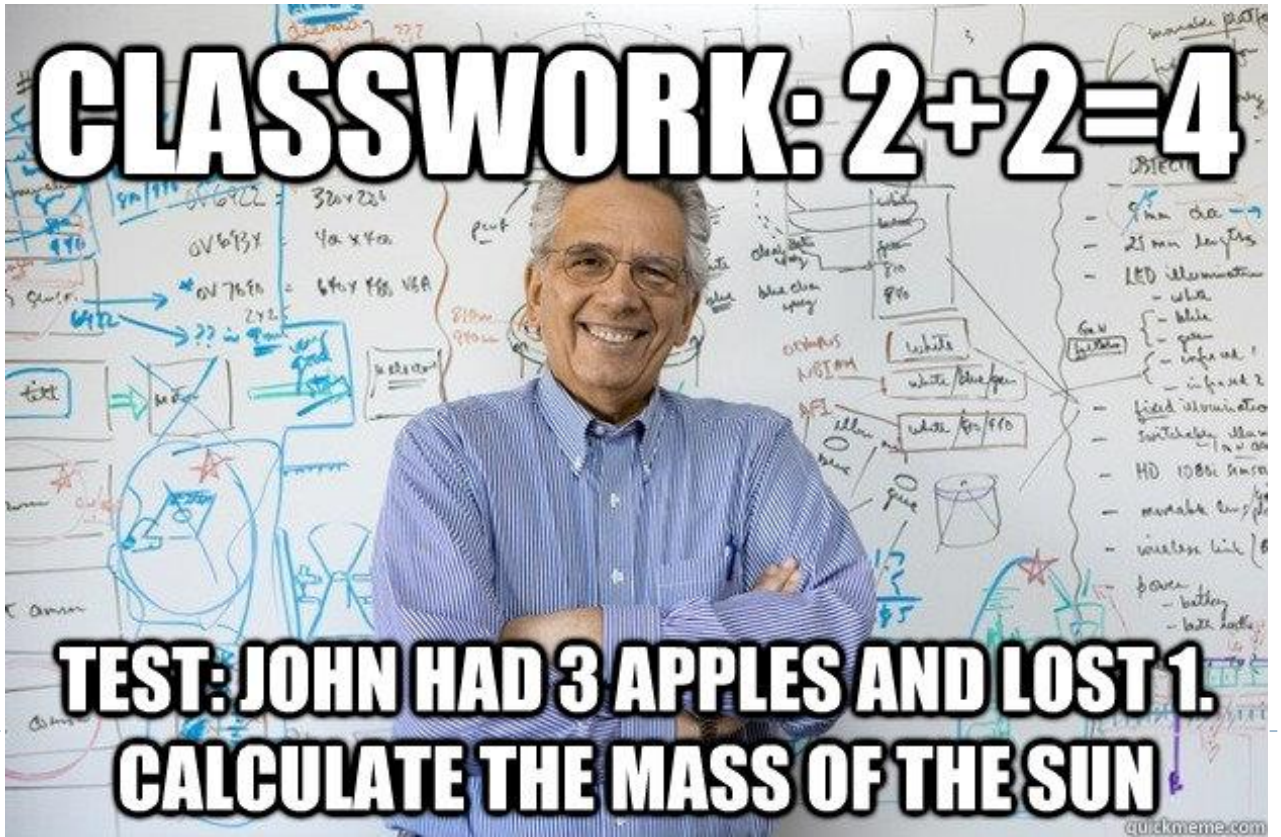
# Notare

- Laborator – 2p
- Teme de casă – 4p
- Examen final – 4p
- Bonus – 0.5p pentru alte activitati legate de programare (concursuri, hackathoane, activitate pe site-uri gen infoarena)
- Condiții de trecere
  - Laborator+teme de casă>=3p &
  - Examen >=2p

# Feedback 2013 - 2015

- Probleme semnalate
  - Explicații insuficiente/nesatisfăcătoare, se avansează prea repede
    - => seminar suplimentar începând din săptămâna 2/3 pentru cei care rămân in urmă
    - frecventa ff redusa (nu stim daca il facem si in 2015)
  - Laboratoare cu prea multe activități
    - Se va pune accent la laborator mai mult pe înțelegerea materiei și mai puțin pe vânat puncte
    - La fiecare laborator se va explica la tablă o problemă și se vor rezolva alte două
  - "Vlad scrie prea urat"
    - Vlad va scrie mai mult la calculator si mai putin la tabla ☺

- Probleme cu temele (prea grele, incorect/incomplet formulate, neclară evaluarea)
  - => 3 teme ce vor fi formulate/rezolvate independent și vor fi publicate din timp cu timp de rezolvare mai lung
  - temele au fost mult mai bine primite anul trecut. Vom avea mai multe subpuncte mai usoare astfel incat rezolvarea sa fie progresiva



# De ce C?

| Oct 2015 | Oct 2014 | Change | Programming Language | Ratings | Change |
|---|---|---|---|---|---|
| 1 | 2 | ↑ | Java | 19.543% | +6.04% |
| 2 | 1 | ↓ | C | 16.190% | -1.47% |
| 3 | 4 | ↑ | C++ | 5.749% | +0.88% |
| 4 | 5 | ↑ | C# | 4.825% | +0.08% |
| 5 | 8 | ↑ | Python | 4.512% | +2.18% |
| 6 | 7 | ↑ | PHP | 2.561% | -0.38% |
| 7 | 13 | ⇈ | Visual Basic .NET | 2.462% | +0.71% |
| 8 | 12 | ⇈ | JavaScript | 2.292% | +0.52% |
| 9 | 9 | | Perl | 2.247% | +0.13% |
| 10 | 16 | ⇈ | Ruby | 1.825% | +0.70% |
| 11 | 11 | | Delphi/Object Pascal | 1.637% | -0.18% |
| 12 | 31 | ⇈ | Assembly language | 1.573% | +1.16% |
| 13 | 14 | ↑ | Visual Basic | 1.515% | -0.05% |

# De ce C?

| Programming Language | 2015 | 2010 | 2005 | 2000 | 1995 | 1990 | 1985 |
|---|---|---|---|---|---|---|---|
| Java | 1 | 1 | 2 | 3 | 31 | - | - |
| C | 2 | 2 | 1 | 1 | 2 | 1 | 1 |
| C++ | 3 | 3 | 3 | 2 | 1 | 2 | 9 |
| C# | 4 | 5 | 7 | 9 | - | - | - |
| Objective-C | 5 | 10 | 43 | - | - | - | - |
| Python | 6 | 6 | 6 | 23 | 15 | - | - |
| PHP | 7 | 4 | 5 | 21 | - | - | - |
| JavaScript | 8 | 8 | 10 | 7 | - | - | - |
| Visual Basic .NET | 9 | 191 | - | - | - | - | - |
| Perl | 10 | 7 | 4 | 4 | 6 | 17 | - |
| Pascal | 17 | 14 | 16 | 18 | 3 | 10 | 6 |

- http://www.tiobe.com

# Caracteristici C

- Dezvoltat intre 1969-1973
- Programare imperativă
  - Programul = secvență de instrucțiuni executată de calculator
- Utilizat pentru a dezvolta sisteme de operare, aplicații embedded, aplicații telecom
- Influențează/stă la baza majorității limbajelor ”moderne”: Java, C#, Javascript, Objective-C, C++

# Versiuni de C

- K&R C (1978)
- **ANSI C – 1990 (cunoscut ca C89 sau C90 sau Standard C**
- C94/C95
- C99 (1999) (tipuri de date noi, vectori de lungime variabila, …)
- C11 (2011) (suport pentru Unicode, programare paralela,…)

# Etapele rezolvării unei probleme

- **Analiza** și prelucrarea datelor de intrare
  - A good programmer is someone who always looks both ways before crossing a one-way street.
  - Identificăm eventualele cazuri speciale ale datelor de intrare și le tratăm separat
    - Ex1: se cere să se citească numere ->ne asigurăm că s-au citit numere și nu alte caractere
    - Ex2: se cere să se calculeze raportul a 2 numere -> ne asigurăm că numitorul este diferit de 0
- Prelucrări necesare (algoritm de rezolvare)
- Prezentarea rezultatelor

# Pseudocod

- Descriere informală a unui algoritm/program
- Exemplu:
  - Program care afișează maximul a 2 numere
    - Se citesc 2 numere intregi a,b
    - Dacă a este mai mare sau egal cu b
      - Afișează a
    - Altfel
      - Afișează b
- După ce elaborăm pseudocodul îl scriem în limbajul de programare preferat ☺ (C)

# Prezentarea rezultatelor

- Rezultatele trebuie să fie inteligibile pentru utilizator

```
vlad@vlad-virtual-machine:~$ ./a.out
4 5
5vlad@vlad-virtual-machine:~$
```

Așa nu

```
vlad@vlad-virtual-machine:~$ ./a.out
introduceti 2 numere intregi:
4 5
maximul lor este: 5
```

Așa da

- Datele cerute de la utilizator trebuie să fie clare
- Eventualele mesaje de eroare trebuie să fie clare

# Compilarea și execuția programelor C

- Editare fisier sursă
- Compilare fișier sursă
  - Generarea unui fișier obiect
    - gcc –c nume_sursa.c => nume_sursa.o

```
vlad@vlad-virtual-machine:~/c1$ ls
max.c
vlad@vlad-virtual-machine:~/c1$ gcc -c max.c
vlad@vlad-virtual-machine:~/c1$ ls
max.c  max.o
```

  - gcc nume_sursa.c –o nume_executabil –alteOptiuni => nume_executabil
  - Link-editare – se generează fișierul executabil pe baza unuia sau mai multe fișiere obiect

# Structura unui program C

```
/* comentarii */
//When I wrote this, only God and I understood what I was doing
//Now, God only knows
/* includere fisiere antet (biblioteci) */
#include <stdio.h>

/* definitii si declarații globale*/
int maxim(int a, int b);

int main()
{
/* definiții și declarații locale */
/* secvență instrucțiuni */
return 0; /* finalizarea execuției + întoarcere rezultat*/
}
```

# Hello world

```
/* includem biblioteca standard I/O */
#include <stdio.h>

/* functia main – intoarce un rezultat intreg */
int main()
{
        printf("Hello world!\n");
        return 0;
}
```

```
{acelasi program in pascal}
Program HelloWorld(output);
begin
        writeln('Hello, world!');
end.
```

# Sintaxa C – elemente de bază

- Limbaj case-sensitive
  - Printf(”hello world”) nu este același lucru cu printf(”hello world”)
- Cuvintele cheie se scriu cu litere mici
- Identificatorii pot conține litere, cifre, underscore (_), nu pot incepe cu cifră
- Comentarii:
  - // comentariu pe o singura linie
  - /* comentariu care se poate întinde pe mai multe linii*/

# Variabile

- Def: un mod de a referi o locație de memorie utilizată într-un limbaj de programare http://cplus.about.com/od/introductiontoprogramming/g/variabledefn.htm
- Pentru a lucra cu mai multe date trebuie să le reținem în memoria calculatorului
- Variabilele sunt folosite pentru a regăsi locul unde am scris datele în memorie
- Variabile simple: au o singură valoare la un moment dat

# Variabile (2)

- Caracterizate de
  - Nume – identificatorul variabilei
  - Tip – tipul variabilei
  - Valoare
  - Adresa
- Ex: int x=5;
- int= tipul variabilei
- x=numele variabilei
- 5=valoarea variabilei
- &x = adresa din memorie unde se află stocată valoarea variabilei cu numele x.

| Tipuri de date simple | Explanation | Descriptor |
|---|---|---|
| char | Cel mai mic tip adresabil. Contine caracterele de baza. Are valorice cu sau fara semn. In general are 1 (un) octet/ 8 biti (mai corect are CHAR_BIT biti) | %c |
| signed char | De aceeasi dimensiune cu char dar garantat cu semn | %c |
| unsigned char | De aceeasi dimensiune cu char dar garantat fără semn => are valori intre $[0,\ 2^{CHAR\_BIT}-1]$ | %c (or %hhu for numerical output) |
| short<br>short int<br>signed short<br>signed short int | Tip de date intreg care are valori minim intre [−32767,+32767] deci cel putin 16 biti | %hi |
| unsigned short<br>unsigned short int | La fel ca short dar fara semn | %hu |
| int<br>signed<br>signed int | Principalul tip intreg. Are valori minim intre [−32767,+32767] dar in general este reprezentat pe 32 de biti | %i sau %d |
| unsigned<br>unsigned int | La fel cu int dar fara semn | %u |
| long<br>long int<br>signed long<br>signed long int | *Intreg lung*. Are valori minim intre [−2147483647,+2147483647] deci cel putin 32 de biti | %li |
| unsigned long<br>unsigned long int | La fel ca long dar fara semn | %lu |

| | | |
|---|---|---|
| long long<br>long long int<br>signed long long<br>signed long long int | *Are minim 64 de biti. Specificat in C99* | %lli |
| unsigned long long<br>unsigned long long int | La fel ca long long dar fara semn | %llu |
| float | Tip de date real, dimensiune nespecificata | %f |
| double | Tip de date real precizie dubla, dimensiune nespecificata | %lf |

# Tipuri de date simple (2)

- C vs. Pascal

| C | Pascal |
|---|---|
| short<br>int<br>long | integer |
| float<br>double | real |
| char | char |
| int /* C has no boolean type */ | boolean |

- nr bytes pentru fiecare tip de date – operatorul sizeof

```
int main()
{
printf("dimensiune char: %d, dimensiune int: %d, dimensiune short: %d,\n"
"dimensiune float: %d, dimensiune double: %d\n", sizeof(char), sizeof(int), sizeof(short),
 sizeof(float), sizeof(double));
return 0;

}
```

```
vlad@vlad-virtual-machine:~/c1$ gcc size.c -o size
vlad@vlad-virtual-machine:~/c1$ ./size
dimensiune char: 1, dimensiune int: 4, dimensiune short: 2,
dimensiune float: 4, dimensiune double: 8
```

# Input/Output (I/O)

- /*discutate oarecum mai devreme intrucat sunt necesare la laborator*/
- int printf ( const char * format, ... );
- parametri:
  - sir de caractere ce poate conține descriptori de format
  - expresii
- descriptor de format:
  - %[flags][width][.precision][length]specifier
  - http://www.cplusplus.com/reference/cstdio/printf/

# I/O (2)

| *specifier* | Output | Example |
|---|---|---|
| d *or* i | Signed decimal integer | 392 |
| u | Unsigned decimal integer | 7235 |
| o | Unsigned octal | 610 |
| x | Unsigned hexadecimal integer | 7fa |
| X | Unsigned hexadecimal integer (uppercase) | 7FA |
| f | Decimal floating point, lowercase | 392.65 |
| e | Scientific notation (mantissa/exponent), lowercase | 3.9265e+2 |
| E | Scientific notation (mantissa/exponent), uppercase | 3.9265E+2 |
| g | Use the shortest representation: %e or %f | 392.65 |
| G | Use the shortest representation: %E or %F | 392.65 |
| c | Character | a |
| s | String of characters | sample |
| p | Pointer address | b8000000 |
| n | Nothing printed.<br>The corresponding argument must be a pointer to a signed int.<br>The number of characters written so far is stored in the pointed location. | |
| % | A % followed by another % character will write a single % to the stream. | % |

# exemple printf

```
printf("valoarea lui x este: %-4.2f\n",3.14);
printf("x=%i, y=%f, x=%o, x=%#x\n",15,3.14,15,15);
printf("c= %c, c=%d\n",'%','%');
printf("sir de caractere: %s\n", "ana are mere");
printf("\\ \" \' \n");
```

```
valoarea lui x este: 3.14
x=15, y=3.140000, x=17, x=0xf
c= %, c=37
sir de caractere: ana are mere
\ " '
```

- secvente escape (caractere speciale)

| | | | |
|---|---|---|---|
| \a | alert (bell) character | \\ | backslash |
| \b | backspace | \? | question mark |
| \f | formfeed | \' | single quote |
| \n | newline | \" | double quote |
| \r | carriage return | *ooo* | octal number |
| \t | horizontal tab | \x*hh* | hexadecimal number |
| \v | vertical tab | | |

# scanf

- int scanf ( const char * format, ... );
- http://www.cplusplus.com/reference/cstdio/scanf/
- similar cu printf
- se transmite ca parametru adresa variabilei citite
- intoarce ca rezultat numărul de valori citite cu succes
- citește într-un buffer – stdin.

```c
int main()
{
        int a, b;
        char c;
        for(;;)
        {
                printf("Introduceti 2 numere intregi\n");
                if(scanf("%d%d",&a,&b)==2)
                        break;
                else
                        while(c=getchar()!='\n' && c!=EOF);


        }
        printf("am citit 2 numere: %d si %d\n",a,b);
        return 0;
}
```

```
vlad@vlad-virtual-machine:~/c1$ ./a.out
Introduceti 2 numere intregi
3 a
Introduceti 2 numere intregi
4 2 3
am citit 2 numere: 4 si 2
```

```
if(scanf("%d%d",&a,&b)==2)
        break;
```

- scanf este o functie. rezultatul sau este numarul de valori citite corect conform descriptorilor.
- daca am reusit sa citim corect 2 valori intrerupem iteratia cu break

```
while(c=getchar()!='\n' && c!=EOF);
```

- se citeste un caracter si se salveaza in variabila c.
- rezultatul atribuirii c=getchar() este chiar caracterul citit si acesta se compara cu \n
- se iese cand s-a intalnit \n sau EOF (character special care semnifica faptul ca nu se mai poate citi din sursa)


Vineri 09 Oct | 18:00 Hol EC

awg.acs.pub.ro

---


Sâmbătă 10 Oct | 10:00 Hol EC

lif.acs.pub.ro

---


Sâmbătă 10 Oct | 16:00 Hol EC

treasure.innovationlabs.ro

---


Duminică 11 Oct | 10:00
Coloana Infinitului

lost.acs.pub.ro