Fundamentele programării

Elena BAUTU & Dorin-Mircea POPOVICI
{ebautu,dmpopovici}@univ-ovidius.ro
Web: http://moodle.univ-ovidius.ro

Cuprins

Variabile

Operatii de intrare/iesire

Unitati lexicale

Operatii

Variabile

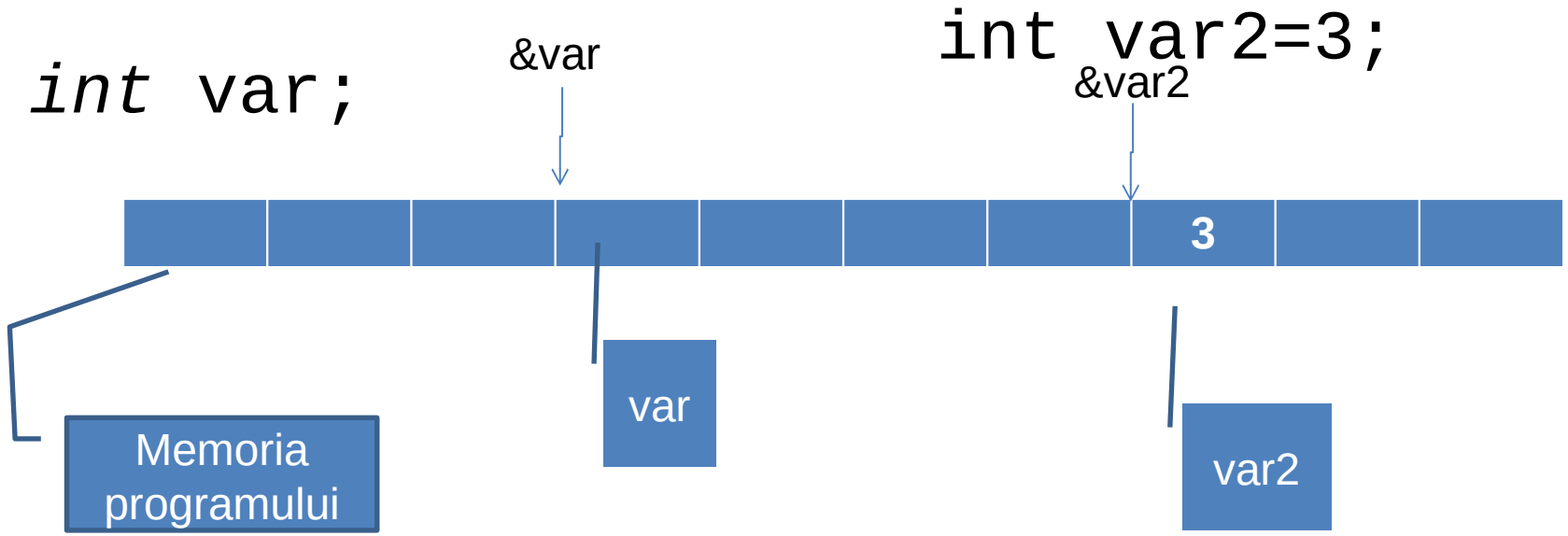
Identificatori

- un **nume** pe care programatorul îl dă unei entităţi in C (o variabilă, funcţie, tip de date, etc) pentru a putea lucra mai uşor cu aceasta.
- identificator
  - secvenţă de caractere alfanumerice şi underscore (_)
  - începe cu o literă sau underscore.
  - literele mari diferă de litere mici

- Găsiţi identificatorii corecţi:
  - `varsta`
  - `1_secret`
  - `greutate`
  - `_secret`
  - `total1980`
  - `_`
  - `1980total`
  - `_1980`

Variabila - declarare

- Variabila = o **zona de memorie** capabila sa retina o informatie de un anumit **tip**, accesibila programatorului printr-un **nume** si calculatorului, printr-o **adresa de memorie**



înainte de a fi folosită orice variabilă trebuie **declarată**
float PI = 3.1415926;
int varsta = 26, greutate; //greutate este **declarata** (alocata fara valoare initiala)
PI si varsta sunt **initializate** (alocate si cu valoare initiala)

Tipul variabilei determină:
operaţiile care se pot efectua asupra valorii variabilei
domeniul de valori a variabilei

Operatii de intrare/iesire

Afişarea datelor

- funcţia **printf** afişează pe ecran un **mesaj** (şir de caractere)
  - mesajul poate să conţină coduri speciale pentru afişarea altor informaţii
    - %d, %i – informaţii cu tip întreg in baza 10
    - %0 – intreg in baza 8, %x intreg in baza 16
    - %f – informaţii cu tip real (float sau double)
    - %c – date de tip caracter
    - %s – date de tip sir (vector) de caractere

```
int varsta = 20;
float pi = 3.141528;
printf("Varsta este %d", varsta);
printf("Pi este aproximativ %f\n", pi);
```

  - printf este alternativa pentru **cout<<“Mesaj”**

Citirea datelor

- funcţia **scanf** citeşte de la tastatură un mesaj (şir de caractere)
  - mesajul poate să conţină coduri speciale pentru citirea unor informaţii în variabile
    - observati folosirea lui & înainte de numele variabilei

```
int varsta;
printf("Ce varsta aveti?");
scanf("%d", &varsta);
```

  - scanf este alternativa pentru **cin>>variabila**.

Unitati lexicale

Unităţi lexicale

- Unităţi lexicale = atomii unui program C
  - **sintaxa** = reguli de îmbinare astfel încât să obţinem programe corecte
  - unităţile lexicale sunt echivalente cu cuvintele unei limbi, iar *sintaxa* limbajului C cu *gramatica*

- Exemple de unităţi lexicale
  - comentarii
  - directive de preprocesare
  - valori constante
  - cuvinte cheie
  - identificatori
  - operatori
  - instrucţiuni

Mulţimea caracterelor

- similar cu alfabetul unei limbi
  - fiecare caracter are un înţeles special pentru compilatorul C
- limbajul C foloseşte caracterele ASCII
  - litere (mici şi mari din alfabetul englez)
  - cifre (de la 0 la 9)
  - caractere de spaţiere (spatiu, tab, enter, etc.)
  - caractere speciale ([{ \ # ~ | % ^ ? etc.)
  - alte caractere (@ ` $ etc.)

Comentarii

- sunt secvenţe din program ignorate de compilator.
  - Programatorul poate scrie aici observaţiile sale privind părţi din program.
- comentariile nu pot fi imbricate (i.e. continute unul in altul)
- comentariu pe o linie: începe cu // şi se termină la sfarşitul liniei

```
// primul meu program C
i++; // valoarea lui i creste cu 1
```

- comentariu bloc: încep cu /* şi se termină la primul */
  - este folosit pentru a nota o secvenţă mai mare/importantă din program

```
/*
  primul meu program C este
  programul Hello world
*/
```

Exerciti

- Găsiţi comentariile corecte:
  - `// un exercitiu pentru voi //`
  - `/* un exercitiu pentru voi`
  - `// un exercitiu pentru voi */`
  - `// un exercitiu`
    `pentru voi`
  - `/* un exercitiu`
    `pentru voi`
    `*/`
  - `/**/`

Directive de preprocesare

- **Preprocesare** = modificarea automată a programului sursă înainte de compilare
- Directivele de preprocesare încep cu #
  - #include, #define #ifdef etc.

```
/* #include este înlocuită AUTOMAT cu
fişierul sursă indicat */
#include <stdio.h>
/* stdio.h contine functii pentru
intrare/iesire */
```

Cuvinte cheie

- secvenţă de caractere rezervate de limbaj
  - nu pot fi folosite ca identificatori

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

Instructiuni

Instrucţiuni C

- instrucţiunile unui program determină *operaţiile* pe care acesta le efectuează
- O instrucţiune este formată din cuvinte cheie, expresii şi/sau alte instrucţiuni.
  - instrucţiunile simple se termină cu punct si virgulă
  - instrucţiunile compuse se scriu între **acolade**
    - Se mai numesc **Bloc** de instructiuni

Instructiunile vida si compusa

- instrucţiunea vidă nu execută nici o operaţie (se scrie doar ;)
- Instructiunea compusa consta dintr-o succesiune de declaratii si instructiuni incluse intre acolade

{

...//instructiuni simple

}

Instrucţiunile expresie si atribuire

- O instructiune expresie C are drept efect evaluarea expresiei

```
x = x+1;
printf(“Salut”);
int A = 5;
```

Operatorul de atribuire **=**

este folosit pentru a seta o valoare pentru *variabila* din partea sa stanga. In partea sa dreapta se afla o *valoare* sau o *expresie* care este evaluata.

Operatori aritmetici

- Operatori unari de păstrare/schimbare a semnului: + şi –
- Operatori binari multiplicativi *, / şi %
- Operatori binari aditivi + şi –
- Exemplu:
  - int i = –2 + 3 * 4 – 5;
  - Este diferit de –(2 + 3 * 4 – 5);
  - Si este diferit de – 2 + 3*(4 – 5);

Instructiuni decizionale

Instrucţiunea decizionala if

- permite execuţia unei instrucţiuni în funcţie de valoarea unei expresii
- **if (*expresie*)**
  - **instructiune 1;**

    else

    instructiune 2;
- Ramura else şi instrucţiune2 sunt opţionale.

- efect: se evaluează **expresie**; dacă rezultatul este nenul, atunci se execută **instrucţiune1**, altfel se execută **instrucţiune2**
- exemplu:

```
if (delta < 0)
printf(“Delta este negativ\n”);
else
        printf(“Delta este 0 sau
pozitiv\n”);

if (bani > 100)
bogat = 1; // oare?
```

Instrucţiunea decizionala if

```
if (current_speed >0 )
   current_speed = current_speed – 1;
else
   printf(“The car is stopped”);
```

Este echivalent cu

```
if (current_speed >0 ) {
   current_speed = current_speed – 1;
}
else{
   printf(“The car is stopped”);
}
```

```
//instructiuni inlantuite

if (punctaj >= 90) {
nota = 10;
} else if (punctaj >= 80) {
nota = 9;
} else if (punctaj >= 70) {
nota = 8;
} else if (punctaj  >= 60) {
nota = 7;
} else if (punctaj >= 55) {
nota = 5;
}
else
Nota =4;
printf(“Nota este %i”, nota);
```

Instrucţiunea decizionala if

```
// instructiuni if imbricate (nested)

//solutia ec. de ord. I: ax+b=0
if(a != 0){
   if(b != 0){
     printf("Solutia este %lf.\d", -b/a);
   }
   else{
     printf("Solutia este 0.\n");
   }
}
else{ //a este 0
   if(b == 0)
      printf("Orice nr real este solutie.\n");
   else // a este 0 dar b este dif. de 0
printf("Ecuatia nu are solutii.\n");
}
```

In prezenta acoladelor, o ramura else se leaga de cel mai apropiat if din blocul in care se gaseste.

Instrucţiunea decizionala if

```
// instructiuni if imbricate (nested)

//solutia ec. de ord. I: ax+b=0
if(a != 0)
   if(b != 0)
     printf("Solutia este %lf.\d", -b/a);
   else
     printf("Solutia este 0.\n");
else //a este 0
   if(b == 0)
      printf("Orice nr real este solutie.\n");
   else // a este 0 dar b este dif. de 0
printf("Ecuatia nu are solutii.\n");
```

Ramura else se leaga de cel mai apropiat if !

Chiar si in absenta { } , compilatorul stie cum sa lege ramurile if-else.

In prezenta acoladelor, o ramura else se leaga de cel mai apropiat if din blocul in care se gaseste.

Exercitiu

Fie codul urmator

```
if (number >= 0)
if (number == 0)
printf("first string\n");
else
printf("second string\n");
printf("third string");
```

Ce output produce codul daca number = 3?

Formatati codul folosind spatii pentru a face fluxul executiei mai usor de urmarit.

Totul depinde de “expresie”

- Operatori relationali

  `<  <=  >  >=  ==  !=`

- Atentie! `a==0` este diferit de `a=0`

Rezultatul evaluarii unei expresii ce contine operatori relationali este un intreg, care codifica o valoare de adevar:

0 = fals, nenul (1) = adevarat

- Operatori logici

  `!` negatie logica

  `&&` SI logic

  `||` SAU logic

Negatia logica

- ! Este operator unar, deci are prioritatea cea mai ridicata
- Tabela de adevar a operatorului

X	!X
≠0	0
0	1

Operatori logici

- && si || sunt operatori binari, au prioritatea mai mica decat operatorii relationali
- Tabelele de adevar ale operatorilor

X	Y	X && Y
0	0	0
0	≠0	0
≠0	0	0
≠0	≠0	1

X	Y	X || Y
0	0	0
0	≠0	1
≠0	0	1
≠0	≠0	1

Ce afiseaza codul urmator?

```
int a=100;
if(a>10 && a<100)
         printf(“Constanta");
else if(a>20)
         printf(“Iasi");
else if(a>30)
           printf(“Bucuresti");
```

```
int a = 10, b = 100;
if (a <10 || b >= 100)
  printf(“Ieri”);
else
  printf(“Maine”);
```

Ce afiseaza codul urmator?

```
int a=5, b=10, c=1;
if(a!=0 && b > c ){
     printf("1");
}
else{
       printf("2");
}
```

```
int a=5, b=10, c=1;
if(a  && b > c ){
     printf("1");
}
else{
       printf("2");
}
```

What will be the output of the following C code?

```
int m=5,n=10,q=20;
if(m != n)
   printf("William Gates");
else if(n == q)
   printf(" Warren Buffet");
else
   printf(" Carlos Slim Helu");
```

```
int m=5,n=10,q=20;
if(q/n*m)
   printf("William Gates");
else if(n/m)
   printf(" Warren Buffet");
else
   printf(" Carlos Slim Helu");
```

Hint.
Arithmetic operations that involve int numbers are evaluated to int values.

Truth values are also encoded as int values in C.
E.g. 10+2 => 12
10/2 => 5
11/2 => 5

Ce afiseaza codul urmator?

```
 if(-1)
   printf("Hu Jintao\n");
if(.92L)
   printf("Emanuel Macron\n");
if(0)
   printf("Ion Popescu\n");
if('W')
   printf("Vladimir Putin\n");
```

```
if(0xA)
   if(052)
            if('\xeb')
                if('\012')
                printf("Tom hanks");
             else;
         else;
     else;
else;
```

Consultati tabela caracterelor din C http://www.codetable.net/asciikeycodes

https://en.wikipedia.org/wiki/Escape_sequences_in_C#Table_of_escape_sequences

Regula “scurtcircuitului”

- Daca primul operand al expresiei in care apare operatorul && este 0, sigur rezultatul final este 0, indiferent de valoarea celui de-al doilea.

- Daca primul operand al expresiei in care apare operatorul || este !=0, sigur rezultatul final este 1, indiferent de valoarea celui de-al doilea

- Expresiile logice in C se calculeaza prin **scurtcircuitare**
  - daca primul operand are valorile de mai sus, corespunzator operandului && sau ||, cel de-al doilea operand nu se mai evalueaza.

Legile lui de Morgan

Daca A si B sunt doua expresii logice:

! (A && B) este echivalent cu !A || ! B

- ! (A || B) este echivalentă cu ! A && ! B

Exercitii

1. Considerati secventa de cod:

```
if (delta < 0 && bani<100)
        printf(“Delta este negativ si” “sigur nu sunt bogat\n”);
```

Modificati instructiunea astfel incat sa trateze si cazul in care “delta e negativ” dar “sunt bogat”

2. Folosind instrucţiunea if-else, scrieţi un program care citeşte un număr intre 1 si 7 şi afişează ziua din săptămână corespunzătoare.

3. Scrieti un program care pentru trei numere naturale citite de la tastatura determină dacă pot fi laturile unui triunghi şi daca acesta este dreptunghic .

Instrucţiunea decizionala switch

- permite execuţia unei instrucţiuni în funcţie de valoarea multivalenta a unei expresii

```
switch (expresie evaluata la int){
        case val1:
                instructiune;
                break; //optional
        case val2:
                instructiune;
                break; //optional
        case valn:
                instructiune;
                break; //optional
        default:
                instructiune;
}
```

efect: se compară rezultatul expresiei cu val1, val2, ...

Dacă una este egală,
atunci se execută instrucţiuni **începând** de la acel caz **pana la** *primul break* intalnit sau pana la sfarsitul blocului }.

Instrucţiunea **break opreşte** execuţia lui switch

Controlul execuției este pasat **următoarei instrucțiuni** de după switch, din program

Dacă nu există o valoare egală se execută instrucţiunile începând de la default (dacă există)

Instrucţiunea decizionala switch

- exemple:

```
int grupa;
grupa =3;
switch(grupa) {
  case  1:
     printf(“Prima grupa\n”);
     break;
  case  2: printf(“A doua grupa\n”); break;
  case  3: printf(“A treia grupa\n”); break;
  default:
     printf(“Grupa necunoscuta\n”);
}
```

Instrucţiunea decizionala switch

- cazurile pot să **nu** conţină instrucţiuni
- cazul implicit (default) este op**ţional**
- exemplu:

```
switch(grupa) {
case  1:
case  2: printf(“Seria 1\n”); break;
case  3:
case 4: printf(“Seria 2\n”); break;
default: printf(“Serie necunoscuta\n”);
}
```

Ce afiseaza codul urmator?

```
int check=2;
     switch(check){
        case 1: printf("1");
        case 2: printf("2");
        case 3: printf("3");
        default:printf("default");
}
```

```
int c = 2;
    switch(c){
                case 1: printf("1");
                case 2: printf("2");
                case 1+1: printf("1+1");
                case 3: printf("3");
                case 4: printf("4");
                }
```

Ce afiseaza codul urmator?

```
switch(5/2*6+3){
       case 3:printf("David Beckham");
            break;
       case 15:printf("Ronaldinho");
            break;
       case 0:printf("Lionel Messi");
            break;
       default:printf("Ronaldo");
    }
```

Ce afiseaza codul urmator?

```
#include<stdio.h>
int main()
{

char c = 280;
switch(c){
    printf("Start");
    case 280: printf("280");
break;
    case 24: printf("24");
break;
    default: printf("default");
    printf("End");
}

}
```

```
int a = 5;
a= (a>=4);
switch(a){
case 0: a=8;
case 1: a=10;
case 2: a=a+1;
case 3:
printf("%d",a);
```

Atribuirea are o prioritate mai mică față de operatorii relaționali.
Expresiile care implică operatori relaționali sunt evaluate la 1 (pentru adevărat) și 0 (pentru fals).

Sumar

Azi am invatat despre:

- operatii de intrare/iesire de baza
    + functiile printf si scanf
- instructiuni
    + simple, expresie, aribuire
    + operatii/operatori
    + instructiunea decizionala if
    + instructiunea decizionala switch

Intrebari?