Programarea calculatoarelor
Limbajul C

CURS 2

Instrucțiuni C

Cuprins

- Operatori, expresii
- Operații
- Funcții intrare-ieşire
- Instrucțiuni
  - if
  - while
  - do.. while
  - for
  - switch
  - break
  - continue
  - goto
  - return

Operatori

- Operatorul de conversie explicită (cast)

(tip) operand

```
int a=10, b=4;
double c;
c=a/b;
c=(float)a/b
```

- Operatorul condițional:

exp1 ? exp2 : exp3;

```
int a=5, b=3,c;
c=a>b ? b : a;
```

Expresii

- Construite cu constante, variabile, operatori
- De mai multe tipuri, ca şi variabilele: 3*x+7, x<y, etc
- La evaluarea expresiilor se ține cont de precedență şi asociativitatea operatorilor!

Exemple:

```
int a,v=0;
a=++v;          //după executarea Instrucțiunii v=1 şi a=1
a=v++;          //a=1 şi v=2
a=v--;          //a=2 şi v=1
a=--v;          //a=0 şi v=0;
float b=5;
b>5 && b<10
int a, b,c,d;
a=b=c=d=1;
```

Conversii implicite

- La evaluarea expresiilor în expresia variabila=expresie
  - Se evaluează prima data expresia, fără a ține cont de tipul variabilei; dacă tipul rezultatului obținut este diferit de cel al variabilei, se realizează conversia implicită la tipul variabilei
  - Dacă o expresie are doar operanzi întregi, ei se convertesc la int
  - Dacă o expresie are doar operanzi reali şi întregi, ei se convertesc la double.

Operații

- Intrare: preluarea unei date de la un dispozitiv de intrare
- Ieşire: trecerea unei date din memorie către un dispozitiv de ieşire
- Atribuire: x=3; y=x; y=x+y
  - Se evaluează expresia din dreapta atribuirii
  - Valoarea obținută este atribuită variabilei din stânga, care îşi pierde vechea valoare
- Decizie

Funcţii de intrare/ieşire

- Funcția printf
- Funcția scanf

Exemple:

```
int i;
float f;
double d;
scanf(“%d%f%lf”,&i, &f, &d);

char c;
printf(“rezultatul este: %c\n”,c);
```

Specificatori format *scanf*

- %d: întreg zecimal cu semn
- %i: întreg zecimal, octal (0) sau hexazecimal (0x, 0X)
- %o: întreg în octal, precedat sau nu de 0
- %u: întreg zecimal fără semn
- %x, %X: întreg hexazecimal, precedat sau nu de 0x, 0X
- %c: orice caracter; nu sare peste spații (doar " %c")
- %s: şir de caractere, până la primul spațiu alb. Se adaugă '\0'.
- %a, %A, %e, %E, %f, %F, %g, %G: real (posibil cu exponent)
- %p: pointer, în formatul tipărit de printf
- %[…]: şir de caractere din mulțimea indicată între paranteze
- %[^…]: şir de caractere exceptând mulțimea indicată între paranteze
- %%: caracterul procent

Specificatori format *printf*

- %d, %i: întreg zecimal cu semn
- %o: întreg în octal, fără 0 la inceput
- %u: întreg zecimal fără semn
- %x, %X: întreg hexazecimal, fără 0x/0X; cu a-f pt. %x, A-F pt. %X
- %c: caracter
- %s: şir de caractere, până la '\0' sau nr. de caractere dat ca precizie
- %f, %F: real fără exp.; precizie implicită 6 poz.; la precizie 0: fără punct
- %e, %E: real, cu exp.; precizie implicită 6 poz.; la precizie 0: fără punct
- %g, %G: real, ca %e, %E dacă exp. < -4 sau precizia; altfel ca %f. Nu tipăreşte zerouri sau punct zecimal în mod inutil
- %%: caracterul procent

Exemple

- printf(“afisez 10 spatii: %*c”,10,’ ‘);
- float a;

  double b;

  scanf(“%f”, &a);

  printf(“%5.2f”,a); /* sunt afisate minim 5 caractere, maxim 2 zecimale*/

  scanf(“%lf”, &b); //citire var double

  printf(“%-4.2lf”,b); // aliniere stanga

  printf(“%+4.2lf”,b); // adaugare semn (+,-)

Tipul void

- nu are constante (valori)
- utilizat atunci când funcțiile nu întorc valori

```
void f(int a)
{
   if (a) a =a/2;
}
```

- sau când funcțiile nu au parametri

```
void f(void)
/*echivalent cu void f()*/
int f(void)
/*echivalent cu int f()*/
```

Alte funcții de intrare/ieşire

int getchar(void);

- returnează codul unui caracter citit de la tastatura sau valoarea EOF (constantă simbolică definită în stdio.h, având valoarea -1) dacă s-a tastat Ctrl/Z.

int putchar(int c);

- tipăreşte pe ecran caracterul transmis ca parametru;
- returnează codul caracterului sau EOF în cazul unei erori.

int puts(const char * şir);

- tipareste şirul primit ca parametru, apoi un NewLine, cursorul trecand la începutul rândului următor.
- returnează codul ultimului caracter din şir.

Alte funcții de intrare/ieşire

int getche(void);

- Citeşte un caracter (aşteaptă apăsarea unei taste, chiar dacă în buffer-ul de intrare mai sunt caractere neprelucrate)
- afişează caracterul pe ecran
- returnează codul; returneaza EOF la tastarea lui Ctrl/Z, respectiv CR ('\r', cu codul 13) la tastarea lui Enter.

int getch(void);

- Analog cu funcția de mai sus, dar caracterul nu se transmite în ecou (nu se afişează pe ecran).

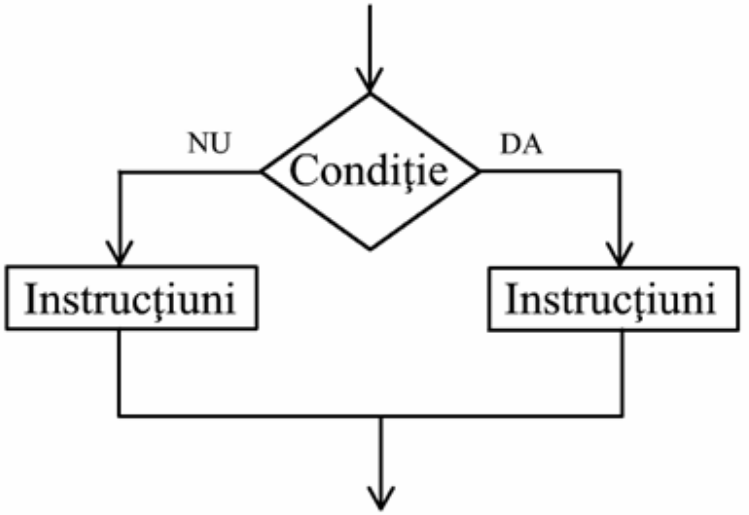
Instrucțiuni C

- Instrucțiunea vidă ;
- Instrucțiunea de atribuire a = 3
- Instrucțiunea compusă (bloc)

  { instr1; instr2; etc }

- Instrucțiunea if

  if (expresie) Instrucțiune1;
  else Instrucțiune2



Instrucțiunea *if*

- Exemple:

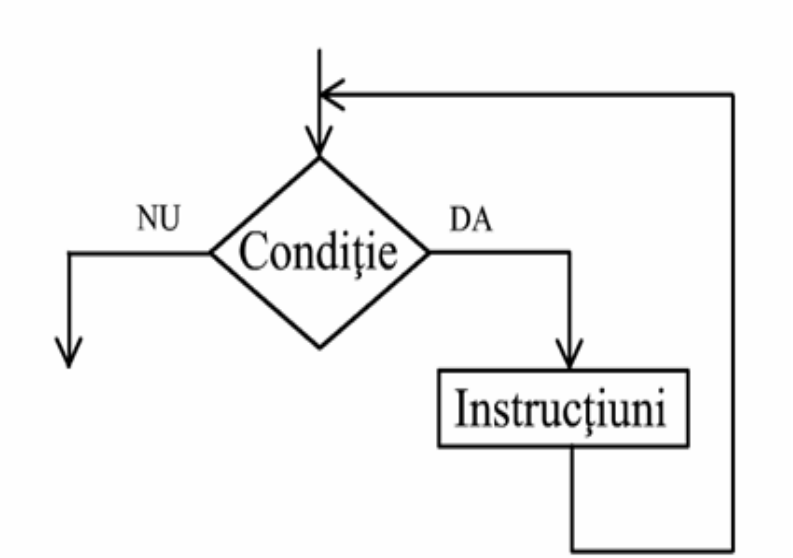
```
if(x)
  if(y) printf(“1”);
  else printf(“2”);

if(x){
    if (y) printf(“1”);
}
else printf(“2”);

If (i=0) printf(“Variabila i are valoarea 0”);
else printf(“Variabila i are o valoare diferita de 0”);
```

Probleme propuse

1. Interschimbul valorilor a două variabile a şi b.
2. Se citesc de la tastatură 3 numere întregi reprezentând lungimile laturilor unui triunghi. Să se calculeze şi să se afişeze aria triunghiului.
3. Să se afişeze în ordine crescătoare valorile a 3 variabile a, b şi c.

Instrucțiunea while

- while (expresie)
  instrucțiune;



- Exemplu:

```
#include<stdio.h>
int main(void)
{
    int a=18, b=12, r;
    while ( a%b != 0 ) {
        r=a%b;
        a=b;
        b=r;
    }
    printf("cmmdc este  %d\n",b);
    return 0;
}
```

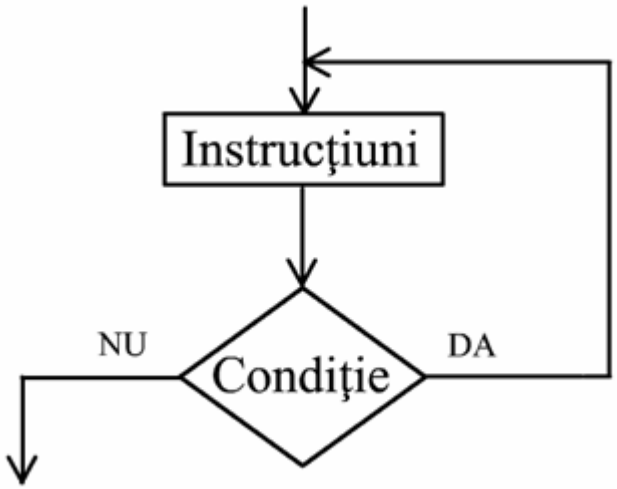
Instrucțiunea while

- Exemplu:

```
#include<stdio.h>
int main()
{
    int a=18, b=12, r;
    while (r=a%b){
        a=b;
        b=r;
    }
    printf("cmmdc este  %d\n",b);
    return 0;
}
```

Instrucțiunea do while

```
do
    instrucțiune;
while(expresie);
```



Exemplu:

```
#include<stdio.h>
int main()
{
    int s=0,i=0,n;
    printf("introduceti n:\n");
    scanf("%d",&n);
    do{
        s+=i;
        i++;
    }while (i<=n);
    printf("suma primelor %d nr. nat. este
                %d\n",n,s);
    return 0;
}
```

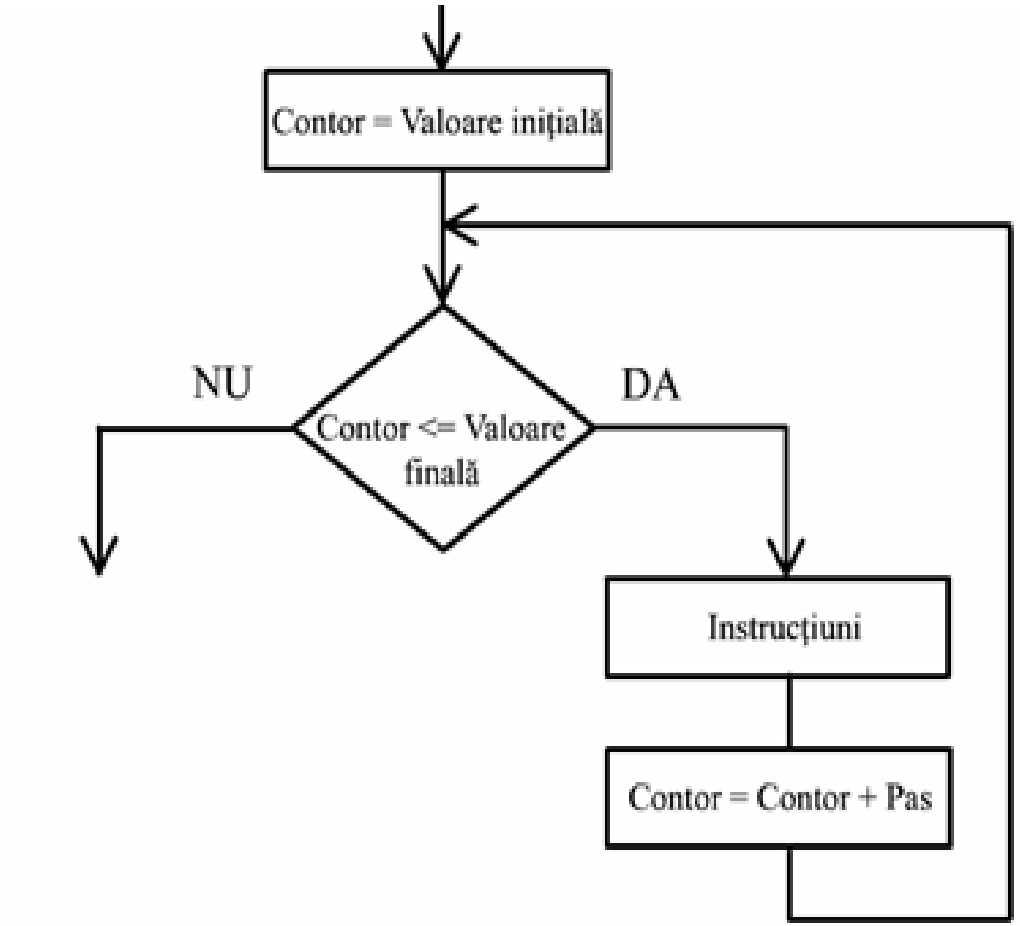
Instrucțiunea do while

- Exemplu - citire repetată până când n ia o valoare între 1 şi 1000

```
do {
    printf(“n= “);
    scanf(“%d”,&n);
} while (n>1000 || n<=0)
```

Instrucțiunea for

for (conditie_start; conditie_continuare; re-evaluare)
instrucțiune;



Instrucțiunea for

- Exemplu:

```
#include<stdio.h>
int main()
{
    int n, fact=1, i;
    printf(“introduceti n: ”);
    scanf(“%d”, &n);
    for(i=1; i<=n; i++) fact*=i;
    printf(“%d! = %d\n”,n,fact);
    return 0;
}
```

Instrucțiunea for

- Oricare din expresii poate fi vidă
- Dacă lipseşte condiția de continuare – ciclu infinit
- Nu pot lipsi ;
- Exemplu:
  - for (i=10; ;i--) instructiune
  - for (; ;) instructiune

  Echivalent cu: while(1) instructiune

Instrucțiunea for

- Expresii compuse – folosind operatorul ,
- Exemplu

```
#include<stdio.h>
int main()
{
    int n, fact, i;
    printf("introduceti n: ");
    scanf("%d", &n);
    for ( fact=1, i=1; i<=n; i++) fact*=i;
    //sau :
    // for ( fact=1, i=1; i<=n; fact*=i, i++);
    printf("%d! = %d\n",n,fact);
    return 0;
}
```