M. Caramihai, © 2020

STRUCTURI DE DATE & ALGORITMI

CURS 2

Reprezentarea datelor. Operatii cu date. Operatori. Expresii.

Declararea variabilelor

Variabilele sunt acele structuri de date ale caror valori se pot modifica in urma parcurgerii unui algoritm.

Fiecare variabila trebuie sa fie declarata, sa aiba un identificator (i.e., un nume) si sa fie de un anumit tip (de date).

d.e: **declararea variabilei “counter” ce contine o variabila numerica:**

```
counter isoftype Num
```

Initializare

- **Fiecare variabila trebuie sa fie initializata cu o valoare corespunzatoare tipului declarat.**

<u>d.e.</u>: variabila *counter* este initializata cu zero:

counter ← 0

Variabilele nu au valori implicite (nu contin informatii inainte de a fi initializate).

Numere

Declaratie:

```
my_num isoftype Num
```

`my_num` poate stoca (integer / real) numere de orice dimensiune.

***Exemple*:**

```
my_num ← 1      // OK
my_num ← -65    // OK
my_num ← 14.5   // OK

my_num ← 'a'    // EROARE
```

Caractere

Declaratie:

your_grade isoftype Char

your_grade poate stoca orice caracter alfanumeric.

Apostroful (*Single-quote*) este utilizat pentru a marca valorile de tip caracter.

Exemple:

your_grade ← 'A' *// OK*
your_grade ← 'A-' *// EROARE*
your_grade ← '9' // OK
your_grade ←9 // *EROARE*

Boolean

Declaratie :

```
this_test isoftype Boolean
```

`this_test` poate contine o valoare booleana (TRUE / FALSE).

Exemple:

```
this_test ← TRUE        //  OK
this_test ← FALSE       //  OK
this_test ← "FALSE"     // EROARE
this_test ← "TRUE"      // EROARE
```

Siruri

Sirurile pot contine mai multe caractere.

Declaratie:

```
this_string isoftype String
```

this_string poate contine un “sir” de numere sau caractere alfanumerice.

Exemple:

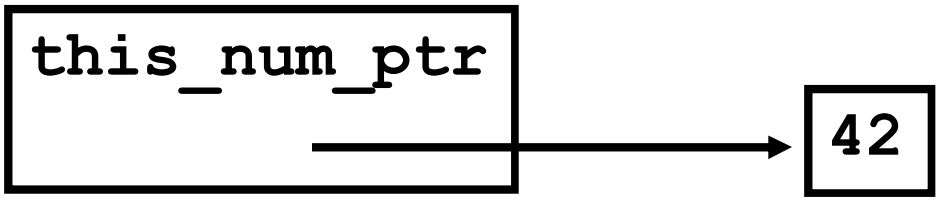
```
this_string ← “hello world” // OK
this_string ← hello world   // EROARE
this_string ← “45”          // OK
this_string ← 45            // EROARE
```

Pointeri

- **Declaratie:**

```
this_num_ptr isoftype ptr toa Num
```

`this_num_ptr` poate "cunoaste" unde o variabila de tip Num "traieste" in memorie.



Despre pointeri si mai tarziu

Exemple (asignare)

```
varsta_mea      ← 43
varsta_ta       ← my_age    // duplicat?
nota            ← '10'
var1            ← TRUE
var2            ← "Hello World"
```

TRUE / FALSE reprezinta fiecare valori booleene singulare.

Alte exemple

`a ← b + c`

la valoarea lui b din memorie

la valoarea lui c din memorie

Aduna cele doua valori

Stocheaza rezultatul in locatia de memorie a

`a ← a + 1`

la valoarea lui a din memorie

Add 1 la aceasta valoare.

Stocheaza rezultatul in locatia de memorie a

Asignarea pointerilor (1)

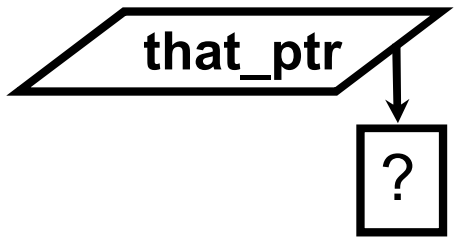
Se da:

```
this_ptr isoftype ptr toa Num
that_ptr isoftype ptr toa Num
```

Then:

```
this_ptr <- new(Num)
that_ptr <- new(Num)
```

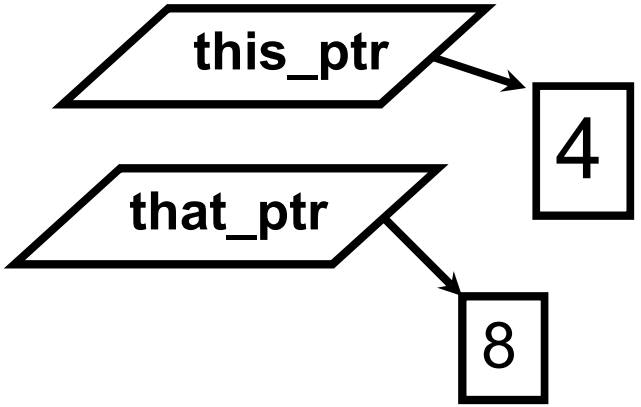
Fiecarei variabile pointer i se asociaza o noua variabila la care se va referi:



Asignarea pointerilor (2)

Rezulta:

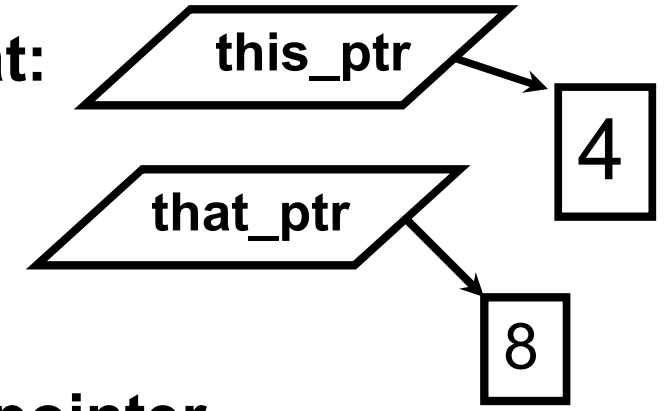
```
this_ptr^ <- 4
that_ptr^ <- 8
```



Asigneaza valori variabilelor la care fac referinta pointerii.

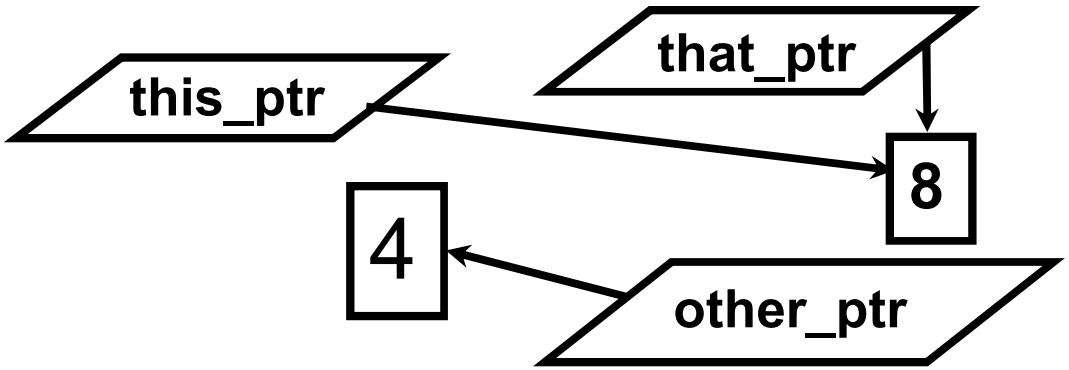
Asignarea pointerilor (3)

Fiind dat:




```
other_ptr <- this_ptr
this_ptr  <- that_ptr
```

Asigna / reasigna variabilele pointer pentru a referentia alte variabile



Operatii aritmetice

- **Operatorii aritmetici sunt utilizati in structurarea expresiilor matematice:**

  - **Adunare** **+**
  - **Scadere** **-**
  - **Inmultire** *****
  - **Impartire** **/**
  - **Impartire intreaga** **DIV**
  - ***Modulo*** **MOD**

Ordinea operatiilor

- **Ordinea operatiilor este cea standard:**
  - **Inmultire & Impartire**
  - **Adunare & Scadere**
  - **De la stanga la dreapta**

Paranteze

- Utilizare pentru controlul ordinei operatiilor.
- De exemplu:

$$x = \frac{a+b}{c-d}$$

- Dar daca scriem: `x <- a + b / c - d` ?

$$x = a + \frac{b}{c} + d$$

Aveti in vedere o solutie ?

DIV si MOD

- **DIV: pastreaza doar catul (intreg) si nu ia in considerare restul**
- **MOD: pastreaza doar restul (intreg) si nu ia in considerare catul**

d.e.:

```
8 DIV 3 = 2
8 DIV 9 = 0
7 MOD 3 = 1
7 MOD 9 = 7
```

DIV si MOD (2)

Alte exemple:

```
7.3333 DIV 1 = 7
-6.3 DIV 2 = -3


0 MOD 3 = 0
1 MOD 3 = 1
2 MOD 3 = 2
3 MOD 3 = 0
4 MOD 3 = 1
5 MOD 3 = 2
6 MOD 3 = 0
```

Input & Output

Operatori I/O permit comunicarea cu lumea din afara algoritmilor.

Operatii standard:

- **Read**
- **Print**

Print

- **Afiseaza itemii pentru user**

Sintaxa:

```
print(item_1, item_2, ..., item_n)
```

Exemple:

```
print("Introduceti parola")
print(num_one, my_char, is_Student)
```

Read

Permite preluarea itemilor de la user

Sintaxa:

```
read(item_1, item_2, ..., item_n)
```

Exemple:

```
read(optiune_meniu)
read(is_Student, name, age)
```

Input & Output – Exemple

```
num_one, num_two, average isoftype Num

// obtinerea a doua numere
   print("Please enter two numbers")
   read (num_one, num_two)

// mesaj si valoarea unei sume
   print ("Sum = ", num_one + num_two)

// sir de caractere si o valoare medie
   average <- (num_one + num_two) / 2
   print ("Average = ", average)
```

Operatori relationali

- Mai mare decat >
- Mai mare sau egal >=
- Egal =
- Mai mic sau egal <=
- Mai mic decat <
- Diferit < >

Utilizarea operatorilor relationali

- Combinand operatorii relationali cu operanzii pot fi generate valori booleene

d.e.:

4 > 5	FALSE
“apple” <= “cat”	TRUE (alfabetic)
5 <> 6	TRUE
42 = 42	TRUE
‘g’ > ‘x’	FALSE

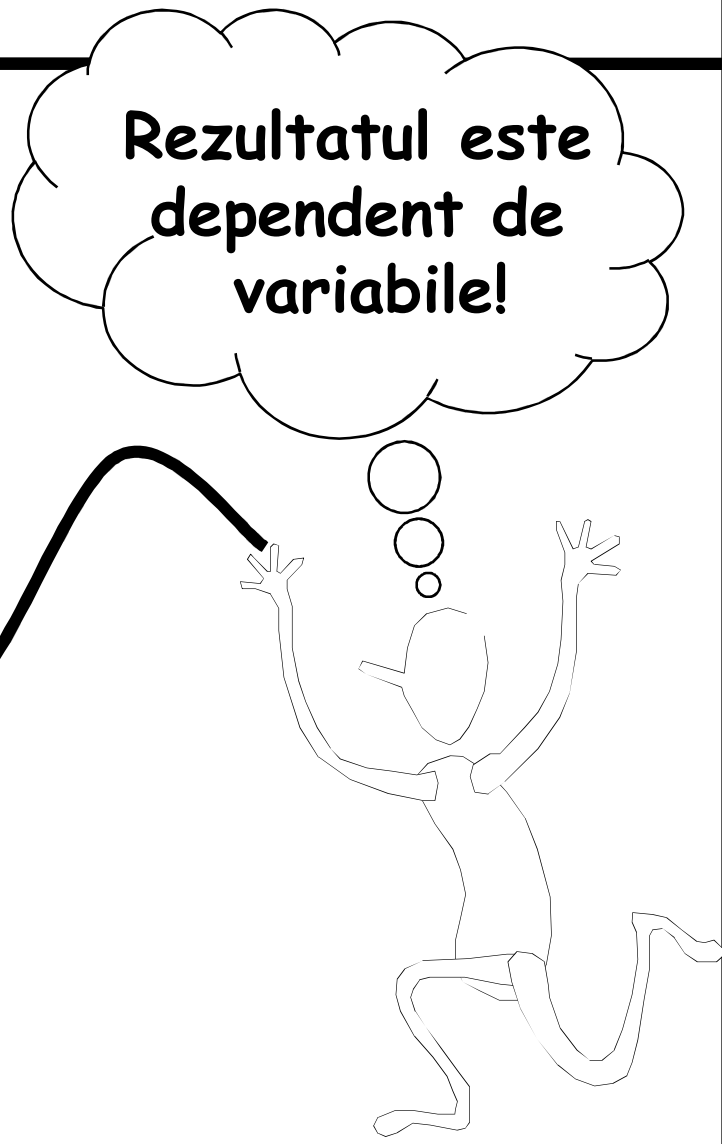
Operatori *booleeni*

- **AND**
- **OR**
- **NOT**

Operatorul AND

- AND – toate conditiile trebuie sa fie *true*

```
((5>4) AND (4>7))  is FALSE
 TRUE       FALSE
```



AND	F	T
F	F	F
T	F	T

Operatorul OR

- **OR – doar o conditie trebuie sa fie adevarata**

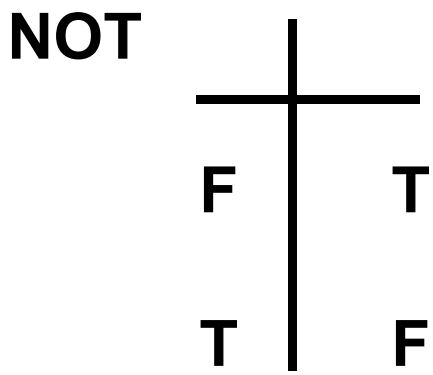
```
((5>4) OR (4>7))    is TRUE
 TRUE      FALSE
```

OR	F	T
F	F	T
T	T	T

Operatorul NOT

- NOT – inverseaza valoarea booleana

```
NOT (4>7)          is TRUE
    FALSE
```



Expresii booleene

Expresiile booleene se compun din:

- **Variabile booleene**

```
(is_too_young) // a boolean
```

- **Expresii cu operator relational**
  - `(10 < y)`
- **Expresii cu operator logic**

```
((10 >= x) AND (x < 20)) OR (y = -4)
```

Observatie:

`(0 < age < 20)` **nu este corecta si ar trebui scrisa** `((0 < age) AND (age < 20))`.

Numele variabilelor booleene

- **Numele variabilelor booleene trebuie ales a.i. sa fie clara semnificatia valorii de adevar**

- **Asa NU**

```
red_or_green
gender
pass_fail
on_off
```

- **Asa DA**

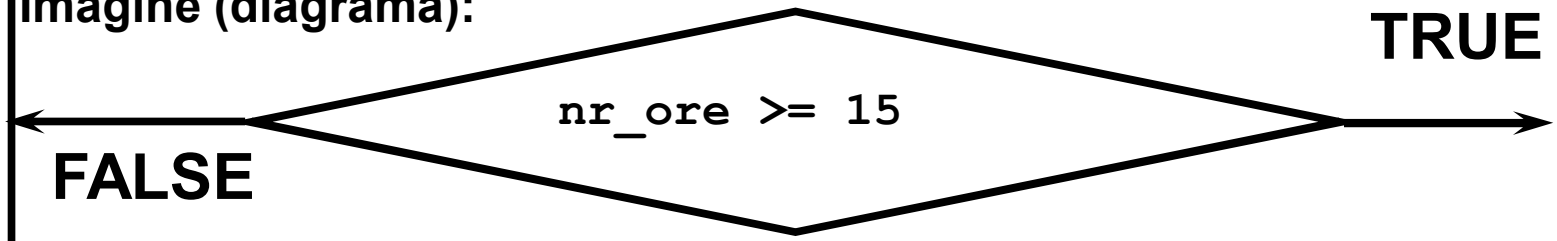
```
is_red
is_male
did_pass
is_on
```

Expresii decizionale

Evaluarea unei conditii permite luarea unei decizii bazate pe o logica elementara. Rezultatele obtinute in urma evaluarii unei conditii vor fi intotdeauna TRUE sau FALSE, i.e., valori booleene.

Limbajul natural:
"Numarul de ore este mai mare sau egal cu 15."

Imagine (diagrama):



Cod:

```
if (nr_ore >= 15) then...
```

Expresii conditionale

If – Then – Endif

Permite executarea conditionata a unor instructiuni.

***Template*:**

```
if (Expresie_booleana) then
  instructiuni
endif
```

Daca `Expresie_booleana` este TRUE, atunci se executa toate instructiunile, dupa care se continua algoritmul.

Daca `Expresie_booleana` este FALSE, atunci instructiunile pana la endif nu se executa, ci se continua algoritmul.

If – Then – Endif: Exemplu

```
wake up

if ( is_raining ) then
  stay inside and play board games
  go to see a movie
  read a book
  call friends
endif

go to sleep
```

If – Then – Else – Endif

```
if (Expresie_booleana) then
  instructiuni daca Expresie_booleana = true
else
 instructiuni daca Expresie_booleana = false
endif
```

Daca `Expresie_booleana` este TRUE, atunci se executa toate instructiunile pana la else, dupa care se continua algoritmul (dupa endif).

Daca `Expresie_booleana` este FALSE, atunci se executa toate instructiunile intre else si endif, dupa care se continua algoritmul.

Observatie:

O singura clauza poate fi executata: if sau else.

If – Then – Else – Endif: Exemplu

```
if ( is_raining ) then
  stay inside and read a book
  go to see a movie
else
  go to the park
  go swimming
endif

go to sleep
```

Expresii conditionale imbricate

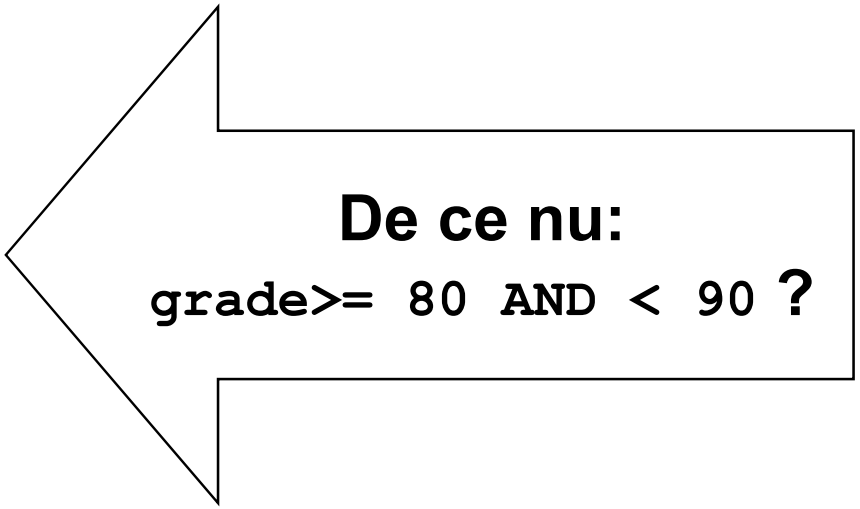
Expresiile conditionale pot fi imbricate unele cu altele:

```
if (today = Sunday) then
  if (NOT raining) then
    go to the park
  else
    stay home & read a book
  endif
else
  go to work
endif
```

Clauza optionala Elseif

In cazul in care trebuie facuta o selectie multipla:

```
if (grade >= 90) then
  print("You get an 'A'")
elseif (grade >= 80) then
  print("You get a 'B'")
elseif (grade >= 70) then
  print("You get a 'C'")
elseif (grade >= 60) then
  print("You get a 'D'")
else // the default
  print("Sorry, you get an 'F'")
endif
```



O reteta de scriere a algoritmilor

- Scriere *shell*
- Structurare logica generala
- Scrierea componentelor principale

- Verificare declaratii
- Verificare constante
- Parcurgere cod
  - erori de sintaxa
  - erori logice

Este inca de lucru dupa scrierea codului!

Un algoritm simplu

```
algorithm  Get_Circumference
// constants
    PI is 3.14159265
// declarations
    radius = 3
    diameter = 6
    circumf = 18.85
// program

// Get the data
    print("Enter the radius")
    read( radius )

// Compute circumference
    diameter <- radius * 2
    circumf <- diameter * PI

// Output the answer
    print("Diameter is: ", diameter)
    print("Circumference is: ", circumf )
endalgorithm // Get_Cirumference
```

- Scriere *shell*
- Name the algorithm
- Structurare logica generala
- Scriere componente principale
- Declarare variabile
- Definire constante
- Parcurgere cod:

```
Introducere raza: 3
Diametru: 6
Circumferinta: 18.85
```

Gata!