

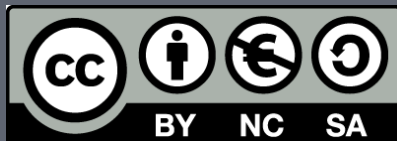


ΑΝΟΙΚΤΑ ακαδημαϊκά
μαθήματα ΠΠ



Εισαγωγή στον Προγραμματισμό Η/Υ για Χημικούς Μηχανικούς

Παρουσίαση Διαλέξεων: 8. Διαδικασίες
Καθηγητής Δημήτρης Ματαράς



Copyright © 2014 by Prof. D. S. Mataras (mataras@upatras.gr). This work is made available under the terms of the Creative Commons Attribution-NonCommercial-NoDerivs 3.0 license, <http://creativecommons.org/licenses/by-nc-nd/3.0/>

'11 IX 2039.

*Από το ανοιχτό παράθυρο άκουσα ένα από
κείνα τα ανόητα λαϊκά τραγούδια:*

Δεν έχουμε γονείς...

Δεν έχουμε ονόματα...

Είμαστε αυτόματα...

Είμαστε αυτόματα...'



Το κυρίως πρόγραμμα

[**PROGRAM** [όνομα προγράμματος]]

...

! δηλώσεις:

...

! αρχή:

...

[**CONTAINS**

[**FUNCTION** όνομα (όρισμα)

...

END FUNCTION όνομα]

[**SUBROUTINE** όνομα (όρισμα)

...

END SUBROUTINE όνομα]

...

END [**PROGRAM** [όνομα προγράμματος]]

Κυρίως Πρόγραμμα

Εσωτερικές Διαδικασίες

Τι είναι οι διαδικασίες;

```
program test  
  real:: a = 2, b
```

```
b = f(a)
```

```
call testsub(a,b)
```

```
  print *, a, b  
end program test
```

```
function f(x)  
  real:: f, x  
  f = x * x  
end function f
```

```
subroutine testsub(x, y)  
  real:: x, y  
  x = y  
  y = x + y  
end subroutine testsub
```

Τι είναι οι διαδικασίες;

```
program test  
  real:: a = 2, b
```

```
b = f(a)
```

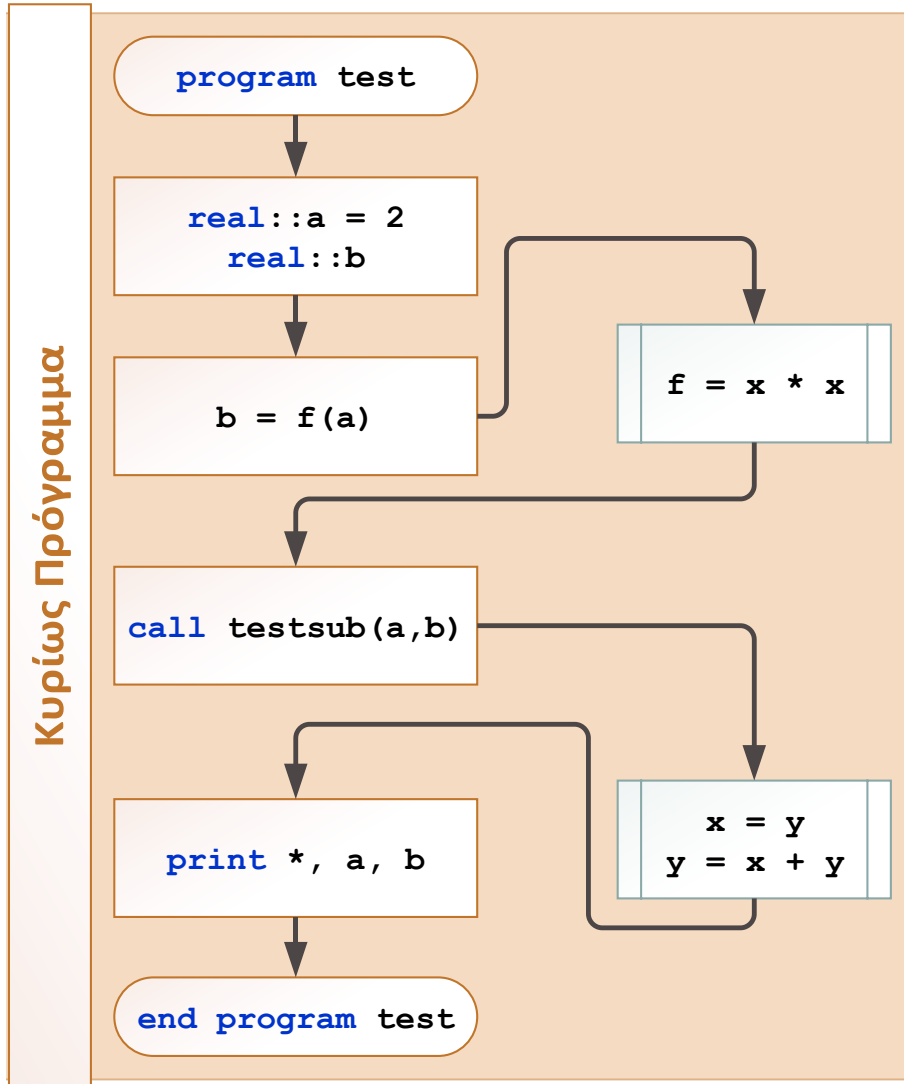
```
call testsub(a,b)
```

```
  print *, a, b  
end program test
```

```
function f(x)  
  real:: f, x  
  f = x * x  
end function f
```

```
subroutine testsub(x, y)  
  real:: x, y  
  x = y  
  y = x + y  
end subroutine testsub
```

Τι είναι οι διαδικασίες;



Μονάδες προγράμματος

για να πάρουμε μια ιδέα...

```
PROGRAM όνομα
  USE όνομα2
  CALL όνομα3( )
  y=f( )
  ...
  CONTAINS
    εσωτερικές διαδικασίες
  FUNCTION f( )
    ...
  END FUNCTION f
  SUBROUTINE όνομα3( )
    ...
  END SUBROUTINE όνομα3
END PROGRAM όνομα
εξωτερικές διαδικασίες
```

```
MODULE όνομα2
  δηλώσεις

  CONTAINS
    διαδικασίες αρθρώματος
END MODULE όνομα2
```

```
εξωτερική διαδικασία

CONTAINS
  εσωτερικές διαδικασίες
END FUNCTION ή END SUBROUTINE
```

Εξωτερική διαδικασία **SUBROUTINE**

για να πάρουμε μια ιδέα...

Εξωτερική Διαδικασία **SUBROUTINE**

SUBROUTINE όνομα [([όρισμα])]

...

! δηλώσεις:

...

! αρχή:

...

[**CONTAINS**]

[**FUNCTION** όνομα (όρισμα)

...

END FUNCTION όνομα]

[**SUBROUTINE** όνομα (όρισμα)

...

END SUBROUTINE όνομα]

Εσωτερικές Διαδικασίες

END [**SUBROUTINE** [όνομα]]

Εξωτερική διαδικασία **FUNCTION**

για να πάρουμε μια ιδέα...

```
[ τύπος ] FUNCTION όνομα ( [ όρισμα ] ) [ RESULT ( όνομα2 ) ]  
...  
! δηλώσεις:  
...  
! αρχή:  
! περιέχει υποχρεωτικά την έκφραση:  
όνομα = ... ! ή όνομα2 = εφόσον χρησιμοποιείται RESULT  
...  
[ CONTAINS ]  
  
[ FUNCTION όνομα ( όρισμα )  
...  
END FUNCTION όνομα ]  
  
[ SUBROUTINE όνομα ( όρισμα )  
...  
END SUBROUTINE όνομα ]  
...  
END [ FUNCTION [ όνομα διαδικασίας ] ]
```

Εξωτερική Διαδικασία **FUNCTION**

Εσωτερικές Διαδικασίες

Το άρθρωμα

για να πάρουμε μια ιδέα...

Άρθρωμα (MODULE)

MODULE όνομα αρθρώματος

...

! δηλώσεις:

...

[CONTAINS]

[**FUNCTION** όνομα (όρισμα)

...

END FUNCTION όνομα]

[**SUBROUTINE** όνομα (όρισμα)

...

END SUBROUTINE όνομα]

...

END [MODULE [όνομα]]

Διαδικασίες Αρθρώματος
(**MODULE PROCEDURES**)

Διαλέγουμε...

`function` ή `subroutine` ;

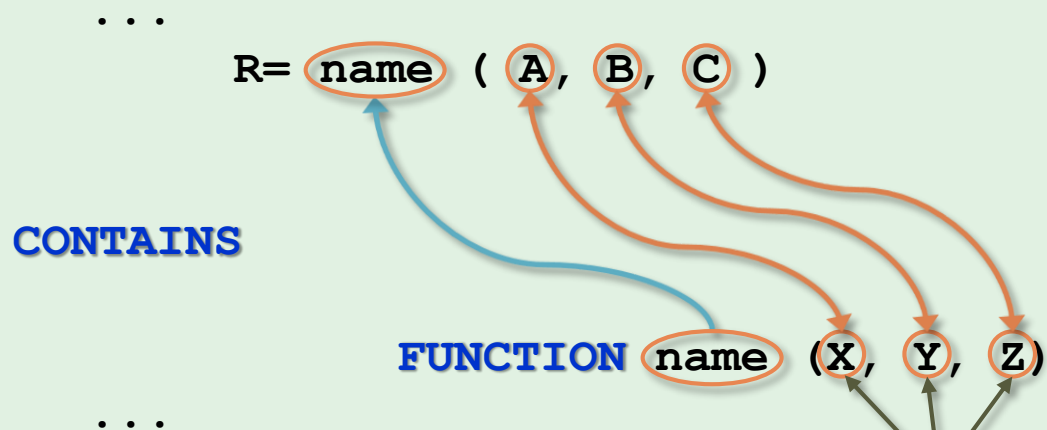
- ▶ Γενικά και οι δύο μπορούν να κάνουν τα ίδια ακριβώς πράγματα, αλλά **καλούνται διαφορετικά**

- ▶ *Ωστόσο αποφασίζουμε (και διατάσσουμε...) τα εξής:*
 - ▶ **FUNCTION** όταν θέλουμε να μας επιστρέφει **μια μοναδική τιμή** (όπως ακριβώς οι εγγενείς συναρτήσεις)
 - ▶ **SUBROUTINE** όταν θέλουμε να επιστρέφονται περισσότερες από μια τιμές

- ▶ **Εξωτερικές** ή **Εσωτερικές** διαδικασίες;
 - ▶ **Εσωτερικές:** μόνο για απλούς υπολογισμούς (ορισμούς συναρτήσεων κλπ) που χρησιμοποιούνται στον ξενιστή
 - ▶ **Εξωτερικές:** για σοβαρότερους, επαναχρησιμοποιήσιμους υπολογισμούς.

Διαδικασία FUNCTION

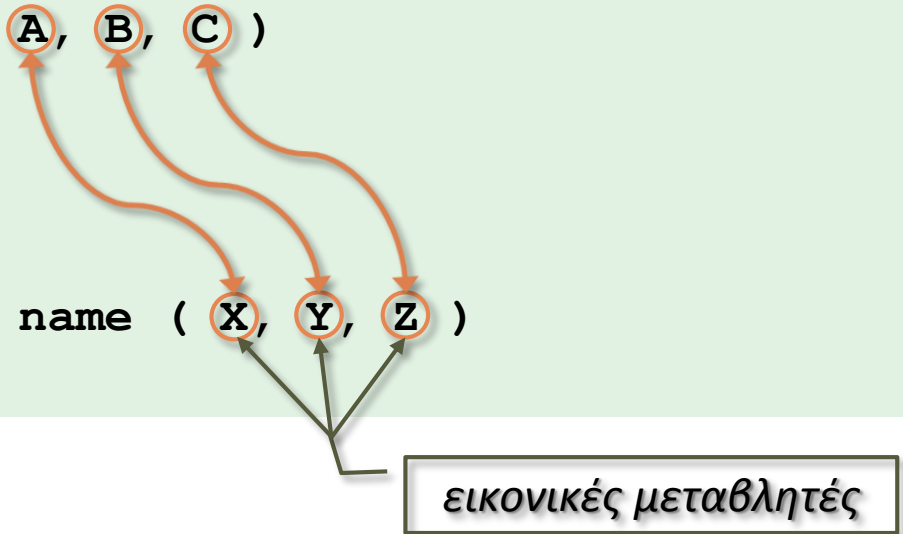
```
[ τύπος ] FUNCTION όνομα ( [ όρισμα ] ) [ RESULT ( όνομα2 ) ]  
...  
! δηλώσεις:  
...  
! εκφράσεις:  
! περιέχει υποχρεωτικά την έκφραση:  
όνομα = ...  
! ή όνομα2 = εφόσον χρησιμοποιείται RESULT  
...  
END [ FUNCTION [ όνομα ] ]
```



Διαδικασία SUBROUTINE

```
SUBROUTINE όνομα [ ([όρισμα]) ]  
...  
! δηλώσεις:  
...  
! εκφράσεις:  
...  
END [ SUBROUTINE [όνομα] ]
```

```
...  
CALL name ( A, B, C )  
...  
CONTAINS  
SUBROUTINE name ( X, Y, Z )  
...
```



Ιδιότητα `INTENT`

για να αποφύγουμε τις παρενέργειες

δήλωση τύπου, `INTENT` (είδος) :: λίστα μεταβλητών

το είδος : μπορεί να είναι:

- ▶ **IN** : για εικονική μεταβλητή εισόδου. Η τιμή της δεν μπορεί να αλλάξει στην έκταση της διαδικασίας. Δηλαδή δεν μπορεί να εμφανίζεται στο αριστερό σκέλος έκφρασης, στην έκταση της διαδικασίας.
- ▶ **OUT** : για εικονική μεταβλητή εξόδου. Πρέπει να λάβει τιμή, δηλαδή πρέπει υποχρεωτικά να εμφανίζεται στο αριστερό σκέλος έκφρασης, στην έκταση της διαδικασίας.
- ▶ **INOUT** : για εικονική μεταβλητή εισόδου-εξόδου. Μπορεί να αλλάξει τιμή στην έκταση της διαδικασίας (είναι το προκαθορισμένο είδος).

PURE FUNCTION όνομα (όρισμα)

! υποχρεωτική δήλωση `INTENT (IN)`

.....

PURE SUBROUTINE όνομα (όρισμα)

! Υποχρεωτική δήλωση `INTENT (IN, OUT ή INOUT)`

.....

Σφάλμα αποκοπής

$$f(x) = \left(1 + \frac{1}{x}\right)^x$$

`selected_real_kind`

```
program truncation
  implicit none
  real::x; integer::i
  integer,parameter::max_power=10
  do i=0, max_power
    x = 10.0**i
    print '(f15.1,2f15.12)', x, f(x), g(x)
  enddo
contains
  real function f(x)
    real,intent(in)::x
    integer,parameter::max_kind=selected_real_kind(max_power+1)
    f = (1 + 1/real(x,max_kind))**x
  end function f
  real function g(x)
    real,intent(in)::x
    g = (1 + 1/x)**x
  end function g
end program truncation
```

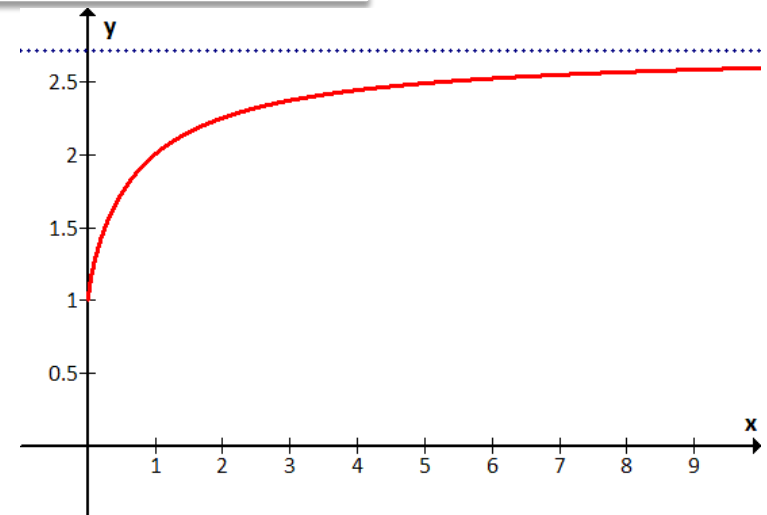
Συσχέτιση μέσω ξενιστή
(Host Association)

Συσχέτιση μέσω ορίσματος
(call association)

Σφάλμα αποκοπής

$$f(x) = \left(1 + \frac{1}{x}\right)^x$$

1.0	2.000000000000	2.000000000000
10.0	2.593742370605	2.593743085861
100.0	2.704813718796	2.704811334610
1000.0	2.716923952103	2.717050790787
10000.0	2.718145847321	2.718596935272
100000.0	2.718268156052	2.721962213516
1000000.0	2.718280553818	2.595226764679
10000000.0	2.718281745911	3.293967723846
100000000.0	2.718281745911	1.000000000000
1000000000.0	2.718281984329	1.000000000000
10000000000.0	2.718281984329	1.000000000000



Σειρά 4 Άσκηση 1 (2012)

Εικονικοί Πίνακες Ρητής Μορφής 1

```
program lab4_ask1
  implicit none
  ! δηλώσεις:
  integer, parameter :: n = 297
  real                :: x(n)
  integer             :: i

  call execute_command_line('chcp 1253')
  ! α) διάβασμα και εκτύπωση πίνακα
  open(1, file = 'data.dat')
    read(1,*) x !διαβάζω όλο τον πίνακα
  close(1)

  print *, (x(i), i = 1, n) !τυπώνω με υπονοούμενη επανάληψη

  ! β) διάταξη και στατιστική
  x = f(x, n)
  call stats(x, n)
```

Σειρά 4 Άσκηση 1 (2012)

Εικονικοί Πίνακες Ρητής Μορφής 2

contains

```
function f(x, n)
  integer, intent(in) :: n
  real, intent(in) :: x(n)
  integer :: i, j, minp
  real :: temp, f(n)
  f = x
  do i = 1, n - 1
    minp = i
    ! βρές το στοιχείο με την μικρότερη τιμή
    do j = i, n
      if(f(j) < f(minp)) minp = j
    end do
    ! αντιμετάθεσε με το στοιχείο με ενδείκτη i
    if(f(minp) /= f(i)) then
      temp = f(i); f(i) = f(minp); f(minp) = temp
    end if
  end do
end function f
```

Σειρά 4 Άσκηση 1 (2012)

Εικονικοί Πίνακες Ρητής Μορφής 3

```
end program lab4_ask1
```

```
subroutine stats(x, n) !εξωτερική διαδικασία
```

```
integer, intent(in) :: n
```

```
real, intent(in) :: x(n)
```

```
real :: median, sdev, average
```

```
average = sum(x) / n
```

```
sdev = sqrt((sum(x * x) - sum(x)**2 / n) / (n + 1))
```

```
median = x(n / 2 + 1)
```

```
print *, 'ο μέσος όρος είναι      :', average
```

```
print *, 'η τυπική απόκλιση είναι:', sdev
```

```
print *, 'η διάμεσος τιμή είναι   :', median
```

```
end subroutine stats
```

Γεννήτρια ψευδοτυχαίων

...σχεδόν σωστά

```
program randomizer  !n τυχαίοι αριθμοί στο διάστημα [a,b]
  implicit none
  ! δηλώσεις:
  integer :: i, n; real:: a, b, harvest

  ! αρχή:
  print *, 'How many numbers'; read *, n
  print *, 'Give the range of numbers: [a,b]'; read *, a, b

  open(1, file='random_numbers.dat')
  call init_random_seed()  !αρχικοποίηση της γεννήτριας
  do i = 1, n
    call random_number(harvest)
    harvest = a + harvest * (b - a)  ![0,1] -> [a,b]
    write (1, *) harvest
  enddo
  close(1)
```

contains

Γεννήτρια ψευδοτυχαίων

...σχεδόν σωστά

contains

```
subroutine init_random_seed() ! αρχικοποίηση γεννήτριας
  integer :: i, n = 12, clock
  integer :: seed(12) = 0

  call system_clock(count = clock)

  seed = [(clock + 37 * i - 1, i = 1, n)]

  call random_seed(put = seed) ! χρήση τυχαίου σπόρου

end subroutine

end program randomizer
```

Waste Allocation Load Lifter – Earth Class



Σημείωμα Αναφοράς

Copyright Πανεπιστήμιο Πατρών, Όνομα μέλους ή μελών ΔΕΠ 2014:
Δημήτριος Ματαράς. «Εισαγωγή στον Προγραμματισμό Η/Υ». Έκδοση: 1.0.
Πάτρα 2014. Διαθέσιμο από τη δικτυακή διεύθυνση:
<https://eclass.upatras.gr/courses/CMNG2178>.

Χρηματοδότηση

- ▶ Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στο πλαίσιο του εκπαιδευτικού έργου του διδάσκοντα.
- ▶ Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο Πανεπιστήμιο Αθηνών**» έχει χρηματοδοτήσει μόνο την αναδιαμόρφωση του εκπαιδευτικού υλικού.
- ▶ Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.



Σημείωμα Αδειοδότησης

Το παρόν υλικό διατίθεται με τους όρους της άδειας χρήσης Creative Commons Αναφορά, Μη Εμπορική Χρήση Παρόμοια Διανομή 4.0 [1] ή μεταγενέστερη, Διεθνής Έκδοση. Εξαιρούνται τα αυτοτελή έργα τρίτων π.χ. φωτογραφίες, διαγράμματα κ.λ.π., τα οποία εμπεριέχονται σε αυτό και τα οποία αναφέρονται μαζί με τους όρους χρήσης τους στο «Σημείωμα Χρήσης Έργων Τρίτων».



[1] <http://creativecommons.org/licenses/by-nc-sa/4.0/>

Ως **Μη Εμπορική** ορίζεται η χρήση:

- που δεν περιλαμβάνει άμεσο ή έμμεσο οικονομικό όφελος από την χρήση του έργου, για το διανομέα του έργου και αδειοδόχο
- που δεν περιλαμβάνει οικονομική συναλλαγή ως προϋπόθεση για τη χρήση ή πρόσβαση στο έργο
- που δεν προσπορίζει στο διανομέα του έργου και αδειοδόχο έμμεσο οικονομικό όφελος (π.χ. διαφημίσεις) από την προβολή του έργου σε διαδικτυακό τόπο

Ο δικαιούχος μπορεί να παρέχει στον αδειοδόχο ξεχωριστή άδεια να χρησιμοποιεί το έργο για εμπορική χρήση, εφόσον αυτό του ζητηθεί.