ΑΝΟΙΚΤΑ ακαδημαϊκά μαθήματα ΠΠ

Εισαγωγή στον Προγραμματισμό Η/Υ για Χημικούς Μηχανικούς

Παρουσίαση Διαλέξεων: 10. Αντικείμενα

Καθηγητής Δημήτρης Ματαράς

'Ο Βασιλιάς όρμησε με όλες τις σκληρές συντεταγμένες του και τις μέσες τιμές του, σκουντούφλησε σε ένα σκοτεινό δάσος από ρίζες και λογαρίθμους, χρειάστηκε να οπισθοχωρήσει, ύστερα συνάντησε το ζώο σε ένα πεδίο άρρητων αριθμών (F_1) και το χτύπησε τόσο άσκημα που έπεσε κατά δύο δεκαδικές θέσεις και έχασε ένα Ε, αλλά το ζώο γλίστρησε γύρω από ένα ασύμπτωτο και κρύφτηκε σε έναν ν-διάστατο ορθογώνιο φασικό χώρο, έπαθε διαστολή και βγήκε αφρίζοντας παραγοντικά, και πέφτοντας επάνω στον Βασιλιά τον χτύπησε πολύ σκληρά.'

Στανισλάβ Λεμ, '*Κυβεριάδα*'

Παράγωγοι Τύποι Δεδομένων

πως δημιουργούνται

```
!δομή δημιουργίας νέου τύπου δεδομένων
TYPE   [,ιδιότητα πρόσβασης][::] όνομα τύπου
        δήλωση1[=αρχική τιμή]
       [δήλωση2[=αρχική τιμή]]
          ...
END TYPE
```

ιδιότητα πρόσβασης	:	**PUBLIC** ή **PRIVATE**
δήλωση1,2,..	:	δηλώσεις τύπου (π.χ. **REAL**, **INTEGER**...)

```
!πρόταση δήλωσης ονομάτων που ανήκουν στο νέο τύπο
TYPE (όνομα τύπου) [,ιδιότητες] :: όνομα [,όνομα,..]
```

ιδιότητες	:	**DIMENSION** ή **PARAMETER**

Απλό παράδειγμα

```
program ddt
   implicit none
   type point
      real:: x, y
   end type point

   type circle
      type (point):: center
      real:: radius
   end type circle

!  δηλώσεις:
   type (point)    :: p
   type (circle)   :: c

!  αρχή:
   c%center%x = 2; c%center%y = 2
   c%radius = 2
   print *, c
```

Απλό παράδειγμα

```
    p = point (0,0)
    c = circle(p,1)
    print *, c

    c = circle(point(1,1),1)
    print *, c

end program ddt
```

```
   2.00000000        2.00000000        2.00000000
   0.00000000        0.00000000        1.00000000
   1.00000000        1.00000000        1.00000000
```

Επέκταση παράγωγων τύπων δεδομένων F03

```
program type_extension
    implicit none
    type :: point
        real :: x = 0, y = 0
    end type
    type, extends(point):: point3d
        real :: z = 0
    end type
!   δηλώσεις:
    character(40)   :: frm
    type (point)    :: my_point
    type (point3d)  :: new_point
!   αρχή:
    write (frm, *) '("point = ",*(f7.4,:,","))'
    my_point  = point(1,2); print frm, my_point
    print frm, new_point
    new_point = point3d(my_point,3); print frm, new_point
! ή new_point = point3d(1,2,3)
end program type_extension
```

```
point =  1.0000, 2.0000
point =  0.0000, 0.0000, 0.0000
point =  1.0000, 2.0000, 3.0000
```

Φακελώστε τους...

I.

```
type student     ! παράγωγος τύπος (derived data type)
   character(10):: name, last_name  !όνομα φοιτητή
   integer      :: am               !αριθμός μητρώου
   real         :: grade            !βαθμός
end type
! δηλώσεις:
 type(student):: a_student
 character(40):: using_form
! αρχή:
 write (using_form,*) '(2a9, i5, f5.1)' ! format
```

```
! κατασκευαστής παράγωγου τύπου
a_student = student('Dimitris', 'Mataras', 1000, 5.0)
print using_form, a_student
```

```
! κατασκευαστής με λέξεις κλειδιά (keywords)
a_student = student(name='Dimitris',last_name='Mataras',&
                  &    am=1000,grade=5.0)
print using_form, a_student
```

Φακελώστε τους...

II.

```
! δίνουμε τιμές σε κάθε συνιστώσα ξεχωριστά
a_student%name      = 'Dimitris'
a_student%last_name = 'Mataras'
a_student%am        = 1000
a_student%grade     = 5.0
print using_form, a_student
```

```
! τυπώνουμε κάθε συνιστώσα ξεχωριστά
print using_form,  a_student%name, a_student%last_name, &
                 &  a_student%am, a_student%grade
```

```
end
```

```
Dimitris Mataras   1000  5.0
Dimitris Mataras   1000  5.0
Dimitris Mataras   1000  5.0
Dimitris Mataras   1000  5.0
```

Φακελώστε τους όλους...

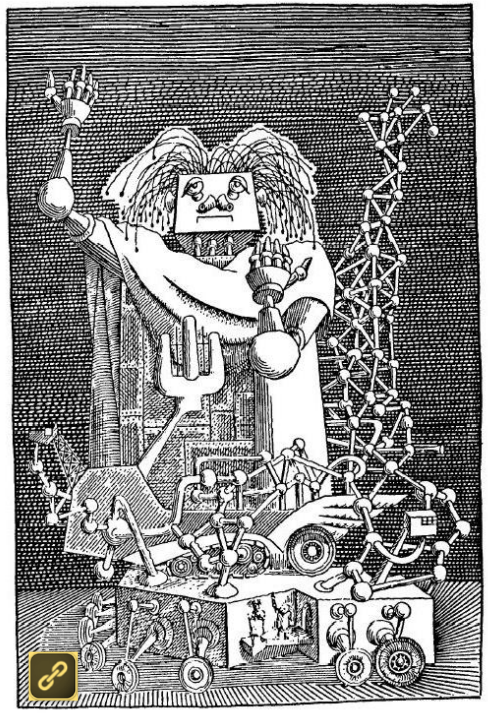
```
type list !Παράδειγμα 9-6
   character(20)::name      !όνομα φοιτητή
   character(20)::surname   !όνομα φοιτητή
   integer       ::am       !αριθμός μητρώου
   real          ::grade    !βαθμός
end type
! δηλώσεις:
   type(list), dimension(5):: students
! αρχή:
   call execute_command_line('chcp 1253')
   do i = 1, 5
      print *,'δώσε το επώνυμο και τον ΑΜ'
      read *, students(i)
   end do
   print '(2a10,i5,f5.1)',students

end
```

```
JAMES     HENDRIX    1000  9.0
ERNESTO   GUEVARA    1001  5.0
BORIS     VIAN       1002  9.5
DAVID     BOWIE      1003  8.0
ROBERT    DYLAN      1004  4.5
```

‘Θα σου χορηγήσω τυχαία πρόσβαση στην καρδιά μου,
Κι εσύ θα μου ψιθυρίσεις όλες τις σταθερές της αγάπης σου
Και έτσι τα δυο μας όλα τα λήμματα του έρωτα θα αποδείξουμε,
Και στην τομή μας δεν θα υπάρξει διαίρεση ποτέ.

Μη με απορρίψεις, όχι ! γιατί τι θα μου απομείνει;
εκθέτες, μερικά δεκαδικά, αρθρώματα, τρόποι,
μια ρίζα του δύο, ένας τόρος και ένας κόμβος:
το αντίστροφο του στίχου μου, μια περιοχή μέτρου μηδέν’

Στανισλάβ Λεμ, ‘*Κυβεριάδα*’



Αρθρώματα

```
MODULE όνομα
   [δηλώσεις]
    ...
[CONTAINS]
   [διαδικασίες αρθρώματος]
    ...
END[MODULE[όνομα]]
```

```
!στο πρόγραμμα ή τη διαδικασία που το χρησιμοποιεί:
   USE όνομα αρθρώματος [,ONLY: λίστα ονομάτων]
```

Περιορισμός πρόσβασης στη χρήση

```
!περιορισμός πρόσβασης στην πηγή:
   PUBLIC [[::] λίστα]
   δήλωση τύπου, PUBLIC::λίστα

   PRIVATE [[::] λίστα]
   δήλωση τύπου, PRIVATE::λίστα
```

Περιορισμός πρόσβασης στην πηγή

Ένα άρθρωμα για τον Ερατοσθένη

I.

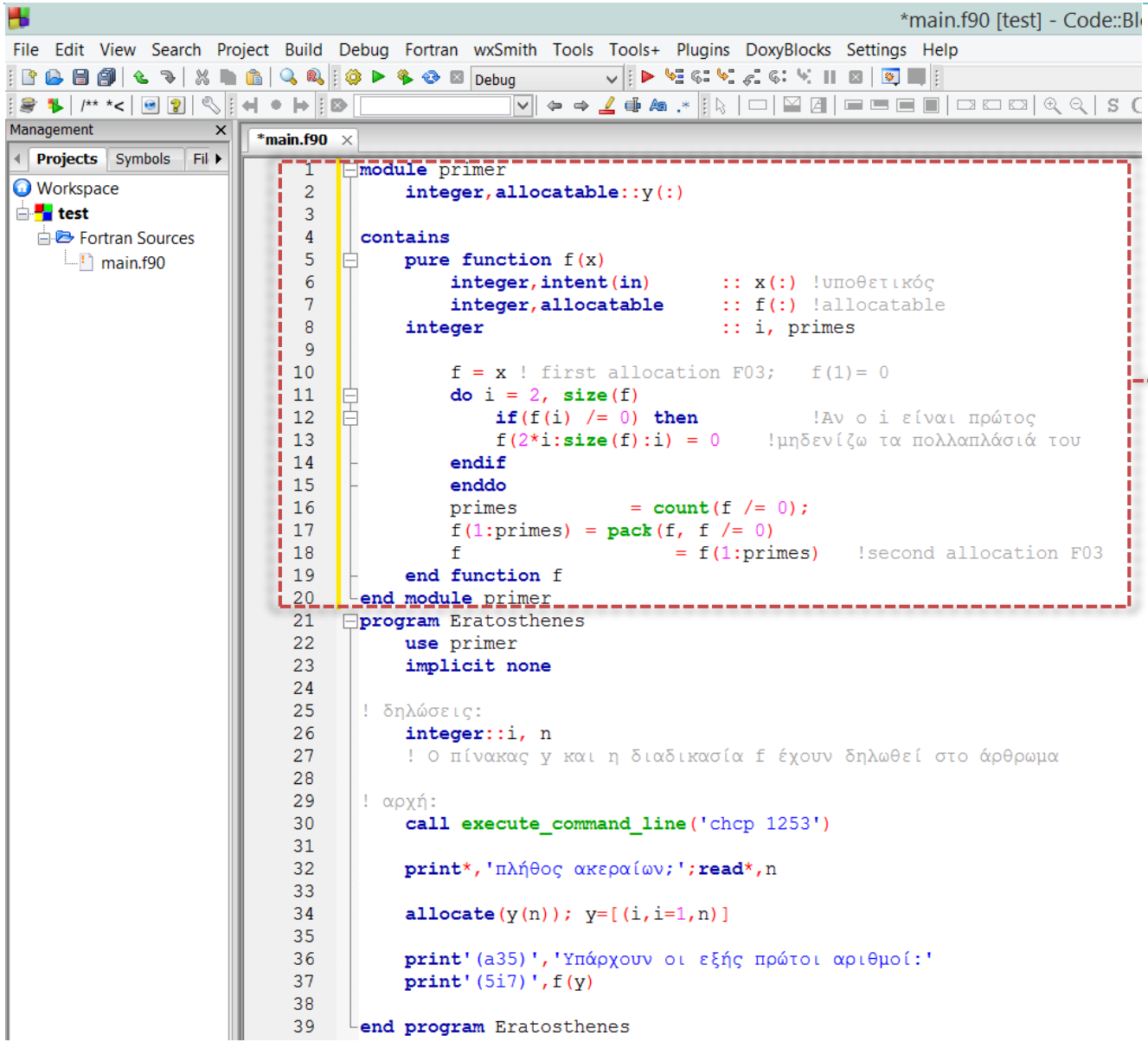
```
program Eratosthenes
   use primer
   implicit none

! δηλώσεις:
   integer::i, n
   ! Ο πίνακας y και η διαδικασία f έχουν δηλωθεί στο άρθρωμα

! αρχή:
   call execute_command_line('chcp 1253')

   print*,'πλήθος ακεραίων;';read*,n

   allocate(y(n)); y=[(i,i=1,n)]

   print'(a35)','Υπάρχουν οι εξής πρώτοι αριθμοί:'
   print'(5i7)',f(y)

end program Eratosthenes
```

Το άρθρωμα του Ερατοσθένη

II.

```
module primer
   integer,allocatable::y(:)

contains
   pure function f(x)
      integer,intent(in)    :: x(:)!υποθετικός
      integer,allocatable   :: f(:)!allocatable
      integer               :: i, primes

      f = x ! first allocation F03;       f(1)= 0
      do i = 2, size(f)
         if(f(i) /= 0) then          !Αν ο i είναι πρώτος
            f(2*i:size(f):i) = 0     !μηδενίζω τα πολλαπλάσιά του
         endif
      enddo
      primes         = count(f /= 0);
      f(1:primes)    = pack(f, f /= 0)
      f              = f(1:primes)   !second allocation F03
   end function f
end module primer
```

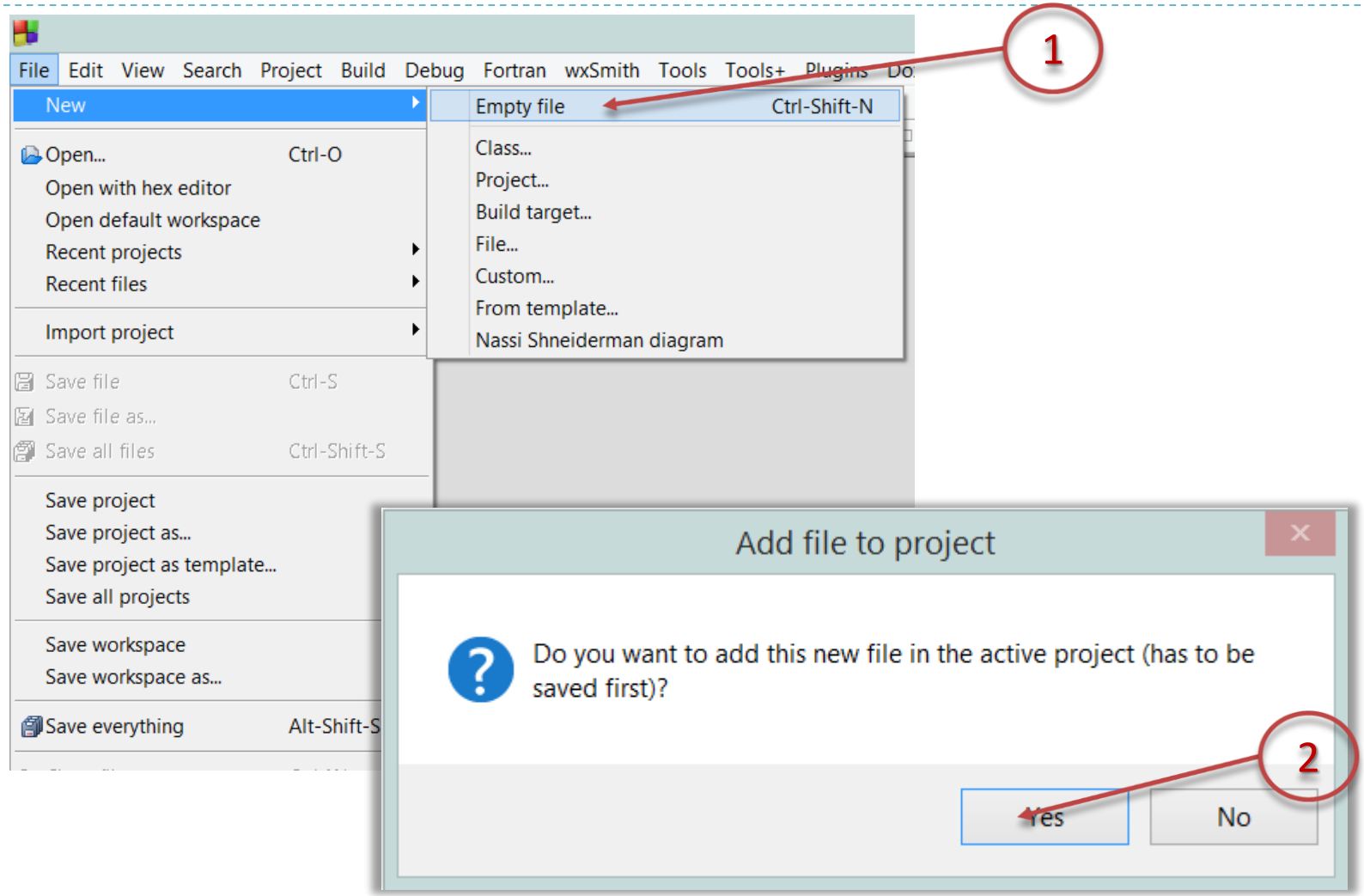
Ή απλά γράφουμε το module στο ίδιο αρχείο με το πρόγραμμα



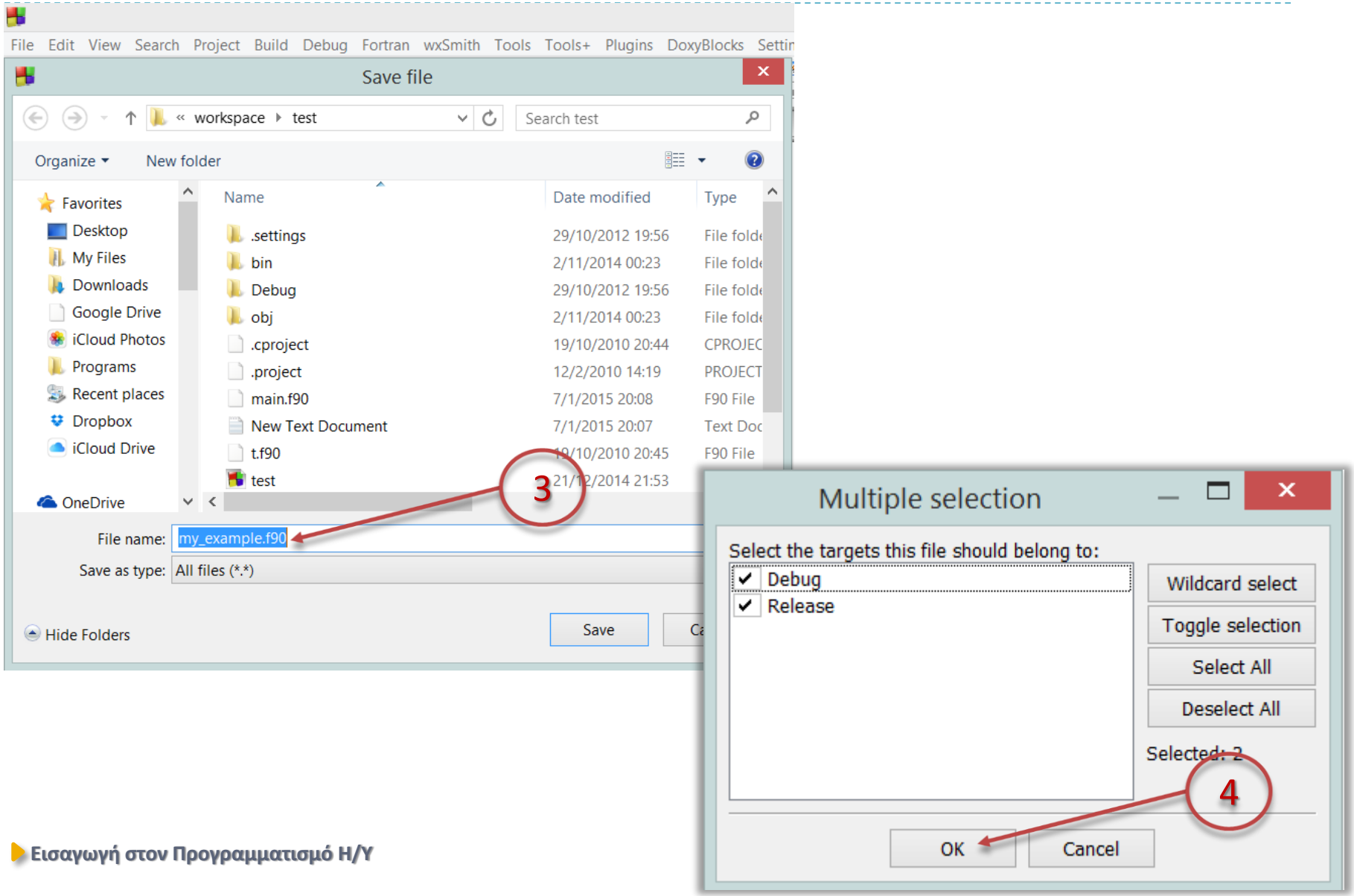
Πρώτα βάζουμε το module,

και μετά βάζουμε το κυρίως πρόγραμμα

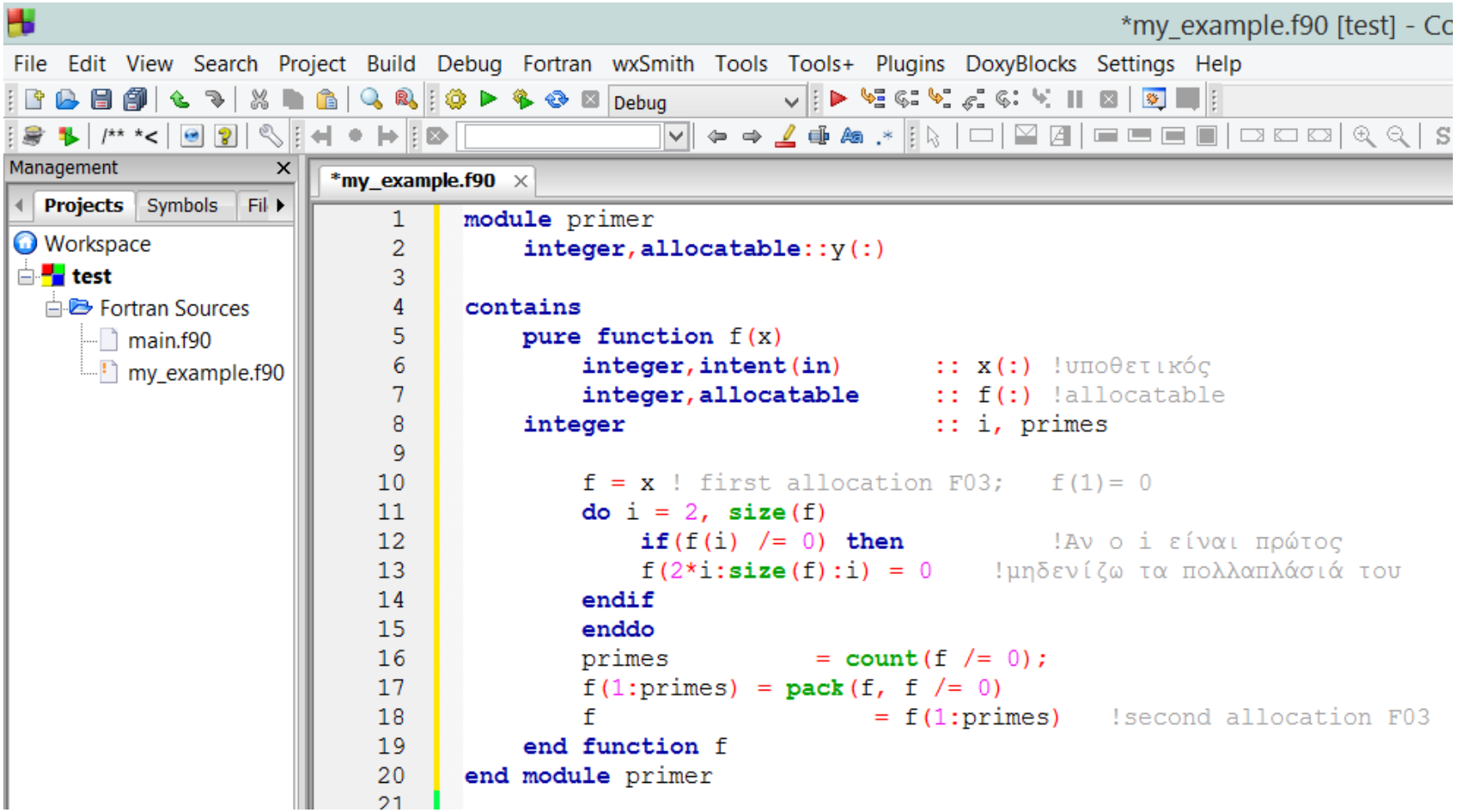
Ή Δημιουργούμε καινούριο module σε ξεχωριστό αρχείο Ι.



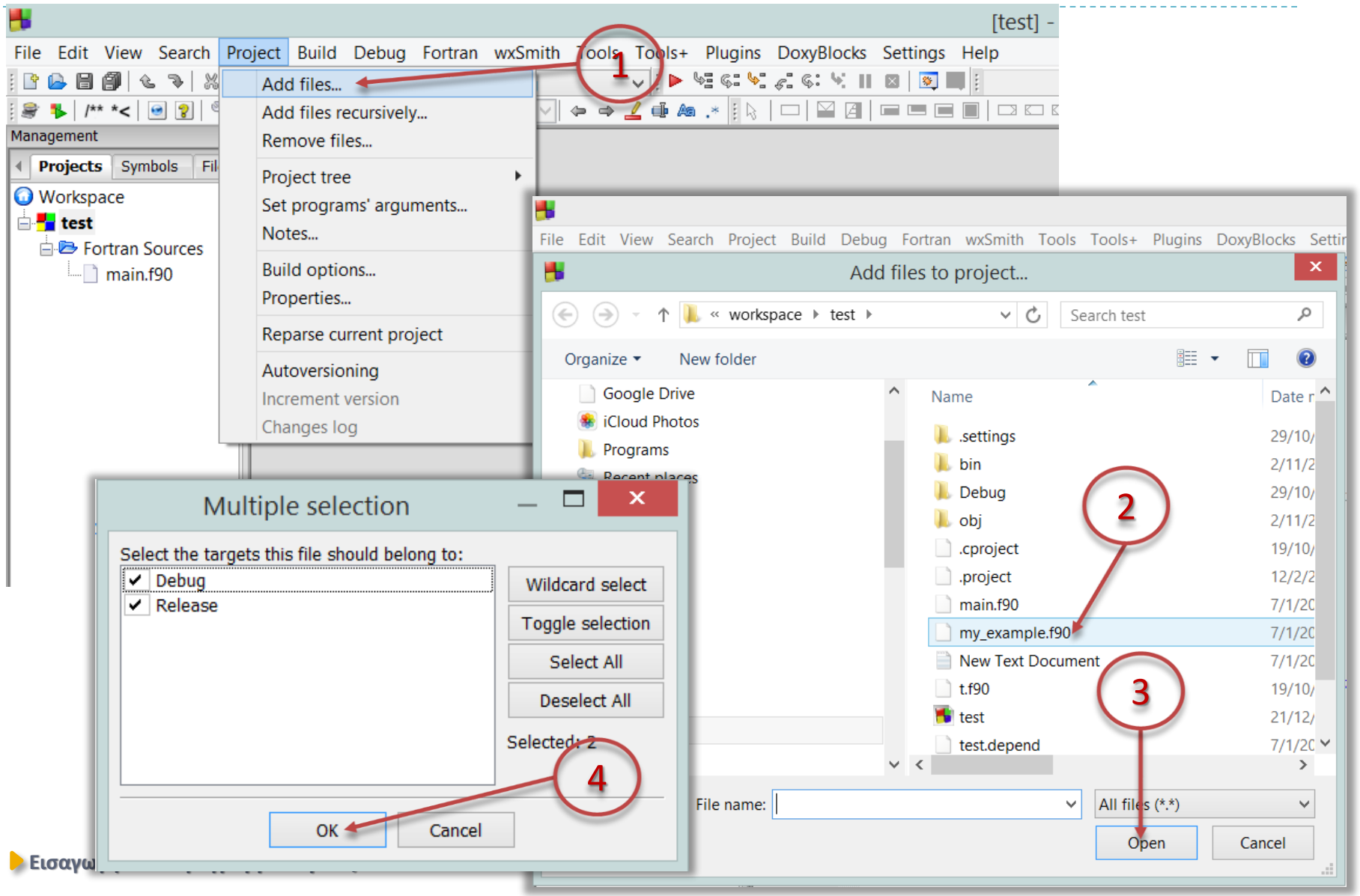
Ή Δημιουργούμε καινούριο module σε ξεχωριστό αρχείο II.



Ή Δημιουργούμε καινούριο module σε ξεχωριστό αρχείο III.



Ή προσθέτουμε ήδη υπάρχον module που βρίσκεται σε ξεχωριστό αρχείο



Διαδικασίες παράγωγων τύπων F03

```
module points
    implicit none
    type:: point
        real :: x = 0 , y = 0
    contains
        procedure, pass :: plus
    end type

contains
    type(point) function plus(this, that)
        class(point) :: this, that
        plus%x = this%x + that%x
        plus%y = this%y + that%y
    end function plus

end module points
```

Διαδικασίες παράγωγων τύπων F03

```
program type_bound
   use points
   implicit none

!  δηλώσεις:
   character(40)   :: frm
   type (point)    :: this, that

!  αρχή:
   write (frm, *) '("point = ",*(f7.4,:,","))'
   this = point(1,2)
   that = point (1,2)

   that = this%plus(that)

   print frm, this
   print frm, that

end program type_bound
```

κλήση διαδικασίας

Straight insertion sort

I.

```
module sorter
   implicit none
contains
   pure function insort(array) result(f)
      real, intent(in)  :: array(:)
      real              :: f(size(array))
      integer :: i, j; real :: temp
      f = array
      do j = 2, size(array)
         temp = f(j)
         do i = j - 1, 1, -1
            if (f(i) <= temp) then
               exit
            else
               f(i+1) = f(i)
               f(i)   = temp
            endif
         enddo; enddo
     end function insort
end module sorter
```

Straight insertion sort

II.

```
module randomizer
    implicit none
contains
    function harvest(n, a, b)
        integer, intent(in) :: n
        real, intent(in)    :: a, b
        real                :: harvest(n)
        call init_random_seed; call random_number(harvest)
        harvest = a + harvest * (b - a)
    contains
        subroutine init_random_seed
            integer :: i, n, clock
            integer,allocatable :: seed(:)
            call random_seed(size = n)
            allocate (seed(n))
                call system_clock(count = clock)
                seed = clock + 37 * [(i - 1, i = 1, n)]
                call random_seed(put = seed)
            deallocate(seed)
        end subroutine init_random_seed
    end function harvest
end module randomizer
```

Straight insertion sort

III.

```
program sort_a_random_array
   use sorter; use randomizer
   real, allocatable :: a(:) !table to be sorted
   real              :: x1 = 0, x2 = 1

   do
      print *,'How many numbers [n > 0 or 0 to stop]'
      read *, n
      if(n > 0) then
         exit
      else
         stop
      endif
   enddo
   do
      print *, 'Give the range of numbers:[x1, x2 &
               & where x2 > x1]'; read *, x1, x2
      if(x2 > x1) exit
   enddo
```

Straight insertion sort


```
   a = harvest(n, x1, x2) !allocation

   print '(/,a)', ' Before Sorting:'; call a_writer(a)

!  call sorting function
   a = insort(a)

   print '(/,a)', ' After Sorting:'; call a_writer(a)

contains

!  write table of size n, 5 values in a row
   subroutine a_writer(array)
      real, intent(in):: array(:)
      character(40)    :: frm = '(5(f10.7,:," "))'
      write(*,frm) array
   end subroutine a_writer

end program sort_a_random_array
```

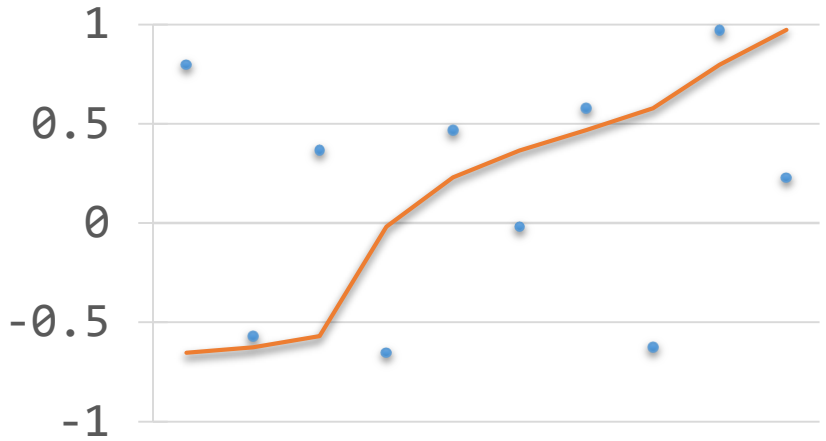
Straight insertion sort

V.

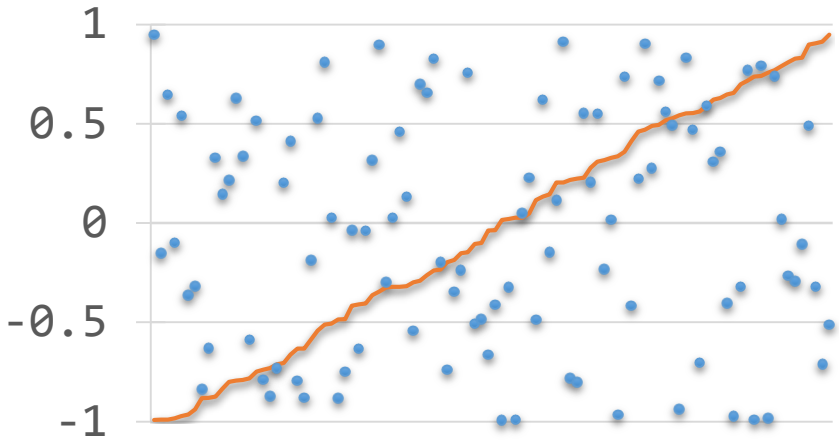
```
 How many numbers [n > 0 or 0 to stop]
25
 Give the range of numbers: [x1, x2 where x2 > x1]
-1 1

 Before Sorting:
 0.3174344 -0.0661023 -0.1160954  0.5332836  0.8618876
-0.8605964 -0.4531953 -0.0687138 -0.4568574 -0.4090136
 0.3322257  0.2716280  0.0816292 -0.3824455 -0.6653461
 0.9477333 -0.1439825 -0.9396762  0.9329839  0.0912217
-0.2051402 -0.2454767  0.5726677  0.4262367 -0.8073467

 After Sorting:
-0.9396762 -0.8605964 -0.8073467 -0.6653461 -0.4568574
-0.4531953 -0.4090136 -0.3824455 -0.2454767 -0.2051402
-0.1439825 -0.1160954 -0.0687138 -0.0661023  0.0816292
 0.0912217  0.2716280  0.3174344  0.3322257  0.4262367
 0.5332836  0.5726677  0.8618876  0.9329839  0.9477333
```

Straight insertion sort

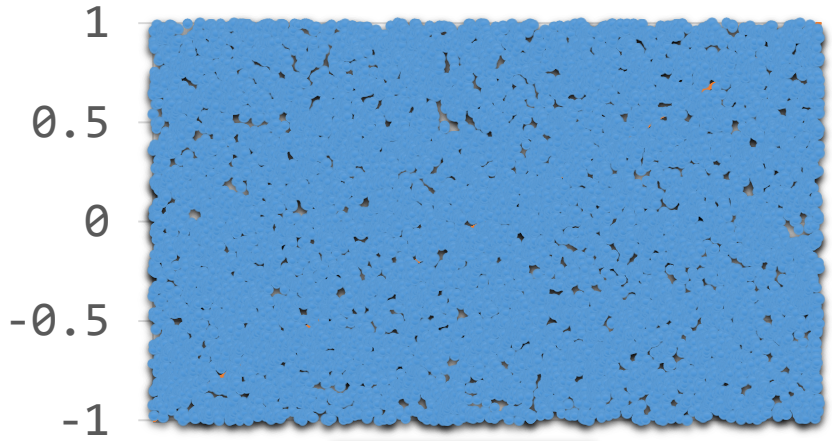
πλοτάρουμε με το excel



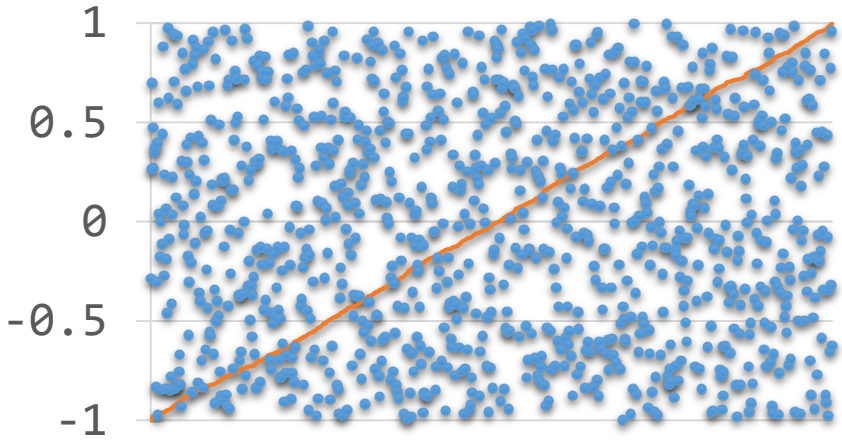
n =10



n =100



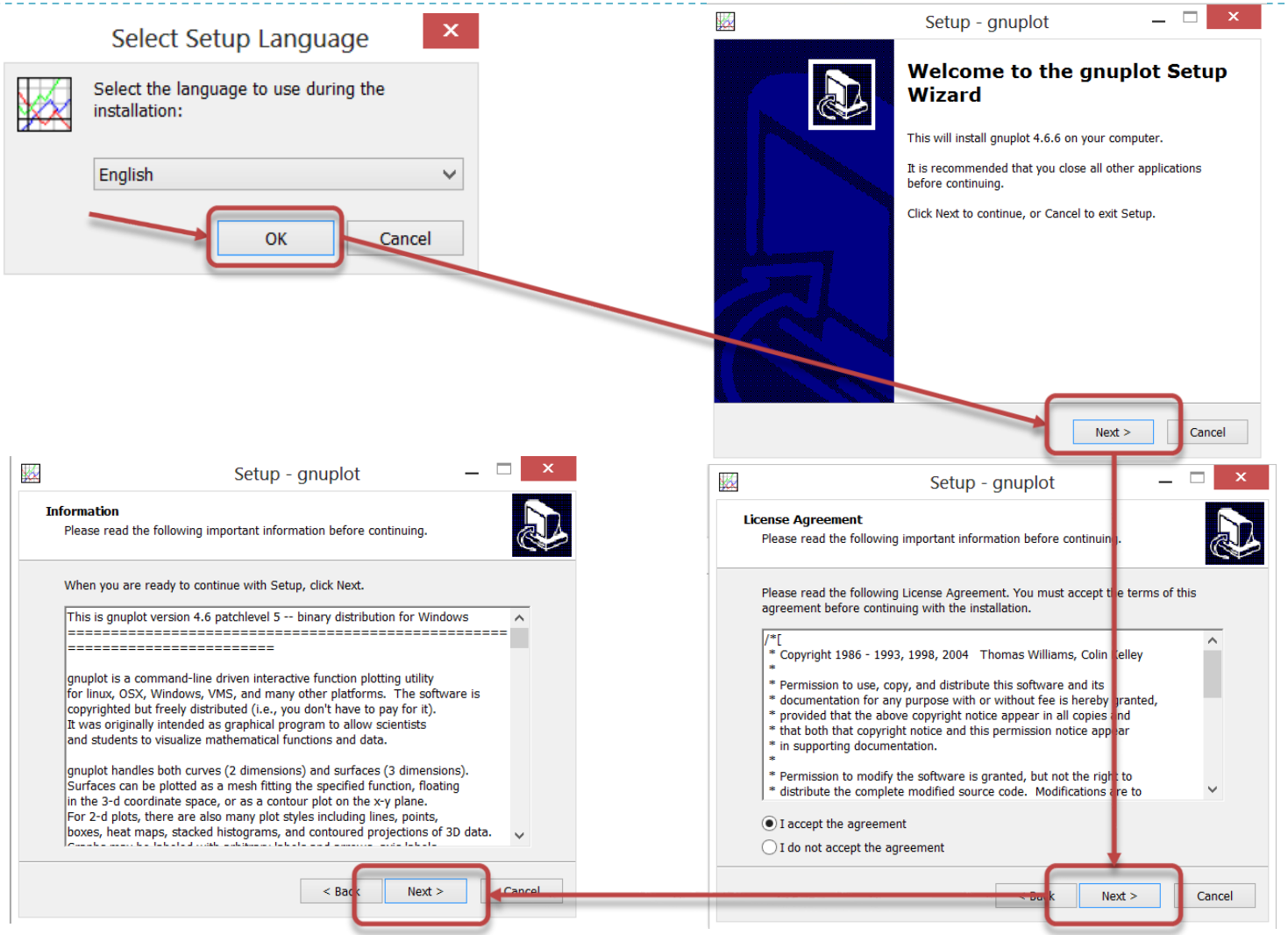
n =10000

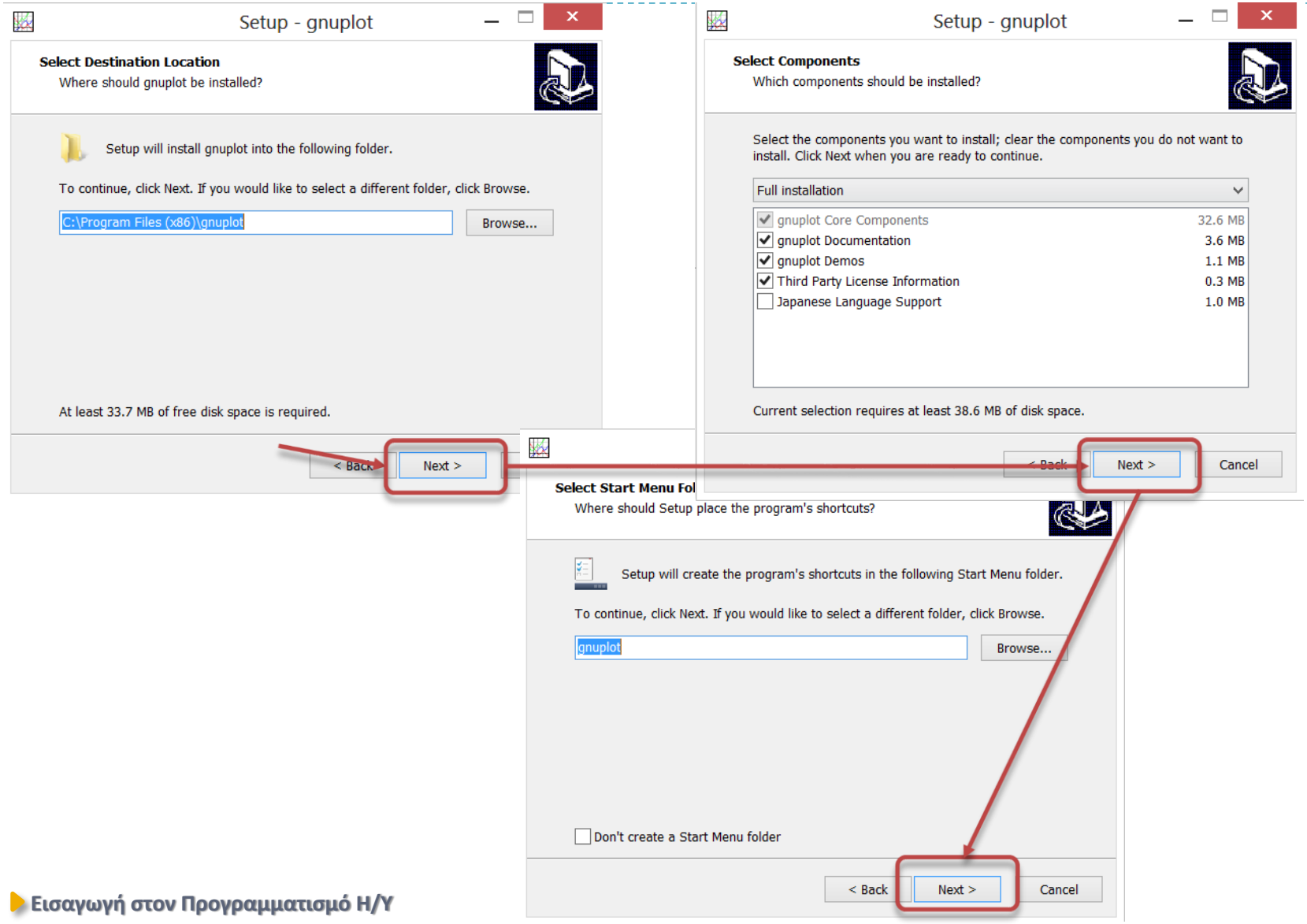


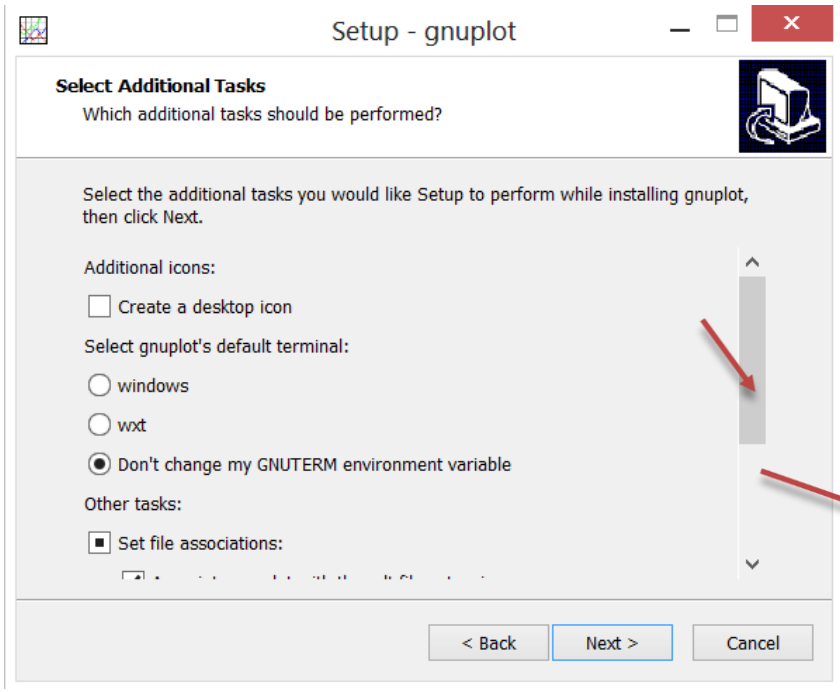
n =1000

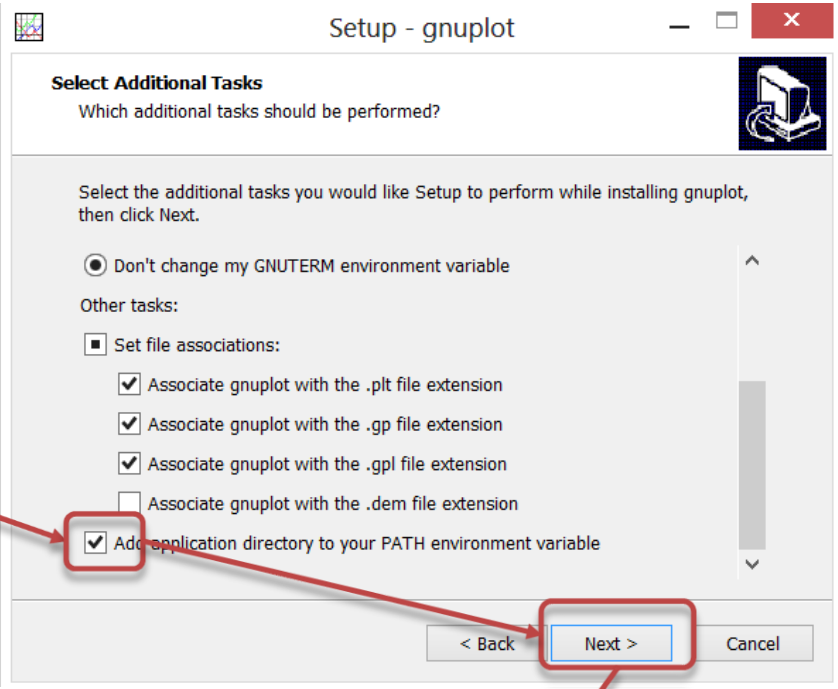
Εγκαθιστούμε το gnuplot

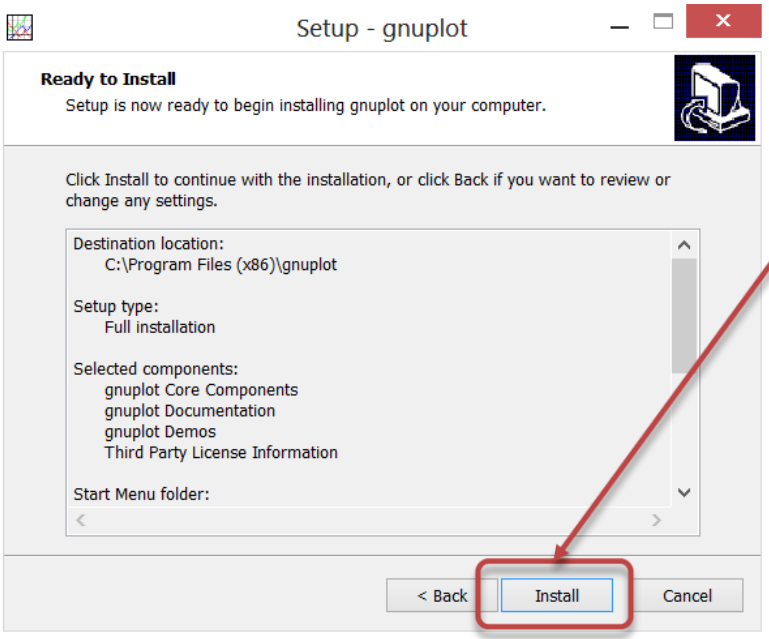
http://sourceforge.net/projects/gnuplot/files/latest/download?source=files

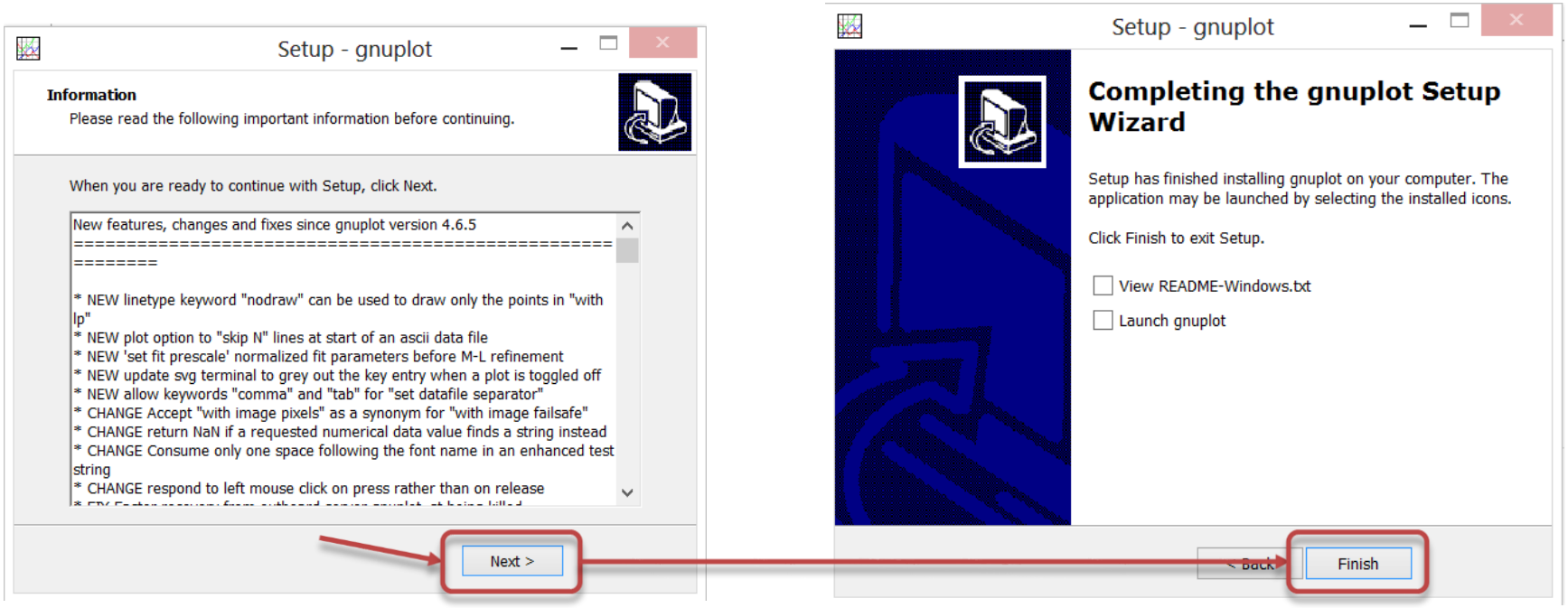



Setup - gnuplot
Select Destination Location
Where should gnuplot be installed?
Setup will install gnuplot into the following folder.
To continue, click Next. If you would like to select a different folder, click Browse.
C:\Program Files (x86)\gnuplot
Browse...
At least 33.7 MB of free disk space is required.
< Back
Next >
Setup - gnuplot
Select Components
Which components should be installed?
Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.
Full installation
gnuplot Core Components 32.6 MB
gnuplot Documentation 3.6 MB
gnuplot Demos 1.1 MB
Third Party License Information 0.3 MB
Japanese Language Support 1.0 MB
Current selection requires at least 38.6 MB of disk space.
< Back
Next >
Cancel
Select Start Menu Fol
Where should Setup place the program's shortcuts?
Setup will create the program's shortcuts in the following Start Menu folder.
To continue, click Next. If you would like to select a different folder, click Browse.
gnuplot
Browse...
Don't create a Start Menu folder
< Back
Next >
Cancel


Setup - gnuplot
Select Additional Tasks
Which additional tasks should be performed?
Select the additional tasks you would like Setup to perform while installing gnuplot, then click Next.
Additional icons:
Create a desktop icon
Select gnuplot's default terminal:
windows
wxt
Don't change my GNUTERM environment variable
Other tasks:
Set file associations:
< Back
Next >
Cancel


Setup - gnuplot
Select Additional Tasks
Which additional tasks should be performed?
Select the additional tasks you would like Setup to perform while installing gnuplot, then click Next.
Don't change my GNUTERM environment variable
Other tasks:
Set file associations:
Associate gnuplot with the .plt file extension
Associate gnuplot with the .gp file extension
Associate gnuplot with the .gpl file extension
Associate gnuplot with the .dem file extension
Add application directory to your PATH environment variable
< Back
Next >
Cancel


Setup - gnuplot
Ready to Install
Setup is now ready to begin installing gnuplot on your computer.
Click Install to continue with the installation, or click Back if you want to review or change any settings.
Destination location:
C:\Program Files (x86)\gnuplot
Setup type:
Full installation
Selected components:
gnuplot Core Components
gnuplot Documentation
gnuplot Demos
Third Party License Information
Start Menu folder:
< Back
Install
Cancel



Αν είναι ανοιχτό το CodeBlocks κάντε του επανεκκίνηση

Πως 'πλοτάρουμε' με το gnuplot

η συνάρτηση κανονικής πιθανότητας I.

```
program gnuplot_demo
   use ogpf                        ! Το απαραίτητο module
   implicit none
   type(gpf)              :: gp     ! Απαραίτητη δήλωση μεταβλητής
   integer, parameter :: n = 21
   real(8)                :: prob(n,2)

   prob = dble(g(x1 = -3.0, x2 = 3.0, step = 0.3))

   ! καθορίζουμε title, xlabel, ylabel και options
   call gp%title   ('normal probability plot')
   call gp%xlabel  ('x')
   call gp%ylabel  ('Φ(x)')
   call gp%options ('set style data linespoints; &
                     set xrange [-3.0:3.0];      &
                     set yrange [0.0:0.45];')
   ! πλοτάρουμε call gp%plot(x, y)
   ! όπου x, y μονοδιάστατοι πίνακες διπλής ακρίβειας
   call gp%plot(prob(:,1), prob(:,2))
```

Πως ‘πλοτάρουμε’ με το gnuplot

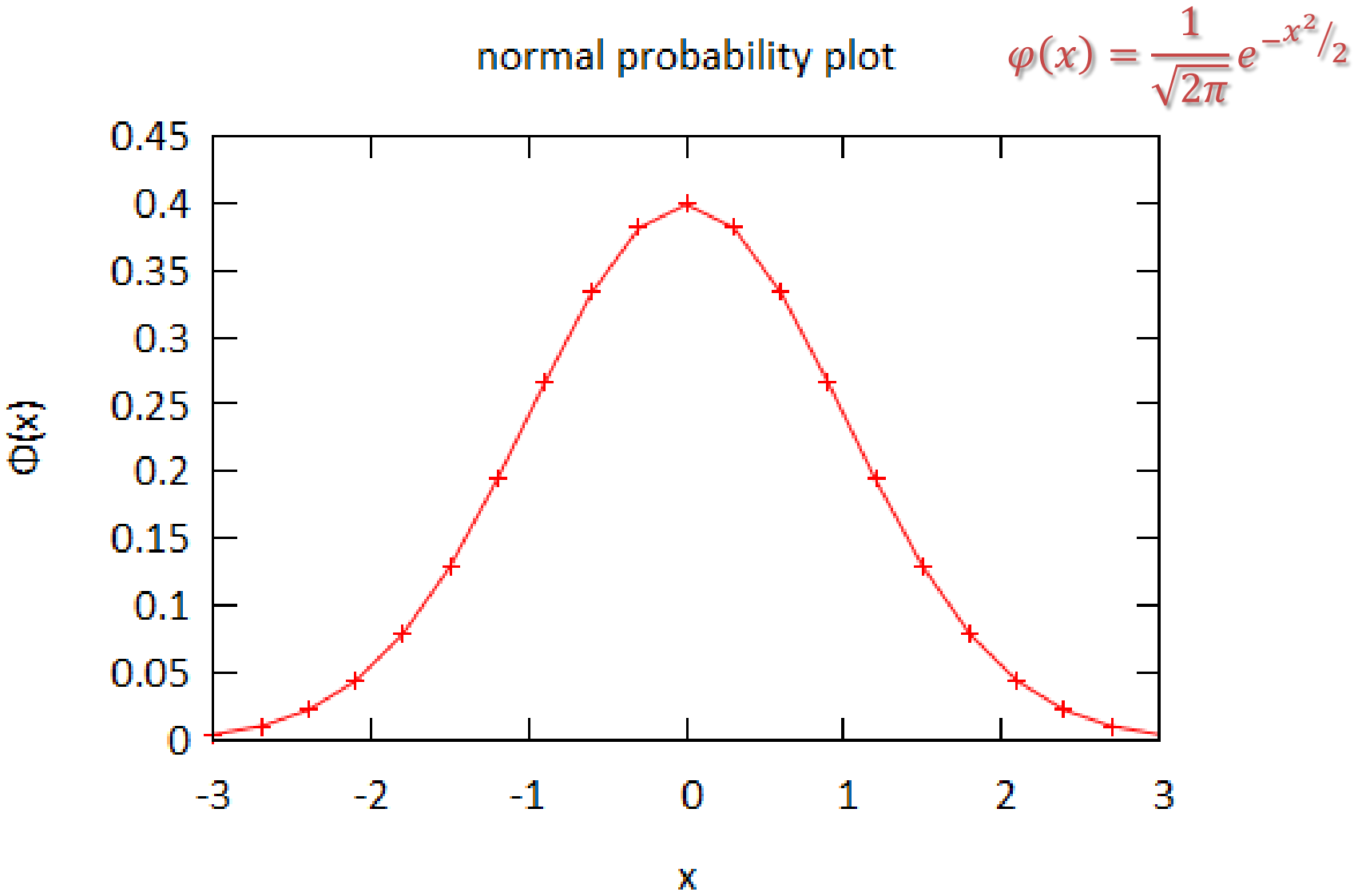
η συνάρτηση κανονικής πιθανότητας II.

$$\varphi(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$$

```
contains

    pure function g(x1, x2, step)
        real,intent(in) :: x1,x2,step
        real            :: g(nint(1 + (x2 - x1) / step),2)
        !locals
        real,parameter  :: pi = 4 * atan(1.)
        integer         :: i
        real            :: y0

        y0 = 1 / sqrt(2 * pi)
        do i = 1, nint(1 + (x2 - x1)/step)
            g(i,1) = x1 + (i - 1) * step
            g(i,2) = y0 * exp(- g(i,1)**2 / 2)
        end do
    end function g

end program gnuplot_demo
```

Πως ‘πλοτάρουμε’ με το gnuplot

η συνάρτηση κανονικής πιθανότητας III.



Σημείωμα Αναφοράς

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