ΑΝΟΙΚΤΑ ακαδημαϊκά μαθήματα ΠΠ

Εισαγωγή στον Προγραμματισμό Η/Υ για Χημικούς Μηχανικούς

Παρουσίαση Διαλέξεων: 6. Πίνακες

Καθηγητής Δημήτρης Ματαράς

Τι είναι οι πίνακες, ορισμοί

ΠΙΝΑΚΑΣ: Συλλογή δεδομένων (μεταβλητών ή σταθερών) του ίδιου τύπου, που εμφανίζονται με κοινό όνομα **το οποίο πρέπει να δηλωθεί υποχρεωτικά**

ΟΝΟΜΑ : Το κοινό όνομα υπό το οποίο δηλώνεται η συλλογή των δεδομένων

ΣΤΟΙΧΕΙΟ ΠΙΝΑΚΑ : Ένα από τα δεδομένα (μεταβλητή ή σταθερά) που περιέχονται στον πίνακα

ΕΝΔΕΙΚΤΗΣ ΣΤΟΙΧΕΙΟΥ : Ακέραιος αριθμός που δείχνει την θέση του στοιχείου στον πίνακα

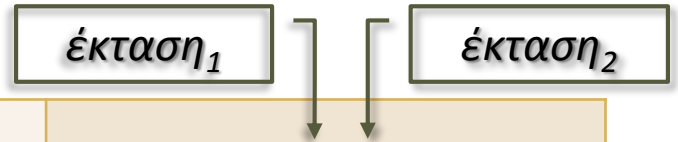
ΤΙΜΗ ΣΤΟΙΧΕΙΟΥ : Η τιμή που περιέχει το στοιχείο του πίνακα

ΙΔΙΟΤΗΤΕΣ ΠΙΝΑΚΑ :

- *Τάξη* : το πλήθος των διαστάσεων του
- *Μέγεθος* : το πλήθος των στοιχείων του δηλ. το γινόμενο των εκτάσεων του
- *Μορφή* : ο τρόπος διάταξης των στοιχείων του

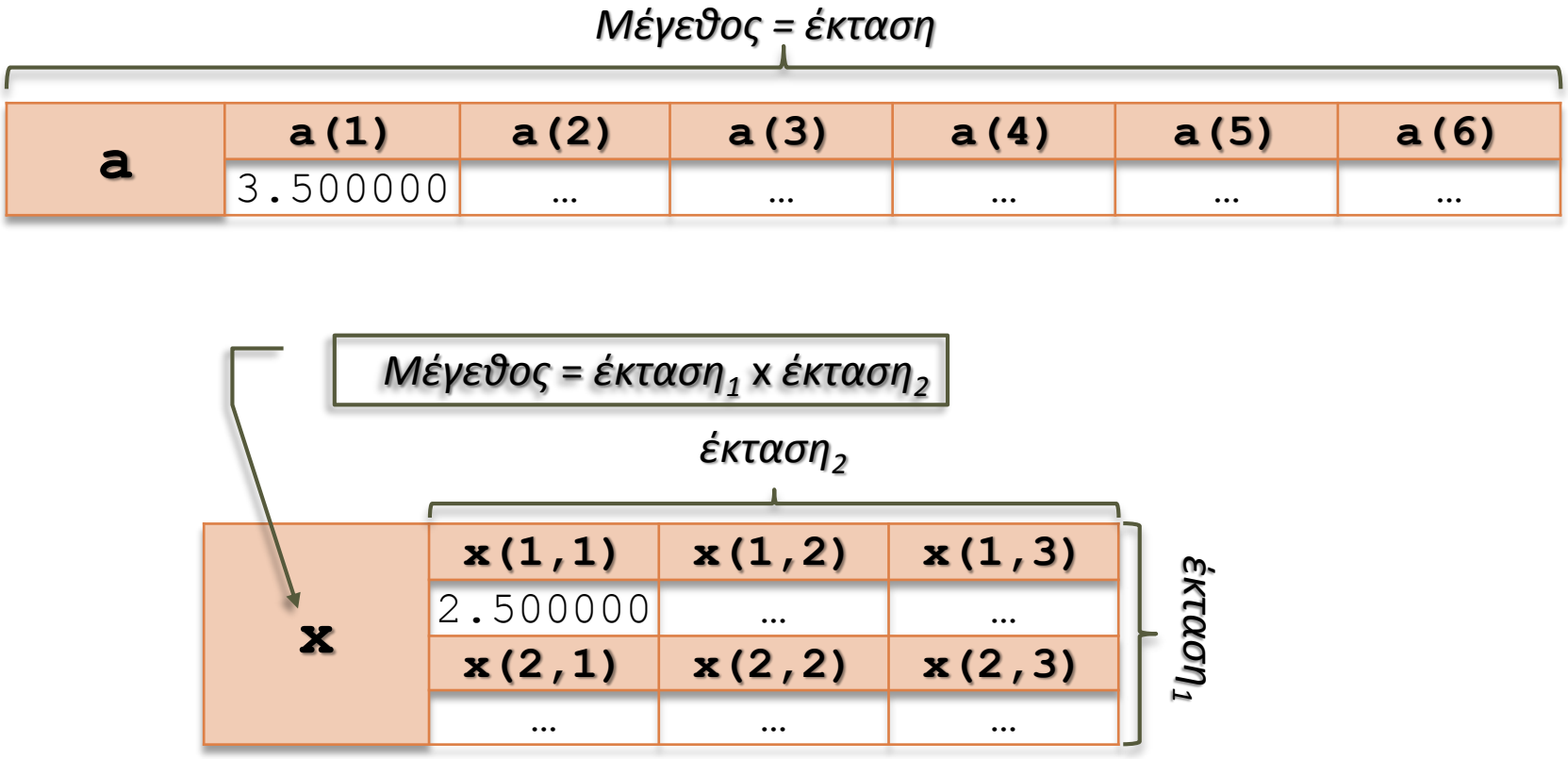
Τι είναι οι πίνακες, παραδείγματα

```
real::a(6),x(2,3)     !δήλωση
a(1) = 3.5; x(1,1) = 2.5
```



ΠΙΝΑΚΑΣ	a(6)	x(2,3)
ΟΝΟΜΑ	A	X
ΣΤΟΙΧΕΙΟ	a(1)	x(1,1)
ΕΝΔΕΙΚΤΗΣ ΣΤΟΙΧΕΙΟΥ	1	1 , 1
ΤΙΜΗ ΣΤΟΙΧΕΙΟΥ	3.5	2.5
Τάξη	1	2
Μέγεθος	6	6
Μορφή	1x6	2x3

Μεγέθη και εκτάσεις



Δήλωση πινάκων

δήλωση τύπου[,ιδιότητες::]όνομα$_1$(έκταση$_1$[,έκταση$_2$,..])&
[,όνομα$_2$(έκταση$_1$[,έκταση$_2$,..]),..

! ή

*δήλωση τύπου,**DIMENSION**(έκταση$_1$[,έκταση$_2$,..])::όνομα$_1$[,όνομα$_2$,..]*

```
REAL::a(10), x(10), y(3,2)
INTEGER::r(5), w(5), g(6)
! ή το ίδιο
REAL, DIMENSION(10) ::a, x, y(3,2)
INTEGER,DIMENSION(5)::r, w, g(6)
```

```
real::a(5)
a(5) = 4.5; a(2) = 2.5; a(3) = 1.
a(1) = a(5) + 2 * a(3)
a(4) = a(5) - a(2)
print*, a     ; end
```

έκταση

ενδείκτης

```
6.500000        2.500000
1.000000        2.000000
4.500000
```

Ότι αναφέρεται σε αυτή την παρουσίαση αφορά ***στατικούς πίνακες*** δηλ. πίνακες των οποίων η *μορφή* και συνεπώς το *μέγεθος* και η *τάξη* είναι γνωστά από την δήλωση τους και *δεν αλλάζουν* κατά την διάρκεια της εκτέλεσης του προγράμματος.

Όπως βλέπετε και από το παράδειγμα τα στοιχεία των πινάκων δεν είναι διαφορετικά από τις κοινές μεταβλητές, *απλά συμβολίζονται διαφορετικά*.

Προσέξτε:

- Άλλο η *έκταση* μιας διάστασης ενός πίνακα και άλλο ο *ενδείκτης* ενός στοιχείου του πίνακα. Η *έκταση* εμφανίζεται συνήθως μόνο στις δηλώσεις ενώ ο *ενδείκτης* οπουδήποτε εκτός από τις δηλώσεις
- Το μέγεθος μονοδιάστατου (ή αλλιώς τάξης 1) πίνακα είναι ίσο με την έκταση
- Η μέγιστη τιμή του *ενδείκτη* δεν πρέπει να ξεπερνά την *έκταση* της διάστασης του πίνακα δηλ. για μονοδιάστατο πίνακα δεν μπορεί να ξεπερνά το μέγεθος του
- Το μέγεθος πολυδιάστατου πίνακα είναι ίσο με το γινόμενο των εκτάσεων του
- Ο πίνακας μπορεί να έχει *μέχρι 7 διαστάσεις*! (δηλ. τάξη 7) Συνήθως όμως χρησιμοποιούνται πίνακες τάξης 1 και 2 ή το πολύ 3.

Δίνουμε τιμές σε όλο τον πίνακα ή σε στοιχεία του

```
real::a(6) = 1.0    !όλες οι τιμές 1.000000
  do i = 1, 6       !με do
    a(i) = i        !τιμές ίδιες με τον ενδείκτη
  end do
  a = sqrt(a)       !αντικατάσταση όλων των τιμών με sqrt(a)
end
```

a	a(1)	a(2)	a(3)	a(4)	a(5)	a(6)
	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

a	a(1)	a(2)	a(3)	a(4)	a(5)	a(6)
	1.000000	2.000000	3.000000	4.000000	5.000000	6.000000

a	a(1)	a(2)	a(3)	a(4)	a(5)	a(6)
	1.000000	1.414213	1.732051	2.000000	2.236068	2.449490

Δίνουμε τιμές από τον χρήστη με READ

```
real::a(6) = 1.0 !όλες οι τιμές 0.0
do i = 1, 6
  print'(a15, i2)', 'δώσε την τιμή', i
  read*, a(i)
end do
print'(6f5.2)', a
end
```

```
 δώσε την τιμή 1
6
 δώσε την τιμή 2
5.5
 δώσε την τιμή 3
5
 δώσε την τιμή 4
4.5
 δώσε την τιμή 5
4
 δώσε την τιμή 6
3.5
 6.00 5.50 5.00 4.50 4.00 3.50
```

a	a(1)	a(2)	a(3)	a(4)	a(5)	a(6)
	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

a	a(1)	a(2)	a(3)	a(4)	a(5)	a(6)
	6.000000	5.500000	5.000000	4.500000	4.000000	3.500000

Δίνουμε τιμές με κατασκευαστή

μόνο σε μονοδιάστατους πίνακες

```
      real::a(5) = (/1.,3.,5.,7.,9./)    !κατασκευαστής
      print 8, a
8     format(5f5.2)

      a = (/2.,4.,6.,8.,10/)             !κατασκευαστής
      print 8, a
      a = [1.5,3.,6.,8.,9.5]             !κατασκευαστής fortran 2003
      print 8, a

      a =(/(i, i = 1,5)/) !κατασκευαστής με υπονοούμενη επανάληψη
      print 8, a
      a =[(i, i = 1,5)]   !το ίδιο στη fortran 2003
      print 8, a

end
```

```
 1.00  3.00  5.00  7.00  9.00
 2.00  4.00  6.00  8.00 10.00
 1.50  3.00  6.00  8.00  9.50
 1.00  2.00  3.00  4.00  5.00
 1.00  2.00  3.00  4.00  5.00
```

Περισσότερη υπονοούμενη επανάληψη

```
  real                    :: pi = 4. * atan(1.)
  real, dimension(5)      :: x = 0, y = 0
5 format(a4,5f9.5,/)
  x=[(pi * i, i = 1, 5)] !κατασκευαστής με υπον. επανάληψη
A print 5, 'A)', x       !εκτύπωση του πίνακα

B print 5,'B)', (pi * i, i = 1, 5)!εκτύπωση με υπον. επανάληψη
                                   C
  y=[(sqrt(1. * j), j = 1, 5)]; print 5, 'C)', y
D print 5,'D)', (sqrt(1. * j), j = 1, 5)

end
```

```
A)  3.14159  6.28319  9.42478 12.56637 15.70796

B)  3.14159  6.28319  9.42478 12.56637 15.70796

C)  1.00000  1.41421  1.73205  2.00000  2.23607

D)  1.00000  1.41421  1.73205  2.00000  2.23607
```

Η εγγενής συνάρτηση `size`

σε δηλώσεις, κατασκευαστές, επαναλήψεις

```
character(50)::string1
real::x(5,5) = 0, a(size(x, 1))  !δήλωση με size

a=[(i, i = 1, size(x,1))] !κατασκευαστής με size

write(string1,*)'(/,2x',size(a),'f4.1,/)'
write(*,'(a,i2,a)')string1 !τυπώνω το format για να το δω
write(*, string1) a !τυπώνω τον πίνακα με αυτό το format

do i = 1, size(a)  !επανάληψη με size
   x(i,i) = i      !δίνει τιμές στη διαγώνιο
   write(6,'(2x, f4.1)', advance='no') x(i, i)
end do
print string1,(x(i,i), i=1,size(x,1))!υπονοούμενη επανάληψη

end
```

Εσωτερικό Αρχείο

```
(/, 2x, 5f4.1,/)

  1.0 2.0 3.0 4.0 5.0

  1.0   2.0   3.0   4.0   5.0
  1.0 2.0 3.0 4.0 5.0
```

ΠΑΡΑΔΕΙΓΜΑ: Έστω δύο μονοδιάστατοι πίνακες μεγέθους 5 και 10 στοιχείων αντίστοιχα που αντιπροσωπεύουν δύο σύνολα ακέραιων αριθμών. α) Δώστε τιμές στους δύο πίνακες χρησιμοποιώντας κατασκευαστή και τυπώστε την τομή των δύο συνόλων. β) Καταχωρίστε σε αρχείο διαδοχικής πρόσβασης την ένωση των δύο συνόλων.

```
program sets !Εξετάσεις Φεβ. 2011 Υ&Α άσκηση 3
  implicit none
  integer::a(5)=[1,5,2,8,15],i,j
  integer::b(10)=[7,20,4,2,1,9,6,8,13,17]; logical::set
  call execute_command_line('chcp 1253')
  open(9,file = 'union.dat')
     print'($, a28)', 'η τομή των 2 συνόλων είναι:'
     ier:do i = 1, size(a)
        set = .false.
        jer:do j = 1, size(b)
           if(a(i) == b(j))then          !κοινά στοιχεία
              print'($,i3)',a(i); set = .true.; exit jer
           endif
        enddo jer
        if(.not.set)write(9,'(i3)') a(i)!μη κοινά στοιχεία
     enddo ier
     write(9,'(i3)')(b(i),i = 1, size(b))
  close(9)
end program sets
```

Αθροίσματα και Γινόμενα

οι εγγενείς συναρτήσεις SUM και PRODUCT

```
real::a(5) = [1.,2.,3.,4.,5.], s = 0.
print*, sum(a)    !με εγγενή συνάρτηση
      ⇕
do i = 1, size(a)!χωρίς εγγενή συνάρτηση
  s = s + a(i)
end do
print*, s
end
```

ΠΡΟΣΕΞΤΕ: η εγγενής συνάρτηση **SUM**
κάνει αθροίσματα πινάκων
όχι αθροίσματα γενικώς!

```
real   :: a(5) = [1.,2.,3.,4.,5.], p = 1.
print*, product(a)  !με εγγενή συνάρτηση
      ⇕
do i = 1, size(a)   !χωρίς εγγενή συνάρτηση
  p = p * a(i)
end do
print*, p
end
```

ΠΡΟΣΕΞΤΕ: η εγγενής συνάρτηση **PRODUCT**
κάνει γινόμενα πινάκων
όχι γινόμενα γενικώς!

Τιμές σε πίνακα τάξης 2

```
integer::x(2,3)
do i = 1, 2
    do j = 1, 3
        x(i,j) = i * 10 + j
        !print'($,2x,i2)',x(i,j)
        write (*, '(2x, i2)', advance='no') x(i,j)
    enddo
    print*
enddo
print'(/, 6i4)',x
end
```

```
  11  12  13
  21  22  23

  11  21  12  22  13  23
```

Όπως βλέπετε στο παράδειγμα τα στοιχεία του πίνακα καταχωρίζονται στη μνήμη *με προτεραιότητα στήλης* και όχι γραμμής. Αυτό φαίνεται καθαρά στην τελευταία πρόταση εκτύπωσης.

Ελάχιστος και Μέγιστος


```
program extremes
   implicit none

!  δηλώσεις:
   integer, parameter:: max_size = 10! μέγιστο πλήθος τιμών
   integer, dimension(max_size):: x  ! τιμές
   integer :: imax                   ! ενδείκτης μέγιστης τιμής
   integer :: imin                   ! ενδείκτης ελάχιστης τιμής
   integer :: i                      ! αριθμητής do
   integer :: n                      ! πλήθος τημών
   integer :: tmax, tmin             ! προσωρινή μνήμη

!  αρχή:
   call execute_command_line('chcp 1253')
```

Ελάχιστος και Μέγιστος


```
   ask: do
      write (*,*) 'πλήθος τιμών;'
      read  (*,*) n

      check:if (n == 0) then
         print *, 'χαιρετίσματα'
         exit ask

      elseif   (n <= max_size) then check

         in: do i = 1, n
            write  (*,10) 'δώσε την τιμή ', i
10          format (' ',a,i3,': ')
            read   (*,*) x(i)
         end do in

         tmax = x(1); imax = 1
         tmin = x(1); imin = 1
```

Ελάχιστος και Μέγιστος


```
      comp: do i = 2, n
         if      (x(i) > tmax) then
            tmax = x(i); imax = i
         elseif (x(i) < tmin) then
            tmin = x(i); imin = i
         end if
      end do comp

      write  (*,11)
11    format (' ','οι τιμές είναι:')
      out: do i = 1, n
         if     (i == imax) then
            write (*,'(1x,i6,2x,a)') x(i), 'Μέγιστος'
         elseif(i == imin) then
            write (*,'(1x,i6,2x,a)') x(i), 'Ελάχιστος'
         else
            write (*,'(1x,i6)') x(i)
         endif
      end do out
```

Ελάχιστος και Μέγιστος


```
      else check
         write  (*,12) max_size
12       format (1x,'δώσε αριθμό < ', i3, ' ή 0 για έξοδο')
      endif check

   enddo ask

end program extremes
```

```
 πλήθος τιμών;
5
 δώσε την τιμή   1:
-3
 δώσε την τιμή   2:
4
 δώσε την τιμή   3:
5
 δώσε την τιμή   4:
7
 δώσε την τιμή   5:
-2
 οι τιμές είναι:
    -3  Ελάχιστος
     4
     5
     7  Μέγιστος
    -2
 πλήθος τιμών;
```

```
 πλήθος τιμών;
12
 δώσε αριθμό <  10 ή 0 για έξοδο
 πλήθος τιμών;
0
 χαιρετίσματα
```

Διάταξη επιλογής (selection sort)

```
program selection   ! διάταξη σε αύξουσα σειρά
    implicit none
!   δηλώσεις:
    integer :: ipos ! ενδείκτης της θέσης του ελάχιστου
    integer :: temp, i, j
    real    :: x(5) = [3, 5, -7, 6, 10]
!   αρχή:
    call execute_command_line('ch
    scan: do i = 1, size(x) - 1
        ipos = i
        in: do j = i + 1, size(x)
            if (x(j) < x(ipos)) ipos = j
        enddo in
        swap: if (i /= ipos) then
            temp = x(i); x(i) = x(ipos); x(ipos) = temp
        endif swap
        print '(a, i2, a, 5f5.1)', ' επανάληψη', i,') ', x
    end do scan
end program selection
```

```
 επανάληψη 1)  -7.0  5.0  3.0  6.0 10.0
 επανάληψη 2)  -7.0  3.0  5.0  6.0 10.0
 επανάληψη 3)  -7.0  3.0  5.0  6.0 10.0
 επανάληψη 4)  -7.0  3.0  5.0  6.0 10.0
```

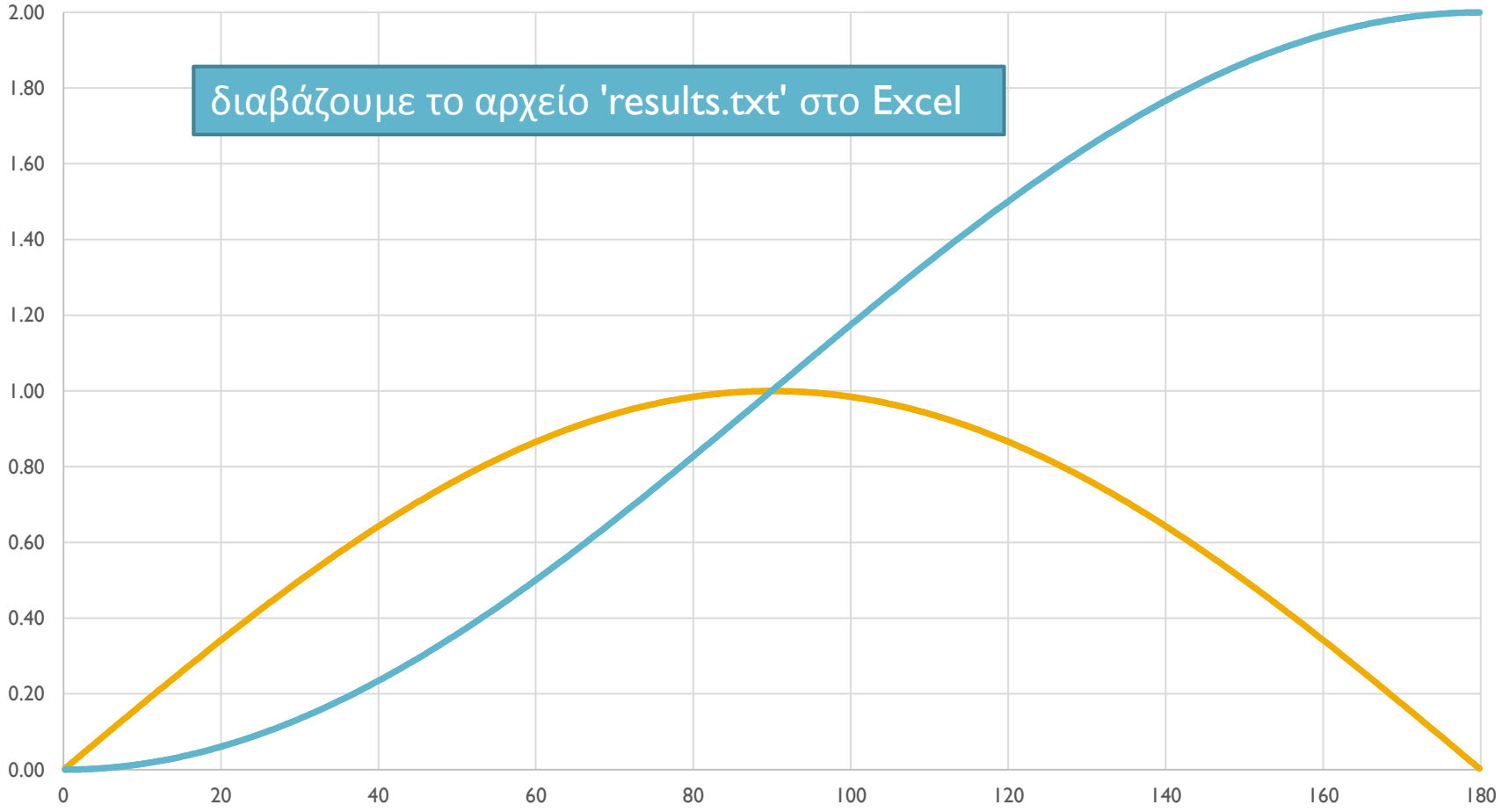
Η μέθοδος του τραπεζίου

υπολογισμός του ορισμένου ολοκληρώματος πινακοποιημένων τιμών

```
program trapezoid
   implicit none
   integer, parameter:: n = 1000
   integer :: i
   real(8) :: h, s, area, x(n), y(n)
   open (10, file = 'function.txt', status = 'old')
   read (10, *)(x(i), y(i), i = 1, n)

   h = (y(n) - y(1))/n; s = (y(1) + y(n))/2
   open (11, file = 'results.txt')

   do i = 2, n
      s = s + (y(i)); area = h * s * n
      write (11,*) x(i), y(i), area
   enddo

   close(10); close(11)
   print *,'the integral of the function is:', area
end program trapezoid
```

Η μέθοδος του τραπεζίου

υπολογισμός του ορισμένου ολοκληρώματος πινακοποιημένων τιμών



Διάταξη αντιμετάθεσης (bubble sort)

```
program bubble   ! διάταξη σε αύξουσα σειρά
   implicit none
!  δηλώσεις:
   logical :: changes ! .true. οταν γίνει αντιμετάθεση
   integer :: temp, i, n = 0
   real    :: x(5) = [3, 5, -7, 6, -10]
!  αρχή:
   call execute_command_line('chcp 1253')
   out:do
      changes = .false.
      in:do i = size(x), 2, -1
         if (x(i-1) > x(i)) then
            temp = x(i-1); x(i-1) = x(i); x(i) = temp
            changes = .true.
         endif
         n=n+1; print'(a,i3,a,5f5.1)',' επανάληψη',n,') ',x
      enddo in
      if(.not.changes) exit out
   enddo out
end program bubble
```

Διάταξη αντιμετάθεσης (bubble sort)

```
program bubble   ! διάταξη σε αύξουσα σειρά
   implicit none
!  δηλώσεις:
   logical :: changes ! .true. οταν γίνει αντιμετάθεση
   integer :: temp, i, n = 0
   real    :: x(5) = [3, 5, -7, 6, -10]
!  αρχή:
   call execute_commar
   out:do
      changes = .false
      in:do i = size(x
         if (x(i-1) > 
            temp = x(i-
            changes = .
         endif
         n=n+1; print'
      enddo in
      if(.not.changes)
   enddo out
end program bubble
```

```
επανάληψη  1)   3.0  5.0 -7.0-10.0  6.0
επανάληψη  2)   3.0  5.0-10.0 -7.0  6.0
επανάληψη  3)   3.0-10.0  5.0 -7.0  6.0
επανάληψη  4) -10.0  3.0  5.0 -7.0  6.0
επανάληψη  5) -10.0  3.0  5.0 -7.0  6.0
επανάληψη  6) -10.0  3.0 -7.0  5.0  6.0
επανάληψη  7) -10.0 -7.0  3.0  5.0  6.0
επανάληψη  8) -10.0 -7.0  3.0  5.0  6.0
επανάληψη  9) -10.0 -7.0  3.0  5.0  6.0
επανάληψη 10) -10.0 -7.0  3.0  5.0  6.0
επανάληψη 11) -10.0 -7.0  3.0  5.0  6.0
επανάληψη 12) -10.0 -7.0  3.0  5.0  6.0
```

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