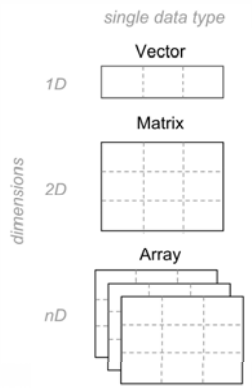
8/11/2022

Αναλυτικές Μέθοδοι στη Γεωπληροφορική

Δομές δεδομένων στο λογισμικό R

Σημερινή ατζέντα μαθήματος ...

- Χρήσεις μονοδιάστατων και πολυδιάστατων (≥2) αντικειμένων στην R
  - Vectors
  - Matrices
  - Arrays
  - Dataframes
  - Factors
- Εισαγωγή δεδομένων από αρχεία



Προηγούμενα αναφερθήκαμε …

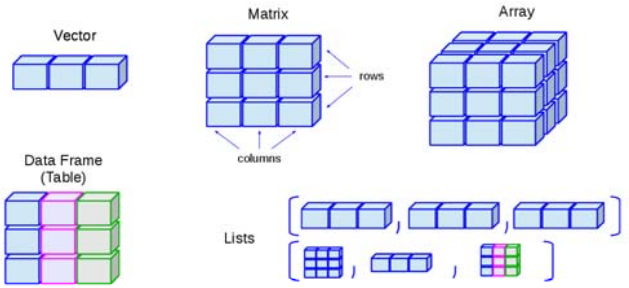
στους τύπους δεδομένων στην R

- Basic Data Types:
  - Integer (ακέραιες τιμές)
  - Double (πραγματικοί δεκαδικοί αριθμοί)
  - Logical (λογικές τιμές)
  - Character (χαρακτήρες)
  - Complex (μιγαδικοί αριθμοί)
  - Raw (Bytes)

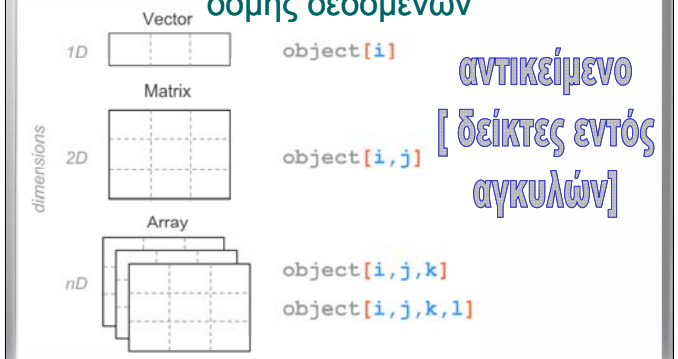
Σήμερα θα επικεντρωθούμε στις δομές τους

- Vectors:
  - atomic vectors
  - functions mode(), typeof(), storage.mode()
  - implicit coercion principles
  - explicit coercion functions
  - vectorization
  - recycling rules
- Arrays and Matrices
  - attributes
  - atomic arrays
- Lists
  - Storing objects in a list
  - Lists as a special type of Non-Atomic vector
- Data Frames & Factors
  - Functions to inspect contents and attributes

… Πέντε βασικοί τύποι δομής δεδομένων (χαρακτηριστικά, ιδιότητες, διάσταση διάταξης τους, ομοιογένεια …)



… Ευρετηρίαση και πρόσβαση στα επιμέρους στοιχεία (υποσύνολα) μιας δομής δεδομένων



αντικείμενο [δείκτες εντός αγκυλών]

Διανύσματα (atomic) vectors)

Data elements: Vectors / Διανύσματα

Είναι ακολουθίες ή διατεταγμένα σύνολα δεδομένων που περιέχουν αντικείμενα (σταθερές ή μεταβλητές ποσότητες) του ίδιου τύπου, και ανάλογα με το είδος τους διακρίνονται σε:

Αριθμητικά Διανύσματα – numeric vectors
Διανύσματα Χαρακτήρων – character vectors
Λογικά Διανύσματα – logical vectors
Διανύσματα Κατηγοριών – categorical vectors

Vectors= Elements of the same type

Με χαρακτηριστικά:

Type, typeof(), what it is.
Length, length(), how many elements it contains.
Attributes, attributes(), additional arbitrary metadata

Data elements: Vectors / Διανύσματα

Τα στοιχεία ενός διανύσματος συχνά αναφέρονται ως συνιστώσες ή μέλη του διανύσματος

```
> c(2, 3.23, 5.09) ; c(2L, 3L, 5L)  # Αριθμητικά Διανύσματα
[1] 2 3.23 5.09
[1] 2 3 5

> c(TRUE, FALSE, TRUE, FALSE, FALSE)   # Λογικά Διανύσματα
[1]  TRUE FALSE  TRUE FALSE FALSE

# Διανύσματα χαρακτήρων
> c("gps", "glonass", "galileo", "beidou", "egnos")
[1] "gps", "glonass", "galileo", "beidou", "egnos"
> length(c("gps", "glonass", "galileo", "beidou", "egnos"))
[1] 5    # το πλήθος των στοιχείων στο διάνυσμα (η διάσταση του)
```

Data elements: Διανύσματα (vectors)

```
# Εκχώρηση αριθμητικών, ονομαστικών ή λογικών τιμών
#  σε διανύσματα γίνεται με την εντολή c(λίστα τιμών). c() είναι
#  ο τελεστής παράθεσης, από τον όρο 'concatenate' (=συνενώνω)

a <- c(1,2,5.3,6,-2,4)   # διάνυσμα αριθμητικών τιμών

# Με το επίθεμα L εκχωρούνται ακέραιες τιμές -αντί δεκαδικές.
int_var <- c(1L, 6L, 10L)

# Διάνυσμα χαρακτήρων
b <- c("gps", "glonass", "galileo", "beidou")
chr_var <- c("these are", "some strings")

log_var <- c(TRUE, FALSE, T, F)   #logical vector
```



```
  # Can delete a vector by simply assigning a NULL to it.
> x
[1] -3 -2 -1  0  1  2
> x <- NULL
> x
NULL
> x[4]
NULL

  # Creating a vector using the seq() function
> seq(1, 3, by=0.2)           # specify step size
[1] 1.0 1.2 1.4 1.6 1.8 2.0 2.2 2.4 2.6 2.8 3.0
> seq(1, 5, length.out=4)     # specify length of the vector
[1] 1.000000 2.333333 3.666667 5.000000
```



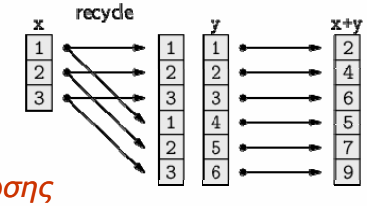
```
# Πράξεις επί ενός διανύσματος αριθμητικών τιμών (ακόμα και * /)
# εκτελούνται μεταξύ όλων των στοιχείων (all element-wise).
> x <- 1:10

> x ; x+1 ; x*2 ; x^2
[1] 1 2 3 4 5 6 7 8 9 10
[1] 2 3 4 5 6 7 8 9 10 11
[1] 2 4 6 8 10 12 14 16 18 20
[1] 1 4 9 16 25 36 49 64 81 100

> x/3
[1] 0.3333333 0.6666667 1.0000000 1.3333333 1.6666667
[6] 2.0000000 2.3333333 2.6666667 3.0000000 3.3333333
> log(x)
[1] 0.0000000 0.6931472 1.0986123 1.3862944 1.6094379
[6] 1.7917595 1.9459101 2.0794415 2.1972246 2.3025851
```


Συνένωση δύο διανυσμάτων

```
vector1 <- c(1, 2, 3)
vector2 <- c(4, 5, 6)
vector3 <- c(vector1, vector2) # η μεταβλητή vector3
                # θα περιέχει έξι στοιχεία, τους αριθμούς 1 έως 6

Αναδιάταξη διανυσμάτων
cbind(vector1) # το διάνυσμα vector1 σαν στήλη
     vector1
[1,] 1
[2,] 2
[3,] 3

rbind(vector1) # το διάνυσμα vector1 σαν γραμμή
        [,1] [,2] [,3]
vector1    1    2    3
```



```
> u <- c(1, 2, 3)
> v <- c(1, 2, 3, 4, 5, 6)
> u + v
[1] 2 4 6 5 7 9
```



Κανόνας ανακύκλωσης

- Εάν δύο διανύσματα τιμών έχουν διαφορετικό μήκος (διαφορετικό πλήθος στοιχείων), το μικρότερο θα ανακυκλωθεί για να ταιριάζει με το μεγαλύτερο διάνυσμα. Για παράδειγμα, το άθροισμά *u* + *v* των διανυσμάτων *u* και *v* υπολογίζεται με τιμές ανακύκλωσης του βραχύτερου διανύσματος *u*

Ευρετηρίαση στοιχείων σε διανύσματα για την απομόνωση συγκεκριμένων καταχωρήσεων ή στοιχείων που πληρούν ορισμένα κριτήρια

Ευρετηρίαση με δείκτες λογικής έκφρασης

```
> a <- c(1,2,3,4,5)
> b <- c(TRUE,FALSE,FALSE,TRUE,FALSE)
> a[b]
[1] 1 4
> max(a[b])
[1] 4
> sum(a[b])
[1] 5
```

Καταχωρήσεις με μη διαθέσιμες τιμές

```
> a <- c(1,2,3,4,NA)
> a
[1]  1  2  3  4 NA
> sum(a)
[1] NA
> sum(a,na.rm=TRUE)
[1] 10
```

Η συνάρτηση is.na ή/και ο λογικός τελεστής «μη» στο R (το σύμβολο !) μπορούν να χρησιμοποιηθούν για να προσδιοριστούν ποια στοιχεία δεν είναι διαθέσιμα σε ένα διάνυσμα τιμών.

```
> a <- c(1,2,3,4,NA)
> is.na(a)
[1] FALSE FALSE FALSE FALSE  TRUE
> !is.na(a)
[1]  TRUE  TRUE  TRUE  TRUE FALSE
> a[!is.na(a)]
[1] 1 2 3 4
```


Ευρετηρίαση με αρνητικούς δείκτες

```
> s = c("aa", "bb", "cc", "dd", "ee")
  > s[3]
  [1] "cc"
> s[-3]
  [1] "aa" "bb" "dd" "ee"
```

- Η χρήση αρνητικών δεικτών επιτρέπει να επιλεχθούν όλα τα στοιχεία του διανύσματος εκτός από αυτά των οποίων η θέση έχει την ίδια απόλυτη τιμή με τον εκάστοτε αρνητικό δείκτη, π.χ. στο παράδειγμα, έχει αφαιρεθεί το τρίτο μέλος του διανύσματος

Κανόνες ευρετηρίασης

Using integer vector as index

```
> x <- c(0, 2, 4, 6, 8, 10)
  [1] 0, 2, 4, 6, 8, 10
> x[c(2, 4)] # access 2nd and 4th element
  [1] 2 6
> x[c(2, -4)]   # cannot mix positive and
  negative integers
Error in x[c(2, -4)] : only 0's may be mixed
  with negative subscripts
> x[c(2.4, 3.54)] # real numbers are truncated
  to integers
  [1] 2 4
```


Τροποποίηση στοιχείων διανύσματος

```
> x <- c(-3, -2, -1, 0, 1, 2) ; x
  [1] -3 -2 -1 0 1 2
> x[2] <- 0; x # αλλαγή του 2ου στοιχείου σε 0
  [1] -3 0 -1 0 1 2
> x[x<0] <- 5; x # αλλαγή τιμής σε 5, όλων
  των στοιχείων μικρότερων από 0
  [1] 5 0 5 0 1 2
> x <- x[1:4]; x # περικοπή του x στα πρώτα 4
  στοιχεία του
  [1] 5 0 5 0
```

Ευρετηρίαση μέσω διανυσμάτων χαρακτήρων

```
# Μπορούμε να ονομάσουμε κάθε στοιχείο ενός διανύσματος
> x <- c("first"=3, "second"=0, "third"=9) ; x
first second third
    3      0     9
> names(x)
[1] "first" "second" "third"
> x["second"]                    > x[2]
 second                          second
      0                               0
> x[c("first", "third")]         > x[c(1, 3)]
 first third                    first third
     3     9                        3     9
```


Out-of-Range Index

```
> s = c("aa", "bb", "cc", "dd", "ee")
  > s[5]
  [1] "ee"
> s [10]
  [1] NA
> s [-6]
  [1] NA
```

- Αν ένας δείκτης είναι εκτός εύρους τιμών, μια τιμή που λείπει θα αναφέρεται μέσω του συμβόλου *NA*

Διάταξη/ταξινόμηση στοιχείων

```
# με τη συνάρτηση sort() ... σε αύξουσα σειρά
> x <- c(7,1,8,3,2,6,5,2,2,4) ;  sort(x)
  [1] 1 2 2 2 3 4 5 6 7 8
# με τη συνάρτηση sort() ... σε φθίνουσα σειρά
> sort(x, decreasing=T)
  [1] 8 7 6 5 4 3 2 2 2 1
> x          # vector x remains unaffected
  [1] 7 1 8 3 2 6 5 2 2 4
# but the vector x is affected when assigning
  the decreasing order elements into x
> x <- sort(x, decreasing=T) ; x
  [1] 8 7 6 5 4 3 2 2 2 1
```


Δείκτες ταξινομημένων στοιχείων

```
# με τη συνάρτηση order() ... σε αύξουσα σειρά
> x <- c(7,1,8,3,2,6,5,2,2,4) ;  order(x)
  [1] 2 5 8 9 4 10 7 6 1 3
# και ... σε φθίνουσα σειρά
> order(x, decreasing=T)
  [1] 3 1 6 7 10 4 5 8 9 2
> x[order(x)]   # this will rearrange x in
  ascending order          sort(x)   #same result
  [1] 1 2 2 2 3 4 5 6 7 8
> x[order(x, decreasing=T)]   # this will also
  rearrange x in descending order
  [1] 8 7 6 5 4 3 2 2 2 1
```

Κατάταξη ≠ ανασχηματισμός

Data elements: Vectors / Διανύσματα

```
# Πρόσβαση σε συγκεκριμένα στοιχεία του διανύσματος
> a <- c(1,2,3,4,5) ; a[1] ; a[2]
[1] 1
[1] 2
> a[0]  # the zero entry is used to indicate how the data is stored
numeric(0)
> a[5] ; a[6]
[1] 5
[1] NA  # Result in trying to get an element past the last position

> typeof(a)  # determine the data type used for a variable
[1] "double"
```

object[index]

Τα στοιχεία ενός διανύσματος δεν μπορεί να είναι διαφορετικών τύπων (π.χ. το ένα να είναι αριθμός και το άλλο συμβολοσειρά)

Επιλογές συγκεκριμένων στοιχείων ενός διανύσματος

```
x <- c(2, 4, 3, 1, -10, 8) ; x     # Output: 2  4  3  1 -10  8
x[c(2,4)]        # 2nd and 4th elements of vector. Output: 4  1
```

Χρησιμοποιώντας αγκύλες [ ] μπορεί να επιλεγεί ένα υποσύνολο των στοιχείων ενός διανύσματος.

```
x[c(2:4, 6)]
[1] 4 3 1 8
x[-2]                    # All elements, except the 2nd
[1]  2  3  1 -10  8
x[x>2]
[1] 4 3 8
x[x<0] <- -x[x<0]  # ίδιο αποτέλεσμα με την εντολή x <- abs(x);
x
[1]  2  4  3  1 10  8
abs(x)
[1]  2  4  3  1 10  8
```


Vector Arithmetic

```
Element-wise    # πράξη μεταξύ αντίστοιχων στοιχείων
u <- 1:3        # στοιχεία: 1, 2, 3
v <- 2:4        # στοιχεία: 2, 3, 4
u*v
[1]  2  6 12            # output: 1*2, 2*3, 3*4
u + v
[1]  3  5  7            # output: 1+2, 2+3, 3+4

Inner-product-wise    # πολλαπλασιασμός δύο διανυσμάτων
                      # με την έννοια του εσωτερικού γινομένου
u <- 1:3  ;  v <- 2:4
u %*% v
     [,1]
[1,]   20
rbind(u)%*%cbind(v)    # το ίδιο εσωτερικό γινόμενο
   v
u 20
```

$\mathbf{u} = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix} \quad \mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}$

$(\mathbf{u},\mathbf{v}) = u_1v_1 + u_2v_2 + \cdots + u_nv_n$

Διαχείριση διανυσμάτων

- *mode()*
- *length()*
- *names()*
- *typeof()*
- *attributes()*
- *is.list()*

Αυτόματος εξαναγκασμός *(automatic coercion)*: Όλα τα στοιχεία ενός ατομικού διανύσματος πρέπει να είναι του ίδιου τύπου, οπότε όταν προσπαθείτε να συνδυάσετε διαφορετικούς τύπους, θα ***εξαναγκαστούν*** στον *πιο ευέλικτο τύπο*. Οι τύποι των στοιχείων διανυσμάτων από τους λιγότερο έως τους πιο ευέλικτους είναι:

logical < integer < numeric < character


```
logical < integer < numeric < character
Σε ένα ατομικό διάνυσμα δεν μπορούν να υπάρχουν τόσο αριθμητικά
στοιχεία όσο και χαρακτήρες. Σε μια τετοια προσπάθεια,
ο τύπος δεδομένων με χαμηλότερη κατάταξη
θα εξαναγκαστεί στον υψηλότερο τύπο.

y <- c(1.5, "hello") ; y
[1] "1.5"   "hello"

y <- c(TRUE, 1.5)  ; y
[1] 1.0 1.5

y <- c(FALSE, 1.5) ; y
[1] 0.0 1.5

y <- c(TRUE, "this_char") ; y
[1] "TRUE"      "this_char"
```

logical < integer < numeric < character

```
> n = c(2, 3, 5)
> s = c("aa", "bb", "cc", "dd", "ee")
> c(n, s)
[1] "2"  "3"  "5"  "aa" "bb" "cc" "dd" "ee"
```

Value Coercion *(κατά τη συνένωση διανυσμάτων)*

- οι αριθμητικές τιμές εξαναγκάζονται σε συμβολοσειρές χαρακτήρων όταν συνδυάζονται δύο διαφορετικού τύπου διανύσματα τιμών

Coercion is from lower to higher types from logical to integer to double to character

```
> x <- c(1, 5, 4, 9, 0)
> typeof(x)
[1] "double"
> length(x)
[1] 5
> x <- c(1, 5.4, TRUE, "done")  ;  x
[1] "1" "5.4" "TRUE" "done"
> typeof(x)
[1] "character"
```


- Coercion – μετατροπή από έναν τύπο σε έναν άλλον
  - συνήθως γίνεται αυτόματα
- Η συνάρτηση **str()**, από τη συντομογραφία του όρου *'structure'*, δίνει μια συμπαγή, κατανοητή περιγραφή οποιασδήποτε δομής δεδομένων

```
str(c("a", 1))
> chr [1:2] "a" "1"

x <- c(FALSE, FALSE, TRUE)
as.numeric(x)
> [1] 0 0 1

    # Total number of TRUEs
sum(x)
> [1] 1

    # Proportion that are TRUE
mean(x)
> [1] 0.3333333
```


Data elements: Χειρισμός διανυσμάτων

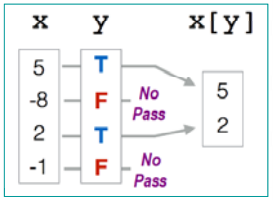
```
> temps <- c(29.3,34.5,22.7,31.8,27.3,28.5)    # numeric vector
> city.names <- c("Athens", "Chania", "Drama",
+ "Patra", "Kerkyra", "Ioannina")   # character vector

# Τα ονόματα αντιστοιχίζονται με τους βαθμούς θερμοκρασίας
> names(temps) <- city.names
> temps   # Το αντικείμενο temps έχει αλλάξει στο μεταξύ,
          #     με την προηγούμενη εντολή
 Athens Chania Drama Patra Kerkyra Ioannina
 29.3 34.5 22.7 31.8 27.3 28.5
> temps > 30
> d = temps > 30              #logical vector
> d
[1] F T F T F F
```


Data elements: Χειρισμός διανυσμάτων

Logical Indexing, Growing Vectors, π.χ.

για το διάνυσμα x (σημ., οι επόμενες δύο εντολές είναι ισοδύναμες)

```
>assign("x",c(1,1,2,3,5,8,13,21)) ; c(1,1,2,3,5,8,13,21)-> x
>is.vector(x)
[1] TRUE
> x<5
[1]  TRUE  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE
> x[x<5]
[1] 1 1 2 3
> x[10] = 55
> x
 [1]  1  1  2  3  5  8 13 21 NA 55
> x %% 2
 [1]  1  1  0  1  1  0  1  1 NA  1
```



Function	Description	Example	Result
`is.na(x)`	Which values in x are NA?	`is.na(c(2, NA, 5))`	FALSE, TRUE, FALSE
`is.finite(x)`	Which values in x are numbers?	`is.finite(c(NA, 89, 0))`	FALSE, TRUE, TRUE
`duplicated(x)`	Which values in x are duplicated?	`duplicated(c(1, 4, 1, 2))`	FALSE, FALSE, TRUE, FALSE
`which(x)`	Which values in x are TRUE?	`which(c(TRUE, FALSE, TRUE))`	1, 3

Συνήθεις διανυσματικές συναρτήσεις

 # Numeric Vectors

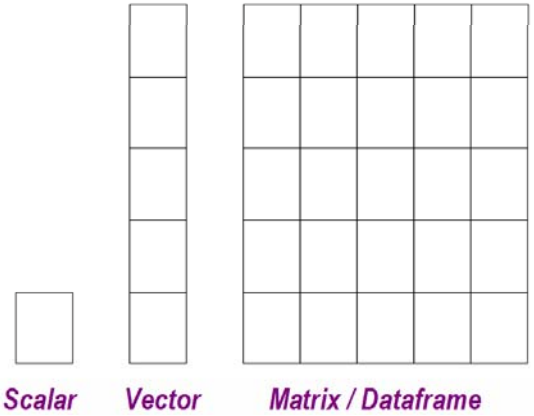
- sum()
- mean()
- sd() (standard deviation)
- max()
- min()
- median()
- range()

```
summary(LakeHuron)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
576.0   578.1   579.1   579.0   579.9   581.9
```

 # Any Vector

- rev() (reverse)
- unique() (list of unique vector values)

Πίνακες (μήτρες) Matrices



Data elements: Διοδιάστατοι πίνακες (matrices)

Είναι η γενίκευση διανυσμάτων σε δύο διαστάσεις.
Οι πίνακες χρησιμοποιούνται για να τακτοποιηθούν τιμές κατά γραμμές και στήλες σε έναν ορθογώνιο πίνακα.

```
> A = matrix(
+   c(2, 4, 3, 1, 5, 7),   # τα περιεχόμενα στοιχεία
+   nrow=2,                # αριθμός των γραμμών
+   ncol=3,                # αριθμός των στηλών
+   byrow = TRUE)          # εκχώρηση των στοιχείων κατά γραμμές
> A                        # print the matrix
     [,1] [,2] [,3]
[1,]    2    4    3
[2,]    1    5    7
```

Σχετικές εντολές για πίνακες είναι: is.matrix και as.matrix.
Σε αυτούς μπορεί να εφαρμοστεί η συνάρτηση διάστασης dim

Matrices - Πίνακες δεδομένων

```
mymatrix <- matrix(vector, nrow=r, ncol=c, byrow=FALSE,
dimnames=list(char_vector_rownames, char_vector_colnames))
# byrow=FALSE υποδηλώνει ότι τα στοιχεία του πίνακας
#          διαμορφώνονται κατά στήλες (default)
# byrow=TRUE στοιχεία του πίνακας
#          διαμορφώνονται κατά γραμμές
# dimnames , κατ' επιλογή ετικέτες για τις στήλες και τις γραμμές
cells <- c(1,26,24,68)
rnames <- c("R1", "R2")
cnames <- c("C1", "C2")
mymatrix <- matrix(cells,nrow=2,ncol=2,byrow=TRUE,
  + dimnames=list(rnames,cnames))
mymatrix
   C1 C2
R1  1 26
R2 24 68
```


Για τη δημιουργία ενός πίνακα, η παροχή τιμών και για τις δύο διαστάσεις *nrow* και *ncol* δεν είναι απαραίτητη. Εάν παρέχεται μια από τις διαστάσεις, η άλλη συνάγεται από το μήκος των δεδομένων

```
> matrix(1:9, nrow = 3, ncol = 3)
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
> matrix(1:9, nrow = 3)  # same result by providing only one dimension
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```


Από προεπιλογή, ένας πίνακας δημιουργείται κατά στήλες. Αυτό μπορεί να αντιστραφεί (δηλ. για δημιουργία στοιχείων ανά γραμμές) ορίζοντας *TRUE* στην παράμετρο *byrow*.

```
> matrix(1:9, nrow=3, byrow=TRUE) # fill matrix row-wise
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
```

- Ωστόσο, σε όλες τις περιπτώσεις, ένας πίνακας αποθηκεύεται εσωτερικά ανά στήλες

Ένας άλλος τρόπος δημιουργίας ενός πίνακα είναι η χρήση των συναρτήσεων *cbind()* και *rbind()* –δηλ. δέσμευση κατά στήλες και δέσμευση κατά γραμμές.

```
> cbind(c(1,2,3),c(4,5,6))
     [,1] [,2]
[1,]    1    4
[2,]    2    5
[3,]    3    6
> rbind(c(1,2,3),c(4,5,6))
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
```


Ένας πίνακας μπορεί επίσης να δημιουργηθεί από ένα διάνυσμα τιμών καθορίζοντας τη διάστασή του χρησιμοποιώντας το όρισμα *dim()*

```
> x <- c(1,2,3,4,5,6) ; x
[1] 1 2 3 4 5 6
> class(x)
[1] "numeric"
> dim(x) <- c(2,3) ;  x
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
> class(x)       # check if x is a matrix or not
[1] "matrix"
```


Ονοματίζοντας τις σειρές και τις στήλες του πίνακα κατά τη δημιουργία του επιτρέπει την άμεση πρόσβαση στα στοιχεία τους ή και για αλλαγές τους

```
> x <- matrix(1:9, nrow = 3, dimnames
  = list(c("X","Y","Z"), c("A","B","C")))
# passing a 2 element list to the argument dimnames
> x
  A B C
X 1 4 7
Y 2 5 8
Z 3 6 9
```



```
> colnames(x)
[1] "A" "B" "C"
> rownames(x)
[1] "X" "Y" "Z"
# It is also possible to change names
> colnames(x) <- c("C1","C2","C3")
> rownames(x) <- c("R1","R2","R3")
> x
   C1 C2 C3
R1  1  4  7
R2  2  5  8
R3  3  6  9
```



```
> x
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```

Ορίζουμε τις σειρές και τις στήλες ως διανύσματα τιμών και τα χρησιμοποιούμε για ευρετηρίαση των στοιχείων του πίνακα

```
> x[c(1,2),c(2,3)] # select rows 1 & 2 and columns 2 & 3
     [,1] [,2]
[1,]    4    7
[2,]    5    8
> x[-1,] # select all rows except first
     [,1] [,2] [,3]
[1,]    2    5    8
[2,]    3    6    9
```

```
> x
[,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
> x[c(3,2),]     # leaving column field blank will select entire columns
[,1] [,2] [,3]
[1,]    3    6    9
[2,]    2    5    8
> x[,]      # leaving row as well as column field blank will select entire matrix
[,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
> x[-1,]      # select all rows except first
[,1] [,2] [,3]
[1,]    2    5    8
[2,]    3    6    9
```



```
> x
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```

Εάν μετά την ευρετηρίαση, το αποτέλεσμα είναι μια σειρά ή μια στήλη, αυτό δίνεται ως ένα διάνυσμα

```
> x[1,]
[1] 1 4 7
> class(x[1,])
[1] "integer"
> x[1,,drop=FALSE]
# now the result is a 1X3
  matrix rather than a vector
     [,1] [,2] [,3]
[1,]    1    4    7
> class(x[1,,drop=FALSE])
[1] "matrix"
```



```
> x
     [,1] [,2] [,3]
[1,]    4    8    3
[2,]    6    0    7
[3,]    1    2    9
```

Η ευρετηρίαση ενός πίνακα μπορεί γίνει με τη χρήση λογικών διανυσμάτων ως δείκτες

```
> x[c(TRUE,FALSE,TRUE),
  c(TRUE,TRUE,FALSE)]
     [,1] [,2]
[1,]    4    8
[2,]    1    2
> x[c(TRUE,FALSE),c(2,3)]
# the 2 element logical
  vector is recycled to 3
  element vector
     [,1] [,2]
[1,]    8    3
[2,]    2    9
```



```
Διαγώνια, πάνω- ή κάτω-τριγωνιακά στοιχεία πίνακα
a <- matrix(c(1:16), ncol = 4) ; a
     [,1] [,2] [,3] [,4]
[1,]    1    5    9   13
[2,]    2    6   10   14
[3,]    3    7   11   15
[4,]    4    8   12   16
diag(a)              # Access the diagonal elements
[1]  1  6 11 16
upper.tri(a)         # Select the upper-triangular elements
      [,1]  [,2]  [,3]  [,4]
[1,] FALSE  TRUE  TRUE  TRUE
[2,] FALSE FALSE  TRUE  TRUE
[3,] FALSE FALSE FALSE  TRUE
[4,] FALSE FALSE FALSE FALSE
a[upper.tri(a)]      # Access the upper-triangular elements
[1]  5  9 10 13 14 15
a[lower.tri(a)]      # Access the lower-triangular elements
[1]  2  3  4  7  8 12
```



```
Ευρετηρίαση με διάνυσμα χαρακτήρων (Character vectors as index)
Είναι δυνατή για μήτρα με όνομα σειράς ή στήλης. Αυτό
μπορεί να γίνει με ανάμιξη ακέραιας ή λογικής ευρετηρίασης.
> x
     A B C
[1,] 4 8 3
[2,] 6 0 7
[3,] 1 2 9
> x[,"A"]
[1] 4 6 1
> x[TRUE,c("A","C")]
     A C
[1,] 4 3
[2,] 6 7
[3,] 1 9
> x[2:3,c("A","C")]
     A C
[1,] 6 7
[2,] 1 9
```

Indexing Vector Elements

Data elements: Δισδιάστατοι πίνακες (matrices)

Matrix Indexing – Μέσα σε αγκύλες, δηλώνονται οι γραμμές, και οι στήλες. Παραλείποντας τη μια από τις δύο παραμέτρους επιστρέφει το διάνυσμα της γραμμής ή στήλης που υποδηλώνει η άλλη παράμετρος.

```
> xmat <- matrix(1:9,ncol = 3) ; xmat
  [,1][,2][,3]
[1,]  1  4  7
[2,]  2  5  8
[3,]  3  6  9
> row.number <-1  ; col.number <-3
> xmat[row.number,col.number]
[1] 7
> xmat[row.number,]
[1] 1 4 7
> xmat[,col.number]
[1] 7 8 9
```


Data elements: Διδιάστατοι πίνακες ή μήτρες (matrices)

```
> A                    # Για τον πίνακα A ...
     [,1] [,2] [,3]
[1,]    2    4    3
[2,]    1    5    7
> A[2, 3]      # Το στοιχείο στη 2η γραμμή, 3η στήλη
[1] 7
> A[2, ]       # Τα στοιχεία στη 2η γραμμή, και
[1] 1 5 7      #    παρόμοια για ολόκληρη τη γραμμή m, A[m, ]
> A[ ,3]       # Τα στοιχεία στην 3η στήλη, και
[1] 3 7        #    παρόμοια για ολόκληρη τη στήλη n, A[ ,n]
A[ ,c(1,3)]    # Στοιχεία από περισσότερες από μια γραμμή ή στήλη,
               #    εδώ την 1η και 3η στήλη
     [,1] [,2]
[1,]    2    3
[2,]    1    7
```



```
A = matrix(
+   c(2, 4, 3, 1, 5, 7),     # τα στοιχεία του πίνακα
+   nrow=2,                  # αριθμός γραμμών
+   ncol=3,                  # αριθμός στηλών
+   byrow = TRUE)      # ο πίνακας στοιχειοθετείται κατά γραμμές

A                    # print the matrix
     [,1] [,2] [,3]
[1,]    2    4    3
[2,]    1    5    7

A[2, 3]     # τα στοιχεία της 2ης γραμμής, 3ης στήλης
[1] 7

Ολόκληρη η m-γραμμή του A μπορεί να εξαχθεί ως A[m, ].
A[2, ]      # η 2η γραμμή
[1] 1 5 7
Παρόμοια, η n-στήλη του A μπορεί να εξαχθεί ως A[ ,n].
A[ ,3]      # η 3η στήλη
[1] 3 7
```


Data elements: Διδιάστατοι πίνακες ή μήτρες (matrices)

Αν έχουμε δώσει ονομασίες στις γραμμές και στήλες, αναφορά στα στοιχεία τους μπορεί να γίνει μέσω των εν λόγω ονομάτων

```
> dimnames(A) = list(
+   c("row1", "row2"),        # ονομασίες γραμμών
+   c("col1", "col2", "col3")) # ονομασίες στηλών

> A              # print A
     col1 col2 col3
row1    2    4    3
row2    1    5    7

> A["row2", "col3"] # στοιχείο στη 2η γραμμή, 3η στήλη
[1] 7
```

Παραδείγμα: πολλαπλασιασμός πινάκων

```
> A <- matrix(c(2,3,4,5,7,6,2,4,1), nrow=3, ncol=3)
> B <- matrix(c(1,5,7,3,1,2,5,7,8), nrow=3)
> w <- c(1,2,3)
> A
     [,1] [,2] [,3]
[1,]    2    5    2
[2,]    3    7    4
[3,]    4    6    1
> B
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    5    1    7
[3,]    7    2    8
> w
[1] 1 2 3
```

```
> A*A     # πολλαπλασιασμός στοιχείο
          #    προς στοιχείο (A2[i,j])
     [,1] [,2] [,3]
[1,]    4   25    4
[2,]    9   49   16
[3,]   16   36    1
> A%*%B
     [,1] [,2] [,3]
[1,]   41   15   61
[2,]   66   24   96
[3,]   41   20   70
```


Παραδείγμα: πολλαπλασιασμός πινάκων

```
> A <- matrix(c(2,3,4,5,7,6,2,4,1), nrow=3, ncol=3)
> B <- matrix(c(1,5,7,3,1,2,5,7,8), nrow=3)
> w <- c(1,2,3)
> A
     [,1] [,2] [,3]
[1,]    2    5    2
[2,]    3    7    4
[3,]    4    6    1
> B
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    5    1    7
[3,]    7    2    8
> w
[1] 1 2 3
```

```
> w%*%A   # πολλαπλασιασμός
          #    διάνυσμα με πίνακα
     [,1] [,2] [,3]
[1,]   20   37   13

> A%*%w   # πολλαπλασιασμός
          #    πίνακα με διάνυσμα
     [,1]
[1,]   18
[2,]   29
[3,]   19
```



```
> t(B)      # υπολογίζεται ο ανάστροφος του πίνακα B
     [,1] [,2] [,3]
[1,]    1    5    7
[2,]    3    1    2
[3,]    5    7    8
```

	[, 1]	[, 2]	[, 3]
a1	3	2	5
a2	2	3	2
a3	5	2	4

```
> solve(A)    # υπολογίζεται ο αντίστροφος του πίνακα A
              a1          a2         a3
[1, ] -0.29629630 -0.07407407  0.4074074
[2, ] -0.07407407  0.48148148 -0.1481481
[3, ]  0.40740741 -0.14814815 -0.1851852
> diag(A)  # η διαγώνιος του πίνακα A
[1] 3 3 4
```



```
> diag(w) # διαγώνιος πίνακας με διαγώνιο το διάνυσμα w=(1,2,3)
     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    2    0
[3,]    0    0    3

> diag(3)  # δίνει μοναδιαίο πίνακα διαστάσεων 3x3
           #    (δοκιμάστε την εντολή για να το επιβεβαιώσετε)
```

Επίσης, δοκιμάστε να επιλύσετε ένα μικρό γραμμικό σύστημα
Au = b, π.χ. το σύστημα των εξισώσεων

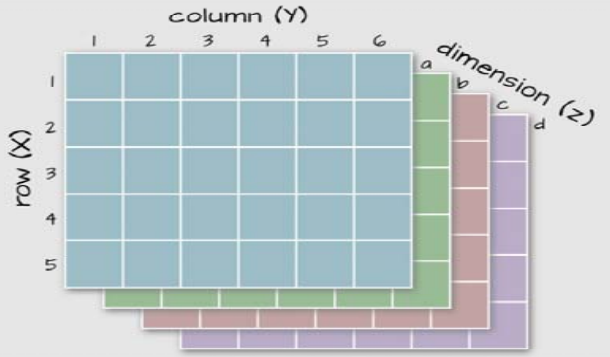
$$\begin{aligned} 2x + 3y + 4z &= 20 \\ 4y + 3z &= 17 \\ x + 2y &= 5 \end{aligned}$$

χρησιμοποιώντας τη συνάρτηση *solve(A,b)* που επιστρέφει τη λύση για το διάνυσμα $u = (x,y,z) = A^{-1}b$


```
> a <- matrix(c(1:16), ncol = 4) ; a
     [,1] [,2] [,3] [,4]
[1,]    1    5    9   13
[2,]    2    6   10   14
[3,]    3    7   11   15
[4,]    4    8   12   16
> diag(a); mode(a); length(a)
[1]  1  6 11 16
[1] "numeric"
[1] 16
> nrow(a); ncol(a); dim(a); names(a)
[1] 4
[1] 4
[1] 4 4
NULL
colnames(a) ; rownames(a) ; dimnames(a)
NULL
NULL
NULL
```

Διαχείριση δεδομένων πινάκων

Συστοιχίες Arrays



Data elements: Πολυδιάστατοι πίνακες ή συστοιχίες (arrays)

```
array(data = a list or expression vector, dim = length(data),
      dimnames = NULL or the names for the dimensions)
# dim: η διάσταση (το μήκος) της συστοιχία που θα δημιουργηθεί.
#      Ένα διάνυσμα μη αρνητικών ακέραιων τιμών (1, 2, ...) που
#      δίνουν τους μέγιστους δείκτες σε κάθε διάσταση του πίνακα

as.array(x, ...)  # Για την εκχώρηση στοιχείων σε συστοιχίες
is.array(x)       # Επιστρέφει TRUE ή FALSE ανάλογα με το αν
                  # το αντικείμενο 'x' είναι ένας πίνακας
```

Μια συστοιχία δύο διαστάσεων είναι το ίδιο όπως ένας πίνακας.

Μονοδιάστατες συστοιχίες συχνά μοιάζουν με διανύσματα, αλλά μπορεί να αντιμετωπιστούν με διαφορετικό τρόπο από ορισμένες συναρτήσεις: π.χ., str κάνει διάκριση μεταξύ τους.


```
# ARRAYS          array(data, dim = (nrow, ncol, nmat), dimnames=names)
> z = 1:24
> z
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
> dim(z) <- c(2,4,3)
> z
, , 1

     [,1] [,2] [,3] [,4]
[1,]    1    3    5    7
[2,]    2    4    6    8

, , 2

     [,1] [,2] [,3] [,4]
[1,]    9   11   13   15
[2,]   10   12   14   16

, , 3

     [,1] [,2] [,3] [,4]
[1,]   17   19   21   23
[2,]   18   20   22   24
```

- The values in the data vector give the values in the array in the same order as *"column major order"* with the first subscript moving fastest and the last subscript slowest.

Data elements: Πολυδιάστατοι πίνακες ή συστοιχίες (arrays)

Arrays γενικεύουν τους πίνακες επεκτείνοντας τις διαστάσεις τους σε παραπάνω από δύο (χωρίς κανένα περιορισμό σε αυτές).
Ένας πολυδιάστατος πίνακας είναι μια ταξινόμηση p΄q΄r΄… τιμών σε p γραμμές, q στήλες, r επίπεδα (ορόφους), … κ.ο.κ.
και πετυχαίνεται με αντιστοίχιση πολλαπλών δεικτών.
Κάθε στοιχείο του έχει δείκτες (i,j,k,…) που κυμαίνονται από 1 έως αντίστοιχα p, q, r, …

```
> x <- seq(1:9) ; x  # Διάνυσμα διαδοχικών τιμών
[1] 1 2 3 4 5 6 7 8 9
# Ένας πίνακας είναι μια συστοιχία δύο διαστάσεων
> x <- array(1:9,c(3,3)) ; x
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```

```
# recycle 1:3 "2 2/3 times"
y <- array(1:3, c(2,4)) ; y
     [,1] [,2] [,3] [,4]
[1,]    1    3    2    1
[2,]    2    1    3    2
```

Data elements: Πολυδιάστατοι πίνακες ή συστοιχίες (arrays)

Με τη χρήση της εντολής dim() μετατρέπεται ένα διάνυσμα τιμών σε μια συστοιχία καθορισμένων διαστάσεων

```
> X <- array(1:32, dim=c(2,4,4)) ; is.array(X) ; X
[1] TRUE
, , 1

     [,1] [,2] [,3] [,4]
[1,]    1    3    5    7
[2,]    2    4    6    8

, , 2

     [,1] [,2] [,3] [,4]
[1,]    9   11   13   15
[2,]   10   12   14   16

, , 3

     [,1] [,2] [,3] [,4]
[1,]   17   19   21   23
[2,]   18   20   22   24

, , 4

     [,1] [,2] [,3] [,4]
[1,]   25   27   29   31
[2,]   26   28   30   32
```

Τα στοιχεία ενός διανύσματος τιμών που εκχωρούνται σε μια συστοιχία στοιχειοθετούνται "κατά στήλες" (γραμμές, ... στήλες, ... επίπεδα)

Data elements: Πολυδιάστατοι πίνακες ή συστοιχίες (arrays)

```
> z = 1:60        # Με τη χρήση της εντολής dim() μετατρέπεται ένα διάνυσμα
> z               # τιμών σε μια συστοιχία καθορισμένων διαστάσεων
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18
[19] 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36
[37] 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54
[55] 55 56 57 58 59 60
> dim(z) <- c(3,5,4)
> z
, , 1

     [,1] [,2] [,3] [,4] [,5]
[1,]    1    4    7   10   13
[2,]    2    5    8   11   14
[3,]    3    6    9   12   15

, , 2

     [,1] [,2] [,3] [,4] [,5]
[1,]   16   19   22   25   28
[2,]   17   20   23   26   29
[3,]   18   21   24   27   30

, , 3

     [,1] [,2] [,3] [,4] [,5]
[1,]   31   34   37   40   43
[2,]   32   35   38   41   44
[3,]   33   36   39   42   45

, , 4

     [,1] [,2] [,3] [,4] [,5]
[1,]   46   49   52   55   58
[2,]   47   50   53   56   59
[3,]   48   51   54   57   60
```


Data elements: Πολυδιάστατοι πίνακες ή συστοιχίες (arrays)

```
#  Τα στοιχεία ενός array μπορούν να επιλεγούν όπως
#  τα στοιχεία των διανυσμάτων και των πινάκων.
> x[1,,]      # οι πρώτες γραμμές κάθε επιπέδου ...
     [,1] [,2] [,3] [,4]
[1,]    1    9   17   25
[2,]    3   11   19   27
[3,]    5   13   21   29
[4,]    7   15   23   31

> x[1,2,]  # από αυτές, η 2η γραμμή, και ...
[1]  3 11 19 27

> x[1,2,1] # από αυτά τα στοιχεία, το 1ο στοιχείο στη σειρά
[1] 3
```


Data elements: Πολυδιάστατοι πίνακες ή συστοιχίες (arrays)

```
#  Τα στοιχεία ενός array μπορούν να επιλεγούν όπως
#  τα στοιχεία των διανυσμάτων και των πινάκων.
> x <- array(1:20, dim=c(4,5)) ; x          # a 4x5 array.
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    5    9   13   17
[2,]    2    6   10   14   18
[3,]    3    7   11   15   19
[4,]    4    8   12   16   20
> i <- array(c(1:3,3:1), dim=c(3,2)) ; i    # a 3x2 index array.
     [,1] [,2]
[1,]    1    3
[2,]    2    2
[3,]    3    1
> x[i]  # Extract those elements
[1] 9 6 3
```

```
# Replace those elements by zeros
> x[i] <- 0  ; x
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    5    0   13   17
[2,]    2    0   10   14   18
[3,]    0    7   11   15   19
[4,]    4    8   12   16   20
```


Data elements: Πολυδιάστατοι πίνακες ή συστοιχίες (arrays)

```
> a <- array(1:24, dim=c(3,4,2)) ; is.array(a) ; a
[1] TRUE
, , 1

     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12

, , 2

     [,1] [,2] [,3] [,4]
[1,]   13   16   19   22
[2,]   14   17   20   23
[3,]   15   18   21   24
```

```
# Element indexing
> a[1,1,1]
[1] 1
> a[2,1,1]
[1] 2
> a[2,4,2]
[1] 23
> a[3,4,2]
[1] 24
```

```
> a[2,,]
     [,1] [,2]
[1,]    2   14
[2,]    5   17
[3,]    8   20
[4,]   11   23
# το ίδιο με την
# εντολή print(a)
> a[, ,]
```


Data elements: Πολυδιάστατοι πίνακες (arrays)

```
> length(ar); dim(ar); mode(ar); class(ar); nrow(ar); ncol(ar)
[1] 24
[1] 2 3 4
[1] "numeric"
[1] "array"
[1] 2
[1] 3
> dimnames(ar) <- list(c("R1","R2"),c("C1","C2","C3"),
+ c("ar1","ar2","ar3","ar4"))
> ar
, , ar1
   C1 C2 C3
R1  1  3  5
R2  2  4  6
, , ar2
   C1 C2 C3
R1  7  9 21
R2  8 10 22
....
```


Data elements: Πολυδιάστατοι πίνακες (arrays)

```
#  Τα στοιχεία ενός array μπορούν να επιλεγούν όπως τα στοιχεία των
   διανυσμάτων και των πινάκων.
> ar[,,1]
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
> ar[2,,1]
[1] 2 4 6
> ar[,2,1]
[1] 3 4
> ar[1,,]
     [,1] [,2] [,3] [,4]
[1,]    1    7   23   29
[2,]    3    9   25   41
[3,]    5   21   27   43
> ar[,2,]
     [,1] [,2] [,3] [,4]
[1,]    3    9   25   41
[2,]    4   10   26   42
```



```
> x
, , 1

     [,1] [,2] [,3] [,4]
[1,]    1    3    5    7
[2,]    2    4    6    8

, , 2

     [,1] [,2] [,3] [,4]
[1,]    9   11   13   15
[2,]   10   12   14   16

, , 3

     [,1] [,2] [,3] [,4]
[1,]   17   19   21   23
[2,]   18   20   22   24

, , 4

     [,1] [,2] [,3] [,4]
[1,]   25   27   29   31
[2,]   26   28   30   32

, , 5

     [,1] [,2] [,3] [,4]
[1,]   33   35   37   39
[2,]   34   36   38   40

, , 6

     [,1] [,2] [,3] [,4]
[1,]   41   43   45   47
[2,]   42   44   46   48

, , 7

     [,1] [,2] [,3] [,4]
[1,]   49   51   53   55
[2,]   50   52   54   56

, , 8

     [,1] [,2] [,3] [,4]
[1,]   57   59   61   63
[2,]   58   60   62   64
```



```
> x
, , 1

     [,1] [,2] [,3] [,4]
[1,]    1    3    5    7
[2,]    2    4    6    8

, , 2

     [,1] [,2] [,3] [,4]
[1,]    9   11   13   15
[2,]   10   12   14   16

, , 3

     [,1] [,2] [,3] [,4]
[1,]   17   19   21   23
[2,]   18   20   22   24

, , 4

     [,1] [,2] [,3] [,4]
[1,]   25   27   29   31
[2,]   26   28   30   32

, , 5

     [,1] [,2] [,3] [,4]
[1,]   33   35   37   39
[2,]   34   36   38   40

, , 6

     [,1] [,2] [,3] [,4]
[1,]   41   43   45   47
[2,]   42   44   46   48
```

```
> x[1,,]
     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
[1,]    1    9   17   25   33   41   49   57
[2,]    3   11   19   27   35   43   51   59
[3,]    5   13   21   29   37   45   53   61
[4,]    7   15   23   31   39   47   55   63

> x[1,2,]
[1]  3 11 19 27 35 43 51 59

> x[1,2,1]
[1] 3
```

```
> x <- 1:64 ; dim(x) <- c(2,4,8)
> x
, , 1
     [,1] [,2] [,3] [,4]
[1,]    1    3    5    7
[2,]    2    4    6    8

, , 2
     [,1] [,2] [,3] [,4]
[1,]    9   11   13   15
[2,]   10   12   14   16

, , 3
     [,1] [,2] [,3] [,4]
[1,]   17   19   21   23
[2,]   18   20   22   24

, , 4
     [,1] [,2] [,3] [,4]
[1,]   25   27   29   31
[2,]   26   28   30   32

, , 5
     [,1] [,2] [,3] [,4]
[1,]   33   35   37   39
[2,]   34   36   38   40

, , 6
     [,1] [,2] [,3] [,4]
[1,]   41   43   45   47
[2,]   42   44   46   48

, , 7
     [,1] [,2] [,3] [,4]
[1,]   49   51   53   55
[2,]   50   52   54   56

, , 8
     [,1] [,2] [,3] [,4]
[1,]   57   59   61   63
[2,]   58   60   62   64
```


Λίστες
Lists

Data elements: Λίστες δεδομένων (lists)

Είναι μια γενικότερη μορφή διανυσμάτων δεδομένων με στοιχεία όχι απαραίτητα του ίδιου τύπου. Μπορούν να περιέχουν ως στοιχεία άλλες δομές αντικειμένων (λίστες, διανύσματα, πίνακες ή ακόμα και συναρτήσεις).

```
> a = c(7, 5, 1) ; a   # διάνυσμα: σειρά δεδομένων του ίδιου τύπου
[1] 7 5 1

# λίστα: μια διατεταγμένη σειρά δεδομένων οποιουδήποτε τύπου.
> (a_list <- list(c(7, 5, 1, 19, 32),month.abb,
matrix(c(3, -8, 1, -3), nrow = 2),asin))
```

Οι λίστες δημιουργούνται με τη συνάρτηση list(), που λειτουργεί όπως και η συνάρτηση c() και προσδιορίζει τα περιεχόμενα στοιχεία, τα οποία διαχωρίζονται μεταξύ τους με κόμμα.


```
(a_list <- list(c(7, 5, 1, 19, 32),month.abb,
  matrix(c(3, -8, 5, -3), nrow = 2),asin))
[[1]]
[1]  7  5  1 19 32

[[2]]
[1] "Jan" "Feb" "Mar" "Apr" "May" "Jun" "Jul" "Aug" "Sep"
[10] "Oct" "Nov" "Dec"

[[3]]
     [,1] [,2]
[1,]    3    5
[2,]   -8   -3

[[4]]
function (x)  .Primitive("asin")
```


Πρόσβαση στα στοιχεία μιας λίστας δεδομένων

```
> y <- list(name="Michael", gender="M", university="NTUA") ; y
y$name          # μέσω των ονομάτων/ετικετών τους
[1] "Michael"
y$gender
[1] "M"
y$university
[1] "NTUA"

y$[[1]]         # ή εναλλακτικά μέσω των δεικτών τους
[1] "Michael"
y$[[2]]
[1] "M"
y$[[3]]
[1] "NTUA"
```

Γενικά, η επιλογή του i-στοιχείου, με ονομασία 'object_name', σε μια λίστα L, γίνεται ως L[[i]] ή L$object_name

```
>length(y)      # το πλήθος των
[1] 3           # στοιχείων της λίστας
```



```
>x <- list(age=28,married=F,location="Athens")
>y <- list(name="Michael", gender="M", university="NTUA")
>z <- list(fee=1000.24,award="yes")
>final_list <- c(x,y,z)    # Συνένωση λιστών
>final_list
$age                    $gender
[1] 28                  [1] "M"

$married                $university
[1] F                   [1] "NTUA"

$location               $fee
[1] "Athens"            [1] 1000.24

$name                   $award
[1] "Michael"           [1] "yes"
```


Data elements: Λίστες δεδομένων (lists)

```
> (a_list <- list(c(7, 5, 1, 19, 32),month.abb,
  matrix(c(3, -8, 5, -3), nrow = 2),asin))

> a_list$[[1]]      # 1ο αντικείμενο της λίστας, και ...
[1]  7  5  1 19 32
> sum(a_list[[1]]) # το άθροισμα των στοιχείων του
[1] 64
> sum(a_list[1])   # σφάλμα εντολής, λόγω απλής αγκύλης
Error in sum(a_list[1])  : invalid 'type' (list) of argument
# Εφαρμογή συναρτήσεων σε λίστα/χρήση της συνάρτησης 'lapply'
> lapply(a_list[[1]], length) ; lapply(a_list[[2]], length)
[1] 5
[1] 12
> lapply(a_list[[2]], mean)
[1] NA
```

lapply(X, FUN, ...)


```
> print(list1)
[[1]]
[1] "Jazz"
[[2]]
[1] "Rock&Roll"
[[3]]
[1] "Blues"
[[4]]
[1] "Pop"
[[5]]
[1] "Folk"
[[6]]
 [1] 1.761643 2.498069 2.400210 2.124113
1.426596 2.578713 1.540195 2.365419
 [9] 2.574228 1.795548
[[7]]
[1] 10 13 16 19 22
[[8]]
[1] TRUE TRUE FALSE TRUE FALSE FALSE
```

Πως εξάγουμε (π.χ. αριθμητικά) στοιχεία από ένα *list* για περαιτέρω χρήση ;


```
> v1 <- unlist(list1)     # convert list to vector
> print(v1)
 [1] "Jazz"          "Rock&Roll"     "Blues"         "Pop"
 [5] "Folk"          "1.76164319321726" "2.49806915918438"
"2.40021026497148"
 [9] "2.12411294667475" "1.42659565789063"
"2.57871302813526" "1.5401945826225"
[13] "2.36541887299244" "2.57422803675644"
"1.79554783736789" "10"
[17] "13"            "16"            "19"            "22"
[21] "TRUE"          "TRUE"          "FALSE"         "TRUE"
[25] "FALSE"         "FALSE"
```

Η εντολή *unlist()* μετατρέπει μια λίστα σε διάνυσμα τιμών

```
> vv1 <- as.numeric(v1[6:12])      # Εξαγωγή στοιχείων ως 'numeric'
> vv1
[1] 1.880632 3.264729 1.922174 1.998035 1.734622 1.996475
1.853654
> typeof(vv1)
[1] "double"
> class(vv1)
[1] "numeric"
>
> v.squared <- vv1^2               # περαιτέρω διαχείριση των στοιχείων
> v.squared
[1]  3.536776 10.658458  3.694754  3.992145  3.008913
3.985913  3.436034
```


Παράγοντες Factors

Αντικείμενα δεδομένων = Κατηγοριοποίηση δεδομένων
□ Αποθήκευση σε επίπεδα

Data elements: Factors / Παράγοντες

- A **factor** stores the nominal values as a vector of integers in the range [1... k] (where k is the number of unique values in the nominal variable), and an internal vector of character strings (the original values) mapped to these integers

```
# variable rating with 20 "old" entries and 30 "new" entries
> rating <- c(rep("old",20), rep("new", 30))
> rating <- factor(rating)
# stores rating as 30 1s and 20 2s and associates 1=new, 2=old internally (alphabetically). R now treats rating as a nominal variable
```


Data elements: Factors / Παράγοντες

Παράγοντες, στην R, *είναι διανύσματα που μπορούν να περιέχουν* μόνο προκαθορισμένες τιμές. Αποτελούν έναν εξαιρετικά χρήσιμο και αποτελεσματικό τρόπο κωδικοποίησης, αποθήκευσης και χειρισμού κατηγορικών δεδομένων με πολλαπλές εισόδους.

Δημιουργούνται από διανύσματα ακέραιων τιμών χρησιμοποιώντας τις ιδιότητες: class() , και factor() -οι οποίες τα κάνουν να συμπεριφέρονται διαφορετικά από τα κανονικά διανύσματα ακέραιων τιμών- και levels(), τα επίπεδα ή στάθμες, που καθορίζουν το σύνολο των επιτρεπόμενων τιμών.

```
> x <factor(c("a", "b", "b", "a")) ; x
[1] a b b a
Levels: a b
class(x) ; levels(x)
[1] "factor"
[1] "a" "b"
```


Factors – Γενικά χαρακτηριστικά τους

- Οι συντελεστές στο R αποθηκεύονται ως διανύσματα τιμών ακέραιων τιμών με ένα αντίστοιχο σύνολο τιμών χαρακτήρων που χρησιμοποιούνται όταν εμφανίζεται ο παράγοντας.
- Το μόνο απαιτούμενο όρισμα για τη συνάρτηση *factor()* είναι ένα διάνυσμα τιμών που θα επιστραφεί ως διάνυσμα τιμών κατηγορικών συντελεστών.
- Τόσο αριθμητικές, όσο και μεταβλητές χαρακτήρων μπορούν να γίνουν παράγοντες, αλλά τα επίπεδα ενός παράγοντα θα είναι πάντα τιμές χαρακτήρων.
- Οι κατηγορικές μεταβλητές εισέρχονται σε στατιστικά μοντέλα διαφορετικά από τις συνεχείς μεταβλητές → Μια από τις σημαντικότερες χρήσεις των παραγόντων είναι η στατιστική μοντελοποίηση

Data elements: Factors / Παράγοντες

```
        # Create a vector as input.
data<-c("East","West","East","South","North","East","West","South")
print(data) ; print(is.factor(data))
[1] "East"  "West"  "East"  "South" "North" "East"  "West"  "South"
[1] FALSE

fdata <- factor(data)     # Apply the factor function.
print(fdata) ; print(is.factor(fdata))
[1] East  West  East  South North East  West  South
Levels: East North South West      □ αλφαβητική σειρά
[1] TRUE
neworder <- factor(fdata,levels = c("East","West","North", "South"))
print(neworder)
[1] East  West  East  South North East  West  South
Levels: East West North South
```


Data elements: Factors / Παράγοντες

Έστω η κατηγορική μεταβλητή gnss με τιμές gps, glonass, egnos
Ο παράγοντας που περιγράφει μια σειρά 5 μετρήσεων εντοπισμού με δορυφόρους των εν λόγω συστημάτων

```
> gnss <- factor(c("gps","glonass","glonass","egnos","gps")) ; gnss
[1] gps glonass glonass egnos gps
Levels: egnos glonass gps
      # Σημ., οι στάθμες του παράγοντα αν και συμβολοσειρές
      # τυπώνονται χωρίς τα εισαγωγικά, και αλφαβητικά
> print.default(gnss)
[1] 3 2 2 1 3       # οι αριθμοί αντιστοιχούν στα levels
> attributes(gnss) # τα ίδια τα ονόματα θεωρούνται ως attributes
$levels
[1] "egnos" "glonass" "gps"
$class
[1] "factor"
```


Data elements: Factors / Παράγοντες

- Πλεονεκτήματα για τη μετατροπή κατηγορηματικών μεταβλητών σε μεταβλητές παράγοντα …
  - Μπορούν να χρησιμοποιηθούν σε στατιστική μοντελοποίηση, όπου θα τους ανατεθεί ο σωστός αριθμός των βαθμών ελευθερίας.
  - Πολύ χρήσιμο σε πολλούς διαφορετικούς τύπους γραφικών
  - Η αποθήκευση μεταβλητών 'αλληλουχίας χαρακτήρων' ως μεταβλητές 'παράγοντες' → μια πιο αποτελεσματική χρήση της μνήμης.

Data elements: Factors / Παράγοντες

```
factor(x = character(), levels, labels = levels,
       exclude = NA, ordered = is.ordered(x))
# x: a vector of data

> v <- c(1,3,5,8,2,1,3,5,3,5)
> is.factor(v)
[1] FALSE

> factor(v)  # Calculates the categorical distribution
 [1] 1 3 5 8 2 1 3 5 3 5
Levels: 1 2 3 5 8

> x <- factor(v)
> x
 [1] 1 3 5 8 2 1 3 5 3 5
Levels: 1 2 3 5 8
> is.factor(x)
[1] TRUE
```

Data elements: Factors / Παράγοντες

```
> x <- factor(v, levels=c(2,1))  # Select levels
> x
 [1] 1    <NA> <NA> <NA> 2    1    <NA> <NA> <NA>
<NA>
Levels: 2 1

> levels(x) <- c("two","one")  # Change the level value
> x
 [1] one  <NA> <NA> <NA> two  one  <NA> <NA>
<NA> <NA>
Levels: two one
```


Data elements: Factors / Παράγοντες

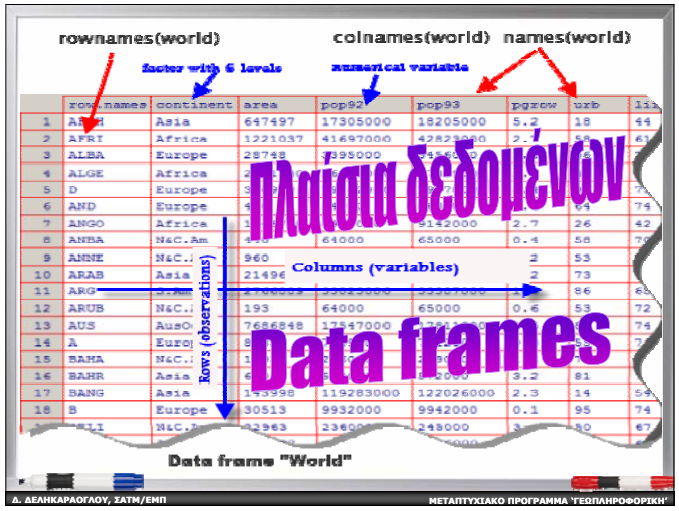
```
> months = c("March","April","January","November","January",
"September","October","September","November","August",
"January","November","November","February","May","August",
"July","December","August","August","September","November",
"February","April")
> months = factor(months)
> table(months)
months
    April    August  December  February   January      July     March       May
        2         4         1         2         3         1         1         1
 November   October September
        5         1         3
# Creating an ordered factor is reflected in the output of the table function
> months = factor(months,levels=c("January","February","March",
+ "April","May","June","July","August","September","October",
"November","December"),ordered=TRUE)
> months[1] < months[2]
[1] TRUE
> table(months)
months
  January  February     March     April       May      June      July    August
        3         2         1         2         1         0         1         4
September   October  November  December
        3         1         5         1
```


Data elements: Factors / Παράγοντες

```
set.seed(124)
          # set the seed to some number, e.g. 124,
          # in order to make the results reproducible
gnss <- sample(0:1, 20, replace = TRUE)
gnss
> [1] 0 0 1 0 0 0 1 0 1 0 1 1 1 1 0 0 1 1 1 0
is.factor(gnss)
> [1] FALSE
is.numeric(gnss)
> [1] TRUE
gnss.f <- factor(gnss, labels = c("gps",
"glonass"))
gnss.f
> [1] gps gps glonass gps gps gps glonass  gps
> [9] glonass gps glonass glonass glonass  glonass
gps gps
> [17] glonass glonass glonass gps
> Levels: gps glonass
is.factor(gnss.f)
> [1] TRUE
```


Data elements: Factors / Παράγοντες

```
fgnss <- c("gps", "glonass", "gps", "beidou", "gps", "beidou",
"glonass", "gps", "glonass", "glonass", "glonass", "glonass",
"beidou", "galileo", "galileo", "gps", "glonass", "glonass",
"gps", "galileo")

is.factor(fgnss)
> [1] FALSE
is.character(fgnss)
> [1] TRUE

gnss.f.order <- factor(fgnss)
is.factor(gnss.f.order)
> [1] TRUE
    # levels are ordered in alphabetical order
levels(gnss.f.order)
> [1] "beidou"   "galileo"   "glonass"    "gps"

    # to indicate the correct ordering of the categories
gnss.c <- factor(gnss, levels = c("gps", "glonass", "beidou",
"galileo"))
is.factor(gnss.c)
> [1] TRUE
levels(gnss.c)
> [1] "gps", "glonass", "beidou", "galileo"
```

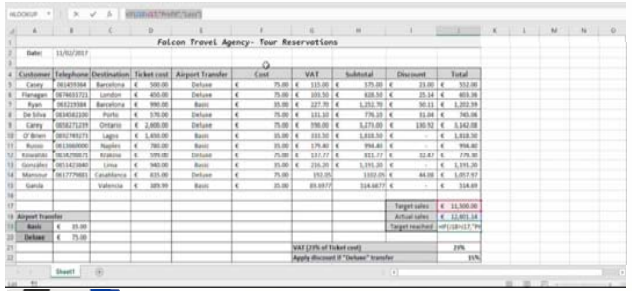
Data elements: Πλαίσια δεδομένων (data frames)

General Definition

- A data frame is a table, or two-dimensional array-like structure, in which each column contains measurements on one variable, and each row contains one case.
- Technically, in R a data frame is a list of column vectors. Unlike an array, the data you store in the columns of a data frame can be of various types: i.e., one column might be a numerical variable, another might be a factor, and a third might be a character variable. All columns have to be the same length (contain the same number of data items).

Data elements: Πλαίσια δεδομένων (data frames)

- Αντιπροσωπεύουν τυπικούς πίνακες δεδομένων – *κάτι ανάλογο με ένα υπολογιστικό φύλλο excel*



Data elements: Πλαίσια δεδομένων (data frames)

- Είναι ουσιαστικά ένας ορθογώνιος πίνακας τιμών σε σειρές και στήλες → *is more general than a matrix*
  - Τα δεδομένα σε κάθε στήλη έχουν τον ίδιο τύπο (π.χ. αριθμός, χαρακτήρες, λογικές μεταβλητές), αλλά διαφορετικές στήλες μπορεί να έχουν δεδομένα με διαφορετικούς τύπους
- Είναι από τις πιο σημαντικές δομές δεδομένων στην R.

- Ένα παράδειγμα πλαισίου δεδομένων είναι το σετ δεδομένων 'iris' που είναι ενσωματωμένο στην R, του οποίου οι στήλες είναι δεδομένα διαφορετικού τύπου

```
> iris[c(1:3,147:150), , ]
```

Τρεις πρώτες και τρεις τελευταίες γραμμές του σετ δεδομένων 'iris'

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
147	6.3	2.5	5.0	1.9	virginica
148	6.5	3.0	5.2	2.0	virginica
149	6.2	3.4	5.4	2.3	virginica
150	5.9	3.0	5.1	1.8	virginica

Data elements: Πλαίσια δεδομένων (data frames)

- Ένα πλαίσιο δεδομένων είναι μία λίστα από διανύσματα, παράγοντες ή άλλα πλαίσια που ***όλα έχουν το ίδιο μήκος*** (στη περίπτωση των πινάκων οι γραμμές έχουν το ίδιο μήκος).
  - Σε ένα πλαίσιο δεδομένων η θεώρηση γίνεται ανά στήλη που είναι και οι μεταβλητές του πλαισίου δεδομένων
  - Είναι οι δομές της R που βρίσκονται πιο κοντά στην έννοια των συνόλων δεδομένων (*datasets*) άλλων λογισμικών στατιστικής ανάλυσης, π.χ. του SPSS

Data elements: Πλαίσια δεδομένων (data frames)

- Είναι ο πιο κοινός τρόπος για την αποθήκευση δεδομένων στο R
  - κάνουν ευκολότερη την ανάλυση των δεδομένων
- Ουσιαστικά επειδή είναι μια δομή 2-διαστάσεων,
  - μοιράζονται τις ιδιότητες τόσο των πινάκων (matrices), όσο και των λιστών (lists).
- Αυτό σημαίνει ότι ένα πλαίσιο δεδομένων έχει names (), colnames (), και rownames ().
- Το μήκος () ενός πλαισίου δεδομένων είναι το μήκος της υποκείμενης λίστας → ncol ().
  - nrow () δίνει τον αριθμό των γραμμών

Data elements: Πλαίσια δεδομένων (data frames)

- Δημιουργούνται με την εντολή *data.frame()*

Ως default η εντολή *data.frame ()* μετατρέπει σειρά χαρακτήρων σε παράγοντες (factors)

```
df <- data.frame(x = 1:3, y = c("a", "b", "c"))
str(df)       # show the structure of the data frame
> 'data.frame':    3 obs. of  2 variables:
> $ x: int  1 2 3
> $ y: Factor w/ 3 levels "a","b","c": 1 2 3
```

καταστολή αυτής της συμπεριφοράς

```
df <- data.frame(
  x = 1:3, y = c("a", "b", "c"),
  stringsAsFactors = FALSE)
str(df)
> 'data.frame':    3 obs. of  2 variables:
> $ x: int  1 2 3
> $ y: chr  "a" "b" "c"
```


Data elements: Πλαίσια δεδομένων (data frames)

```
#  Μερικές χρήσιμες εντολές για το χειρισμό πλαισίων δεδομένων
> length(my.dataframe)
[1] 5
> dim(my.dataframe)
[1] 20 5
> is.data.frame(my.dataframe)
[1] TRUE
> is.list(my.dataframe)
[1] TRUE
> is.matrix(my.dataframe)
[1] TRUE
> is.vector(my.dataframe)
[1] FALSE
> names(my.dataframe)
[1] "my.logic" "my.complex" "my.numeric" "X1" "X2"
```


Data elements: Πλαίσια δεδομένων (data frames)

```
> tmin <- c(50.7,52.8,48.6,53.0,49.9,47.9,54.1,47.6,43.6,45.5)
> tmax <- c(59.5,55.7,57.3,71.5,69.8,68.8,67.5,66.0,66.1,61.7)
> (temps = data.frame(day=1:10, min=tmin, max=tmax))
   min  max
1  50.7 59.5
2  52.8 55.7
3  48.6 57.3
4  53.0 71.5
5  49.9 69.8
6  47.9 68.8
7  54.1 67.5
8  47.6 66.0
9  43.6 66.1
10 45.5 61.7
```

- Για τα μικρά σύνολα δεδομένων, κάθε μία από τις στήλες (μεταβλητές) του πλαισίου δεδομένων μπορούν να εισαχθούν χρησιμοποιώντας τη συνάρτηση *data.frame*
  - Τα ονόματα των στηλών (αν ορίζονται με τη δημιουργία του πλαισίου) εμφανίζονται με τα δεδομένα.
  - Ονόματα μπορούν να προστεθούν και μετά το γεγονός με τη συνάρτηση '*names()*'.

Data elements: Πλαίσια δεδομένων (data frames)

```
> tmin <- c(50.7,52.8,48.6,53.0,49.9,47.9,54.1,47.6,43.6,45.5)
> tmax <- c(59.5,55.7,57.3,71.5,69.8,68.8,67.5,66.0,66.1,61.7)
> (temps = data.frame(day=1:10, min=tmin, max=tmax))
   min  max
1  50.7 59.5
2  52.8 55.7
3  48.6 57.3
4  53.0 71.5
5  49.9 69.8
6  47.9 68.8
7  54.1 67.5
8  47.6 66.0
9  43.6 66.1
10 45.5 61.7
# Οι πληροφορίες για τη μορφή και την κλάση ενός
# πλαισίου δεδομένων δεν είναι πολύ κατατοπιστικές
> mode(temps)
[1] "list"
> class(temps)
[1] "data.frame"

# Η συνάρτηση 'sapply' παρέχει αντίστοιχες
# πληροφορίες για κάθε στήλη του πλαισίου
> sapply(temps,mode)
       date         min     maximum
"numeric" "numeric" "numeric"
```


Data elements: Πλαίσια δεδομένων (data frames)

```
> tmin <- c(50.7,52.8,48.6,53.0,49.9,47.9,54.1,47.6,43.6,45.5)
> tmax <- c(59.5,55.7,57.3,71.5,69.8,68.8,67.5,66.0,66.1,61.7)
> (temps = data.frame(day=1:10, min=tmin, max=tmax))
   min  max
1  50.7 59.5
2  52.8 55.7
3  48.6 57.3
4  53.0 71.5
5  49.9 69.8
6  47.9 68.8
7  54.1 67.5
8  47.6 66.0
9  43.6 66.1
10 45.5 61.7
# Η χρήση ενός μόνο δείκτη, με ένα πλαίσιο δεδομένων,
# αναφέρεται ακριβώς σε αυτή τη στήλη.
 > temps['max']
   max
1  59.5
2  55.7
3  57.3
4  71.5
5  69.8
6  68.8
7  67.5
8  66.0
9  66.1
10 61.7

> temps$max
# ή ισοδύναμα οι εντολές ' temps[,3] ' , ' temps[,"max"] '
# και ' temps[['max']] ' δίνουν το ίδιο αποτέλεσμα,
# δηλ. την εξαγωγή της συγκεκριμένης στήλης δεδομένων
 [1] 59.5 55.7 57.3 71.5 69.8 68.8 67.5 66.0 66.1 61.7
```


Data elements: Πλαίσια δεδομένων (data frames)

```
d <- c(1,2,3,4)
e <- c("red", "white", "red", NA)
f <- c(TRUE,TRUE,TRUE,FALSE)
mydata <- data.frame(d,e,f)
    # variable names
names(mydata) <- c("ID","Color","Passed")
```

Διάφοροι τρόποι για να εκχωρηθούν και να προσδιοριστούν τα στοιχεία ενός πλαισίου δεδομένων

```
    # columns 3,4,5 of data frame
> myframe[3:5]
    # columns ID and Color from data frame
> myframe[c("ID","Color")]
    # variable x1 in the data frame
> myframe$X1
```


Data elements: Πλαίσια δεδομένων (data frames)

```
# create a data frame from scratch
age <- c(25, 30, 56)
gender <- c("male", "female", "male")
weight <- c(160, 110, 220)
mydata <- data.frame(age,gender,weight) # df είναι επίσης ένα
                                        πλαίσιο δεδομένων
```

Στο R υπάρχουν ενσωματωμένα (build-in) πλαίσια δεδομένων για να πειραματιστείτε: π.χ. το *iris* έχει 5 μεταβλητές και 150 παρατηρήσεις

```
> iris[c(1:4,147:150), , ]   # δοκιμάστε την εντολή και θα πάρετε
                             τις τέσσερεις πρώτες και τέσσερεις
                             τελευταίες γραμμές του εν λόγω πλαισίου
                             δεδομένων
```

Data elements: Πλαίσια δεδομένων (data frames)

```
> measrs <- data.frame(gender = c("M", "M",
+     "F"), ht = c(172, 186.5, 165), wt = c(91,
+     99, 74))
> measrs
```

	Gender	ht	wt
1	M	172.0	91
2	M	186.5	99
3	F	165.0	74

Entries in a data.frame are indexed like a matrix:

```
> measrs[1, 2]
[1] 172
```


Data elements: Πλαίσια δεδομένων (data frames)

- Όλα τα στοιχεία σε ένα πλαίσιο δεδομένων μπορεί να εξαχθούν ως διανύσματα με το αντίστοιχο όνομα:

```
> height <- measrs$ht #  as a vector -- in a list
> height
[1] 172.0 186.5 165.0

> attach(measrs)
> wt          #  as a vector with its name
[1] 91 99 74
> detach(measrs)
```


Data elements: Πλαίσια δεδομένων (data frames)

- Components can also be added to a data frame in the intuitive way

```
> measrs$age <- c(28, 55, 43)
> Measrs
```

	gender	ht	wt	age
S1	M	172.0	91	28
S2	M	186.5	99	55
S3	F	165.0	74	43

Data elements: Πλαίσια δεδομένων (data frames)

```
> tmin <- c(50.7,52.8,48.6,53.0,49.9,47.9,54.1,47.6,43.6,45.5)
> tmax <- c(59.5,55.7,57.3,71.5,69.8,68.8,67.5,66.0,66.1,61.7)
> (temps = data.frame(day=1:10, min=tmin, max=tmax))
```

	min	max
1	50.7	59.5
2	52.8	55.7
3	48.6	57.3
4	53.0	71.5
5	49.9	69.8
6	47.9	68.8
7	54.1	67.5
8	47.6	66.0
9	43.6	66.1
10	45.5	61.7

- Αριθμητικοί υπολογισμοί μεταξύ στηλών (εφ' όσον εκφράζουν συναφή μεγέθη) γίνονται παρόμοια με το χειρισμό διανυσμάτων τιμών

```
# Η εντολή ... που μετατρέπει από βαθμούς Farenheit σε Celcius
> 5/9*(temps$max-32) - 5/9*(temps$min-32)
[1]  4.888889  1.611111  4.833333 10.277778 11.055556
[6] 11.611111  7.444444 10.222222 12.500000  9.000000

# μπορεί να συντομευθεί, με τη χρήση της συνάρτησης 'with'
> with(temps,5/9*(max-32) - 5/9*(min-32))
[1]  4.888889  1.611111  4.833333 10.277778 11.055556
[6] 11.611111  7.444444 10.222222 12.500000  9.000000
```


Στη συνέχεια …

Μένει να δούμε πως εισάγουμε δεδομένα στο R από αρχεία διαφόρων μορφοτύπων