Ανάλυση Δεδομένων με χρήση του Στατιστικού Πακέτου R

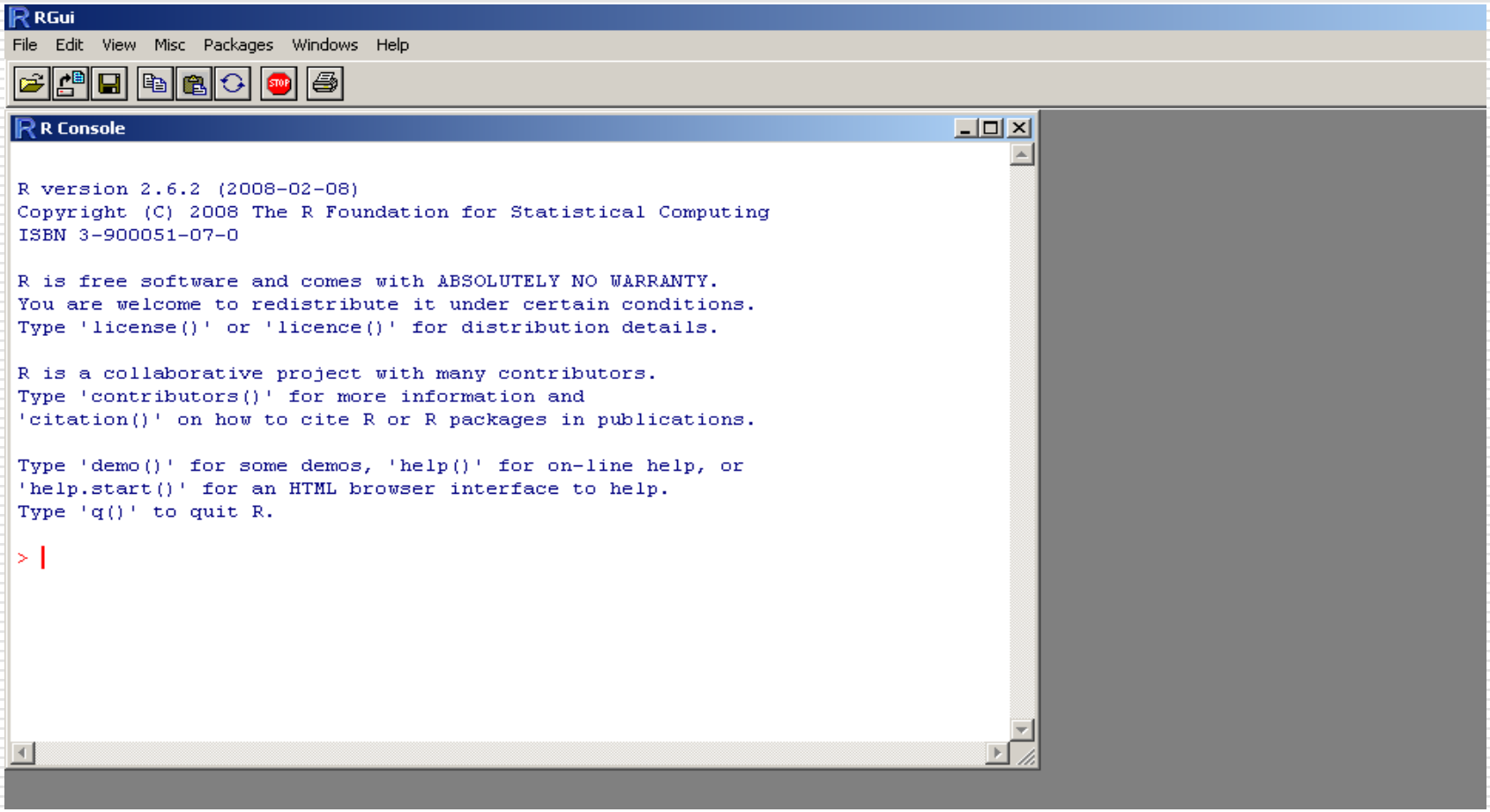
Δημήτρης Φουσκάκης,
Επίκουρος Καθηγητής,
Τομέας Μαθηματικών,
Σχολή Εφαρμοσμένων Μαθηματικών και Φυσικών Επιστημών,
Εθνικό Μετσόβιο Πολυτεχνείο.

Περιεχόμενα

Τι είναι το πακέτο R

- Η R είναι μια γλώσσα προγραμματισμού που χρησιμεύει κυρίως για ανάλυση δεδομένων και εφαρμογή διαφόρων "κλασικών" και σύγχρονων στατιστικών τεχνικών.
- Μπορεί να αποκτηθεί δωρεάν από την ιστοσελίδα http://www.r-projects.org ή από ένα από τα πολλά πρότυπα (mirrors) του CRAN (Comprehensive R Archive) http://cran.r-project.org το οποίο είναι ένα δίκτυο διανομής της R σε πολλά μέρη του κόσμου μέσω διαδικτύου. Υποστηρίζει πολλές πλατφόρμες και λειτουργικά όπως Linux, Mac OS και Windows.
- Μπορεί να χρησιμοποιηθεί είτε με κατευθείαν εντολές που υπάρχουν είτε με προγράμματα που ο χρήστης μπορεί να προγραμματίσει για επίλυση πιο πολύπλοκων στατιστικών προβλημάτων. Επίσης ο χρήστης μπορεί να χρησιμοποιήσει και έτοιμα προγράμματα τα οποία είναι ενσωματωμένα μέσα σε πακέτα τα οποία διατίθενται πάλι ελεύθερα. Οι ποικιλία τέτοιων προγραμμάτων είναι τεράστια.
- Στις συγκεκριμένες σημειώσεις χρησιμοποιούμε την έκδοση 2.6.2.

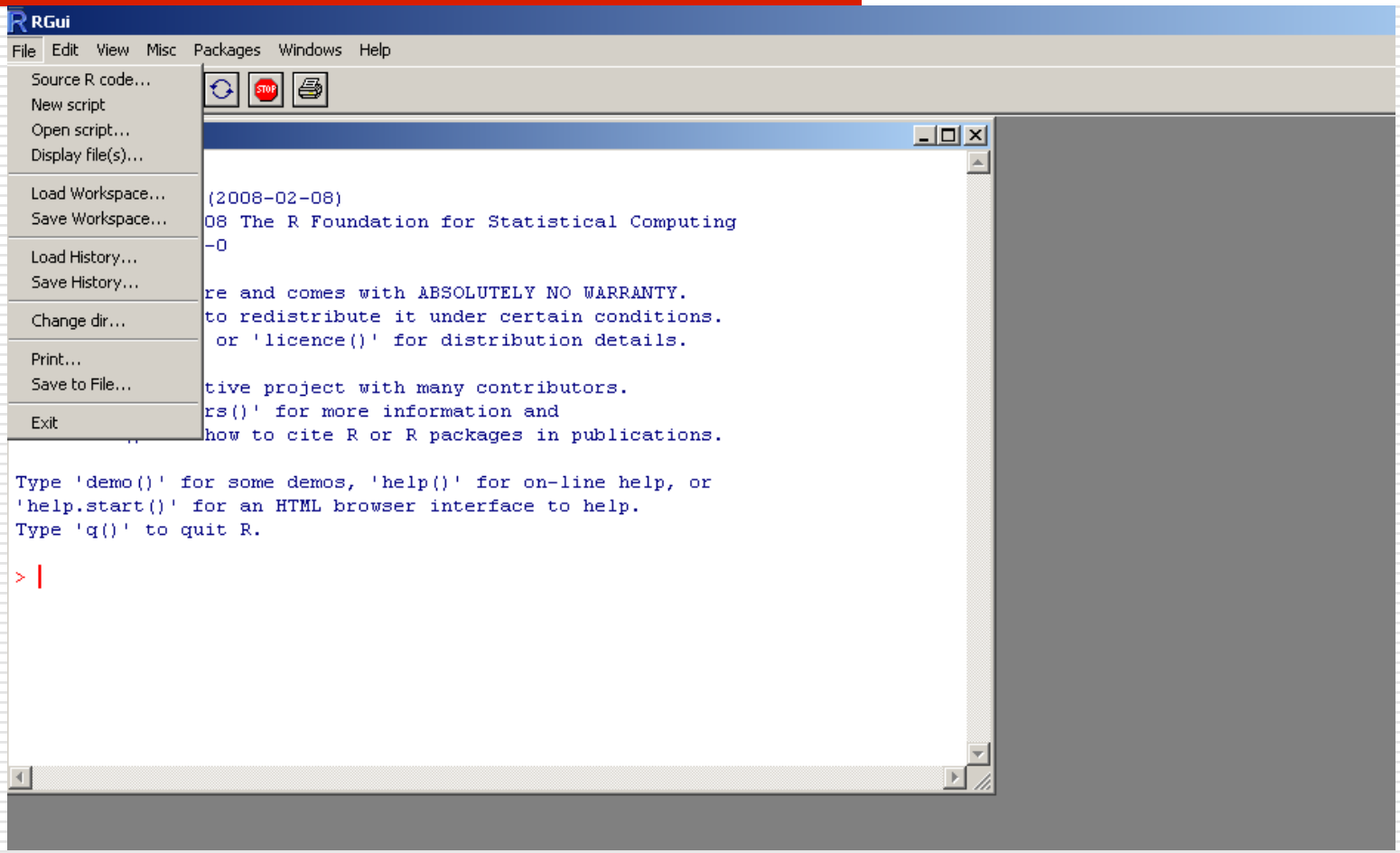
Το περιβάλλον της R



Το περιβάλλον της R

- Η εκκίνηση του προγράμματος γίνεται με διπλό κλικ στο εικονίδιο της R. Τότε εμφανίζεται η βασική οθόνη του προγράμματος (σαν αυτή της προηγουμένης διαφάνειας) στην οποία υπάρχει το παράθυρο εντολών (R-console). Ο κέρσορας βρίσκεται μετά το σύμβολο “>” και το πρόγραμμα περιμένει τις εντολές σας.
- Για να τερματίσετε το πρόγραμμα είτε πληκτρολογήστε q(), είτε κλείστε την οθόνη του προγράμματος (όχι του παραθύρου εντολών) πάνω δεξιά, είτε από το μενού file επιλέξτε το exit. Σε κάθε περίπτωση θα ερωτηθείτε αν θέλετε να αποθηκεύσετε ότι έχετε μέχρι τώρα δημιουργήσει (αντικείμενα, συναρτήσεις, κλπ).

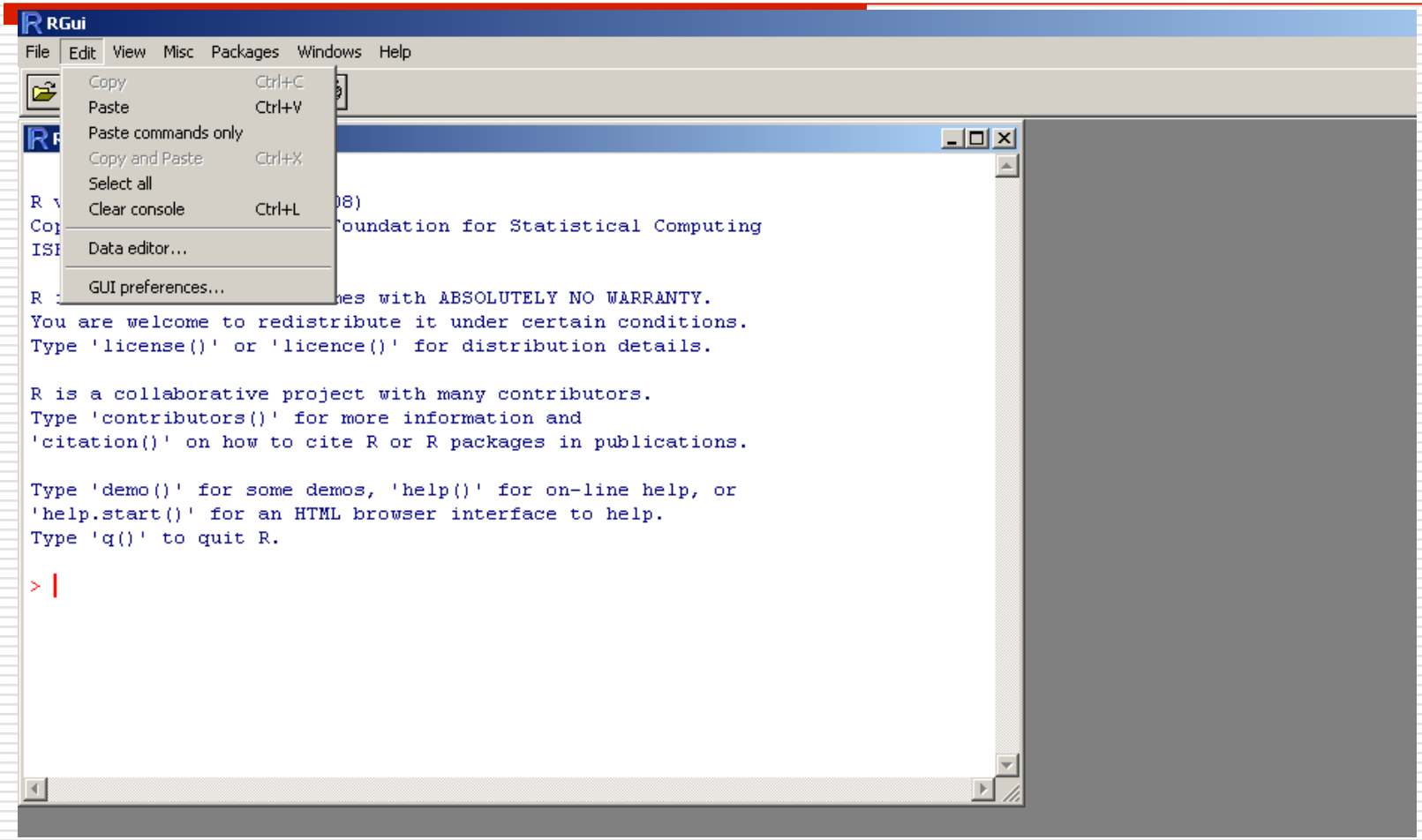
Το μενού File



Το μενού File

- Με την επιλογή αυτή μπορούμε:
  - Να εισάγουμε κώδικα και εντολές από προηγούμενες εφαρμογές μας με το source R code.
  - Να ανοίξουμε έναν νέο συντάκτη (new script) στον οποίο να τυπώσουμε τις εντολές που θέλουμε να εκτελέσουμε. Μαυρίζουμε με το ποντίκι τις εντολές που θέλουμε να εκτελέσουμε και με δεξί κλικ πάνω στον συντάκτη διαλέγουμε την επιλογή run line or selection.
  - Να ανοίξουμε έναν παλιό συντάκτη (open script) από τον οποίο θέλουμε να εκτελέσουμε κάποιες εντολές. Οι εκτέλεση γίνεται όπως και πριν.
  - Να δούμε τα διαθέσιμα R αρχεία του φακέλου που είμαστε (display files).
  - Να εισάγουμε ή να αποθηκεύσουμε επιφάνειες εργασίας (workspace) με αντικείμενα και συναρτήσεις που έχουν δημιουργηθεί (load/save workspace).
  - Να εισάγουμε ή να αποθηκεύσουμε εντολές που ήδη έχουμε χρησιμοποιήσει (load/save history).
  - Να αλλάξουμε τον φάκελο εργασίας μας (change dir).
  - Να εκτυπώσουμε (print), να αποθηκεύσουμε την δουλειά μας σε ένα .txt αρχείο (save to file) και να τερματίσουμε το πρόγραμμα (exit).

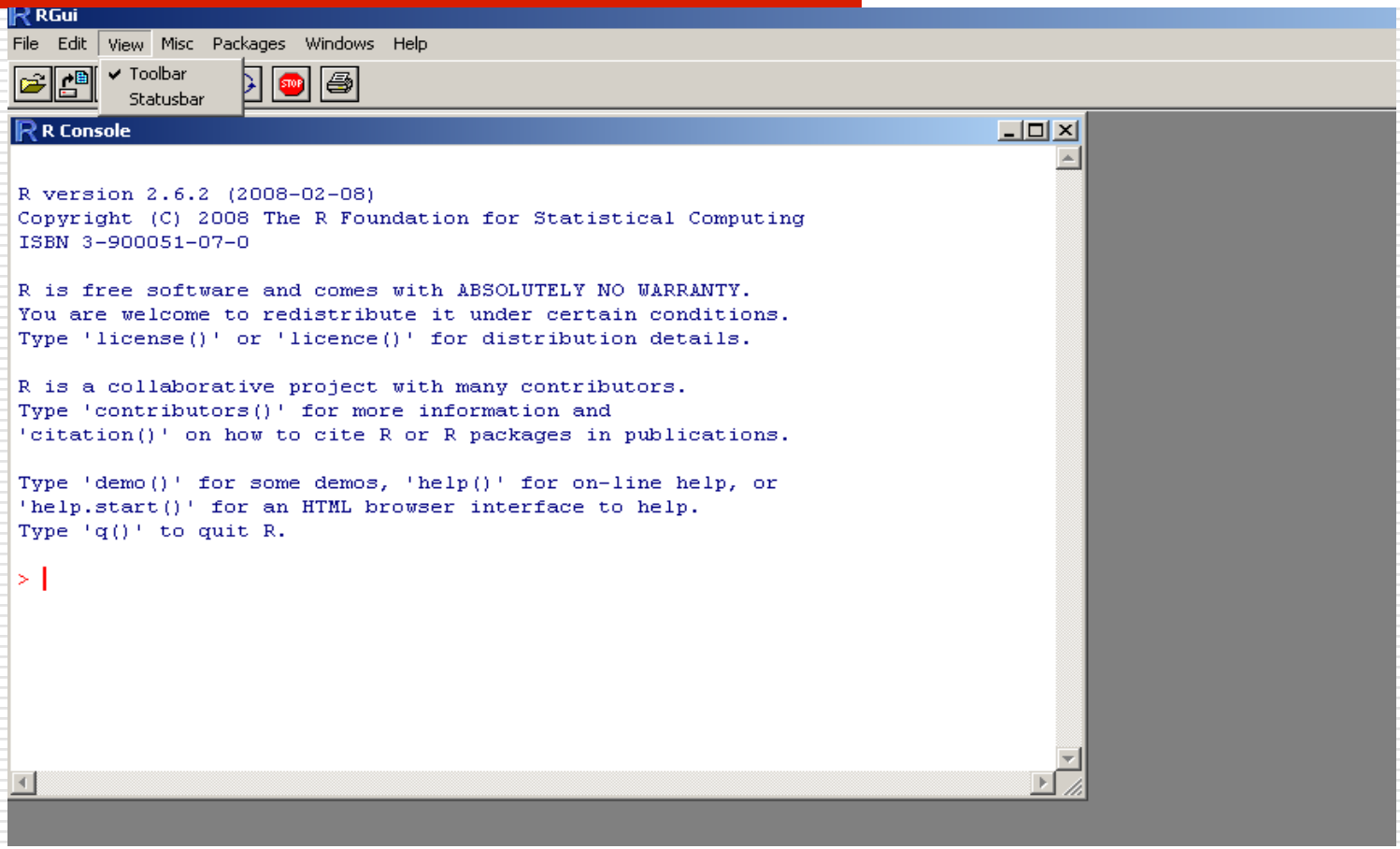
Το μενού Edit



Το μενού Edit

- Εδώ έχουμε τις γνωστές δυνατότητες αντιγραφής (copy) και επικόλλησης (paste), επιλογής όλων όσων έχουμε πληκτρολογήσει (select all), καθαρισμού του παραθύρου εντολών (clear console). Επίσης μπορούμε να ανοίξουμε τον συντάκτη δεδομένων (data editor) για κάποιο σετ δεδομένων που είναι υπό μορφή πλαισίου δεδομένων – data frame (περισσότερα για τα πλαίσια δεδομένων αργότερα) – και να επεξεργαστούμε αυτά τα δεδομένα. Τέλος μπορούμε να αλλάξουμε το τρόπο που φαίνεται το περιβάλλον εργασίας μας (GUI preferences).

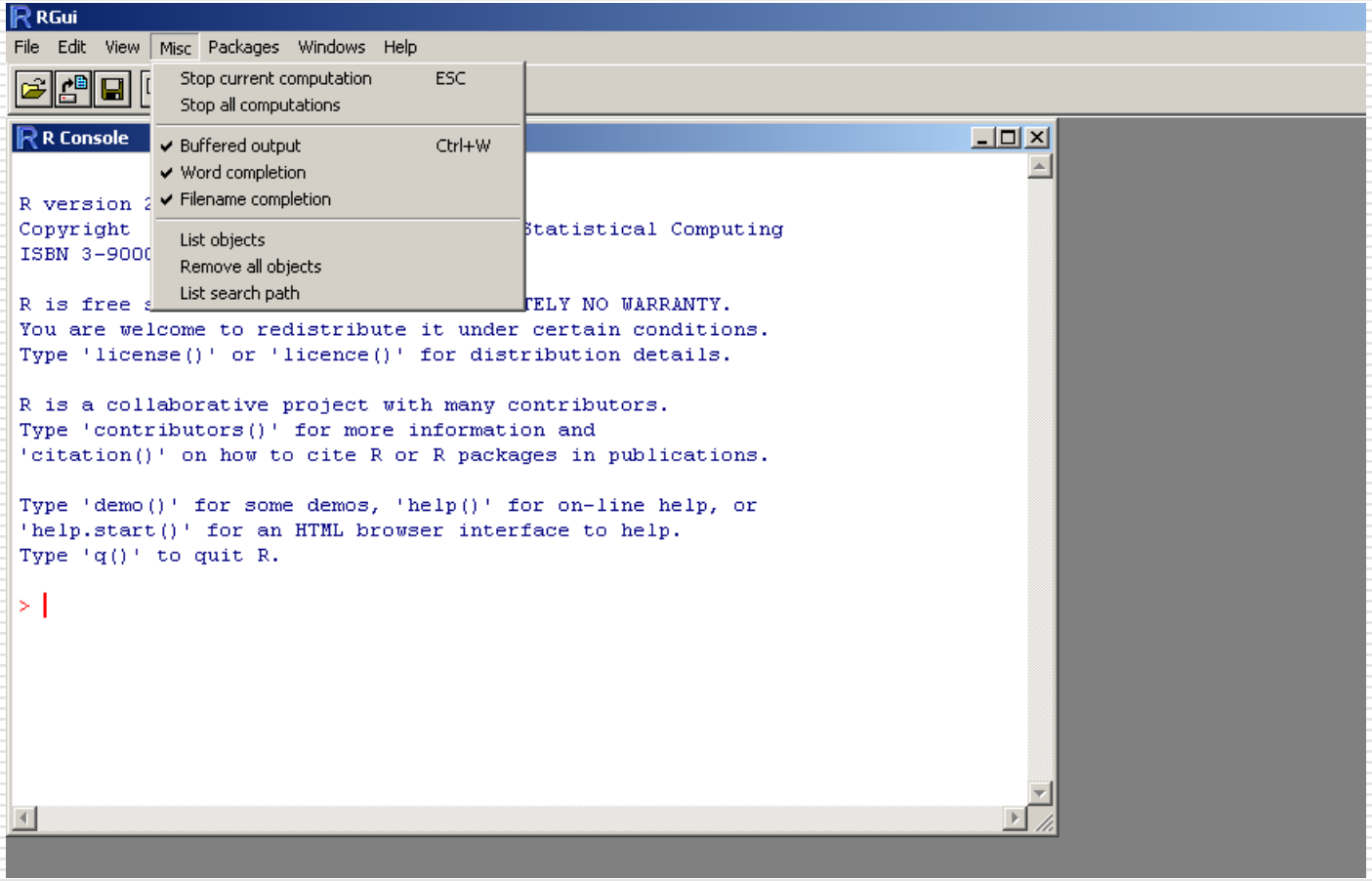
Το μενού view



Το μενού view

- Μπορείτε αν θέλετε να διαγράψετε το toolbar (με όλα τα μενού) από το περιβάλλον εργασίας όπως επίσης και το statusbar (η μπάρα στο κάτω μέρος με πληροφορίες για την έκδοση του προγράμματος που τρέχετε).

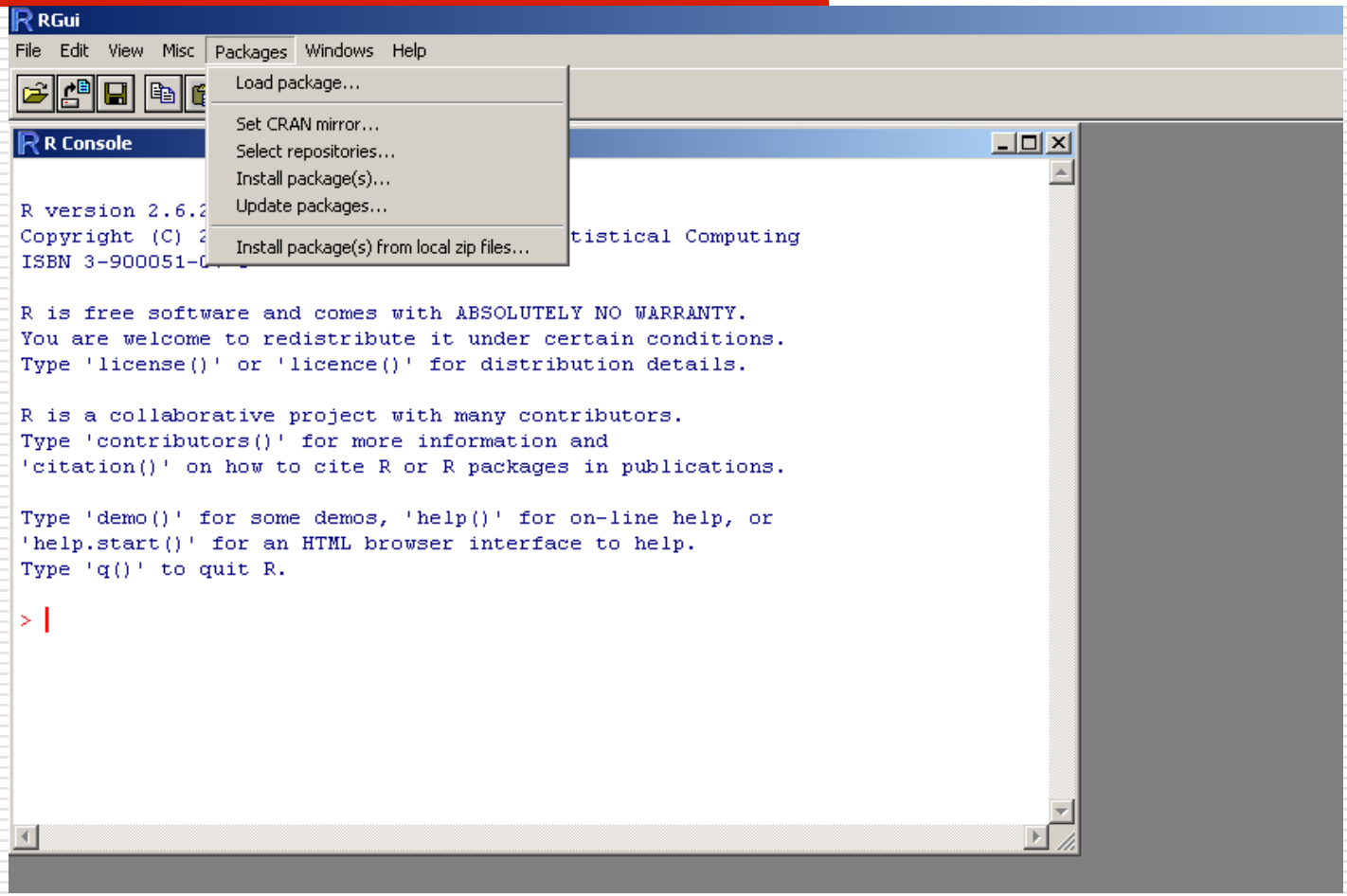
Το μενού misc



Το μενού misc

- Από το μενού misc ο χρήστης μπορεί να σταματήσει το τρέχον ή όλα τα προγράμματα που εκτελούνται (stop current/all computations), να σταματήσει την εκτύπωση αποτελεσμάτων στην οθόνη (buffered output), να δει όλα τα αντικείμενα που έχει δημιουργήσει έως τώρα (list objects) - ισοδύναμα μπορεί να χρησιμοποιήσει την εντολή ls() ή objects(), να διαγράψει όσα αντικείμενα έχει δημιουργήσει έως τώρα (remove all objects) - ισοδύναμα μπορεί να χρησιμοποιήσει την εντολή rm(list=ls(all=TRUE)) και τέλος να δει ποιες βιβλιοθήκες (libraries) και πλαίσια δεδομένων (data frames) επισυνάπτονται στο τρέχον περιβάλλον εργασίας του.

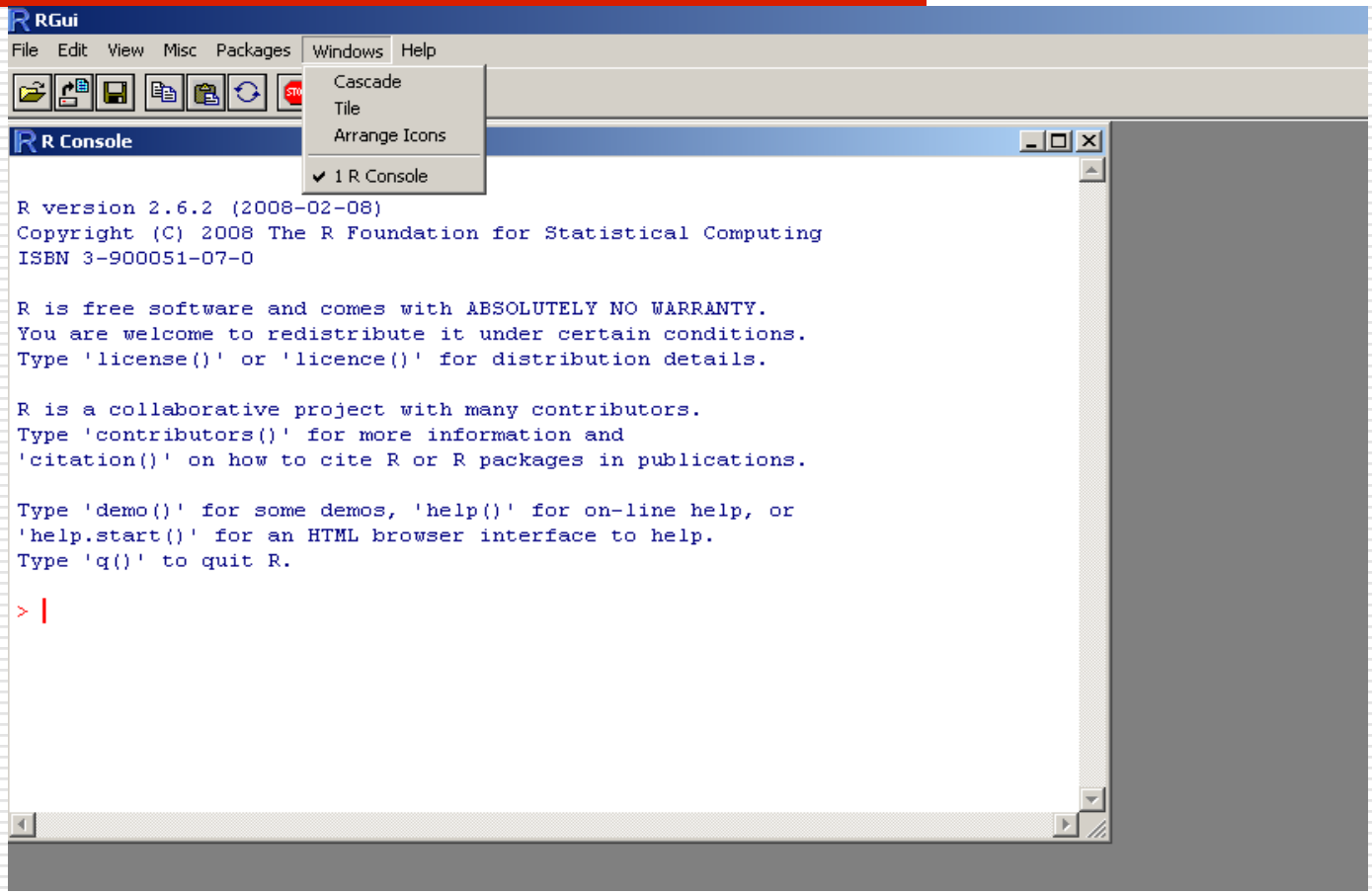
Το μενού packages



Το μενού packages

- Από το μενού packages ο χρήστης μπορεί να φορτώσει βιβλιοθήκες που ήδη έχει κατεβάσει (load package), να κατεβάσει βιβλιοθήκες από διάφορα πρότυπα του CRAN (install package (s)) ή από zip αρχεία του σκληρού του δίσκου (install package(s) from local zip files), να ενημερώσει τις βιβλιοθήκες προσθέτοντας νέες (update packages), να διαλέξει από ποιο μέρος του κόσμου θα κατεβάσει μέσω του CRAN τις βιβλιοθήκες (set CRAN mirror) ή να διαλέξει από ποιον διανομέα (πέραν του CRAN) θέλει να κατεβάσει τις βιβλιοθήκες (set repositories).

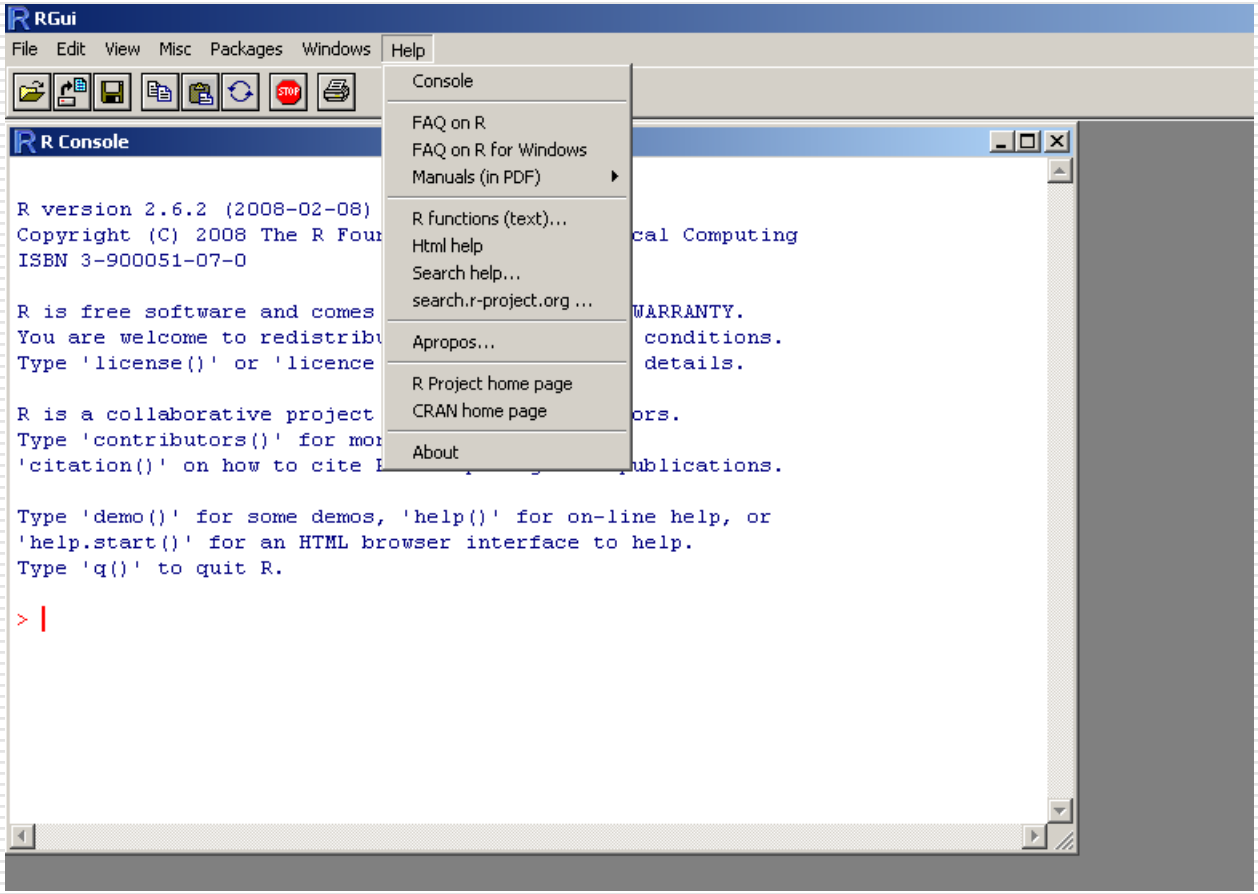
Το μενού windows



Το μενού windows

- Με το μενού windows ο χρήστης μπορεί να μετακινηθεί μεταξύ των ανοιχτών παραθύρων και να τα διατάξει με τον τρόπο που επιθυμεί.

Το μενού help



Το μενού help

- Με το μενού help δίνεται στον χρήστη ένα εγχειρίδιο για όλες τις εντολές και ιδιότητες του πακέτου. Πιο συγκεκριμένα:
  - Console: Πληροφορίες για το πως ο χρήστης μπορεί να χειριστεί την βασική οθόνη του προγράμματος.
  - FAQ on R και FAQ on R for Windows: Απαντήσεις σε συνήθεις ερωτήσεις για την R και για την R για windows.
  - Manuals (in pdf): Βασικό εγχειρίδιο της R σε μορφή pdf.
  - R functions (text): Πληροφορίες για τις εντολές της R που είναι ήδη φορτωμένες (από το βασικό πακέτο ή από τις ήδη φορτωμένες βιβλιοθήκες).

Το μενού help

- Html help: Διαδικτυακός χώρος με πληροφορίες για την R.
- Search help: Αρχεία που σχετίζονται άμεσα ή έμμεσα με την λέξη που αναζητείται από όλες τις διαθέσιμες βιβλιοθήκες.
- Search.r-project.org: Σύνδεσμοι στο διαδίκτυο που σχετίζονται άμεσα ή έμμεσα με την λέξη που αναζητείται.
- Apropos: Εντολές που είναι ήδη φορτωμένες και σχετίζονται άμεσα ή έμμεσα με την λέξη που αναζητείται.
- R project home page: Μεταφέρεσαι στην ιστοσελίδα της R.
- CRAN home page: Μεταφέρεσαι στην ιστοσελίδα της CRAN.
- About: Πληροφορία για την έκδοση και τα δικαιώματα του πακέτου.

Αριθμητικοί Τελεστές της R

Σύμβολο	Πράξη
+	Πρόσθεση
-	Αφαίρεση
*	Πολλαπλασιασμός
/	Διαίρεση
^	Ύψωση σε δύναμη
% / %	Ακέραια Διαίρεση
% %	Υπόλοιπο Διαίρεσης

Αριθμητικοί Τελεστές της R

```
> 3+3  # this is my first command
[1] 6
> 9-2
[1] 7
> 17/3
[1] 5.666667
> 4*2
[1] 8
> 2^3
[1] 8
> 17%/%3
[1] 5
> 17%%3
[1] 2
> 7/0
[1] Inf
```

- Οτιδήποτε ακολουθεί μετά τον χαρακτήρα # αγνοείται.
- Το Inf δηλώνει άπειρο και μπορεί να χρησιμοποιηθεί, π.χ.

```
> exp(-Inf)
[1] 0
```

$\lim_{x \to \infty} e^{-x}$

- Το NaN δηλώνει ότι η πράξη δεν μπορεί να γίνει, π.χ.

```
> log(-2)
[1] NaN
 Warning message:
 In log(-2) : NaNs  produced
```

Τελεστές Εκχώρησης και Σύγκρισης της R

Τελεστής	Ερμηνεία
<-	Εκχωρεί το αποτέλεσμα στο αριστερό μέλος της σχέσης
->	Εκχωρεί το αποτέλεσμα στο δεξί μέλος της σχέσης
>	Μεγαλύτερο από
<	Μικρότερο από
>=	Μεγαλύτερο ή ίσο από
<=	Μικρότερο ή ίσο από
==	Ίσο με
!=	Όχι ίσο με

Τελεστές Εκχώρησης και Σύγκρισης της R

```
> x<-56
> x
[1] 56
> 5*2->y
> y
[1] 10
> x>y
[1] TRUE
> y<4
[1] FALSE
```

```
> x>=56
[1] TRUE
> y<=9
[1] FALSE
> x==56
[1] TRUE
> y!=1
[1] TRUE
```

Περισσότερα για τους Τελεστές Εκχώρησης

- Η R είναι ευαίσθητη στα κεφαλαία γράμματα, δηλαδή το x και το X είναι διαφορετικά αντικείμενα.
- Στην εντολή εκχώρησης το αποτέλεσμα δεν εμφανίζεται στην οθόνη αλλά καταχωρείται στο αντικείμενο. Για να δούμε την τιμή του αντικειμένου πληκτρολογούμε το όνομά του.
- Τα αντικείμενα μπορεί να είναι επίσης διανύσματα, πίνακες, πλαίσια δεδομένων ή λίστες (περισσότερα για τις δομές δεδομένων της R αργότερα).
- Ένα αντικείμενο μπορεί να εμφανιστεί και στα δύο μέρη ενός τελεστή εκχώρησης, αρκεί πριν να το έχουμε ορίσει. Π.χ.

```
> x<-5
> x<-x+3
> x
[1] 8
```

- Σε περίπτωση που το όνομα του αντικειμένου στο οποίο καταχωρούμε μια τιμή υπάρχει, η τιμή του αντικειμένου θα αντικατασταθεί με την καινούργια.

Περισσότερα για τους Τελεστές Σύγκρισης

- Οι τελεστές σύγκρισης ελέγχουν αν ισχύει μια σχέση (π.χ. >) και επιστρέφουν την τιμή TRUE αν ισχύει η FALSE αν δεν ισχύει. Αν θέλουμε να συνδυάσουμε 2 ή περισσότερες συγκρίσεις χρησιμοποιούμε το σύμβολο & (και) για να ισχύουν όλες ή το σύμβολο | (ή) για να ισχύει τουλάχιστον μία. Π.χ.

```
> (5>3) & (8>10)
[1] FALSE
> (5>3) | (8>10)
[1] TRUE
```

- Οι τελεστές σύγκρισης χρησιμοποιούνται επίσης όπως θα δούμε και στις βασικές δομημένες εντολές (*βρόγχοι*), π.χ if, for, κλπ.
- Ο τελεστής ! δηλώνει το αντίθετο της έκφρασης που ακολουθεί. Π.χ.

```
> !(5>3)
[1] FALSE
```

Βασικές Αριθμητικές Συναρτήσεις της R

Συνάρτηση	Πράξη	Συνάρτηση	Πράξη
sqrt()	τετραγωνική ρίζα	asin()	τόξο ημιτόνου
abs()	απόλυτη τιμή	atan()	τόξο εφαπτομένης
log()	φυσικός λογάριθμος	gamma()	συνάρτηση Γάμμα
log2()	λογάριθμος με βάση 2	lgamma()	φυσικός λογάριθμος της συνάρτησης Γάμμα
log10()	λογάριθμος με βάση 10	beta()	συνάρτηση Βήτα
exp()	εκθετική συνάρτηση	floor()	προηγούμενος ακέραιος
cos()	συνημίτονο	ceiling()	επόμενος ακέραιος
sin()	ημίτονο	factorial()	παραγοντικό
tan()	εφαπτομένη	choose()	συνδυασμοί
acos()	τόξο συνημιτόνου	lchoose()	φυσικός λογάριθμος συνδυασμών

Βασικές Αριθμητικές Συναρτήσεις της R

```
> sqrt(16)
[1] 4
> abs(-2)
[1] 2
> log(10)
[1] 2.302585
> log2(10)
[1] 3.321928
> log10(10)
[1] 1
> exp(3)
[1] 20.08554
> cos(pi)
[1] -1
> sin(2*pi)
[1] -2.449213e-16
> tan(pi/2)
[1] 1.633178e+16
```

```
> sin(pi/2)
[1] 1
> tan(0)
[1] 0
> acos(0.2)
[1] 1.369438
> atan(2)
[1] 1.107149
> asin(0)
[1] 0
> gamma(2)
[1] 1
> beta(2,3)
[1] 0.08333333
```

```
> lgamma(4)
[1] 1.791759
> floor(4.9)
[1] 4
> ceiling(4.1)
[1] 5
> factorial(5)
[1] 120
> choose(5,2)
[1] 10
> lchoose(5,2)
[1] 2.302585
```

Βασικοί Τύποι Αντικειμένων

- Κάθε αντικείμενο στην R μπορεί να είναι:
  - Πραγματικός Αριθμός

    ```
    > x<- 3
    ```
  - Μιγαδικός Αριθμός

    ```
    > x<-complex(real=4, imaginary=3)
    > x
    [1] 4 +3i
    ```
  - Δεδομένο Λογικής

    ```
    > x <-3
    > y <- x > 4
    > y
    [1] FALSE
    ```
  - Δεδομένα Χαρακτήρων

    ```
    > x <- 'DIMITRIS'
    > x
    [1] "DIMITRIS"
    ```
- Με την εντολή mode() μπορούμε να δούμε ποιος είναι ο τύπος ενός αντικειμένου.

Βασικές Δομές Αντικειμένων

- Οι κύριες δομές των αντικειμένων στην R είναι:
  - Διανύσματα (vectors).
  - Διδιάστατοι Πίνακες (matrices).
  - Πολυδιάστατοι Πίνακες (arrays).
  - Πλαίσια Δεδομένων (data frames).
  - Λίστες (lists).
- Στις 3 πρώτες δομές τα αντικείμενα πρέπει να είναι του ίδιου τύπου.
- Οι λίστες μπορούν να περιέχουν ως στοιχεία άλλες δομές αντικειμένων.

Διανύσματα

- Τα διανύσματα στην R είναι σύνολα που περιέχουν αντικείμενα του ίδιου τύπου. Ανάλογα με το είδος των αντικειμένων έχουμε
  - Αριθμητικά Διανύσματα.
  - Διανύσματα Χαρακτήρων.
  - Λογικά Διανύσματα.
  - Διανύσματα Κατηγοριών.

Αριθμητικά Διανύσματα

- Ο πιο εύκολος τρόπος δημιουργίας ενός αριθμητικού διανύσματος είναι μέσω της εντολής c(). Τα στοιχεία μέσα στην εντολή διαχωρίζονται με κόμμα. Π.χ.

```
> x<-c(1,2,3,4,5)
> x
[1] 1 2 3 4 5
```

- Επίσης μπορούμε μέσα στην εντολή c να έχουμε ένα ήδη ορισμένο διάνυσμα. Π.χ.

```
> x<-c(1,2,3,4,5)
> y<-c(6,7)
> z<-c(10,x,y)
> z
[1] 10  1  2  3  4  5  6  7
```

Αριθμητικά Διανύσματα

- Χρήσιμες συναρτήσεις για αριθμητικά διανύσματα
  - **Μήκος Διανύσματος**:

    ```
    > x<-c(1,2,3,4,5)
    > length(x)
    [1] 5
    ```
  - **Ελάχιστη/Μέγιστη τιμή**

    ```
    > min(x)
    [1] 1
    > max(x)
    [1] 5
    ```
  - **Άθροισμα/Γινόμενο τιμών**

    ```
    > sum(x)
    [1] 15
    > prod(x)
    [1] 120
    ```

Αριθμητικά Διανύσματα

- **Ταξινόμηση τιμών διανύσματος κατ' αύξουσα/φθίνουσα τάξη μεγέθους**

```
> x<-c(3,5,2,1,6)
> sort(x)
[1] 1 2 3 5 6
> sort(x,decreasing=T)
[1] 6 5 3 2 1
```

- **Σειρά κατάταξης τιμών**

```
> x<-c(3,5,2,1,6)
> rank(x)
[1] 3 4 2 1 5
```

Αριθμητικά Διανύσματα

- **Σειρά κατάταξης τιμών σε περιπτώσεις ισοπαλιών**
  - *Average*: Σε περιπτώσεις ισοπαλιών η τελική σειρά κατάταξης προκύπτει από το μέσο όρο των σειρών κατάταξης των παρατηρήσεων με τις ίδιες τιμές, π.χ.

    ```
    > x2<-c(3,5,2,1,6,3)
    > rank(x2, ties.method="average")
    [1] 3.5 5.0 2.0 1.0 6.0 3.5
    ```
  - *First*: Η πρώτη παρατήρηση σε σειρά εμφάνισης παίρνει την χαμηλότερη σειρά κατάταξης, π.χ.

    ```
    > rank(x2, ties.method="first")
    [1] 3 5 2 1 6 4
    ```
  - *Random*: Τυχαία προκύπτει η σειρών κατάταξης των παρατηρήσεων με τις ίδιες τιμές, π.χ.

    ```
    > rank(x, ties.method="random")
    [1] 4 5 2 1 6 3
    > rank(x, ties.method="random")
    [1] 3 5 2 1 6 4
    ```
  - *Min/Max*: Δίνεται η μικρότερη/μεγαλύτερη σειρά κατάταξης στις παρατηρήσεις με τις ίδιες τιμές, π.χ.

    ```
    > rank(x, ties.method="min")
    [1] 3 5 2 1 6 3
    > rank(x, ties.method="max")
    [1] 4 5 2 1 6 4
    ```

Αριθμητικά Διανύσματα

- **Θέση των ταξινομημένων κατ' αύξουσα τάξη μεγέθους τιμών**

```
> x<-c(3,5,2,1,6)
> order(x)
[1] 4 3 1 2 5
```

(δηλαδή η μικρότερη παρατήρηση βρίσκεται στην 4 θέση, η αμέσως μεγαλύτερη στην 3, κ.λ.π.).

Αριθμητικά Διανύσματα

- **Ορισμός ονομάτων στις τιμές του διανύσματος**

```
> weight<-c(70, 57, 68, 82)
> names(weight)
NULL
> names(weight)<-c("Mary", "Kelly", "Elena",
    "George")
> names(weight)
[1] "Mary"   "Kelly"  "Elena"  "George"
> weight
Mary  Kelly  Elena George
  70     57     68     82
```

Αριθμητικά Διανύσματα

- **Απούσες τιμές**.

```
> x3<-c(1,2,3,NA,9)
> x3
[1]  1  2  3 NA  9
> is.na(x3)
[1] FALSE FALSE FALSE  TRUE FALSE
```

Αριθμητικά Διανύσματα

- **Δημιουργία Αριθμητικών Ακολουθιών**
  - Με την εντολή α:β δημιουργούμε ακολουθίες τιμών από το α στο β με βήμα την μονάδα. Π.χ.

```
> x<-1:10
> x
[1]  1  2  3  4  5  6  7  8  9 10
> x<--4:10
> x
[1] -4 -3 -2 -1  0  1  2  3  4  5  6  7  8  9 10
> x<-6:1
> x
[1] 6 5 4 3 2 1
> x<-3.3:10.3
> x
[1]  3.3  4.3  5.3  6.3  7.3  8.3  9.3 10.3
```

  - Αν η διαφορά των α, β δεν είναι ακέραιος αριθμός, τότε η R δημιουργεί πάλι ακολουθία με βήμα την μονάδα, ξεκινώντας από το α και σταματώντας πριν το β. Π.χ.

```
> x<-3.3:6.9
> x
[1] 3.3 4.3 5.3 6.3
```

Αριθμητικά Διανύσματα

- Αν το βήμα δεν θέλετε να είναι η μονάδα, τότε μπορείτε να χρησιμοποιήσετε την εντολή seq. Ως παραμέτρους η εν λόγω συνάρτηση παίρνει τον πρώτο όρο (from) και τελευταίο όρο (to) της ακολουθίας, το βήμα της ακολουθίας (by), το μήκος της ακολουθίας (length) και το όνομα ενός άλλου διανύσματος (along) έτσι ώστε η ακολουθία να έχει ίδιο μήκος με αυτό το διάνυσμα. Η εν λόγω συνάρτηση χρειάζεται 3 από τις παραπάνω αυτές παραμέτρους, ενώ αν δοθούν μόνο 2 η R θεωρεί την παράμετρο by = 1.

```
> seq(from=1,to=9, by=2)
[1] 1 3 5 7 9
> seq(from=1,to=9, length=3)
[1] 1 5 9
> seq(to=9, length=3)
[1] 7 8 9
> seq(from=1,by=2,length=10)
[1]  1  3  5  7  9 11 13 15 17 19
> y<-1:10
> seq(from=1,by=2,along=y)
[1]  1  3  5  7  9 11 13 15 17 19
```

- Αν το πηλίκο της διαφοράς του τελευταίου από τον πρώτο όρο προς το βήμα δεν είναι ακέραιος αριθμός η R θα σταματήσει πριν τον τελευταίο όρο.

```
> seq(from=1,to=10,by=2)
[1] 1 3 5 7 9
```

Αριθμητικά Διανύσματα

- **Επαναλήψεις τιμών ή διανυσμάτων**. Με την εντολή rep μπορούμε να επαναλάβουμε μια τιμή ή ένα διάνυσμα όσες φορές θέλουμε. Ως παραμέτρους δέχεται πρώτα την τιμή ή το διάνυσμα που θέλουμε να επαναλάβουμε και εν συνεχεία τον αριθμό επαναλήψεων της τιμής ή του διανύσματος (times) ή τον αριθμό επαναλήψεων κάθε στοιχείου του διανύσματος.

```
> rep(2,5)
[1] 2 2 2 2 2
> x<-c(1,2,3)
> rep(x,5)
[1] 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3
> rep(x,each=5)
 [1] 1 1 1 1 1 2 2 2 2 2 3 3 3 3 3
```

Αριθμητικά Διανύσματα

- **Πράξεις Διανυσμάτων**. Μπορούμε να κάνουμε πράξεις μεταξύ διανυσμάτων (προσοχή να είναι ίδιας διάστασης), μεταξύ αριθμών και διανυσμάτων, όπως και να εφαρμόσουμε αριθμητικές συναρτήσεις σε διανύσματα. Π.χ.

```
> x<-c(1,2,3)
> x*3
[1] 3 6 9
> x^2
[1] 1 4 9
> y<-c(4,5,6)
> y/x
[1] 4.0 2.5 2.0
```

Αριθμητικά Διανύσματα

- Μπορούμε εύκολα να επιλέξουμε **συγκεκριμένα στοιχεία** ενός διανύσματος. Αν π.χ. το διάνυσμα είναι το x και θέλουμε το πρώτο στοιχείο του, τότε το καλούμε με x[1].

```
> x<-seq(from=1,to=9,by=2)
> x
[1] 1 3 5 7 9
> x[2]
[1] 3
> x[2:4]
[1] 3 5 7
> x[c(1,3)]
[1] 1 5
> x[-c(1,3)]
[1] 3 7 9
```

Διανύσματα Χαρακτήρων

```
> x<-c('Dimitris', 'Giorgos')
> x
[1] "Dimitris" "Giorgos"
```

- **Μερικές Χρήσιμες συναρτήσεις** (δείτε το help για λεπτομέρειες):
  - character(length)
  - as.character(x)
  - paste
  - strsplit
  - substr
  - substring
  - grep
  - regexpr
  - sub
  - gsub
  - tolower
  - toupper

Λογικά Διανύσματα

- **Βασικές συναρτήσεις**

  > logical(3) (δημιουργεί διάνυσμα με ψευδείς τιμές)

  [1] FALSE FALSE FALSE

  > as.logical(c(0:10)) (μετατρέπει διανύσματα σε λογικά. Η τιμή 0 μετατρέπεται σε False και όλες οι άλλες σε True.)

  [1] FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

Διανύσματα Κατηγοριών

- Κατηγορικές Μεταβλητές δίνονται στην R με την βοήθεια της εντολής factor.

```
> gender<-c('Male', 'Female', 'Male', 'Male', 'Female')
> gender
[1] "Male"   "Female" "Male"   "Male"   "Female"
> factor(gender)
[1] Male   Female Male   Male   Female
Levels: Female Male
> levels(factor(gender))
[1] "Female" "Male"
```

- Αν η μεταβλητή είναι διάταξης τότε χρησιμοποιούμε την εντολή ordered.

```
> opinion<-c('Low', 'Low', 'High', 'High', 'High', 'Medium')
> ordered(opinion, levels=c('Low', 'Medium', 'High'))
[1] Low    Low    High   High   High   Medium
Levels: Low < Medium < High
```

Διδιάστατοι Πίνακες

- Ένας **διδιάστατος πίνακας** (matrix) είναι μια δομή δεδομένων της οποίας τα στοιχεία είναι διατεταγμένα σε γραμμές και στήλες. Για να τους δημιουργήσουμε χρησιμοποιούμε την εντολή matrix με παραμέτρους τα στοιχεία (μπορεί να είναι αριθμοί ή χαρακτήρες) και τον αριθμό γραμμών (nrow) ή στηλών (ncol). Επίσης δηλώνουμε αν θέλουμε τα στοιχεία να διαβαστούν κατά στήλη (προκαθορισμένη τιμή) ή κατά γραμμή (byrow=T).

Διδιάστατοι Πίνακες

```
> x<-1:10
> X<-matrix(x, ncol=2)
> X
     [,1] [,2]
[1,]    1    6
[2,]    2    7
[3,]    3    8
[4,]    4    9
[5,]    5   10
> X<-matrix(x, nrow=5)
> X
     [,1] [,2]
[1,]    1    6
[2,]    2    7
[3,]    3    8
[4,]    4    9
[5,]    5   10
> X<-matrix(x, nrow=5, byrow=T)
> X
     [,1] [,2]
[1,]    1    2
[2,]    3    4
[3,]    5    6
[4,]    7    8
[5,]    9   10
```

Διδιάστατοι Πίνακες

- Η διάσταση του πίνακα δίνεται με την εντολή dim

```
> dim(X)
[1] 5 2
```

- Μπορούμε εύκολα να δούμε κάποιο ή κάποια στοιχεία ενός πίνακα απλά δίνοντας την θέση του μέσα σε [].

```
> X[3,2]
[1] 6
```

- Επίσης μπορούμε να δούμε μια γραμμή ή μια στήλη του πίνακα, παραλείποντας την διάσταση για την οποία δεν ενδιαφερόμαστε

```
> X[3,]
[1] 5 6
> X[,2]
[1]  2  4  6  8 10
```

Διδιάστατοι Πίνακες

- Με τις εντολές rbind και cbind δημιουργούμε πίνακες συνενώνοντας ως στήλες ή ως γραμμές αντίστοιχα διανύσματα.

```
> x1<-1:5
> x2<-6:10
> cbind(x1,x2)
     x1 x2
[1,]  1  6
[2,]  2  7
[3,]  3  8
[4,]  4  9
[5,]  5 10
> rbind(x1,x2)
     [,1] [,2] [,3] [,4] [,5]
x1    1    2    3    4    5
x2    6    7    8    9   10
```

Διδιάστατοι Πίνακες

- Μπορούμε να δημιουργούμε διαγώνιους πίνακες με την εντολή diag

```
> diag(1:5)
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    0    0    0    0
[2,]    0    2    0    0    0
[3,]    0    0    3    0    0
[4,]    0    0    0    4    0
[5,]    0    0    0    0    5
```

Διδιάστατοι Πίνακες

- Για την δημιουργία ενός ταυτοτικού πίνακα χρησιμοποιούμε πάλι την εντολή diag

```
> diag(5)
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    0    0    0    0
[2,]    0    1    0    0    0
[3,]    0    0    1    0    0
[4,]    0    0    0    1    0
[5,]    0    0    0    0    1
```

Διδιάστατοι Πίνακες

- Πράξεις Πινάκων. Μπορούμε να χρησιμοποιήσουμε τους αριθμητικούς τελεστές και τις μαθηματικές συναρτήσεις της R για πράξεις μεταξύ πινάκων ή μεταξύ πινάκων και διανυσμάτων ή μεταξύ πινάκων και αριθμών. Οι μόνοι νέοι τελεστές, ειδικά για πίνακες είναι οι ακόλουθοι:

Διδιάστατοι Πίνακες

Σύμβολο	**Πράξη**
%*%	Πολλαπλασιασμός Πινάκων
t()	Ανάστροφος Πίνακα
solve()	Αντίστροφος Πίνακα

Διδιάστατοι Πίνακες

```
> x<-matrix(c(1,2,3,4,5,6), ncol=2)
> x
     [,1] [,2]
[1,]    1    4
[2,]    2    5
[3,]    3    6
> dim(x)
[1] 3 2
> y<-matrix(c(0,1,1,1), ncol=2)
> y
     [,1] [,2]
[1,]    0    1
[2,]    1    1
```

```
> x%*%y
     [,1] [,2]
[1,]    4    5
[2,]    5    7
[3,]    6    9
> t(x)
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
> solve(y)
     [,1] [,2]
[1,]   -1    1
[2,]    1    0
```

Διδιάστατοι Πίνακες

- Η εντολή apply.

```
> x<-matrix(c(1,2,3,4,5,6), ncol=2)
> x
     [,1] [,2]
[1,]    1    4
[2,]    2    5
[3,]    3    6
> apply(x,1,sum)
[1] 5 7 9
> apply(x,2,sum)
[1]  6 15
```

Άθροισε τα στοιχεία του πίνακα ως προς όλες τις γραμμές.

Άθροισε τα στοιχεία του πίνακα ως προς όλες τις στήλες.

Ιδιοτιμές-Ιδιοδιανύσματα πίνακα

```
> X<-matrix(c(10,-2,-2,-2,1,7,-2,7,1), ncol=3)
> y<-eigen(X, sym=T)
>y
$values
[1] 12  6 -6
$vectors
           [,1]       [,2]       [,3]
[1,]  0.8164966 -0.5773503  0.0000000
[2,] -0.4082483 -0.5773503 -0.7071068
[3,] -0.4082483 -0.5773503  0.7071068
```

Πολυδιάστατοι Πίνακες

- Οι **πολυδιάστατοι πίνακες** (arrays) είναι πίνακες με 3 ή περισσότερες διαστάσεις. Δημιουργούνται με την εντολή array και το μέγεθος κάθε διάστασης δηλώνεται από την παράμετρο dim. Π.χ. αν θέσουμε dim=c (2,3,4), θα έχουμε έναν 3-διάστατο πίνακα με μέγεθος διαστάσεων 2 (nrow), 3 (ncol) και 4.

Πολυδιάστατοι Πίνακες

```
> X<-array(c(1:12,36:48),dim=c(2,3,4))
> X
, , 1

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

, , 2

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

, , 3

     [,1] [,2] [,3]
[1,]   36   38   40
[2,]   37   39   41

, , 4

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

Πλαίσια Δεδομένων

- Τα **πλαίσια δεδομένων** (data frames) είναι διδιάστατοι πίνακες στους οποίους δεν χρειάζεται οι στήλες να είναι όλες του ίδιου τύπου. Συνήθως στα πλαίσια δεδομένων κατοχυρώνουμε τις παρατηρήσεις που έχουμε συλλέξει από ένα δείγμα.
- Ένα πλαίσιο δεδομένων δημιουργείται με την εντολή data.frame().

Πλαίσια Δεδομένων

```
> Gender<-c('Male', 'Male', 'Male', 'Female')
> Gender<-factor(Gender)
> Gender
[1] Male   Male   Male   Female
Levels: Female Male
> Smoking<-c(T, T, F, F)
> Smoking
[1]  TRUE  TRUE FALSE FALSE
> Choresterol<-c(200, 220, 180, 172)
> Choresterol
[1] 200 220 180 172
> sample<-data.frame(Gender, Smoking, Choresterol)
> sample
  Gender Smoking Choresterol
1   Male    TRUE         200
2   Male    TRUE         220
3   Male   FALSE         180
4 Female   FALSE         172
```

Πλαίσια Δεδομένων

- Με την εντολή as.data.frame μπορείτε να μετατρέψετε έναν διδιάστατο πίνακα σε πλαίσιο δεδομένων. Με την εντολή names μπορείτε να δώσετε ονόματα στις στήλες (μεταβλητές) του πλαισίου σας. Επίσης με την παράμετρο row.names της εντολής data.frame μπορείτε να δώσετε ονόματα και στις γραμμές.

Πλαίσια Δεδομένων

```
> x<-matrix(c(1,1,200,1,1,220, 1,0,180,0,0,172),
      ncol=3, byrow=T)
> x
     [,1] [,2] [,3]
[1,]    1    1  200
[2,]    1    1  220
[3,]    1    0  180
[4,]    0    0  172
> x<-as.data.frame(x)
> x
  V1 V2  V3
1  1  1 200
2  1  1 220
3  1  0 180
4  0  0 172
> names(x)
[1] "V1" "V2" "V3"
> names(x)<-c('Gender', 'Smoking', 'Choresterol')
> x
  Gender Smoking Choresterol
1      1       1         200
2      1       1         220
3      1       0         180
4      0       0         172

> x<-data.frame(x, row.names=c('obs1',
      'obs2', 'obs3', 'obs4') )
> x
     Gender Smoking Choresterol
obs1      1       1         200
obs2      1       1         220
obs3      1       0         180
obs4      0       0         172
```

Πλαίσια Δεδομένων

- Ό,τι εντολές χρησιμοποιήσαμε στους πίνακες μπορούμε να χρησιμοποιήσουμε και εδώ.

```
> x
     Gender Smoking Choresterol
obs1      1       1         200
obs2      1       1         220
obs3      1       0         180
obs4      0       0         172
> dim(x)
[1] 4 3
> x[1,]
      Gender Smoking Choresterol
obs1      1       1         200
```

```
> x[1,2]
[1] 1
> x$Gender
[1] 1 1 1 0
> rbind(1,x)
     Gender Smoking Choresterol
1         1       1         1
obs1      1       1       200
obs2      1       1       220
obs3      1       0       180
obs4      0       0       172
> cbind(1,x)
       1 Gender Smoking
     Choresterol
obs1 1      1       1       200
obs2 1      1       1       220
obs3 1      1       0       180
obs4 1      0       0       172
```

Λίστες

- Οι **λίστες** (lists) είναι διανύσματα των οποίων τα στοιχεία δεν είναι ανάγκη να είναι της ίδιας δομής. Δημιουργείται με την εντολή list δίνοντας ως παραμέτρους τα αντικείμενα που θέλουμε να την πλαισιώνουν μαζί με τα ονόματά τους.

Λίστες

```
> Gender<-c('Male', 'Male', 'Male',
      'Female')
> Gender<-factor(Gender)
> Gender
[1] Male   Male   Male   Female
Levels: Female Male
> x<-1:10
> x
 [1]  1  2  3  4  5  6  7  8  9 10
> sample
  Gender Smoking Choresterol
1   Male    TRUE         200
2   Male    TRUE         220
3   Male   FALSE         180
4 Female   FALSE         172
> y<-list(my_sample=sample, x=x,
      the_gender=Gender)
```

```
> y
$my_sample
  Gender Smoking Choresterol
1   Male    TRUE         200
2   Male    TRUE         220
3   Male   FALSE         180
4 Female   FALSE         172

$x
 [1]  1  2  3  4  5  6  7  8  9 10

$the_gender
[1] Male   Male   Male   Female
Levels: Female Male
```

Λίστες

- Με το σύμβολο $ ή την διπλή αγκύλη [[ ]], μπορούμε να δούμε τα επιμέρους στοιχεία μιας λίστας.

```
> y$x
[1]  1  2  3  4  5  6  7  8  9 10
> y[[3]]
[1] Male   Male   Male   Female
Levels: Female Male
> y$x[1:3]
[1] 1 2 3
```

Αποθήκευση Αντικειμένων

- Υπάρχουν αρκετοί τρόποι αποθήκευσης αντικειμένων της R στον σκληρό δίσκο. Αν δώσουμε μόνο το όνομα του αρχείου όπου θα γίνει η αποθήκευση, το εν λόγω αρχείο δημιουργείται στον φάκελο από όπου τρέχουμε την R. Αν θέλουμε η αποθήκευση να γίνει κάπου αλλού τότε πρέπει να δώσουμε την πλήρη διαδρομή. Ένας από τους ευκολότερους τρόπους να αποθηκεύσουμε αντικείμενα είναι με χρήση της εντολής write.

Αποθήκευση Αντικειμένων

- Διανύσματα. Μπορείτε να αποθηκεύσετε διανύσματα (αριθμητικά, χαρακτήρων ή λογικά) με την εντολή write.

```
> Gender<-c("Male", "Male", "Male", "Female")
> write(Gender,file="g.txt", ncol=4)
> x
[1]  1  2  3  4  5  6  7  8  9 10
> write(x,file="x.txt", ncol=length(x))
> Smoking
[1]  TRUE  TRUE FALSE FALSE
> write(Smoking,file="s.txt", ncol=4)
```

1 1 0 0

Αποθήκευση Αντικειμένων

- Διδιάστατοι Πίνακες. Με την εντολή write μπορείτε επίσης να αποθηκεύσετε πίνακες 2 διαστάσεων. Εδώ χρειάζεται **προσοχή** διότι η εντολή αναστρέφει τον πίνακα, οπότε ζητούμε αποθήκευση του ανάστροφου.

```
> X
     [,1] [,2]
[1,]    1    7
[2,]    2    8
[3,]    3    9
[4,]    4   10
[5,]    5   11
[6,]    6   12
> write(t(X), "X.txt", ncol=2)
```

Αποθήκευση Αντικειμένων

- Πλαίσια Δεδομένων. Τα πλαίσια δεδομένων τα αποθηκεύουμε με την εντολή write.table.

```
> sample
  Gender Smoking Choresterol
1   Male    TRUE         200
2   Male    TRUE         220
3   Male   FALSE         180
4 Female   FALSE         172
> write.table(sample, file="sample.txt")
```

Επανάκτηση Δεδομένων

- Μπορούμε εύκολα στην R να διαβάσουμε δεδομένα από ένα αρχείο του σκληρού μας δίσκου. Όπως και στην αποθήκευση έτσι και εδώ πρέπει να δώσουμε την πλήρη διαδρομή του αρχείου από όπου θέλουμε να ανακτήσουμε δεδομένα, εκτός και αν το αρχείο βρίσκεται στον φάκελο που δουλεύουμε την R οπότε το όνομά του αρκεί. Για διανύσματα και διδιάστατους πίνακες χρησιμοποιούμε την εντολή scan.

Επανάκτηση Δεδομένων

```
> x<-scan("x.txt")
Read 10 items
> x
[1]  1  2  3  4  5  6  7  8  9 10
> X<-matrix(scan("XX.txt"), ncol=2, byrow=T)
Read 12 items
> X
     [,1] [,2]
[1,]    1    2
[2,]    3    4
[3,]    5    6
[4,]    7    8
[5,]    9   10
[6,]   11   12
```

Αρχείο από όπου θα πάρει τις τιμές ο πίνακας

Να διαβάσει τις τιμές ανά γραμμή

στηλών του πίνακα

Επανάκτηση Δεδομένων

- Επίσης με την εντολή scan μπορούμε να εισάγουμε δεδομένα και με το πληκτρολόγιο.

```
> z<-scan()
1: 2
2: 4
3: 6
4: 8
5: 2
6:
Read 5 items
> z
[1] 2 4 6 8 2
```

Επανάκτηση Δεδομένων

- Για πλαίσια δεδομένων χρησιμοποιούμε την εντολή read.table.

```
> zz<-read.table("sample.txt")
> zz
     Gender Smoking Choresterol
1   Male    TRUE         200
2   Male    TRUE         220
3   Male   FALSE         180
4 Female   FALSE          172
```

Εισαγωγή Δεδομένων από άλλα Στατιστικά Πακέτα

- Με την βοήθεια του πακέτου foreign μπορούμε να διαβάσουμε δεδομένα από άλλα στατιστικά πακέτα.

```
> install.packages("foreign")
> library("foreign")
> library(help=foreign)
```

Εισαγωγή Δεδομένων από άλλα Στατιστικά Πακέτα

Πακέτο	Εντολή
SPSS	read.spss
S	data.restore, read.S
STATA	read.dta
SAS	read.xport
Epi Info	read.epiinfo
Minitab	read.mtb
Octave	read.octave

Συναρτήσεις στην R

```
> x
 [1]  46 104  94 114  35  70 120  29  19 135
   200 222  89 100  55 214  15  81 118 193
> range<-function(x){
  y<-max(x)-min(x)
  return(y)
  }
> range(x)
[1] 207
```

Συναρτήσεις στην R

```
> calc<-function(a,b=2){
    y<-a^b
    return(y)
   }
> calc(4)
[1] 16
> calc(4,3)
[1] 64
```

b=2 προκαθορισμένη τιμή, εκτός αν δοθεί αλλιώς.

a=4, b=3

Συναρτήσεις στην R

Εντολή	Ερμηνεία
if (A) B	Ελέγχει αν ισχύει το A. Αν ναι εκτελεί το B
if (A) B1 else B2	Ελέγχει αν ισχύει το A. Αν ναι εκτελεί το B1 αλλιώς το B2
ifelse(A, B1, B2)	Ίδιο με πριν
break	Τερματίζει τρέχοντα βρόγχο
next	Τερματίζει τρέχοντα βρόγχο και αρχίζει επόμενη επανάληψη
return(A)	Τερματίζει τρέχουσα συνάρτηση και επιστρέφει A
while(A) B	Ελέγχει κατά επανάληψη αν ισχύει το A. Αν ναι επιστρέφει B
repeat A	Όπως το while
for(index in A) B	Βρόγχος. Εκτελεί το B όσο το index ανήκει στο A

Συναρτήσεις στην R

- Παράδειγμα 1: Έστω ότι θέλουμε να κατασκευάσουμε μια συνάρτηση που να υπολογίζει το x! (x παραγοντικό, όπου x φυσικός αριθμός)

```
fact1<-function(x){
y<-floor(x)
if (y!=x | x<0)
print("Your number is not natural")
else
{
f<-1
if (x<2) return(f)
for (i in 2:x) {
f<-f*i
}
return(f)
}
}
```

```
> fact1(3)
[1] 6
> fact1(1)
[1] 1
> fact1(0)
[1] 1
> fact1(4)
[1] 24
> fact1(2.3)
[1] "Your number is not natural"
```

Συναρτήσεις στην R

```
fact2<-function(x){
    y<-floor(x)
    if (y!=x | x<0)
    print("Your number is not natural")
    else
    {
    f<-1
    t<-x
    while(t>1){
    f<-f*t
    t<-t-1
    }
    return(f)
    }
    }
```

```
> fact2(3)
[1] 6
> fact2(1)
[1] 1
> fact2(0)
[1] 1
> fact2(4)
[1] 24
> fact2(2.3)
[1] "Your number is not natural"
```

Συναρτήσεις στην R

```
fact3<-function(x){
      y<-floor(x)
      if (y!=x | x<0)
      print("Your number is not natural")
     else
      {
     f<-1
     t<-x
     repeat{
     if (t<2) break
     f<-f*t
     t<-t-1
      }
     return(f)
     }
     }
```

```
> fact3(3)
[1] 6
> fact3(1)
[1] 1
> fact3(0)
[1] 1
> fact3(4)
[1] 24
> fact3(2.3)
[1] "Your number is not natural"
```

Συναρτήσεις στην R

- Οι **βρόγχοι** στην R μπορεί να καθυστερήσουν αρκετά μια συνάρτηση και για αυτό τον λόγο καλό είναι να αποφεύγονται αν είναι δυνατόν. Οι βρόγχοι μπορούν να αποφευχθούν κάποιες φορές με χρήση λογικών συναρτήσεων για διανύσματα. Π.χ. ο βρόγχος

```
for(i in 1:length(y)) {if(y[i]<0} y[i]<-0}
```

μπορεί να αντικατασταθεί με την πολύ πιο γρήγορη εντολή

```
y[y<0]<-0
```

Συναρτήσεις στην R

- Καλό είναι επίσης να χρησιμοποιούμε τις έτοιμες συναρτήσεις της R όπου είναι εφικτό. Η συνάρτηση π.χ. cumprod(x) παίρνει ως όρισμα ένα αριθμητικό διάνυσμα και επιστρέφει το αθροιστικό γινόμενο. Π.χ.

```
> cumprod(c(1,2,4))
[1] 1 2 8
```

- Μπορούμε λοιπόν να χρησιμοποιήσουμε την συγκεκριμένη συνάρτηση για τον υπολογισμό του παραγοντικού.

```
fact4<-function(x){
    y<-floor(x)
    if (y!=x|x<0)
    print("Your number is not natural")
    else
    {
    return(max(cumprod(1:x)))
    }
    }
```

```
> fact4(3)
[1] 6
> fact4(1)
[1] 1
> fact4(0)
[1] 1
> fact4(4)
[1] 24
> fact4(2.3)
[1] "Your number is not natural"
```

Συναρτήσεις στην R

- Τέλος θα μπορούσαμε να είχαμε χρησιμοποιήσει την συνάρτηση Γάμμα, μιας και για φυσικούς αριθμούς x!=Γ(x+1) ή την έτοιμη συνάρτηση factorial.

```
> gamma(4)
[1] 6
> gamma(2)
[1] 1
> gamma(1)
[1] 1
> factorial(3)
[1] 6
> factorial(1)
[1] 1
> factorial(0)
[1] 1
```

- Οι συναρτήσεις gamma και factorial επιστρέφουν τιμές και για μη φυσικούς αριθμούς, χωρίς να ερμηνεύονται τότε ως παραγοντικά.

Συναρτήσεις στην R

- Προβλήματα υπερχείλισης: Η R εκτελεί τις πράξεις με την σειρά που αυτές ορίζονται χωρίς να προβαίνει σε απλοποιήσεις. Έτσι υπάρχει περίπτωση να αντιμετωπίσουμε προβλήματα **υπερχείλισης** (overflow). Π.χ.

```
> factorial(200)/(factorial(100)*factorial(100))
[1] NaN
Warning message:
In factorial(200) : value out of range in 'gammafn'
```

$\binom{200}{100}$

- Για να υπολογίσουμε την άνω ποσότητα λοιπόν είναι καλό να κάνουμε εμείς την απλοποίηση και να ζητήσουμε στην R να κάνει τις πράξεις εν συνεχεία.

Συναρτήσεις στην R

Προφανώς

$$\binom{200}{100} = \frac{101 \cdot \ldots \cdot 200}{1 \cdot \ldots \cdot 100}$$

Άρα στην R γράφουμε

```
> prod(101:200)/prod(1:100)
[1] 9.054851e+58
```

$\longrightarrow \quad 9.05 \cdot 10^{58}$

ή ακόμα καλύτερα

```
> x<-1:100
> y<-101:200
> z<-y/x
> prod(z)
[1] 9.054851e+58
```

Συναρτήσεις στην R

- Ένας ακόμα τρόπος να αποφύγουμε προβλήματα υπερχείλισης είναι με χρήση του λογαρίθμου (φυσικού) log.

$$\binom{200}{100} = \exp\left[\log\left\{\binom{200}{100}\right\}\right] = \exp\left[\log(200!) - 2\log(100!)\right]$$

Αλλά

$$\log(n!) = \sum_{i=1}^{n} \log(i)$$

```
> exp(sum(log(1:200))-2*sum(log(1:100)))
[1] 9.054851e+58
```