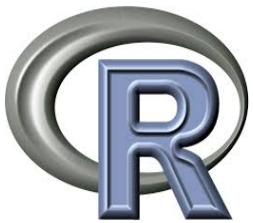
Εισαγωγή στη γλώσσα προγραμματισμού R

Αναστάσιος Κατσιλέρος

Γεωπονικό Πανεπιστήμιο Αθηνών
Εργαστήριο Βελτίωσης Φυτών και Γεωργικού Πειραματισμού

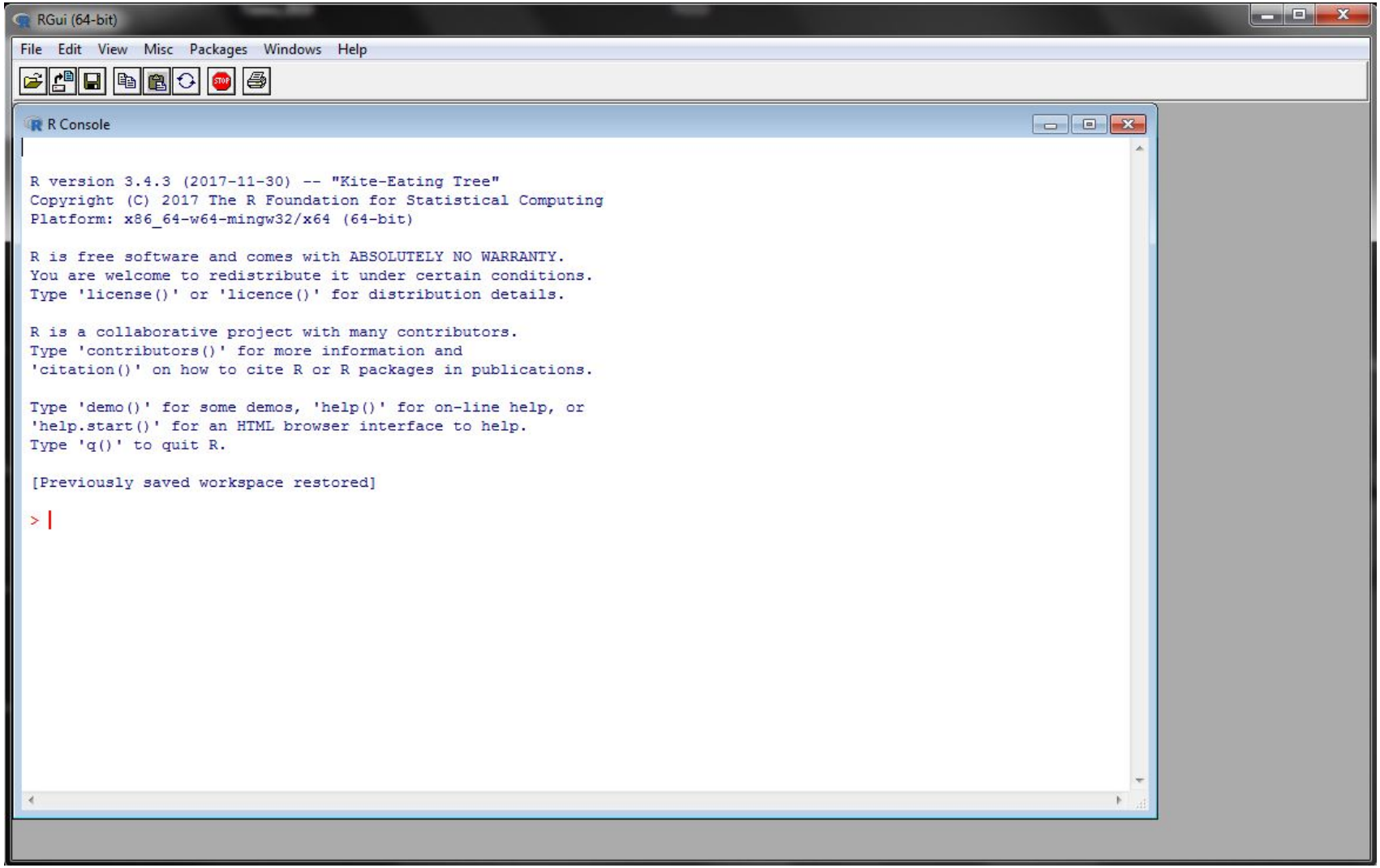
katsileros@aua.gr

Αθήνα 2020



Τι είναι η γλώσσα R

Η R αποτελεί μια γλώσσα προγραμματισμού και ένα περιβάλλον για στατιστικές αναλύσεις και γραφικές απεικονίσεις. Είναι ένα ελεύθερο λογισμικό, ανοικτού κώδικα, το οποίο θεωρείται ευέλικτο και ισχυρό, ενώ υποστηρίζεται από μία μεγάλη κοινότητα χρηστών.
Είναι διαθέσιμο από την ιστοσελίδα http://www.r-project.org/, σε περιβάλλον Windows, Linux και MacOS.


RGui (64-bit)
File Edit View Misc Packages Windows Help
R Console
R version 3.4.3 (2017-11-30) -- "Kite-Eating Tree"
Copyright (C) 2017 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.
[Previously saved workspace restored]
>

Αριθμητικοί Τελεστές

Σύμβολο	**Πράξη**
+	Πρόσθεση
-	Αφαίρεση
*	Πολλαπλασιασμός
/	Διαίρεση
^	Ύψωση σε δύναμη
% / %	Πηλίκο Ακέραιας Διαίρεσης
%%	Υπόλοιπο Διαίρεσης (modulo)

```
> 1 + 1 # πρόσθεση
[1] 2
> 3*8 # πολλαπλασιασμός
[1] 24
> 7/2 # διαίρεση
[1] 3.5
> 5^2 # ύψωση σε δύναμη
[1] 25
> 5%%2 #mod(2)
[1] 1
> 1/0
[1] Inf
> 0/0
[1] NaN
```

Τελεστές Σύγκρισης

Τελεστής	Ερμηνεία
>	Μεγαλύτερο από
<	Μικρότερο από
>=	Μεγαλύτερο ή ίσο από
<=	Μικρότερο ή ίσο από
==	Ίσο με
!=	Όχι ίσο με
&	και
|	ή

Στις τελευταίες εκδόσεις της R οι τελεστές $=$ και $<-$ είναι ισοδύναμοι

```
> x = 5
> y = 2
> x > y
[1] TRUE
> x < y
[1] FALSE
> x == y
[1] FALSE
> x != y
[1] TRUE
> 5 > 2 & 1 < 2
[1] TRUE
> 5 > 2 | 1 > 2
[1] TRUE
```

Βασικές αριθμητικές συναρτήσεις

Συνάρτηση	**Ερμηνεία**
sqrt( )	Τετραγωνική ρίζα
abs( )	Απόλυτη τιμή
sin( ), cos( ), tan( )	Τριγωνομετρικές συναρτήσεις
asin( ), acos( ), atan( )	Τόξα τριγωνομετρικών συναρτήσεων
factorial( )	Παραγοντικό
choose( )	Διωνυμικός συντελεστής
exp( )	Εκθετική συνάρτηση
log( )	Λογάριθμος
gamma( )	Συνάρτηση Γάμμα
floor( )	Μικρότερος ακέραιος
ceiling( )	Μεγαλύτερος ακέραιος
round( )	Στρογγυλοποίηση

```
> sqrt(4)
[1] 2
> abs(-1)
[1] 1
> log(1)
[1] 0
> factorial(4)
[1] 24
> floor(2.8)
[1] 2
> ceiling(2.8)
[1] 3
> pi
[1] 3.141593
> round(pi, 2)
[1] 3.14
```

Τύποι αντικειμένων

- Πραγματικός Αριθμός (numeric)

```
> a = 2
> a
[1] 2
> class(a)
[1] "numeric"
```

- Δεδομένο Χαρακτήρα (character)

```
> b = "Επέμβαση"
> b
[1] "Επέμβαση"
> class(b)
[1] "character"
```

Τύποι αντικειμένων

- Σύνθετος Αριθμός (complex)

```
> c = complex(real = 2, imaginary = 1)
> c
[1] 2+1i
> class(c)
[1] "complex"
```

- Δεδομένο Λογικής (logical)

```
> 5>2
[1] TRUE
```

Δομές αντικειμένων

- Διανύσματα (Vectors)
- Δισδιάστατοι πίνακες (Matrices)
- Πολυδιάστατοι πίνακες (Arrays)
- Πλαίσια δεδομένων (Data frames)
- Λίστες (Lists)

Αριθμητικά διανύσματα

Εισαγωγή δεδομένων αριθμητικού διανύσματος

```
> x = c(5, 1, 4, 6 , 8, 9, 2)
> x
[1] 5 1 4 6 8 9 2
> class(x)
[1] "numeric"
```

Δημιουργία κενού αριθμητικού διανύσματος

```
> x = numeric(7)
> x
[1] 0 0 0 0 0 0 0
> is.mumeric(x)
[1] TRUE
```

Συναρτήσεις αριθμητικών διανυσμάτων

Συνάρτηση	**Ερμηνεία**
length( )	Μήκος διανύσματος
min( ), max( )	Ελάχιστη, μέγιστη τιμή
mean( )	Μέση τιμή
sum( )	Άθροισμα τιμών
prod( )	Γινόμενο τιμών
sort( )	Ταξινόμηση τιμών
rank( )	Κατάταξη τιμών
order( )	Θέση ταξινομημένων τιμών
rev( )	Αντιστροφή σειράς τιμών
sign( )	Πρόσημο τιμών

```
> x = c(5, 1, 4, 6 , 8, 9, 2)
> length(x)
[1] 7
> min(x)
[1] 1
> max(x)
[1] 9
> sum(x)
[1] 35
> rev(x)
[1] 2 9 8 6 4 1 5
> sign(x)
[1] 1 1 1 1 1 1 1
```

```
> x = c(5, 1, 4, 6 , 8, 9, 2)
> sort(x)
[1] 1 2 4 5 6 8 9
> sort(x, decreasing=TRUE)
[1] 9 8 6 5 4 2 1
```

Προσθήκη τιμών στο διάνυσμα

```
> x = c(5, 1, 4, 6 , 8, 9, 2)
> x1=c(x,5)
> x1
[1] 5 1 4 6 8 9 2 5
> rank(x1, ties.method = "average")
[1] 4.5 1.0 3.0 6.0 7.0 8.0 2.0 4.5
> rank(x1, ties.method = "random")
[1] 4 1 3 6 7 8 2 5
> unique(x1)
[1] 5 1 4 6 8 9 2
```

```
> x[2]
[1] 1
> x[-2]
[1] 5 4 6 8 9 2
> x[c(1,7)]
[1] 5 2
> x[x<=5]
[1] 5 1 4 2
> x[x>5]
[1] 6 8 9
> x[x<=4 | x>6]
[1] 1 4 8 9 2
```

```
> x = c(5, 1, 4, 6 , 8, 9, 2)
> y= c(7, 4, 5, 6, 10, 9, 3)
> match(x,y)
[1] 3 NA 2 4 NA 6 NA
> match(y,x)
[1] NA 3 1 4 NA 6 NA
which(x==max(x))
[1] 6
```

```
> x = c(5, 1, 4, 6 , 8, 9, 2)
> y= c(7, 4, 5, 6, 10, 9, 3)
> intersect(x, y)
[1] 5 4 6 9
union(x, y)
[1] 5 1 4 6 8 9 2 7 10 3
> setdiff(x,y)
[1] 1 8 2
```

```
>z=c(1:10)
> z
[1] 1 2 3 4 5 6 7 8 9 10
>z2= seq(from=1,to=10, by=2)
>z2
[1] 1 3 5 7 9
>z3= rep(1:3,time=5)
>z3
[1] 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3
> z4= rep(1:3,each=5)
> z4
[1] 1 1 1 1 1 2 2 2 2 2 3 3 3 3 3
```

```
> x = c(5, 1, 4, 6 , 8, 9, 2)
> mesos=sum(x)/length(x)
> mesos
[1] 5
> s=sort(x)
> n = length(x)
> t = (n+1)/2
> md = (s[floor(t)]+s[ceiling(t)])/2
> md
[1] 5
> median(x)
[1] 5
```

Ορισμός ονομάτων στις τιμές του αριθμητικού διανύσματος

```
> x = c(5, 1, 4, 6 , 8, 9, 2)
> names(x) = c("Nikos", "Maria", "Christina", "Giorgos",
"Eleni", "Agisilaos","Kostas")
> names(x)
[1] "Nikos" "Maria" "Christina" "Giorgos" "Eleni" "Agisilaos"
"Kostas"
> x
 Nikos   Maria Christina Giorgos  Eleni Agisilaos  Kostas
     5       1         4       6       8       9       2
> x[x>=5]
 Nikos Giorgos  Eleni Agisilaos
     5       6       8       9
```

Διανύσματα χαρακτήρων

Εισαγωγή χαρακτήρων σε διάνυσμα

```
> w = c("Athos", "Limnos", "Mexixali", "Papadakis")
w
[1] "Athos" "Limnos" "Mexixali" "Papadakis"
> class(w)
[1] "character"
> a = letters[1:10]
> b = 1:10
> paste(a, b, sep = "")
[1] "a1" "b2" "c3" "d4" "e5" "f6" "g7" "h8" "i9" "j10"
```

Δημιουργία κενού διανύσματος χαρακτηρων

```
> character(length=4)
[1] "" "" "" ""
```

Μετατροπή σε διάνυσμα χαρακτήρων

```
> w1 = c(1:4)
> w1 = as.character(w)
> w1
[1] "1" "2" "3" "4"
> is.character(w1)
[1] TRUE
```

Λογικά διανύσματα

Δημιουργία κενού λογικού διανύσματος

```
> logical(4)
[1] FALSE FALSE FALSE FALSE
```

Μετατροπή σε λογικό διάνυσμα

```
> l=c(0, 1, 0, 1, 1)
> l1=as.logical(l)
> l1
[1] FALSE TRUE FALSE TRUE TRUE
```

Διανύσματα κατηγοριών

Δημιουργία διανύσματος κατηγοριών

```
> f = rep(1:3,each=4)
> f
[1] 1 1 1 1 2 2 2 2 3 3 3 3
> f = factor(f)
> f
[1] 1 1 1 1 2 2 2 2 3 3 3 3
Levels: 1 2 3
> levels(f)
[1] "1" "2" "3"
```

Δισδιάστατοι Πίνακες

Ένας δισδιάστατος πίνακας (matrix) είναι μια δομή δεδομένων της οποίας τα στοιχεία είναι διατεταγμένα σε γραμμές και στήλες.

```
> x= c(1:20)
> X = matrix(x, ncol=4, nrow = 5)
> X
     [,1] [,2] [,3] [,4 ]
[1,]    1    6   11   16
[2,]    2    7   12   17
[3,]    3    8   13   18
[4,]    4    9   14   19
[5,]    5   10   15   20
> class(X)
[1] "matrix"
```

```
> dim(X)
[1] 5 4
> X[1,]
[1] 1 6 11 16
> X[,1]
[1] 1 2 3 4 5
X[5,4]
[1] 20
```

```
> X=c(1:20)
> dim(X)=c(5,4)
> X
     [,1] [,2] [,3] [,4 ]
[1,]    1    6   11   16
[2,]    2    7   12   17
[3,]    3    8   13   18
[4,]    4    9   14   19
[5,]    5   10   15   20
```

```
> colSums(X)
[1] 15 40 65 90
> colMeans(X)
[1] 3 8 13 18
> rowSums(X)
[1] 34 38 42 46 50
> rowMeans(X)
[1] 8.5 9.5 10.5 11.5 12.5
> apply(X,1,mean)
[1] 8.5 9.5 10.5 11.5 12.5
> apply(X,2,mean)
[1] 3 8 13 18
```

```
> Y= c(21, 22, 23, 24, 25)
> X2=cbind(X, Y)
> X2
                          Y
 [1,]   1    6   11   16  21
 [2,]   2    7   12   17  22
 [3,]   3    8   13   18  23
 [4,]   4    9   14   19  24
 [5,]   5   10   15   20  25
```

```
> dimnames(X2)=list(c("R1", "R2","R3","R4","R5"),
c("C1","C2","C3","C4","Y"))
> X2
    C1 C2 C3 C4 Y
R1   1  6 11 16 21
R2   2  7 12 17 22
R3   3  8 13 18 23
R4   4  9 14 19 24
R5   5 10 15 20 25
```

```
> Z=diag(1:5)
> Z
     [,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
```

```
> W=diag(5)
> W
     [,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
```

Τελεστές πινάκων

Σύμβολο	**Πράξη**
%*%	Πολλαπλασιασμός πίνακα
t()	Ανάστροφος πίνακα
solve()	Αντίστροφος πίνακα
eigen()	Ιδιοτιμές και ιδιοδιανύσματα
det()	Ορίζουσα πίνακα

```
> t(X)
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    2    3    4    5
[2,]    6    7    8    9   10
[3,]   11   12   13   14   15
[4,]   16   17   18   19   20
```

```
> X3=matrix(1:4, ncol=2)
> X3
     [,1] [,2]
[1,]    1    3
[2,]    2    4
> X3 + 2
     [,1] [,2]
[1,]    3    5
[2,]    4    6
> X3^2
     [,1] [,2]
[1,]    1    9
[2,]    4   16
> X3==1
      [,1]  [,2]
[1,]  TRUE FALSE
[2,] FALSE FALSE
```

```
> A=matrix(1:6, ncol =2)
> A
     [,1] [,2]
[1,]    1    4
[2,]    2    5
[3,]    3    6
> B=matrix(7:12, ncol=3)
> B
     [,1] [,2] [,3]
[1,]    7    9   11
[2,]    8   10   12
> A%*%B
     [,1] [,2] [,3]
[1,]   39   49   59
[2,]   54   68   82
[3,]   69   87  105
```

```
> eigen(X3)
eigen() decomposition
$values
[1] 5.3722813 -0.3722813
$vectors
[,1] [,2]
[1,] -0.5657675 -0.9093767
[2,] -0.8245648 0.4159736
```

$x + y = 2$
$x - 2y = 8$

```
> X4=c (1,1,1,-2)
> X4=matrix(X4, ncol=2)
> X4
[,1] [,2]
[1,] 1 1
[2,] 1 -2
> Y4=c(2,8)
> solve(X4,Y4)
[1] 4 -2
```

Πολυδιάστατοι Πίνακες

Ένας πολυδιάστατος πίνακας (array) είναι πίνακας με τρεις ή περισσότερες διαστάσεις.

```
> AR=array(1:24, dim = c(3, 4, 2))
> AR
, , 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
> class(AR)
[1] "array"
```

Πλαίσια δεδομένων

Τα πλαίσια δεδομένων (data frames) είναι δισδιάστατοι πίνακες στους οποίους δεν χρειάζεται οι στήλες να είναι όλες του ίδιου τύπου.

```
> x = factor(rep(1:2,each=3))
> y = c(1:6)
> A = data.frame(x, y)
> A
     x   y
 1   1   1
 2   1   2
 3   1   3
 4   2   4
 5   2   5
 6   2   6
> class(A)
[1] "data.frame"
```

```
> y = 1:20
> x = factor(rep(letters[1:5], each = 4))
> A=data.frame(x,y)
> tapply(A$y, A$x, mean)
a b c d e
2.5 6.5 10.5 14.5 18.5
> aggregate(y ~ x, data = A, mean)
x y
1 a 2.5
2 b 6.5
3 c 10.5
4 d 14.5
5 e 18.5
>
```

```
> write.table(A, file = "A.txt")
> fix(A)
> read.table("C:\\Users\\maggie\\Documents\\A.txt",
header=TRUE, dec="")
  x y
1 1 1
2 1 2
3 1 3
4 2 4
5 2 5
6 2 6
```

Συναρτήσεις

Δημιουργία συναρτήσεων

```
> name_function = function(agruments){
    statement
    statement
    ...
    return(value)
  }
```

```
> name_function(x)
```

```
> ginomeno = function(a, b) {
    result= a*b
    return(result)
   }
> ginomeno(2,3)
[1] 6
```

```
> power = function(a, b) {
    result= a^b
    return(result)
   }
> power(2,2)
[1] 4
```

```
> x1= c(5, 1, 4, 6 , 8, 9, 2, 5)
> diamesos = function(x) {
    s=sort(x)
    n = length(x)
    t = (n+1)/2
    diamesos = (s[floor(t)]+s[ceiling(t)])/2
    diamesos
}
> diamesos(x1)
[1] 5
> median(x1)
[1] 5
```

```
> x1= c(5, 1, 4, 6 , 8, 9, 2, 5)
> koryfh = function(x){
    tabulate(x)
    which.max(tabulate(x))
    }
> koryfh(x1)
[1] 5
```

ax^2+bx+c=0

$$x = \frac{-b \pm \sqrt{\Delta}}{2a} \quad \text{όπου } \Delta = b^2 - 4ac$$

```
> quadratic = function(a, b,c) {
    x1 = (-b + sqrt(b^2 -4*a*c))/(2*a)
    x2 = (-b - sqrt(b^2 -4*a*c))/(2*a)
    print(x1)
    print(x2)
    }
> quadratic( 1, 1, 0)
[1] 0
[1] -1
```

Δομές ελέγχου ροής

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
for(index in A) B	Εκτελεί το B όσο το index ανήκει στο A
while(A) B	Ελέγχει κατά επανάληψη αν ισχύει το A. Αν ναι επιστρέφει B
repeat A	Όπως το while (απαιτεί το break)

```
> Proshmo = function(x) {
    if(x > 0){
     print("Non-negative number")
    } else {
     print("Negative number")
    }
    }
> Proshmo(-5)
[1] "Negative number"
```

```
> Elegxos = function(x) {
    if (x > 0) {
    result ="Positive"
    }
    else if (x < 0) {
    result = "Negative"
    }
    else {
    result = "Zero"
    }
   return(result)
    }
> Elegxos(5)
[1] "Positive"
```

```
> x1= c(5, 1, 4, 6 , 8, 9, 2, 5)
> diamesos = function(x) {
    n = length(x)
    s = sort(x)
    if (n %% 2 == 0) {
    (s[n/2]+ s[n/2+1])/2
    } else {s[(n+1)/2]
    }
    }
> diamesos(x1)
[1] 5
```

```
> x1= c(5, 1, 4, 6 , 8, 9, 2, 5)
> athroisma=function(x){
    athroisma=0
    for(i in 1:n)
    {
    athroisma=athroisma+x[i]
    }
    athroisma
    }
> athroisma(x1)
[1] 40
> sum(x1)
[1] 40
```

```
> x1= c(5, 1, 4, 6 , 8, 9, 2, 5)
> mesos=function(x){
    n=length(x)
    mesos=0
    for(i in 1:n)
    {
    mesos=mesos+x[i]/n
    }
    mesos
    }
> mesos(x1)
[1] 5
> mean(x1)
[1] 5
```

$$s^2 = \frac{\sum (Y_i - \bar{Y})^2}{n-1}$$

```
> x1= c(5, 1, 4, 6 , 8, 9, 2, 5)
> diakimansi=function(x){
    n=length(x)
    diakimansi=0
    for(i in 1:n)
    {
    diakimansi=diakimansi+(x[i]-mean(x))^2/(n-1)
    }
    diakimansi
    }
> diakimansi(x1)
[1] 7.428571
> var(x1)
[1] 7.428571
```

```
> x = 4
> paragontiko=function(x){
paragontiko = 1
i = 1
while (i <= x)
{
    paragontiko = paragontiko * i
    i = i + 1
}
    paragontiko
    }
> paragontiko(x)
[1] 24
```

```
x = 1
repeat
{
    print(x)
    x = x + 1
    if(x > 4)
    {
    break
    }
}
[1] 1
[1] 2
[1] 3
[1] 4
```