ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΙΓΑΙΟΥ-ΣΧΟΛΗ ΠΕΡΙΒΑΛΛΟΝΤΟΣ

ΤΜΗΜΑ ΕΠΙΣΤΗΜΩΝ ΤΗΣ ΘΑΛΑΣΣΑΣ

ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ ΣΤΗ ΓΛΩΣΣΑ

Εισαγωγικές Σημειώσεις

(Έκδοση 2.0)

Γ. Τσιρτσής-Β. Κολοβογιάννης

Μυτιλήνη

Φεβρουάριος 2015

Πρόλογος

Οι σημειώσεις αυτές απευθύνονται στους πρωτοετείς φοιτητές του Τμήματος Επιστημών της Θάλασσας στο πλαίσιο του μαθήματος 'Εισαγωγή στην Πληροφορική και Προγραμματισμός', καθώς και στους φοιτητές του ομώνυμου μαθήματος της σειράς ανοιχτών μαθημάτων (open courses) του Πανεπιστημίου Αιγαίου. Από το ακαδημαϊκό έτος 2013-14 διδάσκεται στο Τμήμα Επιστημών της Θάλασσας προγραμματισμός σε γλώσσα R αντί της FORTRAN. Οι λόγοι της επιλογής αυτής είναι οι εξής: (α) Το περιβάλλον R τείνει να γίνει διεθνές standard στην επεξεργασία δεδομένων στις επιστήμες και στην τεχνολογία, έναντι αντίστοιχων εμπορικών ή ελεύθερων λογισμικών, (β) Το περιβάλλον R χρησιμοποιείται πλέον ευρύτατα διεθνώς στη τριτοβάθμια εκπαίδευση σε μαθήματα στατιστικής και προγραμματισμού, (γ) Πρόκειται για ελεύθερο λογισμικό, προσβάσιμο στον κάθε φοιτητή μέσω του διαδικτύου, με πληθώρα συναρτήσεων-έτοιμων προγραμμάτων, πηγών αλλά και εμπλουτιζόμενο με νέες συναρτήσεις-δυνατότητες σε συνεχή βάση και (δ) Το περιβάλλον προγραμματισμού R, σε συνδυασμό με το RStudio, είναι περισσότερο φιλικά στο χρήστη σε σχέση με κλασικές γλώσσες όπως η FORTRAN.

Το μάθημα περιλαμβάνει: Εισαγωγή στον προγραμματισμό και στις έννοιες του αλγορίθμου και του λογικού διαγράμματος, Εγκατάσταση της R και του RStudio, Εισαγωγή δεδομένων από το πληκτρολόγιο (εντολές scan και c) και από αρχείο (ανάγνωση αρχείων csv που έχουν κατασκευαστεί σε LibreOffice Calc ή MS Excel), Δομές επανάληψης (εντολές for, while, apply), Εντολές απόφασης (if, else if), Μεταβλητές με δείκτες μίας (διανύσματα-vectors) ή περισσότερων (πίνακες-matrices) διαστάσεων, Ενσωματωμένες συναρτήσεις (πχ abs, min, max, mean, plot) και συναρτήσεις χρήστη (εντολή function), Εγκατάσταση και χρήση πακέτων συναρτήσεων (R packages), Εντολές γραφικών. Τέλος γίνεται αναφορά σε ειδικότερα θέματα όπως η κατασκευή και διάθεση πακέτων συναρτήσεων και ο παράλληλος προγραμματισμός. Αναλυτικότερη πληροφορία για το μάθημα (περιεχόμενα, βοηθήματα, ασκήσεις) υπάρχουν στους συνδέσμους του μαθήματος στο e-class (https://eclass.aegean.gr/courses/MAR102/ για τους φοιτητές του Τμήματος και https://eclass.aegean.gr/courses/MAR118/ για τους φοιτητές open courses, στους οποίους έχουν πρόσβαση οι εγγεγραμμένοι στο κάθε μάθημα φοιτητές.

Για τις ανάγκες του μαθήματος προσφέρονται οι παρούσες εισαγωγικές σημειώσεις που καλύπτουν τις βασικές εντολές, οι οποίες διδάσκονται στις διαλέξεις του μαθήματος και εφαρμόζονται στις

εργαστηριακές ασκήσεις. Οι Σημειώσεις περιλαμβάνουν 7 κεφάλαια. Στο πρώτο κεφάλαιο γίνεται εισαγωγή στον προγραμματισμό και στις έννοιες του αλγορίθμου και του λογικού διαγράμματος. Επίσης περιγράφεται η διαδικασία εγκατάστασης της R και του RStudio και η χρήση της R από την γραμμή εντολών. Στο δεύτερο κεφάλαιο περιγράφονται οι βασικές εντολές προγραμματισμού, ενώ στο τρίτο εντολές που αυξάνουν σε μεγάλο βαθμό τις δυνατότητες προγραμματισμού. Στο τέταρτο κεφάλαιο περιγράφεται η χρήση των αρχείων εισόδου και εξόδου και οι σχετικές εντολές και στο πέμπτο η χρήση ενσωματωμένων συναρτήσεων και η κατασκευή συναρτήσεων και υποπρογραμμάτων. Το έκτο κεφάλαιο αναφέρεται στη χρήση εντολών κατασκευής γραφικών και τέλος το έβδομο στην εγκατάσταση και χρήση πακέτων συναρτήσεων (R packages). Στο έβδομο κεφάλαιο περιλαμβάνονται επίσης ειδικότερα θέματα όπως η κατασκευή πακέτων συναρτήσεων και ο παράλληλος προγραμματισμός.

ΠΕΡΙΕΧΟΜΕΝΑ

Α. ΕΙΣΑΓΩΓΗ

Α1. Εισαγωγή στον Προγραμματισμό Ηλεκτρονικού Υπολογιστή

Ένας Ηλεκτρονικός Υπολογιστής (ΗΥ) αποτελείται από 2 μέρη, το υλικό (hardware) και το λογισμικό (software). Το υλικό περιλαμβάνει (α) το κουτί του ΗΥ που φιλοξενεί την κεντρική μονάδα επεξεργασίας, τη μνήμη, το τροφοδοτικό ρεύματος, τα αποθηκευτικά μέσα (σκληρό δίσκο, οδηγούς DVD, συσκευών USB, καρτών) και τις διάφορες κάρτες όπως γραφικών, δικτύου κ.λ.π., (β) την οθόνη, (γ) το ποντίκι ή άλλο καταδεικτικό μέσο και (δ) περιφερειακά μέσα όπως εκτυπωτή, σαρωτή, κάμερα, μικρόφωνο και άλλα. Το λογισμικό περιλαμβάνει (α) το λειτουργικό σύστημα (MS Windows, Linux, Mac OS, Unix) που επιτελεί τις βασικές λειτουργίες του ΗΥ και (β) τα προγράμματα, όπως επεξεργασίας κειμένου, φύλλων εργασίας, παρουσιάσεων, βάσεων δεδομένων, πλοήγησης στο διαδίκτυο και ηλεκτρονικού ταχυδρομείου, ζωγραφικής και άλλα πολλά. Και το λειτουργικό σύστημα και τα προγράμματα αποτελούν τον ενδιάμεσο μεταξύ του ΗΥ και του χρήστη, ώστε κάθε ενέργεια του χρήστη (επιλογή με το ποντίκι, πληκτρολόγηση) να μεταφράζεται μέσω σειράς εντολών σε ενέργειες κατανοητές από τον ΗΥ, από τον οποίο και τελικά εκτελούνται. Οι εντολές που βρίσκονται πίσω από κάθε ενέργεια του χρήστη δεν είναι ορατές και έχουν γραφεί από προγραμματιστές με χρήση κάποιας γλώσσας προγραμματισμού (C, C++, Pascal, Fortran, Basic, Visual Basic, Cobol, Python, Java, R). Οι γλώσσες προγραμματισμού είναι γλώσσες με αλφάβητο, λέξεις και συντακτικό όπως οι γλώσσες που μιλάνε οι άνθρωποι. Οι εντολές τους, που είναι συνήθως συνδυασμοί απλών αγγλικών λέξεων (for, do, read, while, if, write, print), μεταφράζονται στην συνέχεια σε αντίστοιχες εντολές κατανοητές από τον ΗΥ και εκτελούνται.

Προγραμματισμός του ΗΥ είναι η διαδικασία συγγραφής σειράς εντολών για την εκτέλεση μίας συγκεκριμένης εργασίας. Η συγγραφή σειράς εντολών, ενός προγράμματος δηλαδή, γίνεται σε κάποια γλώσσα προγραμματισμού από τον προγραμματιστή. Στην συνέχεια ο χρήστης του προγράμματος εκτελεί το πρόγραμμα και επιτελεί την επιθυμητή εργασία. Ο ίδιος ο προγραμματιστής μπορεί να είναι και χρήστης του προγράμματος.

Η R είναι μία γλώσσα προγραμματισμού ελεύθερη, που ξεκίνησε ως λογισμικό στατιστικής και γραφικών, αλλά πολύ γρήγορα εμπλουτίστηκε και εξακολουθεί να εμπλουτίζεται με νέες δυνατότητες, ώστε σήμερα να αποτελεί ένα από τα ευρύτερα χρησιμοποιούμενα λογισμικά για

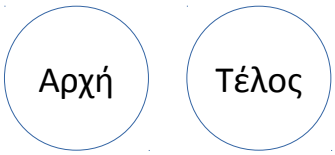
ανάλυση δεδομένων. Υπάρχει στο διαδίκτυο πληθώρα έτοιμων συναρτήσεων για κάθε σχεδόν εργασία, καθώς και άφθονες πηγές πληροφόρησης, που κάνουν την R ιδιαίτερα δημοφιλή. Πολλά εκπαιδευτικά ιδρύματα παγκοσμίως χρησιμοποιούν την R ως ενιαία πλατφόρμα εργασίας για στατιστική, γραφικά, ανάλυση δεδομένων και άλλα.

Α2. Η έννοια του αλγορίθμου και του λογικού διαγράμματος

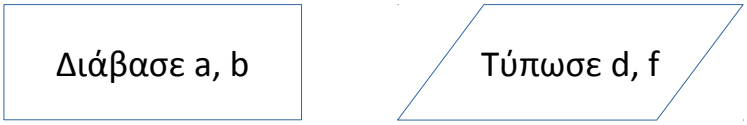
Ως αλγόριθμος στην επίλυση ενός προβλήματος νοείται η σειρά των διαδοχικών ενεργειών-βημάτων που πρέπει να γίνουν για την επίλυση του προβλήματος. Ο αλγόριθμος κατά μία άλλη έννοια είναι η συνταγή επίλυσης του προβλήματος. Η συνταγή χρησιμοποιείται εδώ με την καθημερινή της έννοια, όπως στην ετοιμασία ενός φαγητού, που για να έχει επιτυχία, πρέπει να γίνουν συγκεκριμένα βήματα και με τη σωστή σειρά. Ένας αλγόριθμος, εκτός από το να οδηγεί στην επίλυση ενός προβλήματος, πρέπει να διαθέτει και μία σειρά από χαρακτηριστικά: (α) Σαφήνεια στον ορισμό του κάθε βήματος, που δεν θα επιτρέπει αμφιβολίες κατά την υλοποίησή του, (β) Περατότητα, δηλαδή διασφάλιση της ολοκλήρωσης του επιδιωκόμενου αποτελέσματος μετά από πεπερασμένο αριθμό βημάτων, (γ) Αποτελεσματικότητα στην εξαγωγή του τελικού αποτελέσματος με ακριβή σειρά βημάτων και ακολουθώντας τη συντομότερη οδό και (δ) Επεκτασιμότητα στην αντιμετώπιση συναφών προβλημάτων. Ένας αλγόριθμος, δηλαδή η σειρά των ενεργειών-βημάτων που τον αποτελούν, μπορεί να εκφραστεί μέσω ενός κειμένου ή συνηθέστερα με την μορφή ενός λογικού διαγράμματος ή διαγράμματος ροής.

Ένα διάγραμμα ροής περιγράφει οπτικά έναν αλγόριθμο. Περιέχει μία σειρά από σχήματα που το καθένα εκφράζει μία ενέργεια-βήμα, που στη περίπτωση που θα εκτελεστούν από μία μηχανή όπως ο ΗΥ, συνιστούν τις εντολές προς τη μηχανή. Τα σχήματα συνδέονται μεταξύ τους με βέλη που καθορίζουν τη σειρά που εκτελούνται οι εντολές. Τα σχήματα που χρησιμοποιούνται μπορεί να διαφέρουν κατά περίπτωση (από συγγραφέα σε συγγραφέα). Για τις ανάγκες του παρόντος κειμένου χρησιμοποιούνται τα ακόλουθα σχήματα για τις αντίστοιχες εντολές.

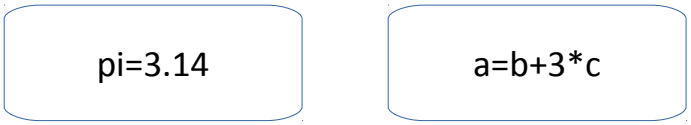
(α) Εντολές αρχής και τέλους: Δηλώνουν την αρχή και το τέλος ενός αλγόριθμου και συμβολίζονται με κύκλο.



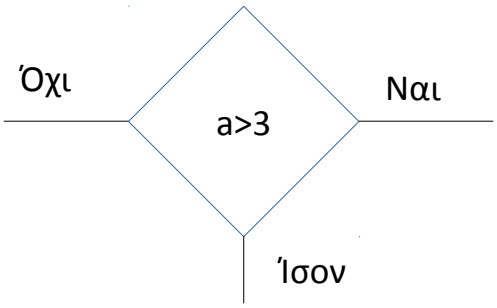
(β) Εντολές εισαγωγής δεδομένων και εκτύπωσης αποτελεσμάτων: Αποδίδουν τιμές σε μεταβλητές του προγράμματος από τον χρήστη μέσω του πληκτρολογίου ή άλλου μέσου εισόδου δεδομένων και τυπώνουν στην οθόνη ή σε άλλο μέσο τιμές μεταβλητών που προκύπτουν από την επεξεργασία που γίνεται μέσα στο πρόγραμμα. Για την εισαγωγή δεδομένων θα χρησιμοποιήσουμε το σχήμα του ορθογώνιου παραλληλόγραμμου και για την εκτύπωση αποτελεσμάτων το παραλληλόγραμμο.



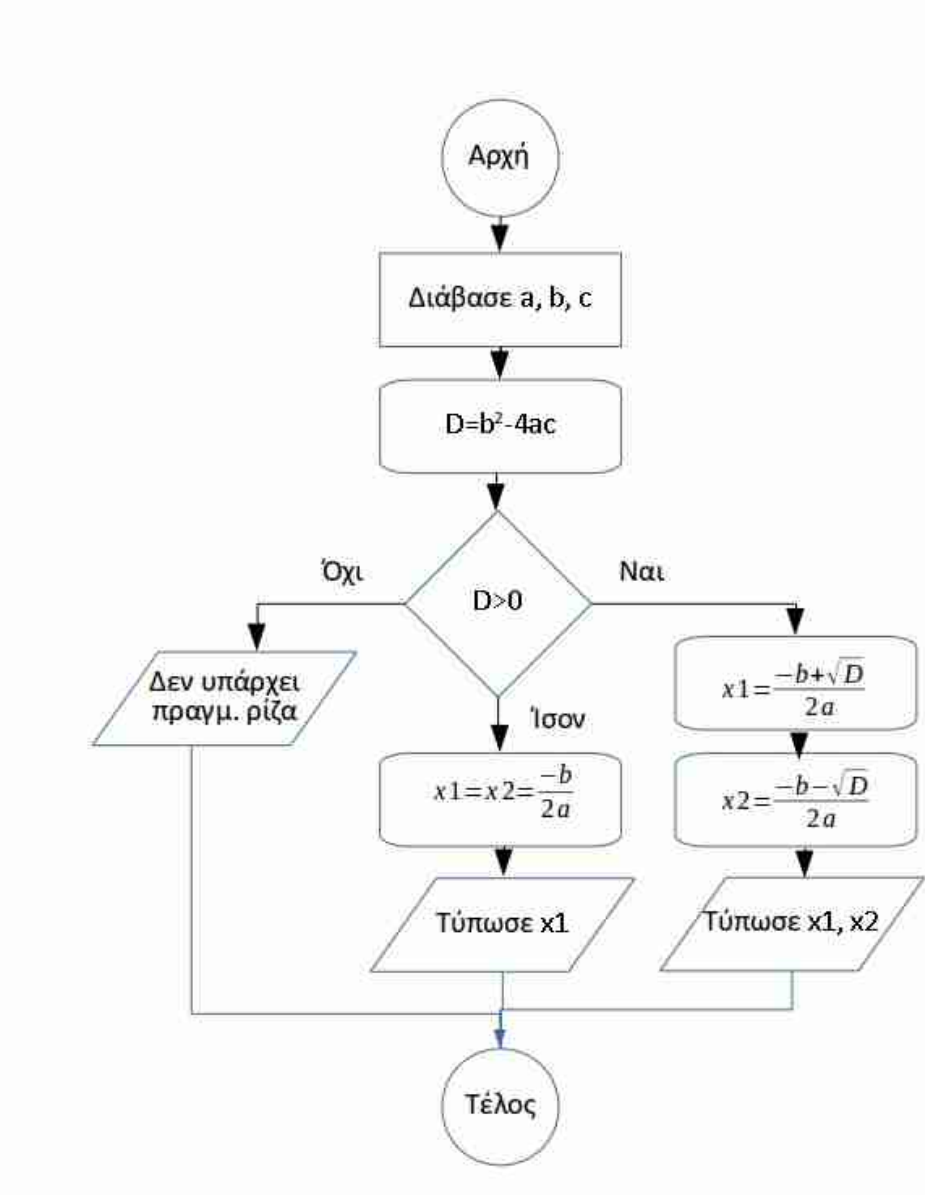
(γ) Εντολές απόδοσης τιμών σε μεταβλητές ή εκτέλεσης πράξεων: Αποδίδουν τιμές σε μεταβλητές κατά τη διάρκεια εκτέλεσης του προγράμματος ή εκτελούν πράξεις. Θα χρησιμοποιηθεί στη περίπτωση αυτή το σχήμα του ορθογωνίου παραλληλογράμμου με στρογγυλεμένες γωνίες.



(δ) Εντολές απόφασης: Λαμβάνουν αποφάσεις κατά την εκτέλεση ενός προγράμματος και διακλαδίζουν τη ροή των εντολών. Χρησιμοποιείται συνήθως το σχήμα του ρόμβου.



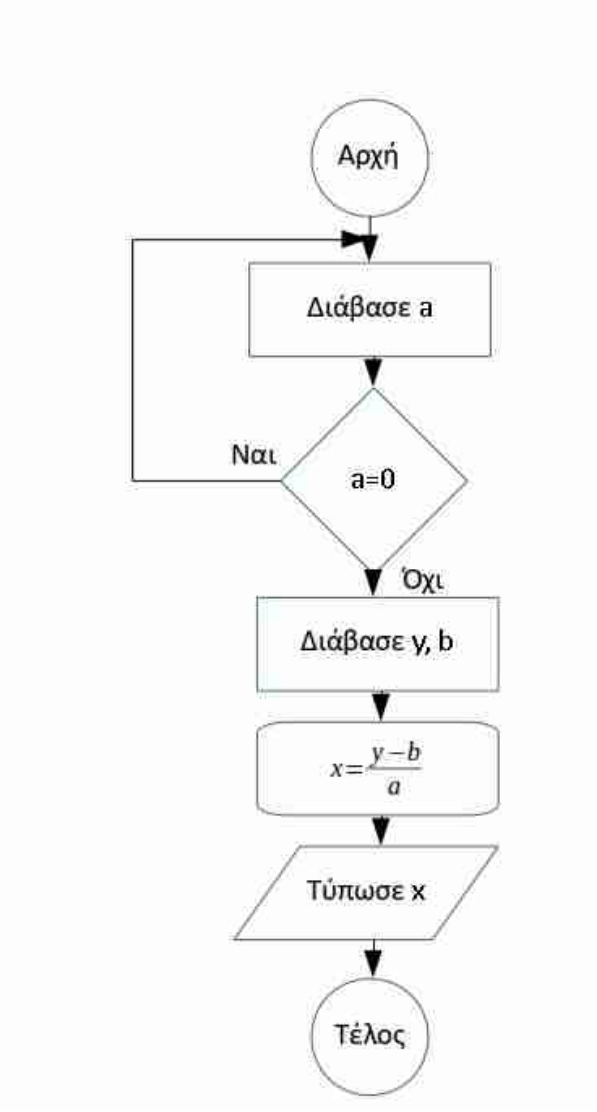
Ένα παράδειγμα λογικού διαγράμματος δίδεται στη συνέχεια για την επίλυση εξίσωσης δεύτερου βαθμού $ax^2+bx+c=0$, με $a\neq0$ (Σχήμα Α1). Στην αρχή ορίζονται οι τιμές των a, b και c. Στη συνέχεια υπολογίζεται η τιμή της διακρίνουσας $D=b^2-4ac$. Με βάση τη τιμή της διακρίνουσας λαμβάνεται απόφαση για τον αριθμό και το είδος των λύσεων της εξίσωσης. Τέλος τυπώνονται οι λύσεις της εξίσωσης.



Σχήμα Α1. *Λογικό διάγραμμα (διάγραμμα ροής) του αλγορίθμου επίλυσης δευτεροβάθμιας εξίσωσης.*

Ένα δεύτερο παράδειγμα αφορά στο πρόβλημα επίλυσης πρωτοβάθμιας εξίσωσης της μορφής y=ax+b, αν δίδονται οι τιμές των y, a και b. Επιπλέον απαιτείται να υπάρχει πρόνοια στον αλγόριθμο στη περίπτωση που ο χρήστης δώσει τη τιμή 0 στο συντελεστή a. Στον σχετικό αλγόριθμο δίδεται κατ' αρχήν η τιμή του a και ελέγχεται αν είναι μηδέν. Αν ναι ζητείται από τον χρήστη νέα τιμή, ενώ σε άλλη περίπτωση και αφού οριστούν οι τιμές των y και b, επιλύεται η εξίσωση ως προς x και

εκτυπώνεται η τιμή του x. Το σχετικό λογικό διάγραμμα δίδεται στη συνέχεια (Σχήμα Α2).



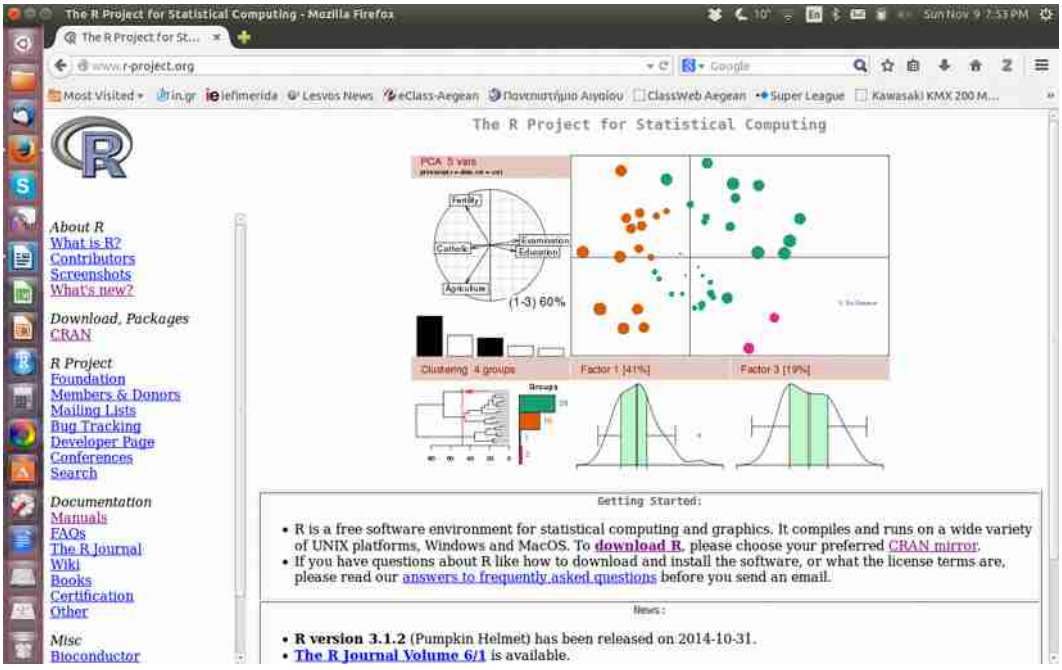
Σχήμα Α2. *Λογικό διάγραμμα (διάγραμμα ροής) του αλγορίθμου επίλυσης πρωτοβάθμιας εξίσωσης με έλεγχο για τη περίπτωση που ο συντελεστής α είναι μηδέν.*

Από τα παραπάνω παραδείγματα φαίνεται ότι ένα λογικό διάγραμμα εκφράζει με σαφήνεια τον αντίστοιχο αλγόριθμο και είναι απλή διαδικασία στη συνέχεια η μετατροπή του λογικού διαγράμματος σε πρόγραμμα (σειρά εντολών), γραμμένο σε οποιαδήποτε γλώσσα προγραμματισμού. Κατά συνέπεια το λογικό διάγραμμα αποτελεί το ενδιάμεσο βήμα μεταξύ του αλγορίθμου επίλυσης ενός προβλήματος και του αντίστοιχου προγράμματος ΗΥ. Αν και το ενδιάμεσο

αυτό βήμα είναι συχνά χρήσιμο, δεν είναι απαραίτητο. Εναπόκειται στη κρίση και στην εμπειρία του προγραμματιστή η απόφαση αν θα κατασκευάσει λογικό διάγραμμα, ή αν θα περάσει απευθείας στη συγγραφή του αλγορίθμου σε μορφή προγράμματος σε κάποια γλώσσα προγραμματισμού.

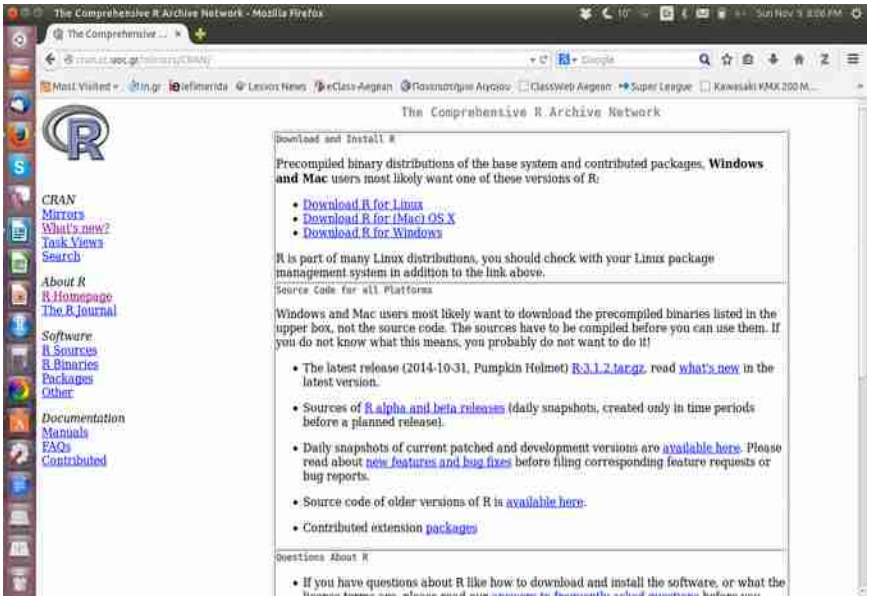
Α3. Γενικά εισαγωγικά στοιχεία για την χρήση της R

Για να χρησιμοποιήσουμε τη γλώσσα προγραμματισμού R πρέπει να επισκεφθούμε τον δικτυακό τόπο http://www.r-project.org/ (Εικόνα Α3) και να κατεβάσουμε δωρεάν την έκδοση της R που αντιστοιχεί στο λειτουργικό σύστημα του ΗΥ μας. Επιλέγοντας Download R, οδηγούμαστε σε ιστοσελίδα όπου επιλέγουμε ένα mirror site για τη λήψη, προτιμότερο από Ελλάδα (http://cran.cc.uoc.gr/mirrors/CRAN/) για μεγαλύτερη ταχύτητα. Η R είναι διαθέσιμη για όλα τα συνηθισμένα λειτουργικά, όπως Windows, Linux, Mac OS (Εικόνα Α4). Κατεβάζοντας και εγκαθιστώντας την R έχουμε όλες τις δυνατότητες που περιγράφονται παρακάτω. Σημειώνεται ότι με την παραπάνω διαδικασία εγκαθίσταται το βασικό πακέτο εντολών και συναρτήσεων της R (base package). Δυνατότητες μπορούν να προστεθούν κατεβάζοντας πακέτα (packages) που επιτελούν πιο εξειδικευμένες εργασίες.

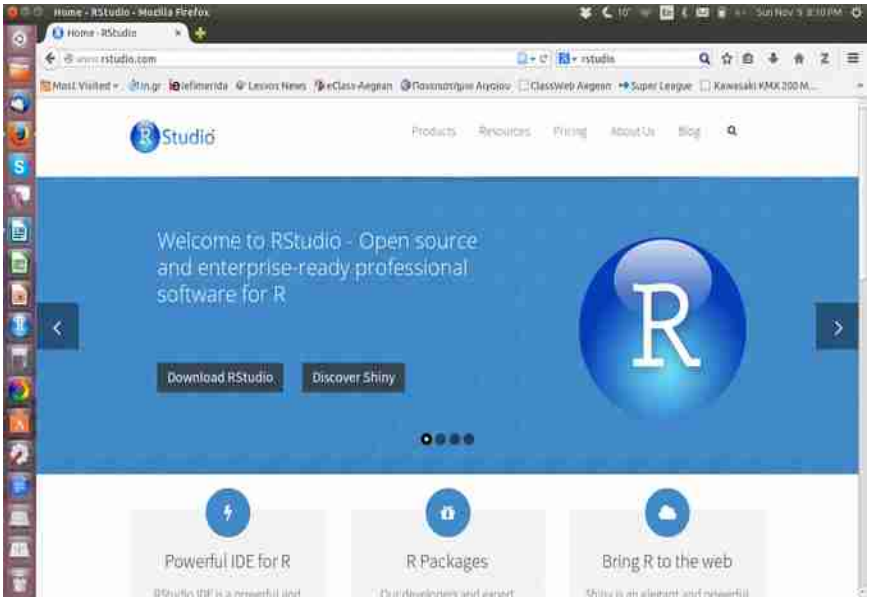


Εικόνα Α3. *Ο επίσημος δικτυακός τόπος της R.*

Επιπλέον δυνατότητες, κυρίως ως προς την ευκολία χρήσης της R, παρέχει το πρόγραμμα RStudio που μπορεί να κατεβάσει κάποιος δωρεάν από τον δικτυακό τόπο http://www.rstudio.com/ (Εικόνα Α5). Το RStudio αποτελεί ένα ενδιάμεσο (interface) που κάνει τη χρήση της R ευκολότερη και πιο φιλική. Σημειώνεται ότι η R λειτουργεί αυτόνομα και παρέχει όλες τις δυνατότητες που περιγράφονται παρακάτω, ενώ το RStudio δεν λειτουργεί αυτόνομα και προϋποθέτει την εγκατάσταση της R.



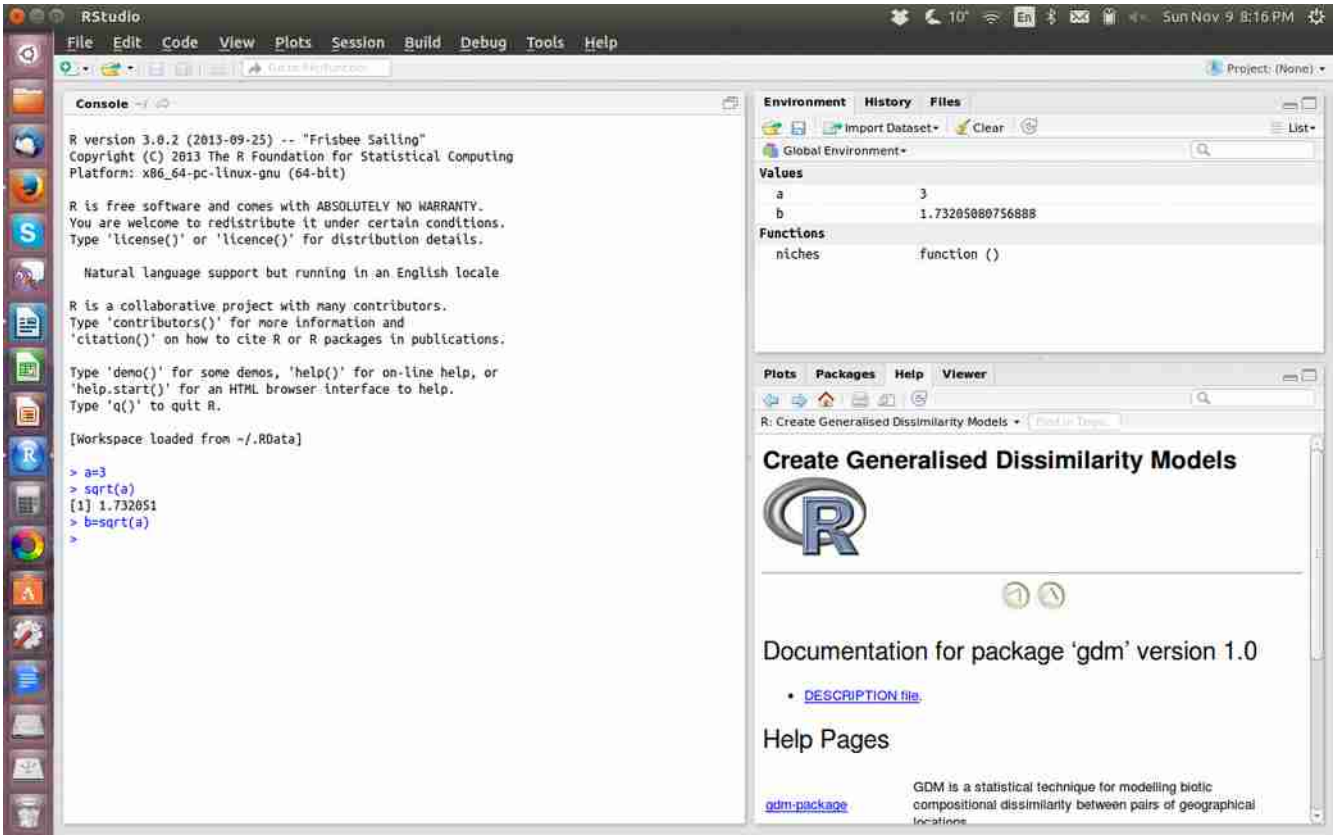
Εικόνα Α4. *Λήψη της R στην έκδοση του λειτουργικού του HY μας από επιλεγμένο mirror site.*



Εικόνα Α5. *Ο δικτυακός τόπος του RStudio.*

Α4. Χρήση της R από την γραμμή εντολών

Για να χρησιμοποιήσουμε την R (είτε απευθείας, είτε μέσω του RStudio) μπορούμε να γράψουμε στην γραμμή εντολών μία μεμονωμένη εντολή που εκτελείται μόλις πατήσουμε Enter και αλλάξουμε γραμμή (Εικόνα Α6). Π.χ. αν γράψω >a=3 και πατήσω Enter καταχωρείται στη μνήμη στη θέση της μεταβλητής a η τιμή 3. Αν στην επόμενη γραμμή γράψω >sqrt(a) και Enter υπολογίζεται και τυπώνεται η τετραγωνική ρίζα του a, ενώ αν γράψω >b=sqrt(a) υπολογίζεται η τετραγωνική ρίζα του a και η τιμή της καταχωρείται στη μνήμη στη θέση της μεταβλητής b. Στο RStudio υπάρχει το παράθυρο Environment (συνήθως πάνω δεξιά) όπου φαίνονται οι μεταβλητές και οι τιμές τους κάθε στιγμή. Αν χρησιμοποιούμε για τις παραπάνω εντολές την R και όχι το RStudio, η εικόνα που βλέπουμε είναι ακριβώς αυτή που φαίνεται στο παράθυρο Console του RStudio (Εικόνα Α6, αριστερά), δεν εμφανίζονται δηλαδή ούτε το μενού επάνω, ούτε τα υπόλοιπα παράθυρα.



Εικόνα Α6. *Χρήση της R μέσω του RStudio από τη γραμμή εντολών. Αν χρησιμοποιούμε την R και όχι το RStudio βλέπουμε μόνο το παράθυρο Console (αριστερά).*

Για να γράψω ένα πρόγραμμα (σειρά εντολών και όχι μεμονωμένες εντολές) ακολουθώ όσα

περιγράφονται αμέσως παρακάτω στην επόμενη ενότητα.

ΑΣΚΗΣΕΙΣ

Α1. Δίδεται η αριθμητική τιμή μεταβλητής x και ζητείται ο υπολογισμός της απόλυτης τιμής της. Εκφράστε σε μορφή λογικού διαγράμματος τον αλγόριθμο επίλυσης του προβλήματος.

Α2. Δίδονται οι αριθμητικές τιμές μεταβλητών x και y και ζητείται ο υπολογισμός της τιμής της παράστασης A=|x-2|-|y-3|. Εκφράστε σε μορφή λογικού διαγράμματος τον αλγόριθμο επίλυσης του προβλήματος.

Α3. Δύο παίκτες (Α και Β) ρίχνουν ένα νόμισμα και ο ένας ποντάρει στην μία πλευρά (κορώνα), ενώ ο άλλος στην άλλη (γράμματα). Όποιος προβλέψει σωστά το αποτέλεσμα κερδίζει έναν πόντο σε κάθε ρίψη, ενώ τελικά νικητής είναι αυτός που θα συλλέξει 50 σωστές προβλέψεις. Να αναπτύξτε αλγόριθμο σε μορφή λογικού διαγράμματος στον οποίο θα δίδεται ο παίκτης που προέβλεψε σωστά (Α ή Β) σε κάθε ρίψη και θα βρίσκεται τελικά ο νικητής του παιχνιδιού.

Α4. Εισάγετε στην γραμμή εντολών της R ή του RStudio τις αριθμητικές τιμές δύο μεταβλητών a και b, υπολογίστε το άθροισμά τους, την απόλυτη τιμή του αθροίσματος (συνάρτηση abs) και στην συνέχεια την τετραγωνική ρίζα της απόλυτης τιμής (συνάρτηση sqrt).

Β. ΤΑ ΒΑΣΙΚΑ ΤΟΥ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ ΣΕ R

Β1. Κατασκευή προγράμματος σε R

Υπάρχουν δύο τρόποι κατασκευής προγράμματος (σειράς εντολών και όχι μίας μεμονωμένης) στην R. Εκ των δύο προτείνεται ο δεύτερος που θα χρησιμοποιηθεί κατ' εξοχήν παρακάτω.

1ος τρόπος: Στη γραμμή εντολών > εισάγουμε άγκιστρο { και στην συνέχεια πατάμε Enter. Στις επόμενες γραμμές γράφουμε τις εντολές του προγράμματος και όταν τελειώσουμε εισάγουμε στην τελευταία γραμμή άγκιστρο } (Εικόνα Β1).

> {

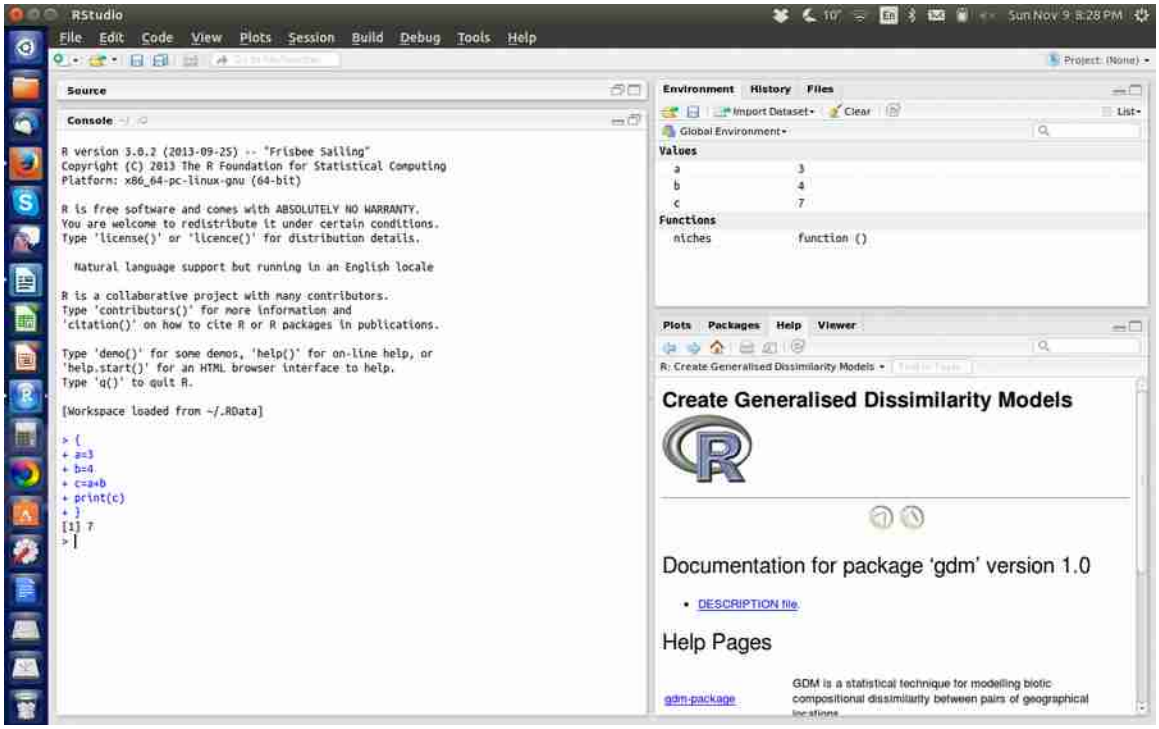
εντολή 1

εντολή 2

…

}

Αμέσως μετά το τέλος της κατασκευής του προγράμματος με την εισαγωγή του }, το πρόγραμμα εκτελείται. Μειονέκτημα του τρόπου αυτού είναι ότι μετά την εκτέλεση, το πρόγραμμα δύσκολα αποθηκεύεται και ανακτάται, για να γίνουν αλλαγές.

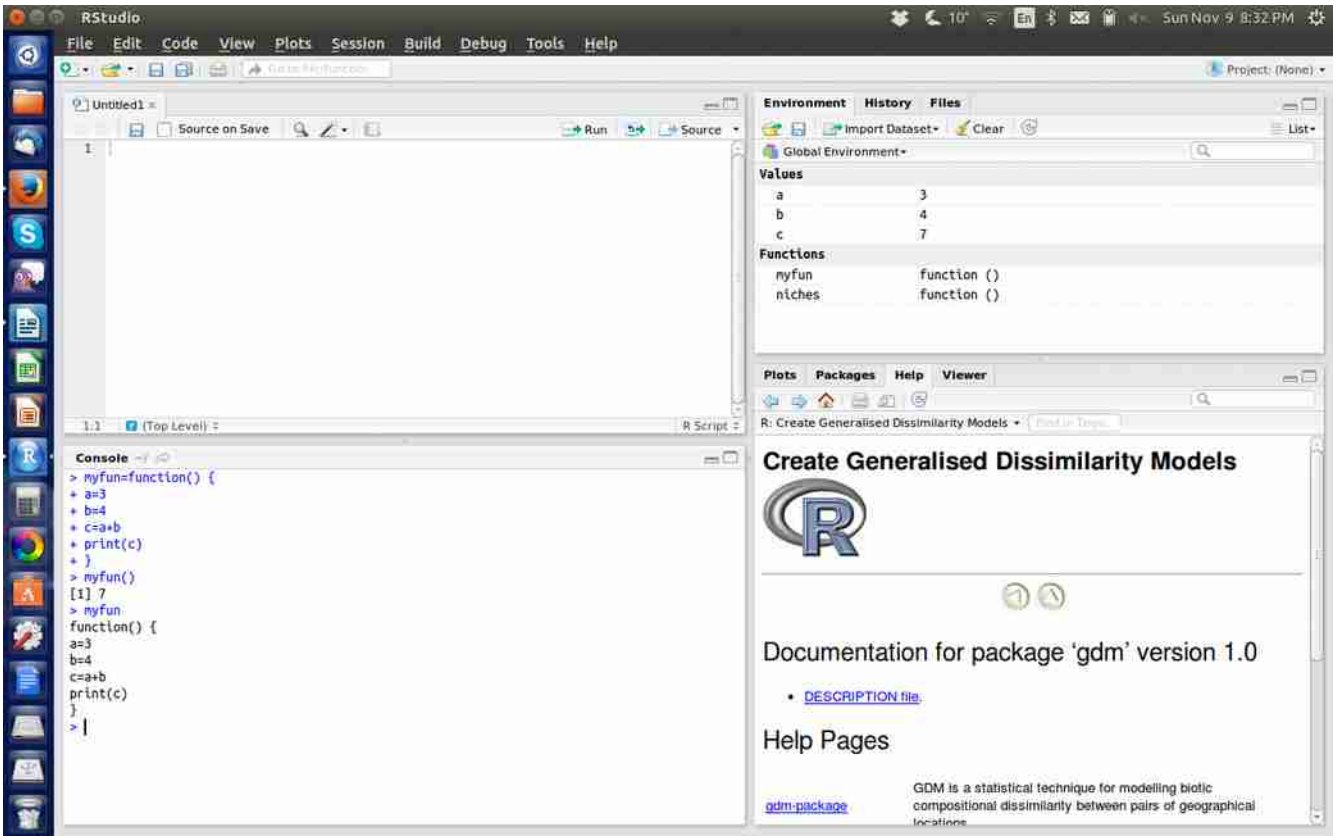


Εικόνα Β1. *Κατασκευή και εκτέλεση απλού προγράμματος στην R χρησιμοποιώντας άγκιστρα {…}.*

2ος τρόπος: Κατασκευάζουμε το πρόγραμμά μας με τη μορφή συνάρτησης (function) της R, ως εξής:

Στην γραμμή εντολών > γράφουμε την εντολή myfun=function() { και πατάμε Enter, όπου myfun το όνομα της συνάρτησης που μπορεί να είναι οποιοδήποτε (π.χ. askisi1, askisi2, test). Στις επόμενες γραμμές εισάγουμε τις εντολές και μετά την τελευταία κλείνουμε το άγκιστρο } (Εικόνα Β2).

```
> myfun=function() {
  εντολή1
  εντολή2
  …
  }
```

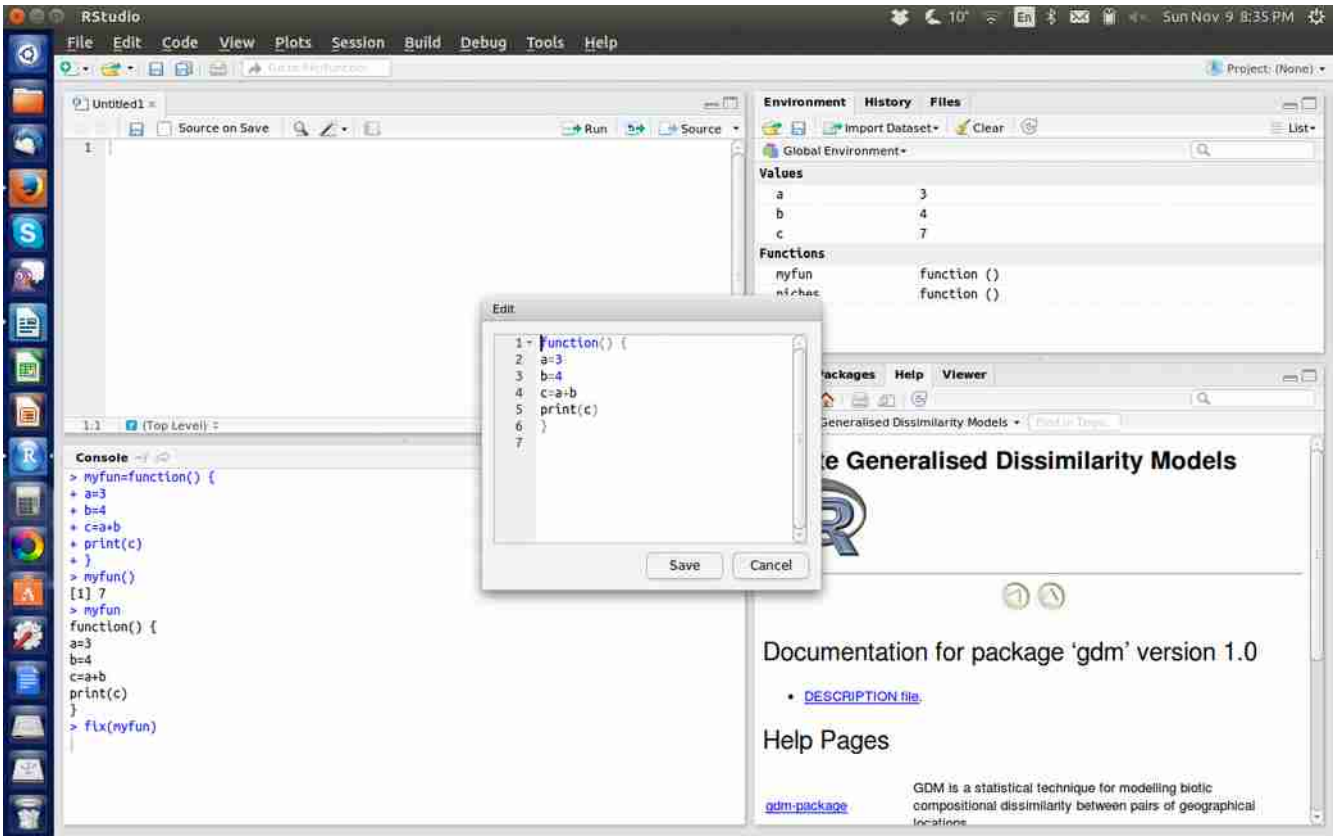


Εικόνα Β2. *Κατασκευή και εκτέλεση προγράμματος στην R χρησιμοποιώντας την εντολή function().*

Για την εκτέλεση του προγράμματος στη συνέχεια πρέπει ο χρήστης να γράψει το όνομά του ακολουθούμενο από παρενθέσεις στη γραμμή εντολών, π.χ. >myfun() (Εικόνα Β2).

Αν γράψει >myfun χωρίς παρενθέσεις θα εμφανιστεί ο κώδικας αλλά δεν θα εκτελεστεί το πρόγραμμα (Εικόνα Β2), ενώ με την εντολή >fix(myfun) ανοίγει το πρόγραμμα για επεξεργασία

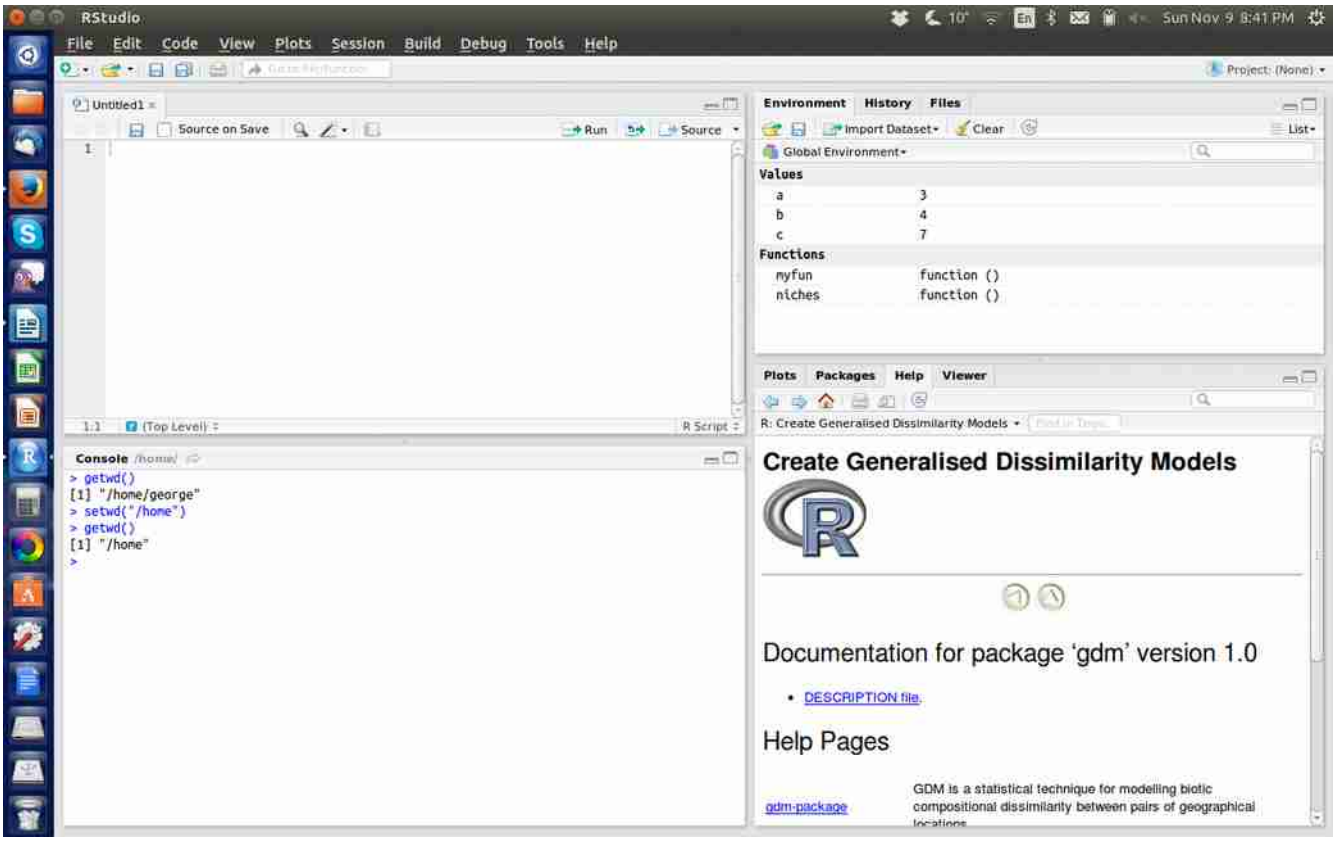
(προσθήκη, αλλαγή εντολών) σε έναν φιλικό editor (Εικόνα Β3). Στον editor μπορούν να γίνουν οι οποιεσδήποτε αλλαγές στον κώδικα του προγράμματος και να αποθηκευθούν στην συνέχεια πατώντας Save.



Εικόνα Β3. *Επεξεργασία του προγράμματος με την εντολή fix(onoma_tou_programmatos) σε έναν φιλικό editor.*

Για να αποθηκευθεί το πρόγραμμα μόνιμα σε κάποιο αποθηκευτικό μέσο (σκληρό δίσκο, USB stick), ώστε να μπορεί να ανακτηθεί αργότερα προς επεξεργασία, πρέπει πρώτα να οριστεί ο φάκελος μέσα στο οποίο θα γίνει η αποθήκευση. Κάθε στιγμή η R (ή το RStudio) χρησιμοποιεί ένα φάκελο εργασίας (working directory) μέσα στον οποίο γράφει ή από τον οποίο διαβάζει προγράμματα ή δεδομένα. Για να ελέγξουμε στην R ποιος είναι ο τρέχων φάκελος εργασίας γράφουμε την εντολή >getwd() (Εικόνα Β4). Για να αλλάξουμε τον φάκελο εργασίας και να καθορίσουμε νέο, γράφουμε την εντολή >setwd("C:\\temp\\test") όπου ανάμεσα στα εισαγωγικά είναι ένα παράδειγμα διαδρομής στον σκληρό δίσκο. Επισημαίνεται ότι στα MS Windows χρησιμοποιείται διπλή κάθετος \\ για να οριστούν οι φάκελοι και υποφάκελοι, ενώ στα Linux, Unix η ανάποδη κάθετος /, π.χ. /usr/temp. Η επιλογή του

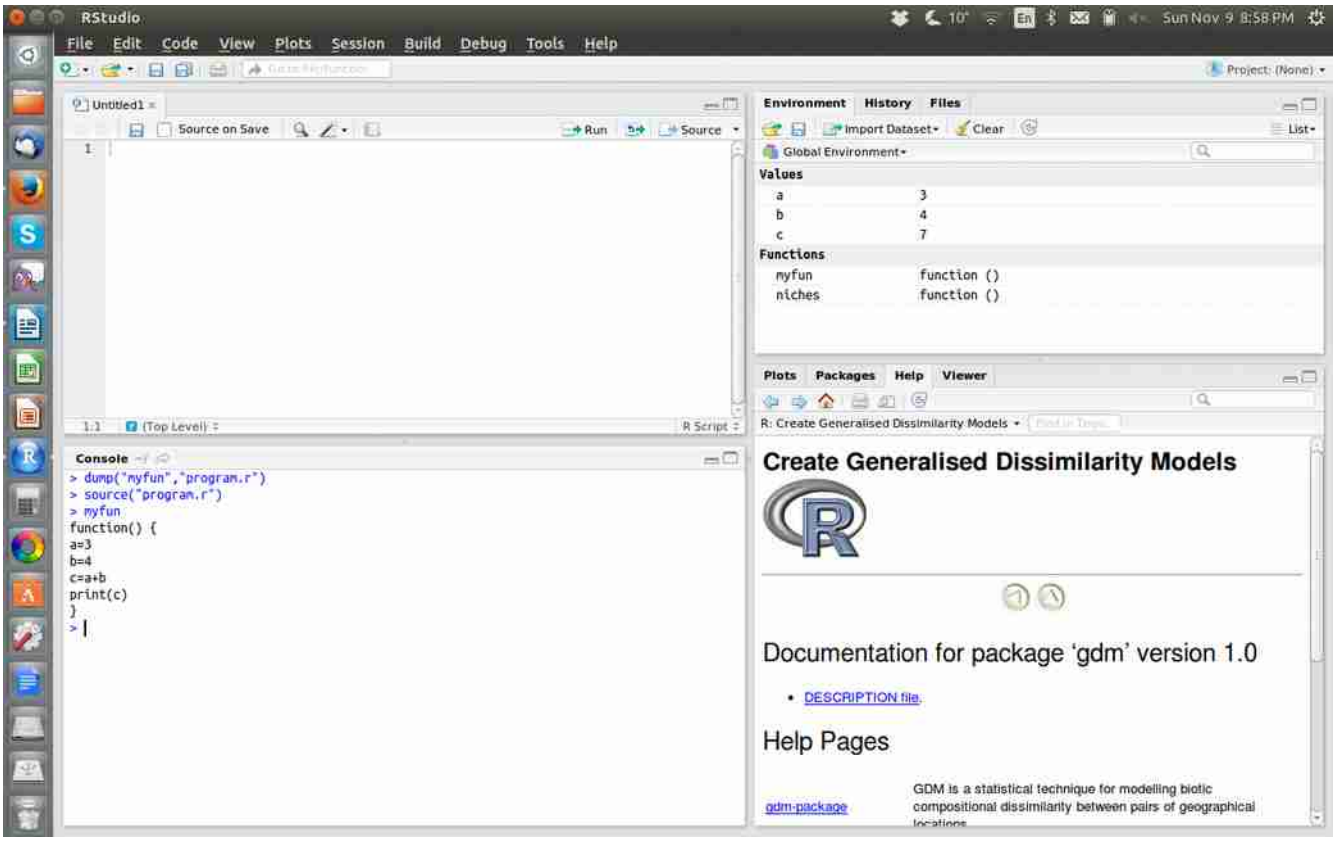
φακέλου εργασίας μπορεί να γίνει πολύ ευκολότερα στο RStudio, χωρίς να χρησιμοποιηθούν οι παραπάνω εντολές, από το menu με την επιλογή Session-Set Working Directory, Choose Directory.



Εικόνα Β4. *Εντολές για την εύρεση και αλλαγή του τρέχοντος φακέλου εργασίας στην κονσόλα της R.*

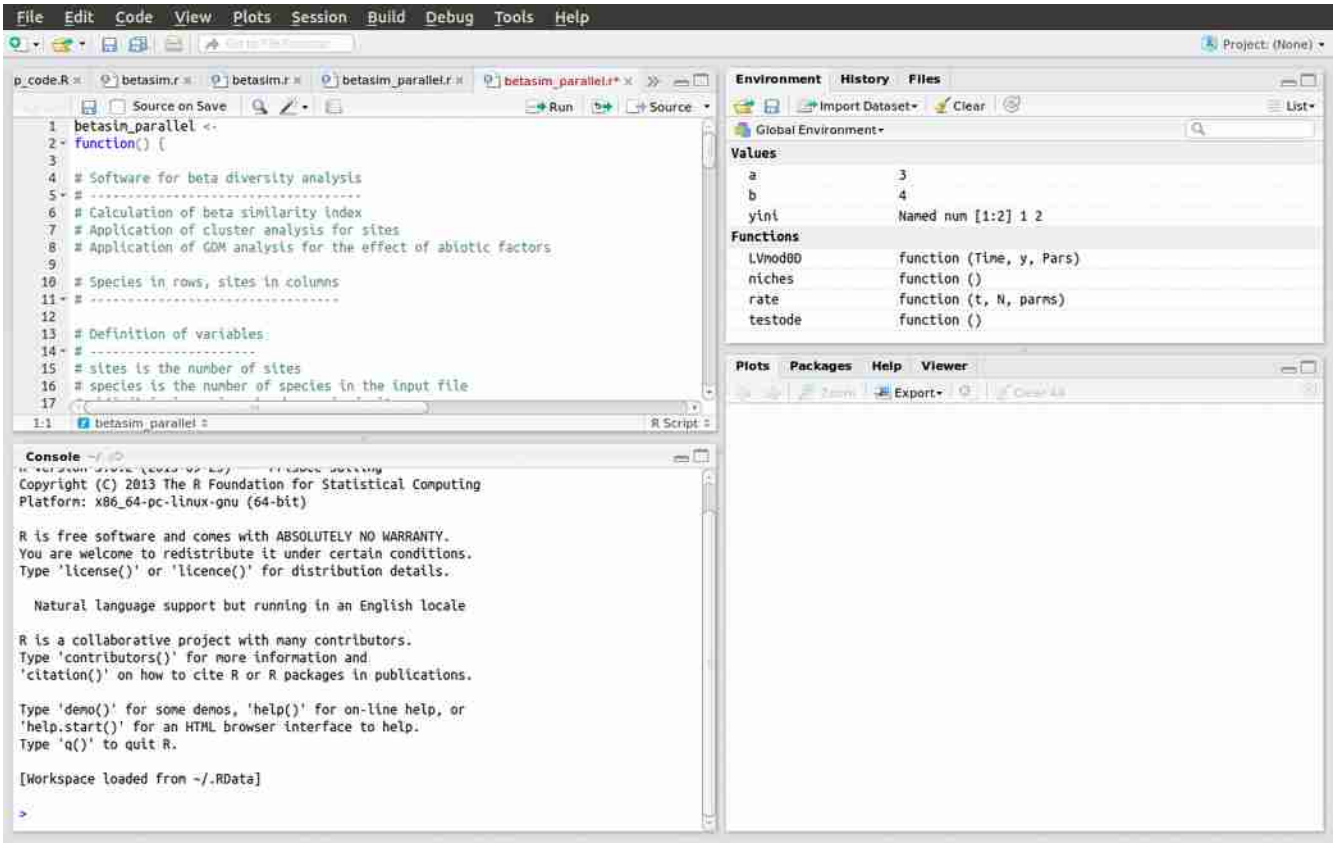
Αφού ορίσουμε λοιπόν ως φάκελο εργασίας τον φάκελο μέσα στον οποίο θέλουμε να αποθηκεύσουμε το πρόγραμμα-συνάρτηση που κατασκευάσαμε, γράφουμε την εντολή dump("myfun","program.r"), όπου myfun το πρόγραμμα-συνάρτηση που κατασκευάσαμε και program.r, παράδειγμα ονόματος αρχείου στο οποίο θα αποθηκευθεί το πρόγραμμα-συνάρτηση. Το program είναι ένα τυχαίο όνομα (μπορεί να είναι οποιοδήποτε όνομα) αλλά η προέκταση πρέπει να είναι .r. Όταν θέλουμε αργότερα να εργαστούμε με συνάρτηση-πρόγραμμα που έχουμε αποθηκεύσει, ορίζουμε αρχικά ως φάκελο εργασίας τον φάκελο μέσα στον οποίο βρίσκεται το πρόγραμμα με την εντολή setwd() στην γραμμή εντολών της R ή με την επιλογή Session-Set Working Directory, Choose Directory στο μενού του RStudio. Στην συνέχεια επιλέγουμε και ενεργοποιούμε το πρόγραμμα-συνάρτηση για επεξεργασία ή εκτέλεση με την εντολή >source("program.r") στην γραμμή εντολών της R (ή του RStudio), όπου program.r το όνομα του αρχείου. Η επεξεργασία του

προγράμματος μπορεί να γίνει με την εντολή >fix(myfun) και η εκτέλεση με την εντολή >myfun(), όπου myfun το όνομα της συνάρτησης (Εικόνα Β5).



Εικόνα Β5. *Χρήση των εντολών dump() για αποθήκευση προγράμματος-συνάρτησης στον φάκελο εργασίας σε μορφή αρχείου με προέκταση .r και ανάκτηση του αρχείου σε μορφή συνάρτησης με την εντολή source().*

Εναλλακτικά το αποθηκευμένο πρόγραμμα-συνάρτηση μπορεί να ανακτηθεί προς εκτέλεση ή επεξεργασία στο RStudio, μέσω του μενού. Αφού οριστεί ο φάκελος που βρίσκεται το αρχείο μέσω της επιλογής Session-Set Working Directory, Choose Directory όπως αναφέρθηκε παραπάνω, ανοίγεται το αρχείο με το πρόγραμμα (π.χ. program.r) με τη επιλογή File-Open file του μενού. Τότε ο κώδικας (οι εντολές του προγράμματος) εμφανίζονται στο παράθυρο Source (συνήθως πάνω αριστερά). Μέσα στο παράθυρο αυτό είναι δυνατή η επεξεργασία του προγράμματος, όπως στον editor (Εικόνα Β6). Για να εκτελεστεί όμως το πρόγραμμα από την γραμμή εντολών με το όνομά του (π.χ. >myfun()), πρέπει πρώτα να μεταφερθεί στη γραμμή εντολών πιέζοντας το πλήκτρο Source που βρίσκεται πάνω δεξιά στο παράθυρο Source (Εικόνα Β6).



Εικόνα Β6. *Ανάκτηση και εμφάνιση του προγράμματος προς επεξεργασία στο παράθυρο Source του RStudio (πάνω δεξιά).*

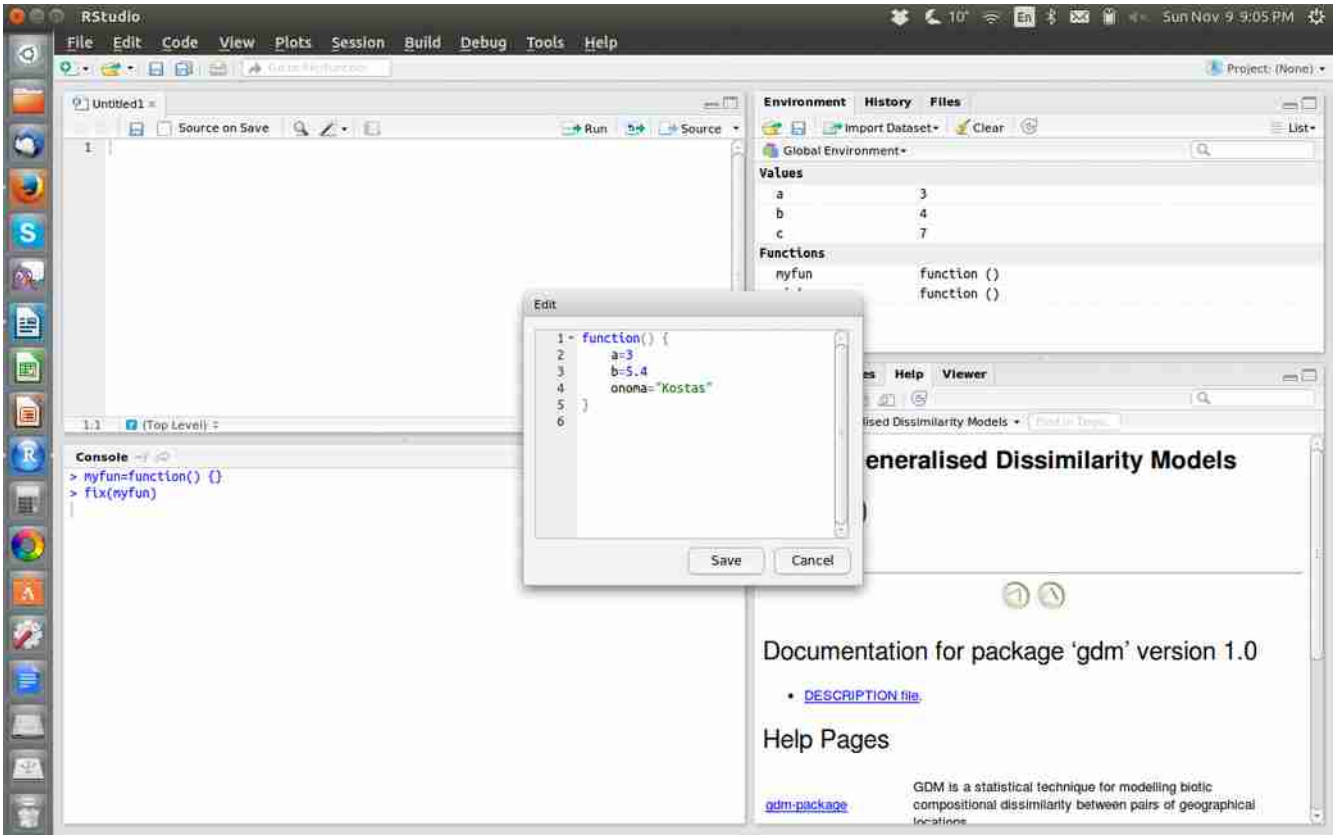
Συνοψίζοντας, από τους δύο τρόπους κατασκευής προγράμματος στην R που περιγράφηκαν στην αρχή της ενότητας, προτείνεται ανεπιφύλακτα ο δεύτερος. Συγκεκριμένα γράφοντας στην γραμμή εντολών >myfun=function() {} κατασκευάζεται και αποθηκεύεται προσωρινά μία κενή συνάρτηση, η οποία στην συνέχεια ανοίγει για επεξεργασία με την εντολή >fix(myfun) σε έναν φιλικό editor. Συνιστάται κατά την κατασκευή του προγράμματος, αυτό να αποθηκεύεται ανά διαστήματα με τους τρόπους που περιγράφηκαν αναλυτικά παραπάνω.

Αποθήκευση της εργασίας που έγινε στην R είναι επίσης δυνατή κατά το κλείσιμο του RStudio, αν στην επιλογή Save workspace image to ~Rdata, ο χρήστης επιλέξει Save. Η αποθήκευση με τον τρόπο αυτό γίνεται σε τοπικό φάκελο του υπολογιστή που εργαζόμαστε.

Β2. Απόδοση τιμών σε μεταβλητές της R

Στην R όπως και σε κάθε γλώσσα προγραμματισμού, χρησιμοποιούνται μεταβλητές για την καταχώρηση τιμών. Οι συνηθέστερες μορφές μεταβλητών είναι οι αριθμητικές-numeric που παίρνουν τιμές πραγματικών αριθμών (π.χ. 3, 4.2, -67.8) και οι αλφαριθμητικές ή χαρακτήρων που παίρνουν ως τιμές ονόματα που μπορεί να περιλαμβάνουν χαρακτήρες ή αριθμούς (π.χ. kostas, abc, a13). Τα ονόματα των μεταβλητών πρέπει να αρχίζουν από γράμμα και μπορεί να περιλαμβάνουν αριθμούς και το σύμβολο της κάτω παύλας (_ underscore).

Με τις απλές εντολές a=3 ή b=5.4 αποδίδονται στις μεταβλητές a και b (τυχαία ονόματα) οι αριθμητικές τιμές 3 και 5.4 αντίστοιχα, από τον προγραμματιστή κατά την διάρκεια κατασκευής του προγράμματος. Ομοίως με την εντολή onoma="kostas" αποδίδεται στην αλφαριθμητική μεταβλητή (μεταβλητή χαρακτήρων) η τιμή kostas (Εικόνα Β7).



Εικόνα Β7. *Απόδοση τιμών σε αριθμητικές και αλφαριθμητικές μεταβλητές από τον προγραμματιστή σε πρόγραμμα σε R.*

Αν ο προγραμματιστής επιθυμεί οι παραπάνω τιμές να δίδονται από τον χρήστη κατά την διάρκεια

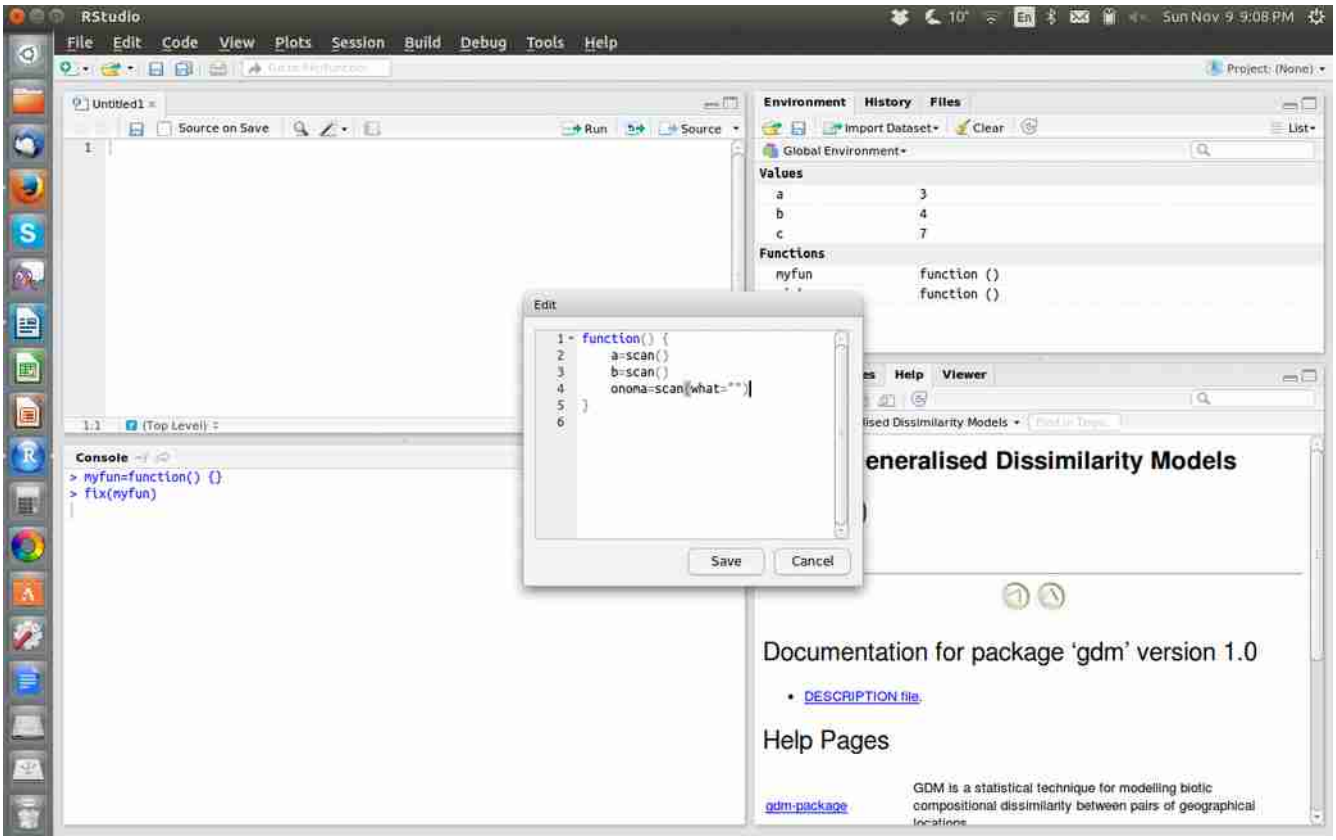
εκτέλεσης του προγράμματος γράφει τις εντολές:

a=scan() για αριθμητική μεταβλητή

b=scan() για αριθμητική μεταβλητή

onoma=scan(what="") για μεταβλητή χαρακτήρων (Εικόνα Β8)

οπότε κατά την εκτέλεση του προγράμματος, αυτό σταματά και περιμένει από τον χρήστη να πληκτρολογήσει τις τιμές. Σημειώνεται ότι όταν ο χρήστης πληκτρολογήσει τη τιμή π.χ. για την a και πατήσει Enter, μετά πρέπει να ξαναπατήσει Enter στο κενό για να πάει στην επόμενη εντολή (Εικόνα Β9).



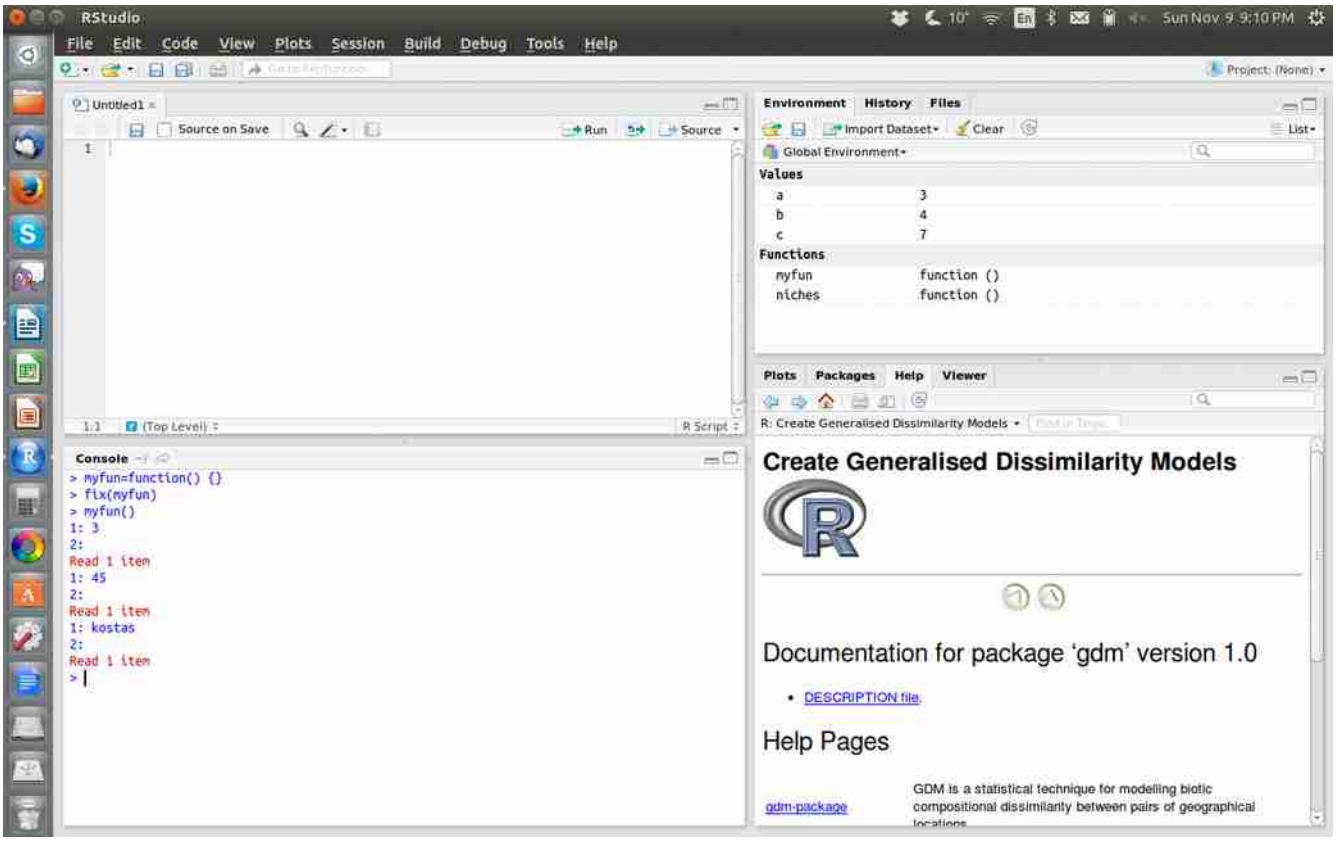
Εικόνα Β8. *Χρήση της εντολής scan() για την απόδοση τιμών σε αριθμητικές και αλφαριθμητικές μεταβλητές από τον χρήστη προγράμματος σε R.*

Σε μεταβλητές της R μπορεί να αποδοθεί τιμή με τη χρήση εντολών εκτέλεσης πράξεων, π.χ.:

c=a+b

c=a*b+a^2

όπου χρησιμοποιούνται τα σύμβολα +, -, *, / για τις τέσσερις βασικές πράξεις (πρόσθεση, αφαίρεση, πολ/σμό, διαίρεση) και το σύμβολο ^ για την δύναμη. Για τις πράξεις υπάρχει προτεραιότητα της ύψωσης σε δύναμη, του πολλαπλασιασμού/διαίρεσης και τελευταία της πρόσθεσης/αφαίρεσης. Αν έχουμε οποιαδήποτε αμφιβολία για την σειρά των πράξεων, χρησιμοποιούμε παρενθέσεις. Στην εκτέλεση πράξεων μπορούν να χρησιμοποιηθούν και ενσωματωμένες συναρτήσεις της R, οι οποίες θα αναλυθούν περισσότερο σε επόμενο κεφάλαιο. Χαρακτηριστικές ενσωματωμένες συναρτήσεις είναι: sqrt(x) τετραγωνική ρίζα του x, abs(x) απόλυτη τιμή του x, sin(x), cos(x) ημίτονο, συνημίτονο του x και πολλές άλλες.



Εικόνα Β9. *Απόδοση τιμών σε αριθμητικές και αλφαριθμητικές μεταβλητές από τον χρήστη κατά την εκτέλεση προγράμματος σε R.*

Β3. Εκτύπωση τιμών μεταβλητών στην οθόνη

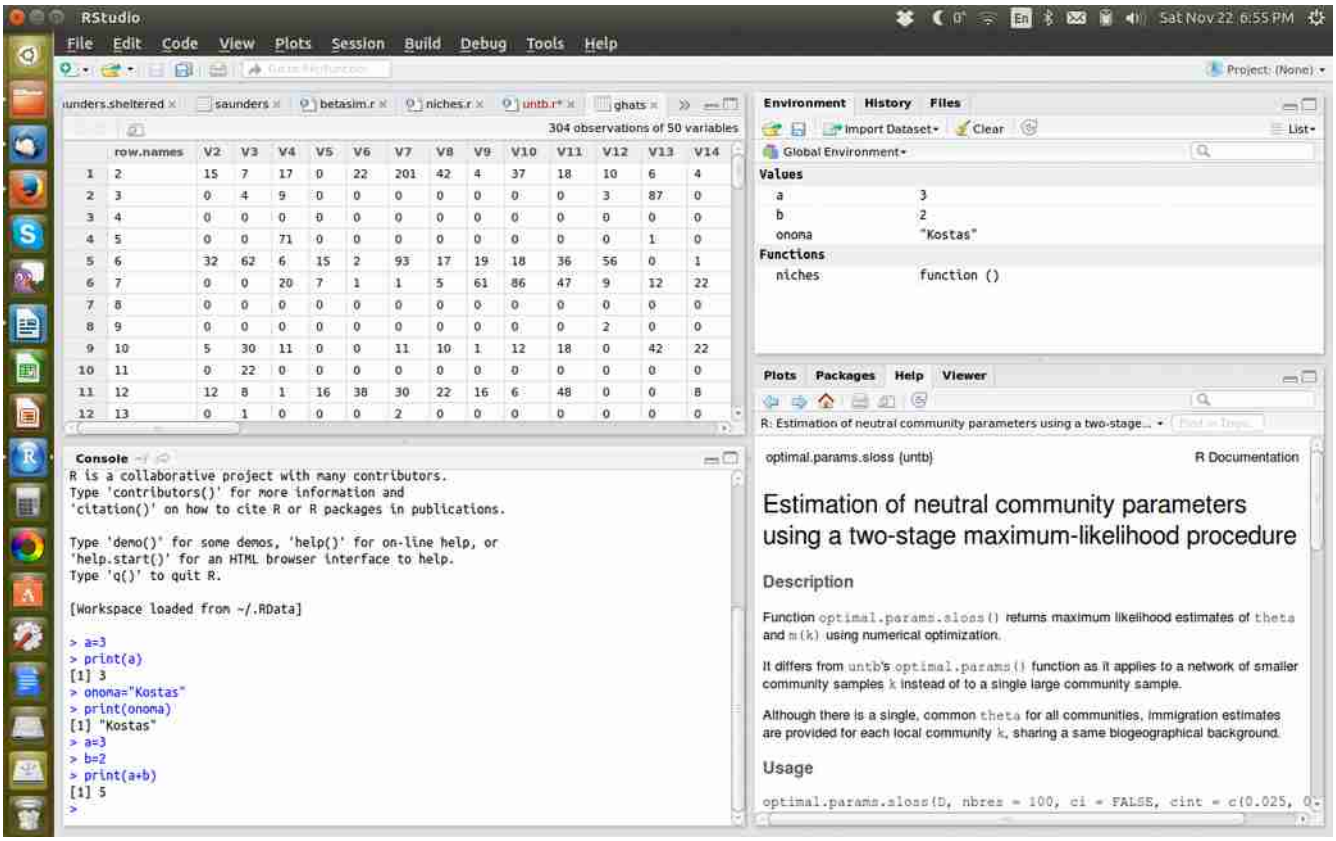
Για την εκτύπωση τιμών μεταβλητών στην οθόνη χρησιμοποιείται συνήθως η εντολή print (Εικόνα Β10). Με την εντολή αυτή εμφανίζεται η τιμή της μεταβλητής είτε είναι αριθμητική, είτε

αλφαριθμητική (κείμενο ή λέξη). Π.χ.

Αν a=3 με την εντολή print(a) θα εμφανιστεί στην οθόνη το 3.

Αν onoma=”kostas” με την εντολή print(onoma) θα εμφανιστεί στην οθόνη “kostas”

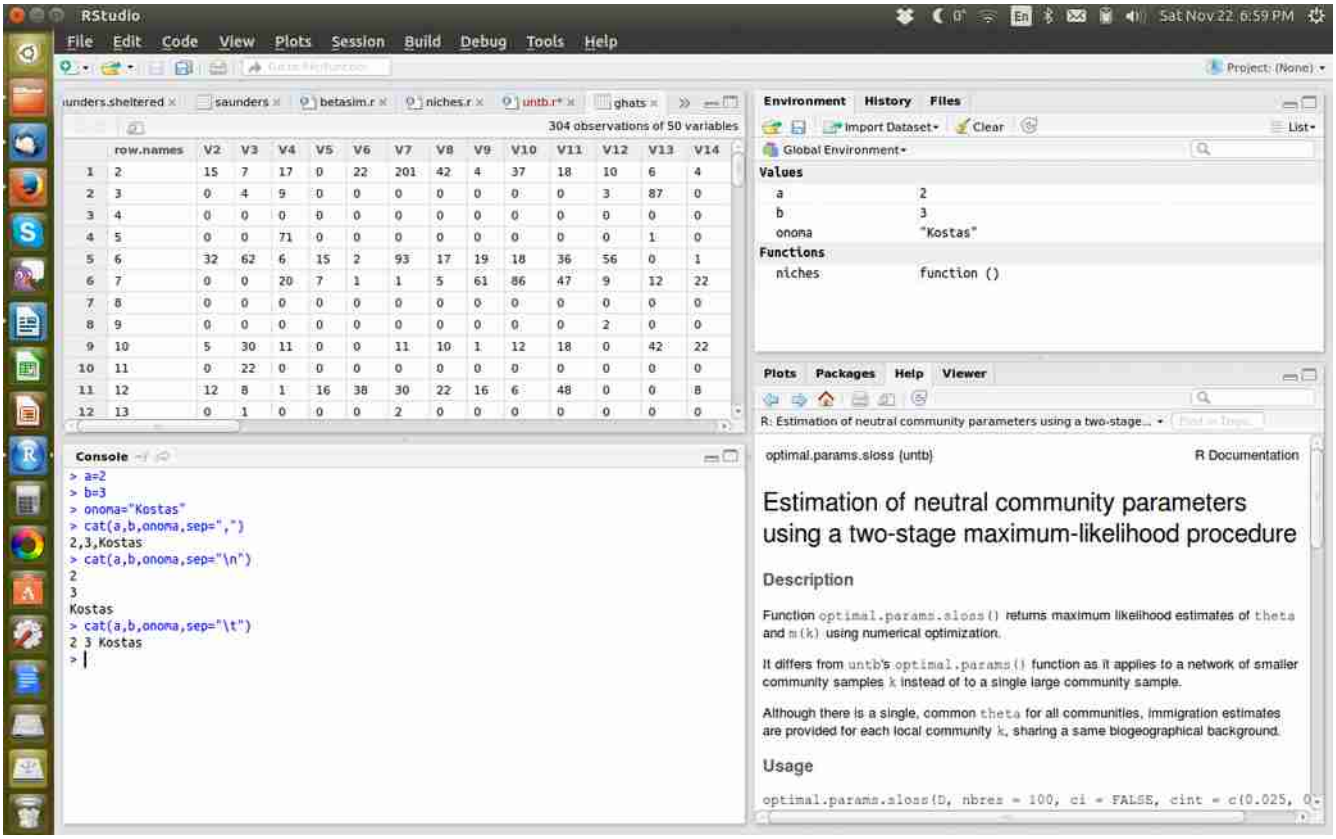
Αν θέλω να τυπώσω κάποιο κείμενο, το γράφω στην παρένθεση μέσα σε εισαγωγικά. Για παράδειγμα αν δώσω την εντολή print(“Define the name of the file”), θα τυπωθεί στην οθόνη το κείμενο “Define the name of the file”



Εικόνα Β10. *Χρήση της εντολής εκτύπωσης print για την εκτύπωση αριθμητικών τιμών και τιμών μεταβλητών χαρακτήρων.*

Εκτός από την εντολή print, διατίθεται και η εντολή cat, η οποία μάλιστα δίνει αρκετές επιπλέον δυνατότητες, με κυριότερη την δυνατότητα εκτύπωσης πολλών τιμών μεταβλητών στην ίδια γραμμή (Εικόνα Β11). Η γενική μορφή σύνταξής της είναι cat(……., sep=”…”), όπου στην αρχή της παρένθεσης τα …… αφορούν τις μεταβλητές που θα τυπωθούν και μετά το sep ανάμεσα στα εισαγωγικά διάφορα σύμβολα που θα περιγραφούν στη συνέχεια. Για παράδειγμα αν x είναι ένα διάνυσμα με τρεις τιμές (3.4, 7.6, 8.2) και y ένα όνομα π.χ. y=”test”, γράφοντας την εντολή cat(x, y) θα γραφεί στην οθόνη σε

μία γραμμή: 3.4 7.6 8.2 test, με κενό μεταξύ τους. Με το όρισμα sep="..." δηλώνεται τι επιθυμούμε να διαχωρίζει τις τιμές. Π.χ. αν δώσω την εντολή cat(x, y, sep=",") θα τυπωθεί στην οθόνη: 3.4,7.6,8.2,test. Άλλα χρήσιμα διαχωριστικά είναι: το sep="\t" που βάζει ένα διαχωριστικό tab ανάμεσα στις τιμές, το sep="\n" που αλλάζει γραμμή μετά από κάθε τιμή που εκτυπώνεται και το sep="\r" που επιστρέφει την επόμενη εκτύπωση πίσω στην αρχή της ίδιας γραμμής, γράφει δηλαδή πάνω από την προηγούμενη εκτύπωση. Η εντολή cat μπορεί να χρησιμοποιηθεί και για την εγγραφή-αποθήκευση τιμών σε αρχείο, όπως θα περιγραφεί παρακάτω.



Εικόνα Β11. *Χρήση της εντολής εκτύπωσης cat για την εκτύπωση αριθμητικών τιμών και τιμών μεταβλητών χαρακτήρων με διάφορες επιλογές του ορίσματος sep.*

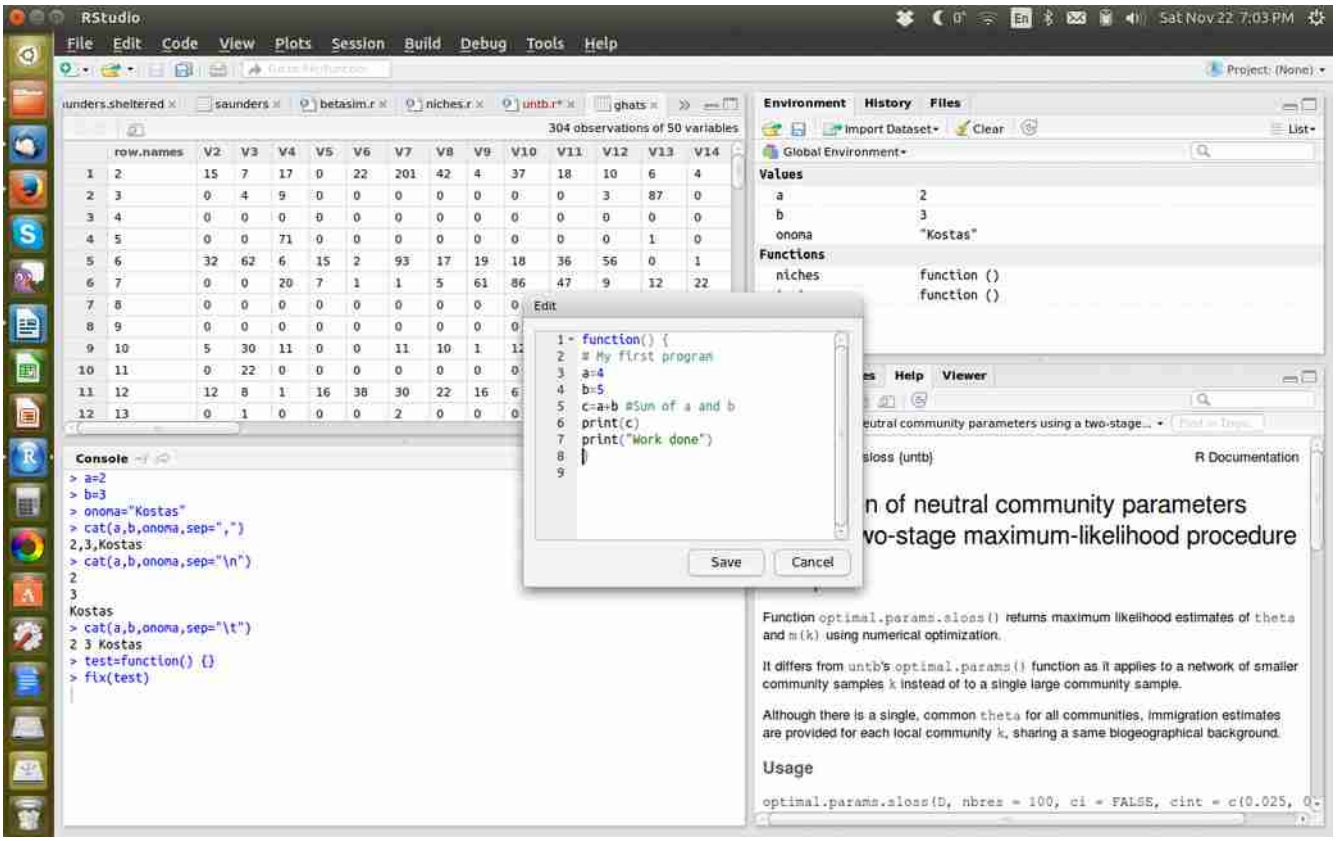
Β4. Εισαγωγή σχολίων

Τα σχόλια είναι κείμενα που εισάγονται σε ένα πρόγραμμα, δεν εκτελούνται, δεν είναι ορατά στον χρήστη και απλά δίνουν στον προγραμματιστή χρήσιμα στοιχεία για την δομή του προγράμματος. Σχόλιο μπορεί να αποτελεί μία ολόκληρη γραμμή αν ξεκινάει με #, π.χ. μπορώ να γράψω σε μία

γραμμή # My first program.

Σχόλιο μπορεί να προστεθεί και σε μία γραμμή εντολής, αν μετά την εντολή γραφεί το σύμβολο #. Π.χ. μπορώ να γράψω σε μία γραμμή: c=a*b # Product of a and b. Εκτελείται κανονικά η εντολή c=a*b και ακολουθεί το μη εκτελέσιμο σχόλιο (Εικόνα Β12).

Συνιστάται κατά την κατασκευή προγραμμάτων η εισαγωγή σχολίων που υπενθυμίζουν στον προγραμματιστή στοιχεία για την ροή του προγράμματός του.



Εικόνα Β12. *Σχόλια μέσα σε ένα πρόγραμμα για τη διευκόλυνση του προγραμματιστή.*

Παραδείγματα απλών προγραμμάτων σε R στα οποία χρησιμοποιούνται οι βασικές εισαγωγικές εντολές που περιγράφηκαν στο κεφάλαιο Β είναι τα ακόλουθα:

Παράδειγμα 1. Δίδονται από το πληκτρολόγιο οι τιμές των μεταβλητών a και b, υπολογίζεται και τυπώνεται στην οθόνη το άθροισμά τους.

```
exampleb1=function() {
  # Example B1 Sum of two numbers
```

```
 print(“Example B1 Sum of two numbers”)
 print(“Define the value of a”)
 a=scan()
 print(“Define the value of b”)
 b=scan()
 c=a+b
 print(“The sum of a and b is”)
 print(c)
}
```

Παράδειγμα 2. Δίδονται τα μήκη των 2 κάθετων πλευρών ενός ορθογωνίου τριγώνου από το πληκτρολόγιο και υπολογίζεται και τυπώνεται στην οθόνη το μήκος της τρίτης πλευράς (υποτείνουσας).

```
exampleb2=function() {
 # Example B2 Calculations on a right triangle
 cat(“Example B2 Calculations on a right triangle”,”\n”)
 cat(“Define the length of side b”,”\n”)
 b=scan()
 cat(“Define the length of side c”,”\n”)
 c=scan()
 a=sqrt(b^2+c^2)
 cat(“The length of side a is ”,c)
}
```

ΑΣΚΗΣΕΙΣ

Β1. Να γραφεί πρόγραμμα σε R στο οποίο θα ζητείται από τον χρήστη η εισαγωγή τριών αριθμητικών τιμών μεταβλητών, και θα υπολογίζονται και τυπώνονται στην οθόνη το άθροισμά τους, η απόλυτη τιμή του αθροίσματος και η τετραγωνική ρίζα του αθροίσματος.

Β2. Να γραφεί πρόγραμμα σε R που να υπολογίζει και τυπώνει στην οθόνη το βάρος συμπαγούς πλαστικού σωλήνα σωλήνα με την χρήση των τύπων: B=d x V, όπου B το βάρος του σωλήνα, d η πυκνότητα του πλαστικού και V ο όγκος του σωλήνα. Ο όγκος του σωλήνα υπολογίζεται απ' την σχέση: $V = \frac{\pi \, x \, Di^2 \, x \, h}{4}$, όπου π=3.14159.., Di η διάμετρος του σωλήνα και h το ύψος του. Τα d, Di και h δίδονται από τον χρήστη.

Γ. ΕΝΤΟΛΕΣ ΕΛΕΓΧΟΥ ΚΑΙ ΛΟΓΙΚΗΣ, ΕΝΤΟΛΕΣ ΕΠΑΝΑΛΗΨΗΣ, ΔΙΑΝΥΣΜΑΤΑ, ΠΙΝΑΚΕΣ

Γ1. Εντολές ελέγχου και λογικής

Οι εντολές ελέγχου και λογικής επιτρέπουν τη λήψη αποφάσεων και τη διακλάδωση του προγράμματος με βάση την ισχύ ή όχι κάποιας συνθήκης. Οι εντολές ελέγχου και λογικής θεωρούνται οι κομψότερες εντολές στον προγραμματισμό και μεταφέρουν στοιχεία της ανθρώπινης κρίσης και λογικής στη μηχανή-ΗΥ.

Οι βασικότερες εντολές λήψης απόφασης είναι δύο:

(α) Η if (συνθήκη) εντολή, που σημαίνει αν ισχύει η συνθήκη μέσα στη παρένθεση τότε εκτέλεσε την εντολή που ακολουθεί, π.χ. if(x>2) a=x^2

(β) Η if (συνθήκη1) {

εντολές1

} else if (συνθήκη2) {

εντολές2

} else if (συνθήκη3) {

εντολές3

.....

} else {

εντολές6

}

Αν ισχύει η συνθήκη1 εκτελούνται οι εντολές1 (μία ή περισσότερες σε διαφορετικές γραμμές), αν η συνθήκη2 οι εντολές2 κλπ, αλλιώς (else) οι εντολές6.

Αν ο έλεγχος αφορά την ισχύ μίας μόνο συνθήκης παίρνει την απλούστερη μορφή:

if (συνθήκη) {

εντολές

}

και αν έχουμε 2 περιπτώσεις δηλαδή αν ισχύει η συνθήκη κάνε τις εντολές1 αλλιώς κάνε τις εντολές2, παίρνει την μορφή:

if (συνθήκη) {

εντολές1

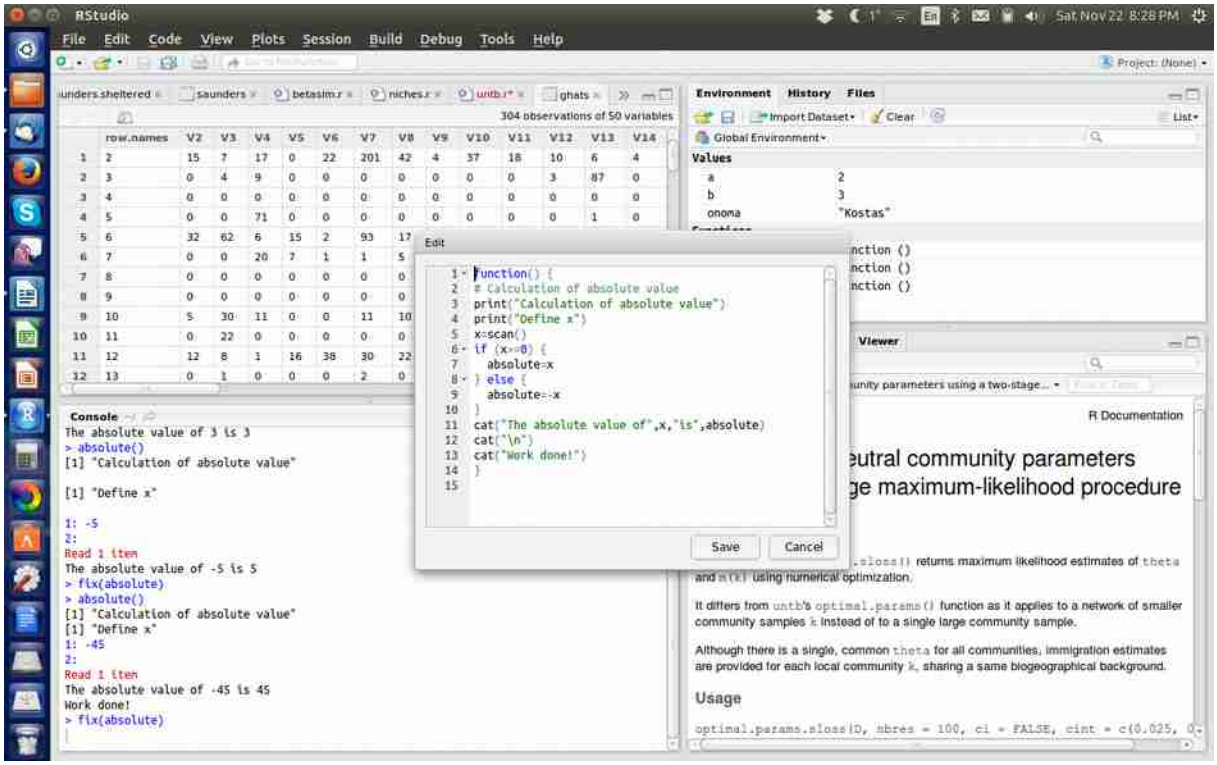
} else {

εντολές2

}

Στις συνθήκες χρησιμοποιούνται συνήθως οι τελεστές ίσον (==, προσοχή στην R για το ίσον μέσα σε συνθήκη γράφω 2 ίσον δίπλα-δίπλα), μεγαλύτερο (>), μικρότερο (<), μεγαλύτερο ή ίσο (>=), μικρότερο ή ίσο (<=) και διάφορο (!=). Μπορούν να χρησιμοποιηθούν και σύνθετες συνθήκες με χρήση συνήθως των τελεστών AND (&) και OR (|).

Π.χ. Η συνθήκη x μεταξύ 2 και 3 γράφεται στην R: if(x>2 & x<3) και η συνθήκη x<0 ή x>10 γράφεται στην R: if(x<0 | x>10).



Εικόνα Γ1. *Πρόγραμμα για τον υπολογισμό απόλυτης τιμής με τη χρήση της εντολής if, else.*

Ένα ολοκληρωμένο πρόγραμμα υπολογισμού της απόλυτης τιμής που βασίζεται στη χρήση της εντολής if, else δίδεται στην Εικόνα Γ1.

Γ2. Εντολές επανάληψης

Οι εντολές επανάληψης επιτρέπουν την εκτέλεση ομάδας εντολών πολλές φορές, τόσες όσες ορίζει ένας δείκτης (περίπτωση α, εντολή for) ή η ισχύς μίας συνθήκης (περίπτωση β, εντολή while). Αν οι εντολές ελέγχου και λογικής θεωρούνται οι κομψότερες εντολές στον προγραμματισμό, οι εντολές επανάληψης είναι οι περισσότερο δυναμικές, εφόσον αφορούν στη μαζική εκτέλεση μεγάλου αριθμού ενεργειών (πράξεων) σε μικρό χρόνο και χωρίς λάθη.

(α) Εντολή for

```
for (i in 1:100) {
  εντολές
}
```

Οι εντολές μέσα στα άγκιστρα εκτελούνται για π.χ. 100 φορές με τον δείκτη i να είναι 1 στον πρώτο κύκλο-επανάληψη, 2 στον δεύτερο κλπ. Μετά την ολοκλήρωση της εκτέλεσης, το πρόγραμμα μεταβαίνει στην εντολή που βρίσκεται αμέσως μετά το άγκιστρο τέλους του for. Ο δείκτης i μπορεί να χρησιμοποιηθεί και ως μεταβλητή στις εντολές. Αν θέλουμε το βήμα αύξησης του δείκτη να μην είναι 1 αλλά π.χ. 2, γράφουμε for (i in seq(1,100,2)), οπότε ο δείκτης ξεκινάει από το 1 και αυξάνεται με βήμα 2, δηλαδή είναι 1 στον πρώτο κύκλο, 3 στον δεύτερο, 5 στον τρίτο κ.λ.π. μέχρι να φτάσει στο 99 στην περίπτωση αυτή.

(β) Εντολή while

```
while(συνθήκη) {
  εντολές
}
```

Οι εντολές μέσα στα άγκιστρα εκτελούνται όσο ισχύει η συνθήκη. Μόλις πάψει να ισχύει το πρόγραμμα μεταβαίνει στην εντολή ακριβώς κάτω από το άγκιστρο τέλους της while. Η συνθήκη

γράφεται με τον τρόπο που εξηγήθηκε παραπάνω στις εντολές ελέγχου και λογικής.

Γ3. Μεταβλητές με δείκτη (διανύσματα-vectors) ή δείκτες (πίνακες-matrices)

Πολύ συχνά στον προγραμματισμό υπάρχει η ανάγκη μαζικής επεξεργασίας μεταβλητών. Για παράδειγμα αν έχουμε 100 αριθμούς και θέλουμε να βρούμε τον μεγαλύτερο, πρέπει κατ' αρχήν να καταχωρίσουμε τους αριθμούς στην μνήμη με την μορφή μεταβλητών και στην συνέχεια να τις συγκρίνουμε μεταξύ τους για να βρούμε τον μεγαλύτερο. Γενικότερα ομοειδείς μεταβλητές που πρόκειται να τις επεξεργαστούμε μαζικά, τις καταχωρούμε στην μνήμη με την μορφή μεταβλητής με δείκτη ή δείκτες.

Μεταβλητή με δείκτη (ή διάνυσμα-vector) είναι μία μεταβλητή x[i] όπου x το όνομά της και i ένας δείκτης που μπορεί να παίρνει διάφορες ακέραιες τιμές π.χ. x[1], x[2], x[3] κ.λ.π. Κάθε μία από τις μεταβλητές αυτές είναι μία διαφορετική μεταβλητή και έχει τη δική της θέση στη μνήμη. Απλά έχουν κοινό όνομα και μεταβάλλοντας τον δείκτη i μπορεί κάποιος να τις επεξεργαστεί μαζικά.

Στην R γράφοντας μία εντολή που αναφέρει μόνο το κοινό όνομα της μεταβλητής (π.χ. x), γίνεται αναφορά σε όλες τις μεταβλητές (δηλ. x[1], x[2], x[3] κ.λ.π.). Π.χ. γράφοντας max(x) βρίσκεται ο μεγαλύτερος όλων των x[1], x[2], x[3] κ.λ.π. ή γράφοντας print(x) τυπώνονται στην οθόνη σε μία σειρά οι τιμές των x[1], x[2], x[3] κ.λ.π.

Μεταβλητή με δείκτες (ή πίνακας-matrix) είναι μία μεταβλητή x[i,j] όπου x το όνομά της και i, j δείκτες που μπορεί να παίρνουν διάφορες ακέραιες τιμές π.χ. x[1,1], x[2,3], x[3,1] κ.λ.π. Επειδή μία τέτοια μεταβλητή έχει τις τιμές της σε μορφή πίνακα, ο πρώτος δείκτης i δηλώνει την γραμμή και ο δεύτερος j την στήλη του πίνακα. Όπως και στη περίπτωση μεταβλητής-διάνυσμα, με μία εντολή που αναφέρει μόνο το κοινό όνομα της μεταβλητής (π.χ. x), γίνεται αναφορά σε όλες τις μεταβλητές. Π.χ. γράφοντας max(x) βρίσκεται ο μεγαλύτερος όλων των x[i,j] ή γράφοντας print(x) τυπώνονται στην οθόνη σε σειρές και στήλες όλες οι τιμές των x[i,j], δηλαδή ολόκληρος ο πίνακας.

Αν θέλουμε να αναφερθούμε σε όλες τις τιμές μίας γραμμής (π.χ. της δεύτερης) γράφουμε x[2,], όπως max(x[2,] ή print(x[2,]). Ομοίως αν θέλουμε να αναφερθούμε σε όλες τις τιμές μίας στήλης (π.χ. της τρίτης) γράφουμε x[,3], όπως max(x[,3] ή print(x[,3]).

Μία μεταβλητή με δείκτες μπορεί, ομοίως με την δεύτερη περίπτωση, να έχει πάνω από δύο διαστάσεις, π.χ. τρεις δηλαδή x[i,j,k].

Για να δώσει ο προγραμματιστής τιμές σε μία μεταβλητή με δείκτη-διάνυσμα μπορεί να χρησιμοποιήσει την εντολή c, π.χ.:

x=c(0.23, 56.7, 3.2) όπου x το όνομα της μεταβλητής με δείκτη και στην παρένθεση οι τιμές της, δηλαδή x[1]=0.23, x[2]=56.7, x[3]=3.2.

Για να δώσει ο χρήστης τιμές σε μία μεταβλητή με δείκτη-διάνυσμα μπορεί επίσης να χρησιμοποιηθεί η εντολή scan(), π.χ.:

a=scan()

Ο χρήστης πληκτρολογεί την πρώτη τιμή π.χ. 3.2 πατάει Enter, μετά την δεύτερη π.χ. 78.9 πατάει Enter, μετά την τρίτη π.χ. -9.3 και όταν τελειώσει πατάει Enter κενό. Τότε θα καταχωρηθούν οι τιμές a[1]=3.2, a[2]=78.9 και a[3]=-9.3.

Οι μεταβλητές με δείκτη δεσμεύουν στη μνήμη έναν αριθμό από θέσεις μνήμης. Π.χ. στα δύο παραδείγματα μεταβλητών διανύσματος παραπάνω, δεσμεύθηκαν 3 θέσεις μνήμης για την μεταβλητή x και 3 για την μεταβλητή a. Ομοίως αν η μεταβλητή έχει πάνω από μία διαστάσεις (μεταβλητή πίνακα) θα δεσμεύσει στη μνήμη τόσες θέσεις όσο το γινόμενο των γραμμών επί τις στήλες της. Στις περιπτώσεις που μεταβλητές με δείκτη (διανύσματα) ή με δείκτες (πίνακες) παίρνουν τιμές με τις εντολές c και scan που αναφέρθηκαν παραπάνω ή οι τιμές τους διαβάζονται από αρχείο με την εντολή read.csv (αμέσως παρακάτω) δεσμεύεται αυτόματα ο απαραίτητος χώρος μνήμης και δεν χρειάζεται να χρησιμοποιηθεί κάποια εντολή γι αυτό. Σε κάθε άλλη περίπτωση, προτού χρησιμοποιηθεί μία μεταβλητή διάνυσμα ή πίνακας μέσα σε ένα πρόγραμμα, πρέπει προηγουμένως να δεσμευθούν γι αυτήν οι απαραίτητες θέσεις στην μνήμη. Αυτό γίνεται με τις εντολές:

x=vector(length=60) όπου π.χ. για την μεταβλητή-διάνυσμα x (με έναν δείκτη) δεσμεύονται 60 θέσεις μνήμης, δηλαδή θέσεις για x[1], x[2], x[3], ..., x[60].

y=matrix(nrow=10, ncol=20) όπου για την μεταβλητή-πίνακα π.χ. y (με δύο δείκτες) δεσμεύονται 10x20=200 θέσεις μνήμης, δηλαδή θέσεις για τιμές y[i,j] σε 10 γραμμές και 20 στήλες. Αν σε ένα

πρόγραμμα δεν γνωρίζουμε εκ των προτέρων τον ακριβή αριθμό θέσεων μνήμης που χρειαζόμαστε για τις μεταβλητές με δείκτη μπορούμε, χωρίς να δημιουργείται κάποιο πρόβλημα, να δεσμεύσουμε μεγαλύτερο αριθμό θέσεων που θεωρούμε επαρκή.

Για την μαζική επεξεργασία τιμών πινάκων χρήσιμη είναι η εντολή apply. Η εντολή χρησιμοποιείται για να γίνει μαζικά πράξη σε όλες τις τιμές κάθε στήλης ή κάθε γραμμής ή όλων των τιμών ενός πίνακα. Για παράδειγμα αν m είναι ένας πίνακας, γράφοντας την εντολή apply(m, 1, mean) υπολογίζεται η μέση τιμή των τιμών κάθε γραμμής του πίνακα m. Γράφοντας apply(m, 2, mean) υπολογίζεται η μέση τιμή των τιμών κάθε στήλης του πίνακα m, ενώ γράφοντας apply(m, 1:2, mean) υπολογίζεται η μέση τιμή όλων των τιμών του πίνακα m. Η mean είναι ενσωματωμένη συνάρτηση της R και υπολογίζει μέση τιμή. Αν θέλουμε να κάνουμε κάποια άλλη πράξη στις τιμές του πίνακα, μπορούμε τη πράξη αυτή να την ορίσουμε ως function, γράφοντας για παράδειγμα: apply(m, 1:2, function(x) x/2). Τότε όλες οι τιμές του πίνακα διαιρούνται με το 2.

ΑΣΚΗΣΕΙΣ

Γ1. Να γραφεί πρόγραμμα σε R που να υπολογίζει και τυπώνει στην οθόνη τις λύσεις τριωνύμου της μορφής $ax^2+bx+c=0$. Στο πρόγραμμα να ζητείται από τον χρήστη να ορίσει τις τιμές των a, b και c (με $a\neq0$), να υπολογίζεται η διακρίνουσα και στη συνέχεια οι πραγματικές ρίζες (μία, δύο ή καμία) ανάλογα με το πρόσημο της διακρίνουσας.

Γ2. Να γραφεί πρόγραμμα σε R στο οποίο να ζητείται από τον χρήστη να ορίσει τη τιμή της μεταβλητής x και να υπολογίζει και τυπώνει στην οθόνη τη τιμή της παράστασης |x-2|+|3-x|.

Γ3. Δύο παίκτες ρίχνουν από ένα ζάρι. Εκείνος που θα φέρει μεγαλύτερο νούμερο κερδίζει έναν πόντο. Αν φέρουν το ίδιο νούμερο ξαναρίχνουν. Αυτό επαναλαμβάνεται 50 φορές. Νικητής είναι εκείνος που στο τέλος θα έχει σημειώσει τους περισσότερους πόντους. Δεν αποκλείεται η περίπτωση της ισοπαλίας. Να γραφεί πρόγραμμα σε R στο οποίο ο χρήστης θα εισάγει τα νούμερα που ρίχνει κάθε παίκτης κάθε φορά και θα υπολογίζει και τυπώνει στην οθόνη τους τελικούς πόντους των δύο παικτών και τον νικητή.

Γ4. Να γραφεί πρόγραμμα σε R στο οποίο θα εισάγονται από τον χρήστη n αριθμοί και θα υπολογίζεται ο μεγαλύτερος, ο μικρότερος και ο μέσος όρος τους.

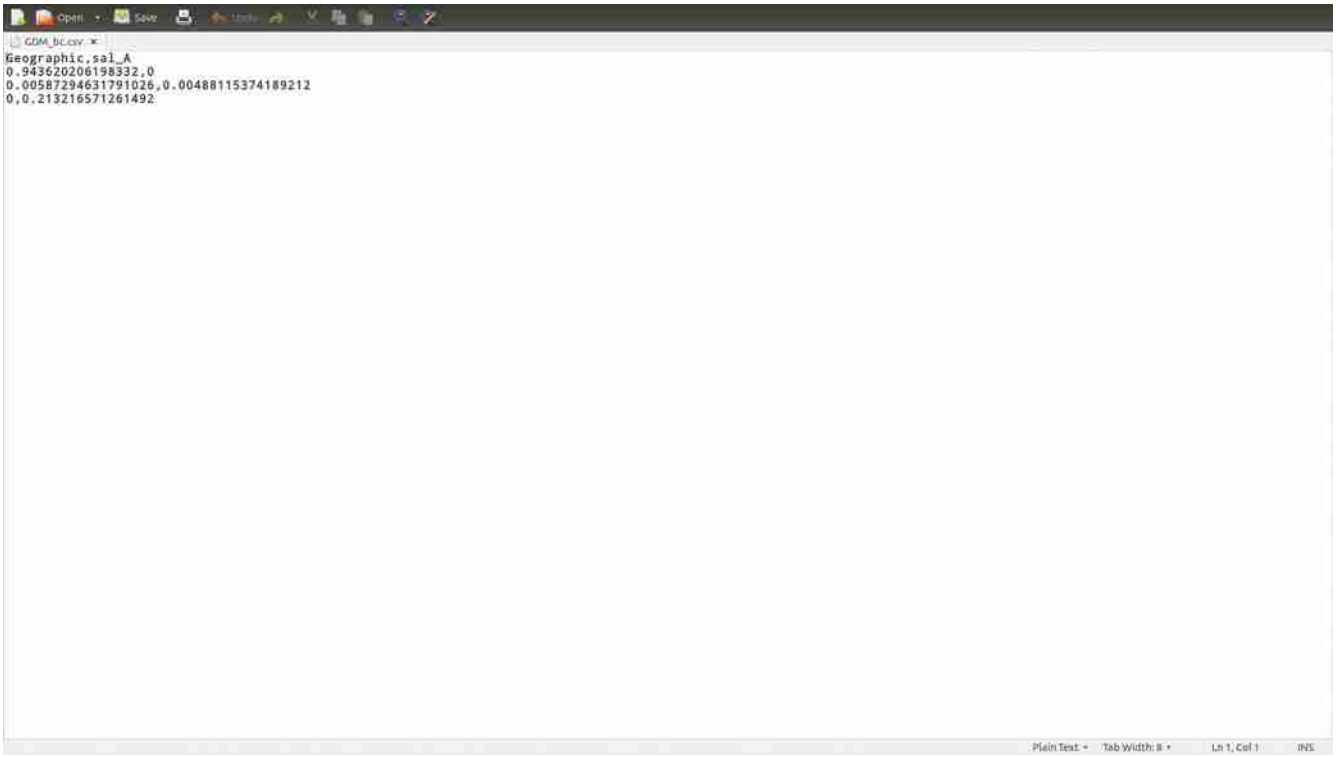
Γ5. Να γραφεί πρόγραμμα σε R για τον υπολογισμό και την εκτύπωση στην οθόνη της τιμής της παράστασης $K=\sum_{J=1}^{15} J^2+\prod_{J=10}^{26} J$.

Γ6. Να γραφεί πρόγραμμα σε R για τον υπολογισμό εμβαδού τριγώνου με μήκος πλευρών a, b και c, όπου a η μεγαλύτερη πλευρά. Αν το τρίγωνο είναι ορθογώνιο, το εμβαδόν υπολογίζεται απ' τη σχέση $E=\frac{b\,x\,c}{2}$, ενώ αν δεν είναι, απ' τη σχέση: $E=\sqrt{t\,x\,(t-a)\,x\,(t-b)\,x\,(t-c)}$, όπου t είναι η ημιπερίμετρος.

Δ. ΑΡΧΕΙΑ ΕΙΣΟΔΟΥ ΔΕΔΟΜΕΝΩΝ ΚΑΙ ΕΞΟΔΟΥ ΑΠΟΤΕΛΕΣΜΑΤΩΝ

Δ1. Ανάγνωση δεδομένων από αρχείο εισόδου

Ο τύπος αρχείου που είναι ευκολότερο να διαβάσει και να γράψει η R είναι τα αρχεία csv (comma separated values ή κείμενο διαχωρισμένο με κόμματα). Τα αρχεία αυτά κατασκευάζονται σε λογισμικό φύλλου εργασίας (π.χ. LibreOffice Calc ή MS Excel) συμπληρώνοντας τιμές στις γραμμές και στήλες ενός φύλλου εργασίας. Στη συνέχεια το αρχείο αποθηκεύεται ως αρχείο τύπου csv (Εικόνα Δ1). Αν ο υπολογιστής έχει ελληνική ρύθμιση, ως δεκαδικό σύμβολο χρησιμοποιείται το κόμμα, οπότε επιλέγουμε ως διαχωριστικό των τιμών των κελιών της ίδιας γραμμής το ελληνικό ερωτηματικό (;). Αν ο υπολογιστής έχει αγγλική ρύθμιση, ως δεκαδικό σύμβολο χρησιμοποιείται η τελεία, οπότε επιλέγουμε ως διαχωριστικό των τιμών των κελιών της ίδιας γραμμής το κόμμα (,). Στην περίπτωση που σε σειρά κελιών είναι γραμμένα ονόματα αντί αριθμών, χρησιμοποιείται ως διαχωριστικό το “”.



Εικόνα Δ1. *Αρχείο csv όπως εμφανίζεται σε έναν κειμενογράφο. Στη πρώτη γραμμή εμφανίζονται οι επικεφαλίδες των 2 στηλών και στις επόμενες γραμμές οι τιμές σε 3 γραμμές και 2 στήλες. Οι τιμές μεταξύ τους διαχωρίζονται με κόμμα και δεκαδικό σύμβολο είναι η τελεία.*

Για να διαβάσει η R τις τιμές ενός αρχείου πρέπει κατ' αρχήν να οριστεί η διαδρομή, ο φάκελος

δηλαδή που περιέχει το αρχείο. Όπως αναφέρθηκε στην Εισαγωγή, κάθε στιγμή η R χρησιμοποιεί ένα συγκεκριμένο φάκελο για να διαβάσει και να γράψει αρχεία που λέγεται working directory (φάκελος εργασίας). Για να ελέγξουμε ποιος είναι ο φάκελος εργασίας γράφουμε στην γραμμή εντολών την εντολή >getwd(). Για να αλλάξουμε το φάκελο εργασίας και να καθορίσουμε νέο στον οποίο βρίσκεται και το αρχείο δεδομένων που θέλουμε να διαβάσουμε γράφουμε την εντολή >setwd("C:\\temp\\test") όπου ανάμεσα στα εισαγωγικά είναι ένα παράδειγμα διαδρομής στον σκληρό δίσκο. Επισημαίνεται ότι στα MS Windows χρησιμοποιείται διπλή κάθετος \\ για να οριστούν οι φάκελοι και υποφάκελοι, ενώ στα Linux, Unix η ανάποδη κάθετος /, π.χ. /usr/temp. Στα ονόματα διαδρομών και αρχείων προτιμότερο είναι να χρησιμοποιούμε απλά ονόματα χωρίς κενά ή σύμβολα και οπωσδήποτε στα αγγλικά.

Αν επιθυμούμε να ορίσει ο χρήστης κατά την διάρκεια εκτέλεσης του προγράμματος τη διαδρομή γράφουμε τις εντολές:

print("Please define the path") ή κάτι αντίστοιχο

workdir=scan(what="")

setwd(workdir) όπου workdir τυχαίο όνομα μεταβλητής στην οποία ο χρήστης καταχωρεί το όνομα της διαδρομής.

Αφού οριστεί η διαδρομή μπορεί ο χρήστης να διαβάσει τα περιεχόμενα του αρχείου και να τα καταχωρίσει στην μεταβλητή με δείκτη με όνομα π.χ. x με την εντολή:

x=read.csv(file="test.csv", sep=";", dec=",", header=FALSE) όπου "test.csv" παράδειγμα ονόματος αρχείου δεδομένων. Το ερωτηματικό στο sep=";" είναι το διαχωριστικό μεταξύ τιμών της ίδιας γραμμής που χρησιμοποιήσαμε κατά την αποθήκευση του αρχείου csv και το κόμμα στο dec="," είναι το δεκαδικό σύμβολο του αρχείου csv. Αν το αρχείο csv έχει αποθηκευθεί σε αγγλική έκδοση και έχει δεκαδικό σύμβολο την τελεία και διαχωριστικό το κόμμα, θα γράφαμε αντίστοιχα:

x=read.csv(file="test.csv", sep=",", dec=".", header=FALSE). Η επιλογή header=FALSE χρησιμοποιείται αν θέλουμε να διαβάζονται δεδομένα από την πρώτη γραμμή του αρχείου. Αν το αρχείο έχει στην πρώτη γραμμή τις επικεφαλίδες των στηλών, αρχίζουμε να διαβάζουμε τιμές δεδομένων από την δεύτερη γραμμή, οπότε γράφουμε header=TRUE.

Αν το αρχείο δεδομένων περιέχει μία σειρά τιμών (ένα διάνυσμα) σε μία γραμμή αυτές θα καταχωρηθούν στις μεταβλητές x[1], x[2], x[3] κλπ. Αν περιέχει πίνακα τιμών (τιμές σε γραμμές και στήλες) αυτές θα καταχωρηθούν στις μεταβλητές x[i,j] όπου i η γραμμή και j η στήλη. Σημειώνεται ότι αν το αρχείο περιέχει τιμές μόνο σε μία στήλη, αυτές θα καταχωρηθούν σε πίνακα (δύο διαστάσεις) ως π.χ. y[1,1], y[2,1], y[3,1] κ.λ.π. με τη δεύτερη διάσταση να είναι 1.

Αν επιθυμούμε να ορίσει ο χρήστης κατά τη διάρκεια εκτέλεσης του προγράμματος το όνομα του αρχείου γράφουμε τις εντολές:

print(“Please define the name of the file”) ή κάτι αντίστοιχο

filename=scan(what=””)

x=read.csv(file=filename, sep=”,”, dec=”.”, header=FALSE) όπου filename τυχαίο όνομα μεταβλητής στην οποία ο χρήστης καταχωρεί το όνομα του αρχείου.

Δ2. Εκτύπωση αποτελεσμάτων σε αρχείο εξόδου

Για να αποθηκεύσουμε αποτελέσματα που βρίσκονται σε μορφή μεταβλητής, διανύσματος ή πίνακα σε αρχείο εξόδου χρησιμοποιούμε την εντολή:

write.table(x, file=”output.csv”, row.names=FALSE, col.names=FALSE) όπου x μεταβλητή την τιμή της οποίας θέλουμε να αποθηκεύσουμε (μπορεί να είναι και πολλές τιμές αν πρόκειται για μεταβλητή με δείκτη), “output.csv” παράδειγμα ονόματος αρχείου εξόδου. Τις επιλογές row.names=FALSE και col.names=FALSE τις χρησιμοποιούμε όταν δεν θέλουμε να αποθηκευθούν στο αρχείο εξόδου οι επικεφαλίδες-ονόματα των γραμμών και στηλών τις οποίες αυτόματα δημιουργεί η R για κάθε μεταβλητή. Αν τις θέλουμε γράφουμε TRUE αντί FALSE.

Και στην περίπτωση αυτή αν επιθυμούμε, μπορεί να ορίσει ο χρήστης κατά την διάρκεια εκτέλεσης του προγράμματος το όνομα του αρχείου γράφοντας τις εντολές:

print(“Please define the name of the output file”)

filename=scan(what=””)

write.table(x, file=filename, row.names=FALSE, col.names=FALSE)

Αν χρησιμοποιήσουμε την εντολή write.table για δεύτερη φορά μέσα σε ένα πρόγραμμα και δώσουμε το ίδιο όνομα αρχείου, η εγγραφή θα γίνει πάνω στα περιεχόμενα του αρχείου, δηλαδή τα προηγούμενα περιεχόμενα θα σβηστούν. Για παράδειγμα αν κάπου σε ένα πρόγραμμα είναι γραμμένη η εντολή write.table(x, file=”output.csv”, row.names=FALSE, col.names=FALSE), θα γραφεί στο αρχείο output.csv η τιμή ή οι τιμές της μεταβλητής x. Αν παρακάτω στο ίδιο πρόγραμμα υπάρχει η εντολή write.table(y, file=”output.csv”, row.names=FALSE, col.names=FALSE) θα γραφεί στο αρχείο output.csv η τιμή ή οι τιμές της μεταβλητής y και θα σβηστούν οι προηγούμενες τιμές x. Αν θέλουμε να κρατηθούν οι τιμές x και να γραφούν οι τιμές της y πιο κάτω (στη συνέχεια), χρησιμοποιούμε την δυνατότητα append=TRUE και γράφουμε την δεύτερη εντολή ως:

write.table(y, file=”output.csv”, append=TRUE, row.names=FALSE, col.names=FALSE)

Εκτός από την παραπάνω εντολή, μπορεί να χρησιμοποιηθεί και η εντολή cat για εκτύπωση τιμών σε αρχείο. Η γενική μορφή σύνταξής της είναι cat(....., file=filename, sep=”...”, append=TRUE), όπου τα ……. στην αρχή της παρένθεσης αφορούν στις μεταβλητές που θα τυπωθούν, το filename είναι το όνομα του αρχείου στο οποίο θα γίνει η εγγραφή, τα …. μετά το sep διάφορα σύμβολα που θα περιγραφούν στη συνέχεια και το append=TRUE ή FALSE την έννοια που έχει και στην write.table. Για παράδειγμα αν x είναι ένα διάνυσμα με τρεις τιμές (3.4, 7.6, 8.2) και y ένα όνομα π.χ. y=”test”, γράφοντας την εντολή cat(x, y, file=”output.csv”, sep=”,”, append=TRUE) θα γραφούν σε μία γραμμή του αρχείου με όνομα output.csv, τα 3.4,7.6,8.2,test, με κόμμα ανάμεσά τους (επειδή sep=”,”) και αν στο αρχείο ήδη υπήρχαν εγγραφές, αυτά θα γραφούν παρακάτω χωρίς να σβήσουν οι υπάρχουσες εγγραφές (επειδή append=TRUE). Όπως αναφέρθηκε προηγουμένως στην χρήση της cat για εκτύπωση στην οθόνη, με το όρισμα sep=”....” δηλώνεται τι επιθυμούμε να διαχωρίζει τις τιμές. Χρήσιμα διαχωριστικά είναι το κόμμα για αρχεία .csv, δηλαδή το όρισμα sep=”,”, το sep=”\t” που βάζει ένα διαχωριστικό tab ανάμεσα στις τιμές, το sep=”\n” που αλλάζει γραμμή μετά από κάθε τιμή που εγγράφεται.

ΑΣΚΗΣΕΙΣ

Δ1. Κατασκευάσετε αρχείο csv σε λογισμικό φύλλων εργασίας (LibreOffice Calc ή MS Excel) που περιέχει πραγματικούς αριθμούς σε m γραμμές και n στήλες (πχ m=5 και n=10). Στην συνέχεια

κατασκευάστε πρόγραμμα σε R στο οποίο αφού διαβάζονται οι αριθμοί από το αρχείο, να καταχωρούνται με την μορφή μεταβλητής με δείκτη και να υπολογίζονται και τυπώνονται στην οθόνη (α) ο μεγαλύτερος, ο μικρότερος και ο μέσος όρος κάθε γραμμής και (β) το άθροισμα των αριθμών κάθε γραμμής.

Δ2. Κατασκευάστε αρχείο csv σε λογισμικό φύλλων εργασίας (LibreOffice Calc ή MS Excel) που περιέχει 10 μη μηδενικούς πραγματικούς αριθμούς. Στην συνέχεια κατασκευάστε πρόγραμμα σε R στο οποίο να διαβάζονται οι αριθμοί από το αρχείο και να καταχωρούνται με την μορφή μεταβλητής με δείκτη. Στην συνέχεια το πρόγραμμα να υπολογίζει και να τυπώνει στην οθόνη (α) τη ρίζα, το τετράγωνο και τον φυσικό λογάριθμο (ln) κάθε αριθμού x, αν ο αριθμός είναι θετικός και (β) την κυβική ρίζα, τον κύβο και το e^x, αν ο αριθμός x είναι αρνητικός.

Δ3. Να γραφεί πρόγραμμα σε R που να διαβάζει από ένα αρχείο τα 50 στοιχεία μεταβλητής με δείκτη $y_1, y_2, y_3, ..., y_{50}$ και να υπολογίζει και τυπώνει στην οθόνη τις παραστάσεις $Z=\sum_{i=1}^{50-2} y_i x y_{i+1} x y_{i+2}$ και $T=\sum_{i=1}^{50}\prod_{j=1}^{i} y_j$.

Δ4. Κατασκευάστε αρχείο φύλλου εργασίας (MS Excel ή LibreOffice Calc) που περιέχει πραγματικούς αριθμούς σε n γραμμές και n στήλες (π.χ. n=5). Να γραφεί πρόγραμμα σε R που να ζητά από τον χρήστη να καθορίσει το n, να διαβάζει τους n x n (π.χ. 25) αριθμούς από το αρχείο και να τους καταχωρεί με την μορφή μεταβλητής με δείκτη. Στην συνέχεια να υπολογίζει και τυπώνει στην οθόνη το άθροισμα και το γινόμενο των διαγώνιων στοιχείων. Διαγώνια είναι τα στοιχεία (οι αριθμοί) που ο αριθμός της γραμμής και ο αριθμός της στήλης που βρίσκονται, ταυτίζονται. Π.χ. για n=5 το ζητούμενο άθροισμα είναι S=x[1,1]+x[2,2]+x[3,3]+x[4,4]+x[5,5] και το γινόμενο Π=x[1,1]x[2,2]x[3,3]x[4,4]x[5,5]. Στο πρόγραμμα τα S και Π να υπολογίζονται για οποιοδήποτε n με την χρήση δομής επανάληψης.

Δ5. Κατασκευάστε αρχείο φύλλου εργασίας (MS Excel ή LibreOffice Calc) που περιέχει πραγματικούς αριθμούς (θετικούς, αρνητικούς και μηδενικά) σε n γραμμές και m στήλες. Να γραφεί πρόγραμμα σε R που να διαβάζει τους n x m αριθμούς από το αρχείο και να τους καταχωρεί με την μορφή μεταβλητής με δείκτη. Στην συνέχεια να υπολογίζει και τυπώνει σε αρχείο εξόδου και στην οθόνη το

πλήθος των θετικών, αρνητικών και μηδενικών αριθμών της κάθε γραμμής.

Ε. ΣΥΝΑΡΤΗΣΕΙΣ-ΥΠΟΠΡΟΓΡΑΜΜΑΤΑ ΣΤΗΝ R

Ε1. Χρήση συναρτήσεων-υποπρογραμμάτων στην R

Έως τώρα χρησιμοποιήθηκε η έννοια της συνάρτησης ως ταυτόσημη του προγράμματος που δημιουργήσαμε σε R, που στην συνέχεια θα ονομάζουμε κυρίως πρόγραμμα για να το ξεχωρίζουμε από τις συναρτήσεις και τα υποπρογράμματα, που θα εξετάσουμε στο κεφάλαιο αυτό με διαφορετική έννοια. Συνήθως όταν λέμε συνάρτηση ή υποπρόγραμμα εννοούμε ένα έτοιμο πρόγραμμα (μία σειρά από εντολές) που κάποιος έχει δημιουργήσει ή που εμείς δημιουργούμε και το έχουμε στη διάθεσή μας για να το χρησιμοποιήσουμε μέσα σε ένα κυρίως πρόγραμμα αναφερόμενοι σ' αυτό με το όνομά του. Π.χ. αν κάπου σε ένα κυρίως πρόγραμμα θέλουμε να υπολογίσουμε την τετραγωνική ρίζα του 5 και την τιμή της να την καταχωρήσουμε στην μεταβλητή b, γράφουμε την εντολή b=sqrt(5). Το sqrt είναι το όνομα μιας ενσωματωμένης (built-in) συνάρτησης που είναι έτοιμη προς χρήση στην R και η τιμή 5 που βάλαμε μέσα στην παρένθεση λέγεται όρισμα (argument) της συνάρτησης. Ομοίως αν θέλουμε να υπολογίσουμε την απόλυτη τιμή της μεταβλητής a και την τιμή της να την καταχωρήσουμε στην μεταβλητή b, γράφουμε στο κυρίως πρόγραμμα την εντολή b=abs(a). Το abs είναι το όνομα της συνάρτησης που υπολογίζει την απόλυτη τιμή. Πίσω από το όνομα κάθε συνάρτησης κρύβεται μία ολόκληρη σειρά από εντολές (συνάρτηση ή υποπρόγραμμα ή υπορουτίνα) που εκτελούν τον σχετικό υπολογισμό.

Οι συναρτήσεις sqrt και abs είναι βασικές συναρτήσεις ενσωματωμένες στην βασική εγκατάσταση της R (base package), ενώ επιπλέον έτοιμες συναρτήσεις για πιο εξειδικευμένες εργασίες περιέχονται στα πακέτα (packages) που είναι διαθέσιμα για κατέβασμα και εμπλουτισμό των δυνατοτήτων της R.

Για να κατασκευάσουμε μια απλή δική μας συνάρτηση, που καλώντας την θα υπολογίζει για παράδειγμα το άθροισμα 2 αριθμών, δημιουργούμε κατ' αρχήν την συνάρτηση με κάποιο όνομα, όπως κάναμε παραπάνω για ένα πρόγραμμα, π.χ. myfunction=function(a,b) {}. Η διαφορά σε σχέση με την χρήση της function που κάναμε για να κατασκευάσουμε ένα κυρίως πρόγραμμα στο κεφάλαιο Β, είναι ότι μέσα στην παρένθεση τώρα υπάρχουν τα a και b που είναι τα ορίσματα της συνάρτησης.

Οι εντολές της συνάρτησης που υπολογίζει το άθροισμα των 2 αριθμών είναι:

```
myfunction=function(a,b) {
c=a+b
```

}

Αν τώρα μέσα στο κυρίως πρόγραμμα γράψω την εντολή g=myfunction(5,6), η μεταβλητή g θα πάρει την τιμή 11. Χρησιμοποιείται δηλαδή η myfunction όπως και οι ενσωματωμένες συναρτήσεις sqrt και abs προηγουμένως.

Η παραπάνω συνάρτηση έχει δύο ορίσματα και επιστρέφει ένα αποτέλεσμα, το άθροισμα των a και b. Μπορούμε να έχουμε όμως επιστροφή και περισσότερων αποτελεσμάτων από μία συνάρτηση και στην περίπτωση αυτή είναι καταλληλότερη η λέξη υποπρόγραμμα ή υπορουτίνα αντί της συνάρτησης. Για παράδειγμα η συνάρτησή μας (υποπρόγραμμα) myfunction μπορεί να υπολογίζει και το άθροισμα και τη διαφορά των a και b. Γράφουμε τότε τις εξής εντολές:

```
 myfunction=function(a,b) {
c=a+b
d=a-b
return(list(out1=c,out2=d))
}
```

όπου out1 και out2 ονόματα των τιμών εξόδου (τα ονόματα είναι τυχαία και βέβαια μπορεί να είναι οτιδήποτε), που εμπεριέχονται σε μία λίστα (list) της R.

Αν τώρα μέσα στο κυρίως πρόγραμμα γράψουμε την εντολή g=myfunction(5,6), το υποπρόγραμμα υπολογίζει για την μεταβλητή g δύο τιμές το άθροισμα 11 και τη διαφορά -1. Το άθροισμα καταχωρείται στην μεταβλητή με όνομα g$out1 και η διαφορά στην μεταβλητή με όνομα g$out2, δηλαδή εδώ g$out1=11 και g$out2=-1. Οι μεταβλητές αυτές μπορούν πλέον να χρησιμοποιηθούν με τα ονόματα αυτά όπως και οι άλλες μεταβλητές του κυρίως προγράμματος, σε πράξεις, υπολογισμούς κ.λ.π.

Σε πολλές περιπτώσεις είναι βολικό οι μεταβλητές εξόδου μίας συνάρτησης-υποπρογράμματος να είναι σε μορφή μεταβλητής με δείκτη. Τροποποιούμε τη συνάρτηση myfynction ως εξής:

```
myfunction=function(a,b) {
c=vector(length=2)
```

```
c[1]=a+b

c[2]=a-b

return(list(out=c))

}
```

όπου οι τιμές εξόδου εμφανίζονται με την μορφή μεταβλητής με δείκτη με το όνομα out (τυχαίο όνομα που μπορεί να είναι οτιδήποτε). Αν δηλαδή μέσα στο κυρίως πρόγραμμα γράψουμε την εντολή g=myfunction(8,2), το υποπρόγραμμα υπολογίζει δύο τιμές για την μεταβλητή g το άθροισμα 10 και τη διαφορά 6. Το άθροισμα καταχωρείται στην μεταβλητή g$out[1] και η διαφορά στην μεταβλητή g$out[2], δηλαδή εδώ g$out[1]=10 και g$out[2]=6. Η μεταβλητή με δείκτη g$out μπορεί να χρησιμοποιηθεί στη συνέχεια όπως και οι άλλες μεταβλητές του κυρίως προγράμματος, σε πράξεις, υπολογισμούς κ.λ.π.

Με τον ίδιο τρόπο που χρησιμοποιούμε μέσα σε ένα κυρίως πρόγραμμα συναρτήσεις (υποπρογράμματα) όπως η myfunction (καλώντας την με το όνομά της, δίνοντας τιμές στα ορίσματά της και παίρνοντας ως επιστροφή τις τιμές εξόδου σε μορφή απλών μεταβλητών ή μεταβλητών με δείκτη), μπορούμε να χρησιμοποιήσουμε τις εκατοντάδες ή και χιλιάδες συναρτήσεις (υποπρογράμματα) που είναι διαθέσιμες στην R. Πληροφορίες γι αυτές μπορούμε εύκολα να βρούμε στο διαδίκτυο σε ότι αφορά στην εργασία που επιτελούν, στα ορίσματα που απαιτούν και στα αποτελέσματα που επιστρέφουν. Οι επιπλέον αυτές συναρτήσεις που επιτελούν πιο εξειδικευμένες εργασίες είναι οργανωμένες σε πακέτα (packages) της R που πρέπει κάποιος να κατεβάσει και να εγκαταστήσει, πέρα από τη βασική έκδοση. Περισσότερες πληροφορίες για την εγκατάσταση πακέτων και την χρήση συναρτήσεων πακέτων δίδονται σε επόμενο κεφάλαιο.

ΣΤ. ΓΡΑΦΙΚΑ ΣΤΗΝ R

Μια πρώτη επαφή με τις δυνατότητες δημιουργίας γραφημάτων στο RStudio μπορεί να πάρει κάποιος πληκτρολογώντας στη γραμμή εντολών:

```
> demo(graphics)

> demo(image)

> demo(persp)
```

Τα γραφήματα εμφανίζονται είτε στο κάτω δεξιά τμήμα της διεπιφάνειας του RStudio (καρτέλα plots, πατώντας Enter δημιουργείται το επόμενο διάγραμμα), είτε σε ξεχωριστό παράθυρο που εμφανίζεται με την εκτέλεση της εντολής (για λειτουργικό windows):

```
> windows()
```

όπου με αριστερό κλικ δημιουργείται το επόμενο διάγραμμα.

Η αναζήτηση βοήθειας για μία εντολή-συνάρτηση (σύνταξη, ορίσματα που δέχεται, κλπ) ή για ένα σύνολο δεδομένων (dataset) γίνεται πληκτρολογώντας την εντολή με ? να προηγείται, π.χ.:

```
> ?plot

> ?points

> ?CO2        #info for R's data set CO2
```

Παραδείγματα για το τι κάνει μία εντολή:

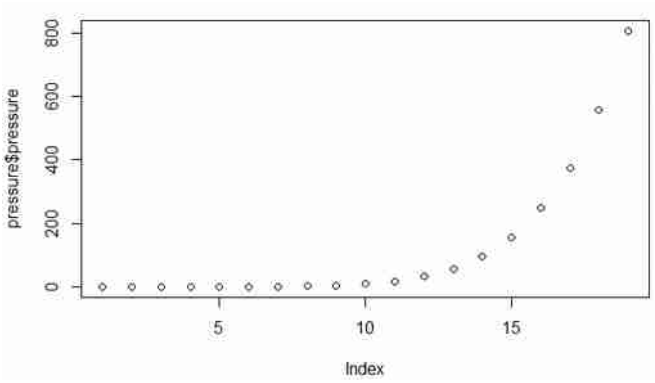
```
> example(plot)

> example(points)

> example(CO2)      #example for R's data set CO2
```

Μια βασική εντολή δημιουργίας γραφήματος στην R είναι η **plot()**. Ανάλογα με το είδος των ορισμάτων μέσα στην παρένθεση, μπορούν να δημιουργηθούν διάφοροι τύποι διαγραμμάτων.

<u>Παράδειγμα 1</u>

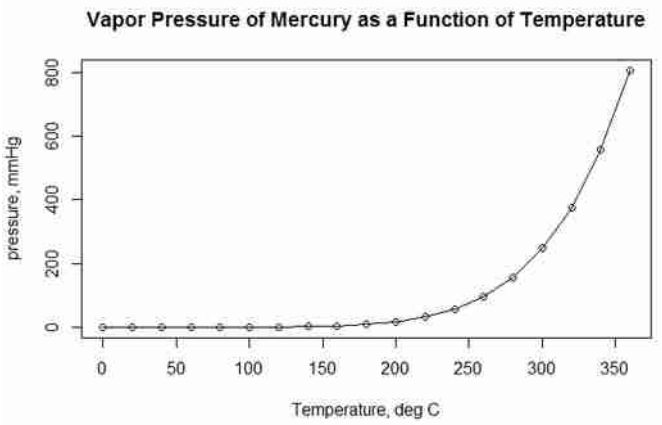
Διάγραμμα διασποράς (scatter plot) μιας μεταβλητής-διανύσματος x ως προς τον δείκτη που έχει κάθε στοιχείο του διανύσματος και μεταβλητής y ως προς μεταβλητή x:

```
> data(pressure)    #loads R's dataset 'pressure'

> pressure          # prints the dataset onscreen

   temperature pressure
1            0   0.0002
2           20   0.0012
3           40   0.0060
4           60   0.0300
5           80   0.0900
6          100   0.2700
7          120   0.7500
8          140   1.8500
9          160   4.2000
10         180   8.8000
11         200  17.3000
12         220  32.1000
13         240  57.0000
14         260  96.0000
15         280 157.0000
16         300 247.0000
17         320 376.0000
18         340 558.0000
19         360 806.0000

> plot(pressure$pressure)

> # produces scatter plot
```



Σχήμα ΣΤ1. *Scatter plot of data column 'pressure' as a function of its elements indices.*

```
> plot(pressure$temperature, pressure$pressure,
type='o', xlab='Temperature, deg  C',
ylab='pressure, mmHg', main="Vapor Pressure of
Mercury as a Function of Temperature")
```



Σχήμα ΣΤ2

Με την τελευταία εντολή, η πίεση ατμών υδραργύρου παριστάνεται συναρτήσει της θερμοκρασίας, ενώ προστίθεται γραμμή που ενώνει τα σημεία, τίτλοι στους άξονες και στο διάγραμμα (Σχήμα ΣΤ2).

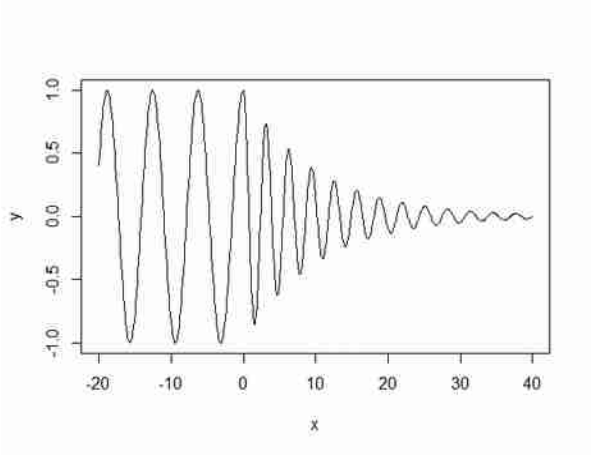
Παράδειγμα 2

Γραφική παράσταση συνάρτησης y ως προς x. Τα 2 διανύσματα x, y πρέπει να έχουν τις ίδιες διαστάσεις. Για παράδειγμα, η συνάρτηση

$$f(x) = \begin{cases} \cos(x)' & -20 < x < 0 \\ e^{(-x/10)}\cos(2x)' & 0 \le x < 40 \end{cases}$$

μπορεί να παρασταθεί γραφικά με τον παρακάτω κώδικα:

```
> x1 = seq(-20,0,by=0.1)
> x2 = seq(0,40,length=500)
> y1 = cos(x1)
> y2 = exp(-x2/10)*cos(2*x2)
> x = c(x1,x2)
> y = c(y1,y2)
> plot(x,y, type='l')
```



Σχήμα ΣΤ3

```
>#Εναλλακτικά χρησιμοποιείται η εντολή
>#plot(y~x, type='l')
```

Βιβλιογραφία και ιστότοποι για εμβάθυνση:

Κεντρικός ιστότοπος της R (manuals > Introduction to R), http://www.r-project.org/

Karline Soetaert and Filip Meysman, 2013, Scientific computing in R,

http://www.rforscience.com/wpmain/wp-content/uploads/2014/05/scienceR.pdf

http://www.rforscience.com/

Φωκιανός Κ. - Χαραλάμπους Χ., 2010, Εισαγωγή στην R – πρόχειρες σημειώσεις,

http://cran.r-project.org/doc/contrib/mainfokianoscharalambous.pdf

Quick-R, http://www.statmethods.net/graphs/

Creating publication quality graphs in R,

http://teachpress.environmentalinformatics-marburg.de/2013/07/creating-publication-quality-graphs-in-r-7/

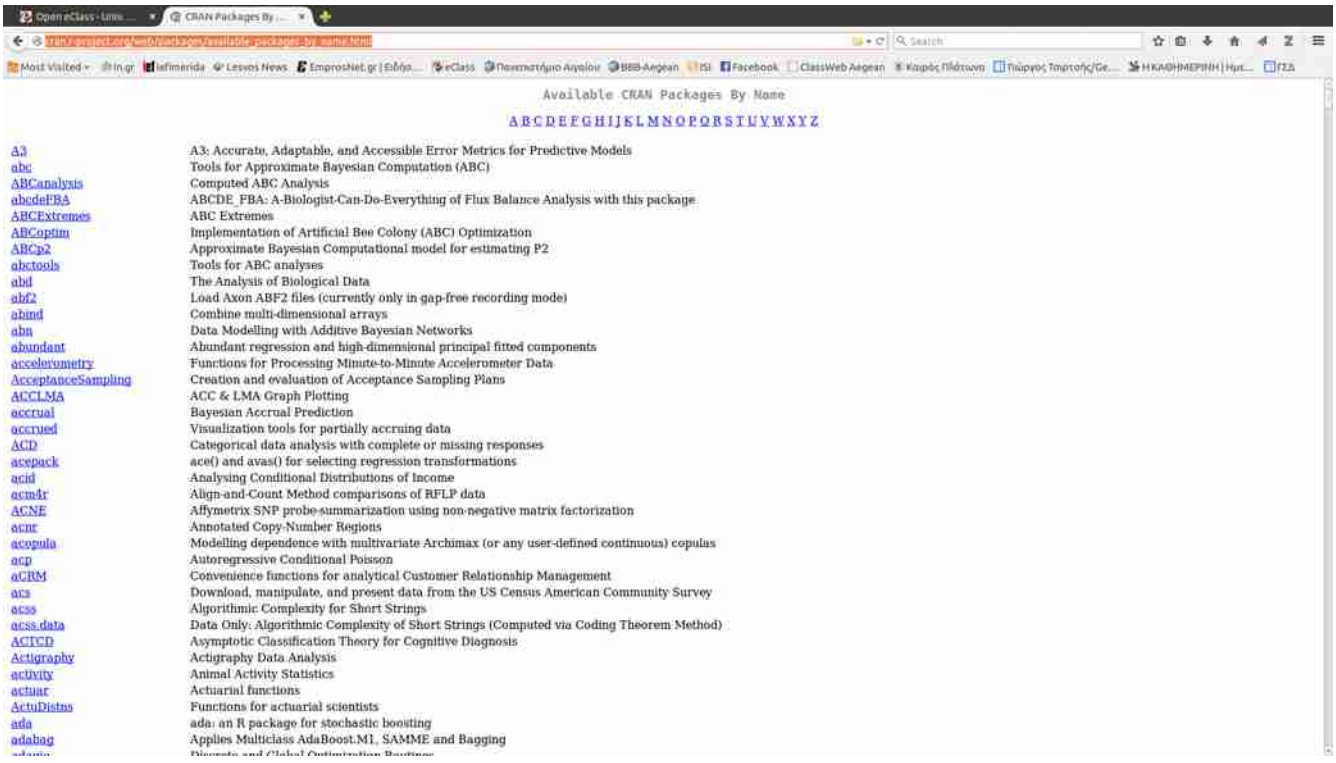
stackoverflow : ιστότοπος ερωταπαντήσεων και επίλυσης απορριών για προγραμματιστές,

http://stackoverflow.com/

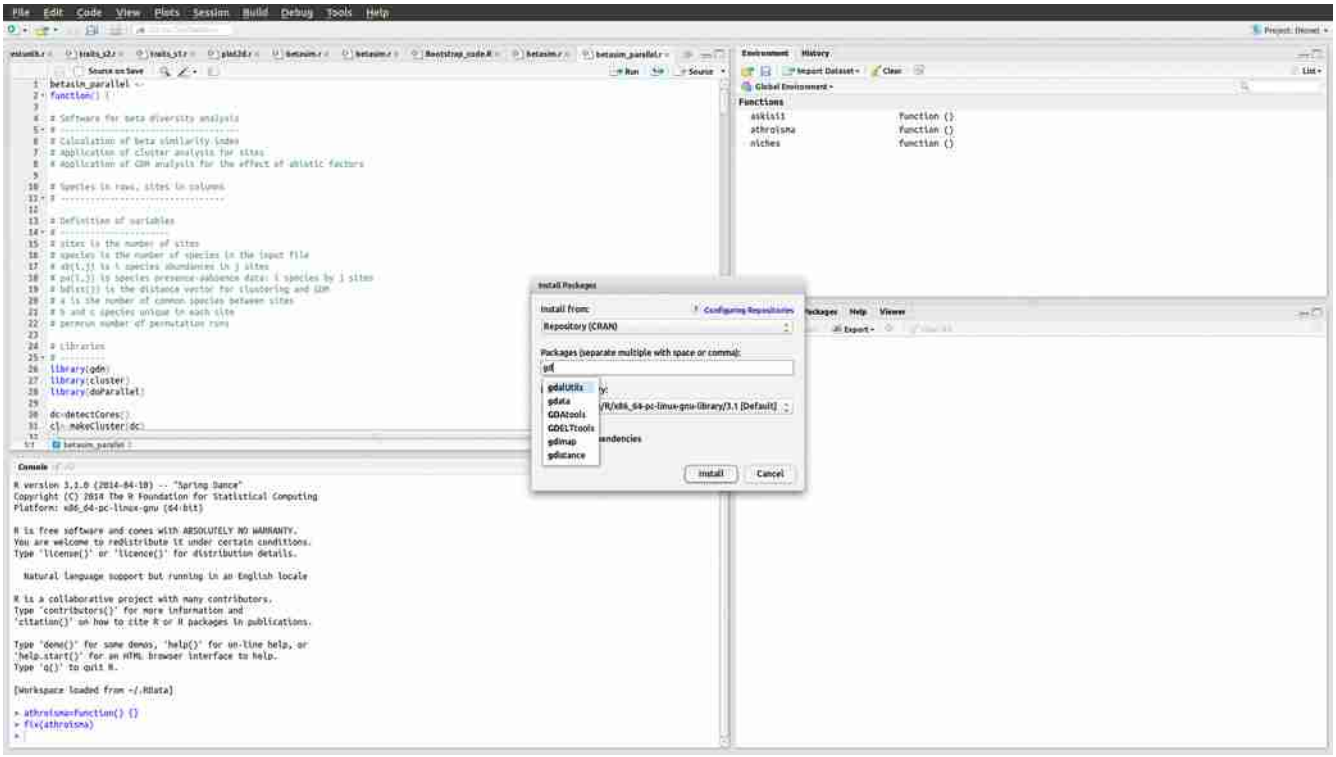
Z. ΠΑΚΕΤΑ ΣΥΝΑΡΤΗΣΕΩΝ-ΠΑΡΑΛΛΗΛΟΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ

Z1. Εγκατάσταση και χρήση πακέτων συναρτήσεων

Όπως προαναφέρθηκε η μεγάλη δύναμη της R και το συγκριτικό της πλεονέκτημα έναντι άλλων εμπορικών ή ελεύθερων αντίστοιχων λογισμικών βρίσκεται στη πληθώρα συναρτήσεων (υποπρογραμμάτων) που είναι διαθέσιμες μέσω του διαδικτύου και είναι ομαδοποιημένες με βάση την εργασία που επιτελούν σε πακέτα (packages). Πληροφορίες για τα πακέτα και τις περιεχόμενες συναρτήσεις μπορούμε εύκολα να βρούμε στο διαδίκτυο σε ότι αφορά στην εργασία που επιτελούν, στα ορίσματα που απαιτούν και στα αποτελέσματα που επιστρέφουν, π.χ. στο δικτυακό τόπο http://cran.r-project.org/web/packages/available_packages_by_name.html (Εικόνα Z1). Στη βασική έκδοση της R (base package) ήδη περιέχεται μεγάλος αριθμός συναρτήσεων που επιτελούν βασικές λειτουργίες (π.χ. abs, max, min, mean, sum). Για να χρησιμοποιήσουμε τις επιπλέον συναρτήσεις για πιο εξειδικευμένες εργασίες, πρέπει να κατεβάσουμε και να εγκαταστήσουμε τα πακέτα (packages) της R που τις περιέχουν, πέρα από τη βασική έκδοση.



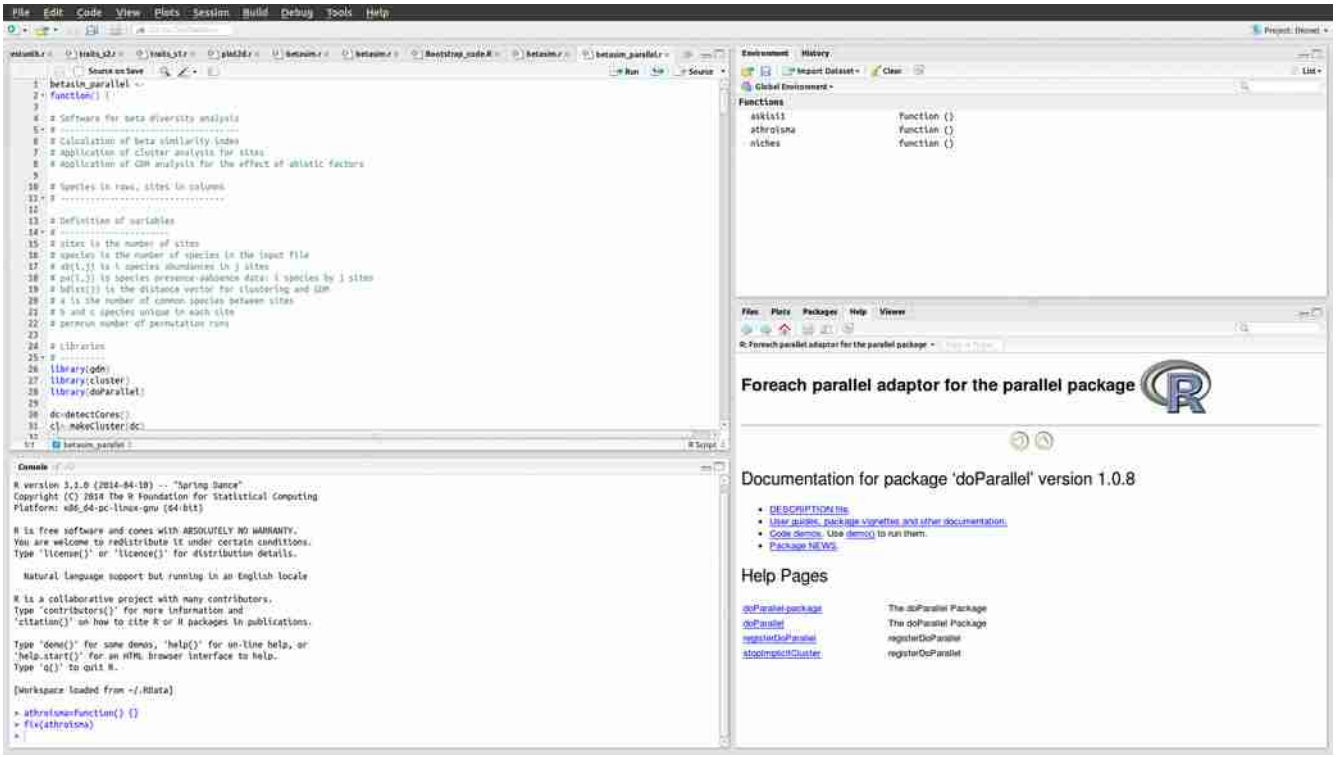
Εικόνα Z1. *Διαθέσιμα πακέτα συναρτήσεων αλφαβητικά από τον επίσημο δικτυακό τόπο της R.*



Εικόνα Z2. *Εγκατάσταση πακέτου συναρτήσεων από το μενού του RStudio, επιλογή Tools-Install packages.*

Η εγκατάσταση πακέτων συναρτήσεων γίνεται εύκολα μέσω του RStudio με την επιλογή Tools-Install packages. Στο παράθυρο που ανοίγει επιλέγουμε Install from: Repository (CRAN) και στη συνέχεια το πακέτο που επιθυμούμε. Πληκτρολογώντας τα 1, 2 πρώτα γράμματα του ονόματος του πακέτου, εμφανίζεται λίστα με όσα πακέτα αρχίζουν από τα αρχικά αυτά, διευκολύνοντας την επιλογή (Εικόνα Z2). Το πεδίο Install dependencies στο παραπάνω παράθυρο πρέπει να είναι επιλεγμένο ώστε να εγκατασταθούν και όποια άλλα πακέτα είναι απαραίτητα για την σωστή λειτουργία αυτού που επιλέξαμε. Στην συνέχεια γίνεται η εγκατάσταση του πακέτου.

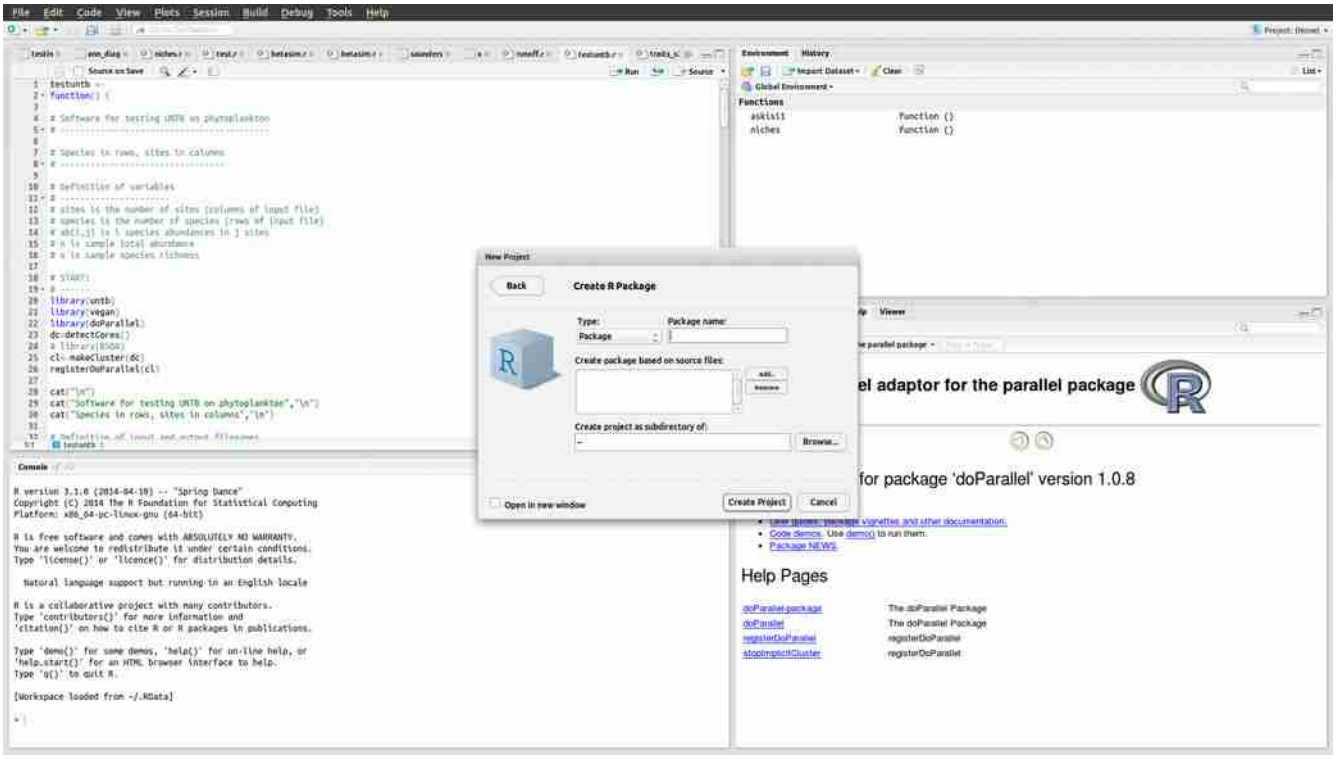
Όταν ένα πακέτο εγκατασταθεί πρέπει στην γραμμή εντολών ή μέσα στο πρόγραμμά μας να γράψουμε την εντολή library(onoma_paketou), π.χ. library(Rcmdr), όπου Rcmdr το όνομα ενός πακέτου στατιστικής. Με τον τρόπο αυτό ενεργοποιείται το πακέτο και μπορούν να χρησιμοποιηθούν στη συνέχεια οι συναρτήσεις του πακέτου. Σημειώνεται ότι το όνομα του πακέτου στην εντολή library πρέπει να είναι γραμμένο επακριβώς, γιατί στην R καταχωρούνται ως διαφορετικά γράμματα τα κεφαλαία και τα πεζά. Δηλαδή αν γράφαμε library(rcmdr) η R δεν θα αναγνώριζε την εντολή αφού το όνομα είναι Rcmdr και όχι rcmdr.



Εικόνα Ζ3. *On-line βοήθεια για το πακέτο συναρτήσεων και τις συναρτήσεις του στο παράθυρο Help του RStudio, που βρίσκεται συνήθως κάτω και δεξιά.*

Πολύ σημαντικό στην περίπτωση χρήσης έτοιμων συναρτήσεων είναι να γνωρίζει ο χρήστης επακριβώς τι απαιτεί η συνάρτηση σε ότι αφορά: (α) στα δεδομένα εισόδου και στην μορφή τους και (β) στα αποτελέσματα που παρέχει και στην μορφή τους. Βοήθεια σε ότι αφορά στο πακέτο και στις συναρτήσεις του μπορεί να βρει ο χρήστης on-line στο RStudio στο παράθυρο Help που εμφανίζεται συνήθως κάτω δεξιά (Εικόνα Ζ3). Βέβαια πολύ περισσότερη βοήθεια και παραδείγματα μπορεί να αναζητήσει ο χρήστης στο διαδίκτυο γράφοντας R και στη συνέχεια το όνομα του πακέτου ή της συνάρτησης.

Σε προηγούμενο κεφάλαιο περιγράφηκε αναλυτικά η διαδικασία αποθήκευσης και στη συνέχεια ανάκτησης για επεξεργασία ή εκτέλεση ενός προγράμματος-συνάρτησης που κατασκευάζεται στην R (Κεφάλαιο Β, Ενότητα Β1). Στη περίπτωση όμως που ένας προγραμματιστής έχει κατασκευάσει πολλά προγράμματα-συναρτήσεις είναι πιο πρακτικό να ομαδοποιήσει αυτά τα προγράμματα κατασκευάζοντας το δικό του πακέτο. Έτσι γράφοντας, είτε στη γραμμή εντολών της R, είτε μέσα σε ένα πρόγραμμα, την εντολή library(onoma_paketou) έχει πρόσβαση στη χρήση όλων των συναρτήσεων που έχει ενσωματώσει στο πακέτο.

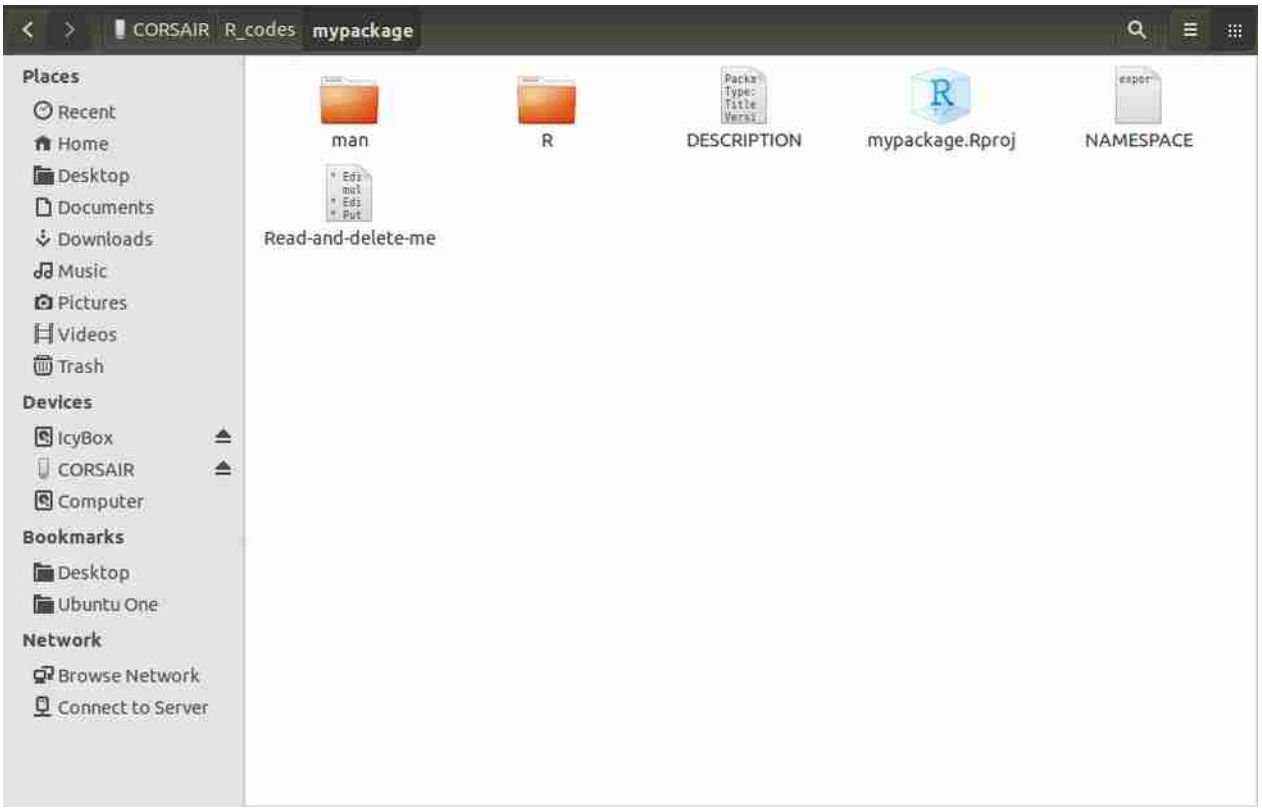


Εικόνα Z4. *Κατασκευή πακέτου χρήστη από το μενού του RStudio ακολουθώντας τις επιλογές File-NewProject-New Directory-R Package.*

Το πακέτο του προγραμματιστή μπορεί να περιλαμβάνει on-line βοήθεια στο παράθυρο Help, που αφορά στο πακέτο συνολικά ή στις επιμέρους συναρτήσεις του. Δηλαδή ένα τέτοιο πακέτο αν κατασκευαστεί έχει όλες τις ιδιότητες ενός πακέτου συναρτήσεων της R, όπως αυτά που διατίθενται στο διαδίκτυο. Επιπλέον, επειδή αποθηκεύεται όλη η πληροφορία (κώδικες συναρτήσεων, βοήθεια κ.λ.π.) σε ένα φάκελο, ο προγραμματιστής μπορεί να διανείμει τον φάκελο σε κάποιον συνεργάτη του εύκολα. Ακόμα ευκολότερη διαδικασία για την διανομή και εγκατάσταση του πακέτου είναι η κατασκευή συμπιεσμένου αρχείου με τα περιεχόμενα του πακέτου, όπως ακριβώς διατίθενται τα πακέτα συναρτήσεων μέσω του διαδικτύου. Η κατασκευή του συμπιεσμένου αρχείου γίνεται από την R και θα περιγραφεί παρακάτω.

Η διαδικασία κατασκευής πακέτου συναρτήσεων είναι σχετικά απλή στο RStudio. Στο Μενού επιλέγουμε File-New Project, στο παράθυρο που εμφανίζεται New Directory και στο επόμενο παράθυρο R Package (Εικόνα Z4). Στο επόμενο παράθυρο εισάγουμε τα χαρακτηριστικά του πακέτου: (α) το όνομά του, (β) τα αρχεία που θα περιέχει (αρχεία .r προγραμμάτων-συναρτήσεων και ενδεχομένως αρχεία δεδομένων) και (γ) τον φάκελο του HY μέσα στον οποίο θα δημιουργηθεί ο

υποφάκελος που θα περιέχει όλη την πληροφορία για το πακέτο. Στην συνέχεια πιέζοντας το πλήκτρο Create Package δημιουργείται ο υποφάκελος με όλη την πληροφορία για το πακέτο (Εικόνα Z5). Ο υποφάκελος περιέχει μεταξύ άλλων ένα αρχείο κειμένου με όνομα Read-and-delete-me που καθοδηγεί για τα επόμενα βήματα. Το σημαντικότερο είναι να προστεθεί πληροφορία στα αρχεία βοήθειας τα οποία θα εμφανίζονται στο παράθυρο Help του RStudio όταν εγκαθίσταται και χρησιμοποιείται το πακέτο. Η πληροφορία αυτή αφορά στο αντικείμενο του προγράμματος-συνάρτησης (τι κάνει το πρόγραμμα), το όνομα του συγγραφέα, τον τύπο και την μορφή των δεδομένων εισόδου και των αποτελεσμάτων, ένα παράδειγμα, τυχόν αναφορές, λέξεις κλειδιά. Όλα τα παραπάνω δεν είναι απαραίτητα πλην μίας γραμμής που πρέπει να συμπληρωθεί και επισημαίνεται στο παράδειγμα παρακάτω. Τα αρχεία βοήθειας είναι τόσα όσα τα αρχεία προγράμματος που προστέθηκαν στο πακέτο και ένα επιπλέον για το ίδιο το πακέτο και βρίσκονται στον υποφάκελο man του φακέλου που δημιουργήθηκε για το πακέτο. Η επεξεργασία των αρχείων βοήθειας γίνεται με έναν απλό κειμενογράφο.



Εικόνα Z5. *Τα περιεχόμενα του υποφακέλου που δημιουργείται με την εντολή Create Package του RStudio.*

Παράδειγμα αρχείου βοήθειας δίνεται στη συνέχεια, όπου έχει υπογραμμιστεί η γραμμή που πρέπει απαραίτητα να συμπληρωθεί. Η γραμμή αυτή αφορά στον τίτλο του προγράμματος και εμφανίζεται στο παράθυρο Help όταν επιλεγεί η βοήθεια της συνάρτησης. Όπως αναφέρθηκε παραπάνω, αν και η γραμμή που πρέπει να συμπληρωθεί απαραίτητα είναι μία, το αρχείο βοήθειας μπορεί να περιέχει μεγάλο όγκο πληροφορίας που βέβαια βοηθά τον χρήστη της συνάρτησης on-line. Για να προστεθεί πληροφορία σε έναν κειμενογράφο ο κατασκευαστής του πακέτου σβήνει την γραμμή που αρχίζει με % και προσθέτει το κείμενό του (Εικόνα Z6). Για παράδειγμα στη θέση details παρακάτω μπορεί να σβήσει το %% ~~ If necessary, more details than the description above ~~ και να προσθέσει το κείμενο που επιθυμεί. Το ίδιο μπορεί να κάνει σε οποιοδήποτε άλλο πεδίο αρχίζει με %.

```
\name{plot2d}

\alias{plot2d}

%- Also NEED an '\alias' for EACH other topic documented here.

\title{

 plot2d  # Η γραμμή πρέπει απαραίτητα να συμπληρωθεί με μία λέξη ή κάποιο σύντομο κείμενο

}

\description{

  plots 2d distributions

}

\usage{

plot2d()

}

%- maybe also 'usage' for other objects documented here.

\details{

%%  ~~ If necessary, more details than the description above ~~

}
```

```
\value{
%%  ~Describe the value returned
%%  If it is a LIST, use
%%  \item{comp1 }{Description of 'comp1'}
%%  \item{comp2 }{Description of 'comp2'}
%% ...
}
\references{
%% ~put references to the literature/web site here ~
}
\author{
 George  # Το όνομα του συγγραφέα
}
\note{
%%  ~~further notes~~
}
%% ~Make other sections like Warning with \section{Warning }{....} ~
\seealso{
%% ~~objects to See Also as \code{\link{help}}, ~~~
}
\examples{
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--      or do  help(data=index)  for the standard data sets.
```

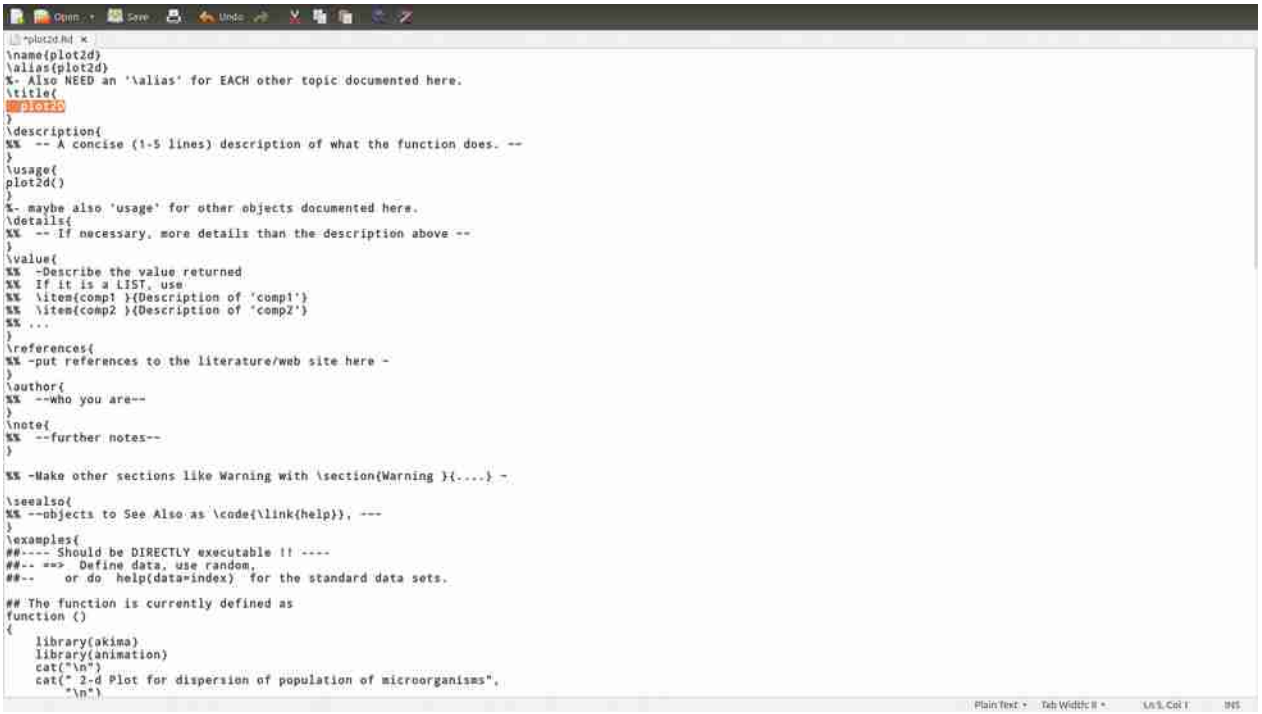
The function is currently defined as

% Add one or more standard keywords, see file 'KEYWORDS' in the

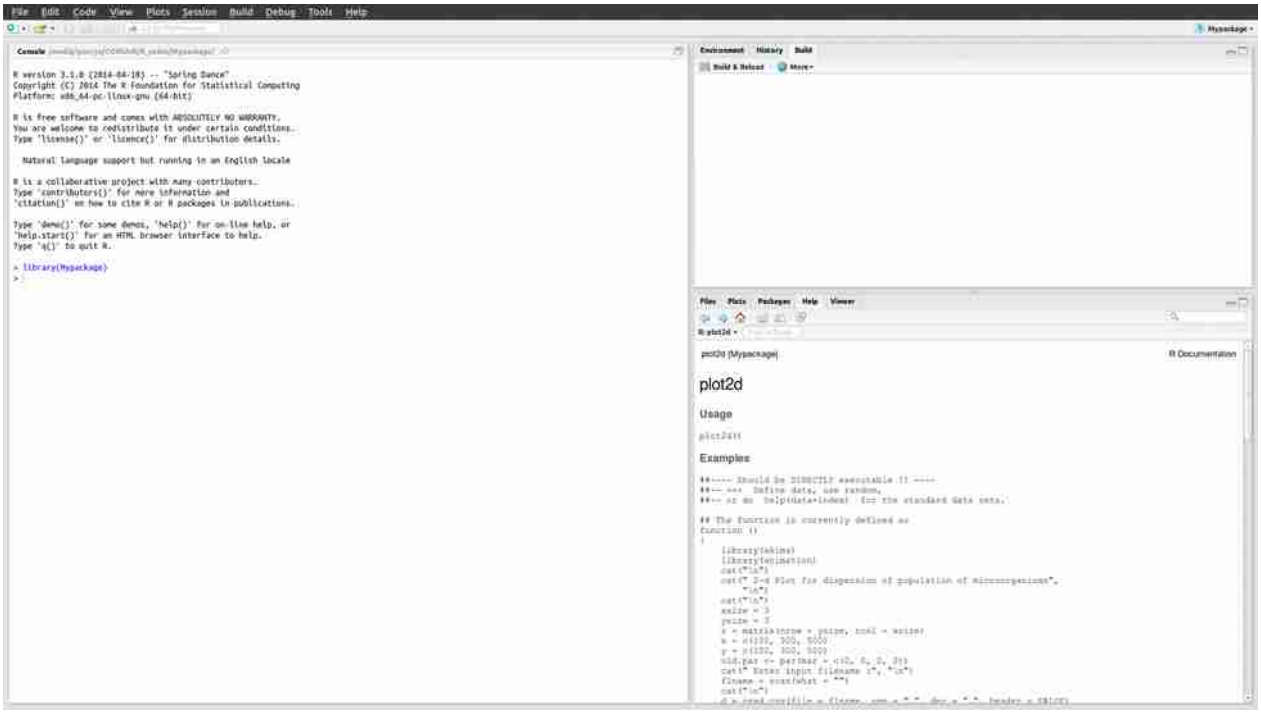
% R documentation directory.

\keyword{ ~kwd1 }

\keyword{ ~kwd2 }% __ONLY ONE__ keyword per line



Εικόνα Z6. *Το αρχείο βοήθειας συνάρτησης του πακέτου όπως εμφανίζεται στον κειμενογράφο. Με πορτοκαλί σημειώνεται η γραμμή που οπωσδήποτε πρέπει να προστεθεί πληροφορία.*



Εικόνα Ζ7. *Ανάκτηση προς χρήση των συναρτήσεων χρήστη με την εντολή library(Mypackage) και παράθυρο χρήστη με την βασική πληροφορία για μία από τις συναρτήσεις κάτω δεξιά.*

Αφού γίνουν οι αλλαγές στα αρχεία βοήθειας (τουλάχιστον οι απαραίτητες), το πακέτο εγκαθίσταται στον ΗΥ με την επιλογή Build-Build and Reload του RStudio. Από κει και πέρα κάθε φορά που ανοίγει το RStudio με την εντολή library(onoma_paketou) στη γραμμή εντολών ή μέσα σε ένα πρόγραμμα ανακτώνται όλες οι συναρτήσεις του πακέτου και μπορούν να χρησιμοποιηθούν στη συνέχεια (Εικόνα Ζ7). Επίσης με την εντολή Build-Build Binary Package δημιουργείται ένα συμπιεσμένο αρχείο με όλα τα περιεχόμενα του πακέτου που μπορεί να διανεμηθεί οπουδήποτε και να εγκατασταθεί σε άλλον ΗΥ κανονικά όπως κάθε άλλο πακέτο με την εντολή Tools-Install Packages του RStudio.

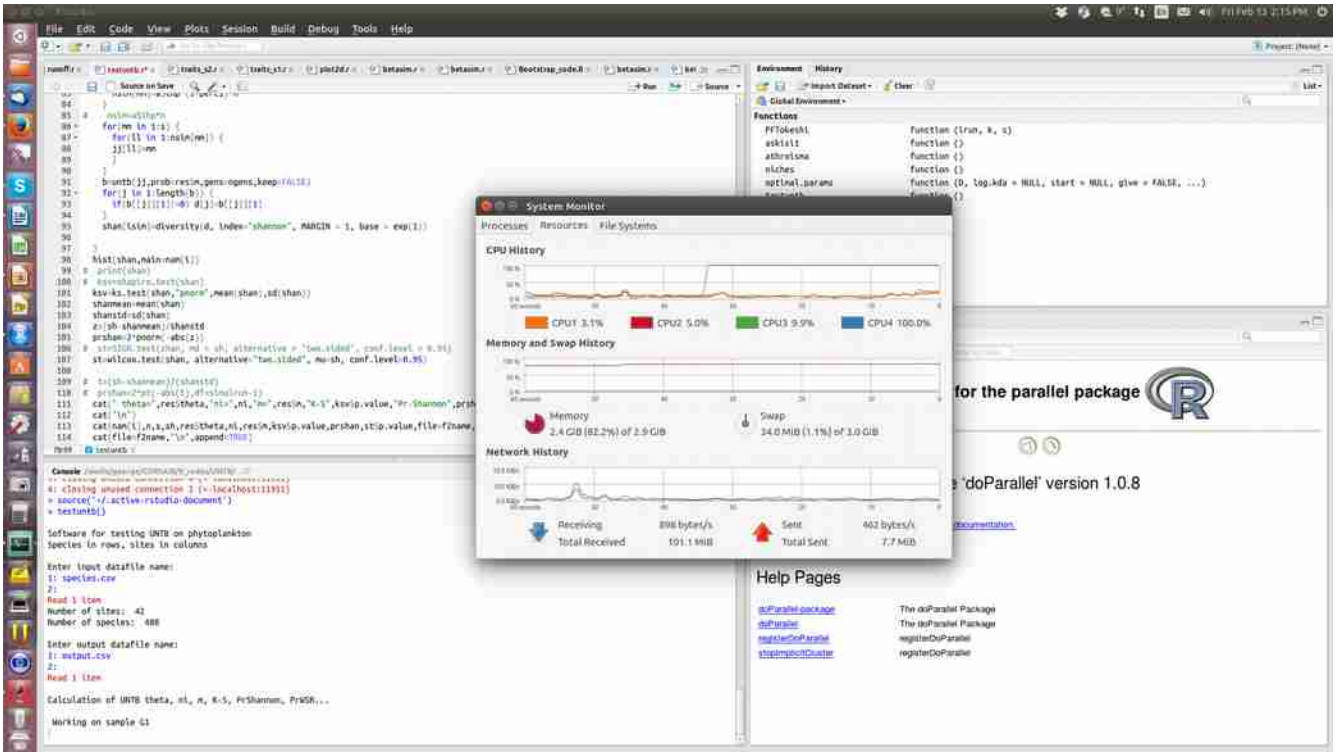
Ζ2. Παράλληλος προγραμματισμός

Η ενότητα αυτή έχει ως στόχους να αποτελέσει εισαγωγή στη λογική του παράλληλου προγραμματισμού και να επιδείξει βασικές εντολές του στην R. Σε καμία περίπτωση δεν εμβαθύνει στο αντικείμενο που είναι ευρύτατο και από τα πλέον εξειδικευμένα στο προγραμματισμό.

Παλαιότερα η πολυεπεξεργασία, δηλαδή η δυνατότητα να εκτελούνται πολλές εργασίες ταυτόχρονα σε έναν ΗΥ από έναν χρήστη ή και περισσότερους, ήταν προνόμιο των υπολογιστών υψηλών

δυνατοτήτων που κατείχαν μεγάλα εκπαιδευτικά και ερευνητικά ιδρύματα ή μεγάλες εταιρείες. Τα τελευταία χρόνια οι προσωπικοί υπολογιστές (φορητοί ή επιτραπέζιοι) είναι εφοδιασμένοι με επεξεργαστές πολλών πυρήνων, που παρέχουν υψηλές δυνατότητες πολυεπεξεργασίας. Όταν σε έναν ΗΥ εκτελούνται συγχρόνως πολλά προγράμματα, το λειτουργικό σύστημα κατανέμει την εργασία στους πυρήνες, κατά κανόνα τις εργασίες κάθε προγράμματος σε ένα πυρήνα, αν βέβαια υπάρχουν ελεύθεροι. Σε περίπτωση που οι πυρήνες είναι όλοι δεσμευμένοι, οι εργασίες των διαφορετικών προγραμμάτων διανέμονται κατά τον βέλτιστο τρόπο μεταξύ των πυρήνων.

Όταν εκτελείται ένα πρόγραμμα σε R (ή σε άλλη γλώσσα προγραμματισμού), ο φόρτος αποδίδεται από το λειτουργικό σύστημα σε έναν μόνο πυρήνα. Αυτό σημαίνει ότι αν ο ΗΥ διαθέτει επεξεργαστή πολλών πυρήνων (ή πολλούς επεξεργαστές) και την ίδια στιγμή δεν εκτελούνται άλλα προγράμματα, ένας ή περισσότεροι πυρήνες (ή επεξεργαστές) μένουν αχρησιμοποίητοι. Στην περίπτωση ενός πολύ απαιτητικού σε υπολογιστική ισχύ και σε χρόνο εκτέλεσης προγράμματος, είναι σκόπιμη η κατανομή των εργασιών του προγράμματος σε περισσότερους πυρήνες, ώστε να μειωθεί αντίστοιχα ο χρόνος εκτέλεσης. Ένα πρόγραμμα που κατανέμει τις εργασίες του σε περισσότερους από έναν πυρήνες ή επεξεργαστές λέγεται παράλληλο και η διαδικασία προγραμματισμού παράλληλος προγραμματισμός (parallel programming). Στις Εικόνες Z8 και Z9 φαίνεται η διαφορά στην εκμετάλλευση της διαθέσιμης υπολογιστικής ισχύος από το ίδιο πρόγραμμα R, αν αυτό εκτελείται κανονικά σε ένα πυρήνα (Εικόνα Z8) και παράλληλα σε τέσσερις πυρήνες (Εικόνα Z9). Στην συνήθη μη παράλληλη εκτέλεση χρησιμοποιείται ένας πυρήνας σε ποσοστό 100%, ενώ οι τρεις άλλοι λειτουργούν σε ποσοστά κάτω του 10%. Στην παράλληλη εκτέλεση λειτουργούν και οι τέσσερις σε ποσοστό 100%.



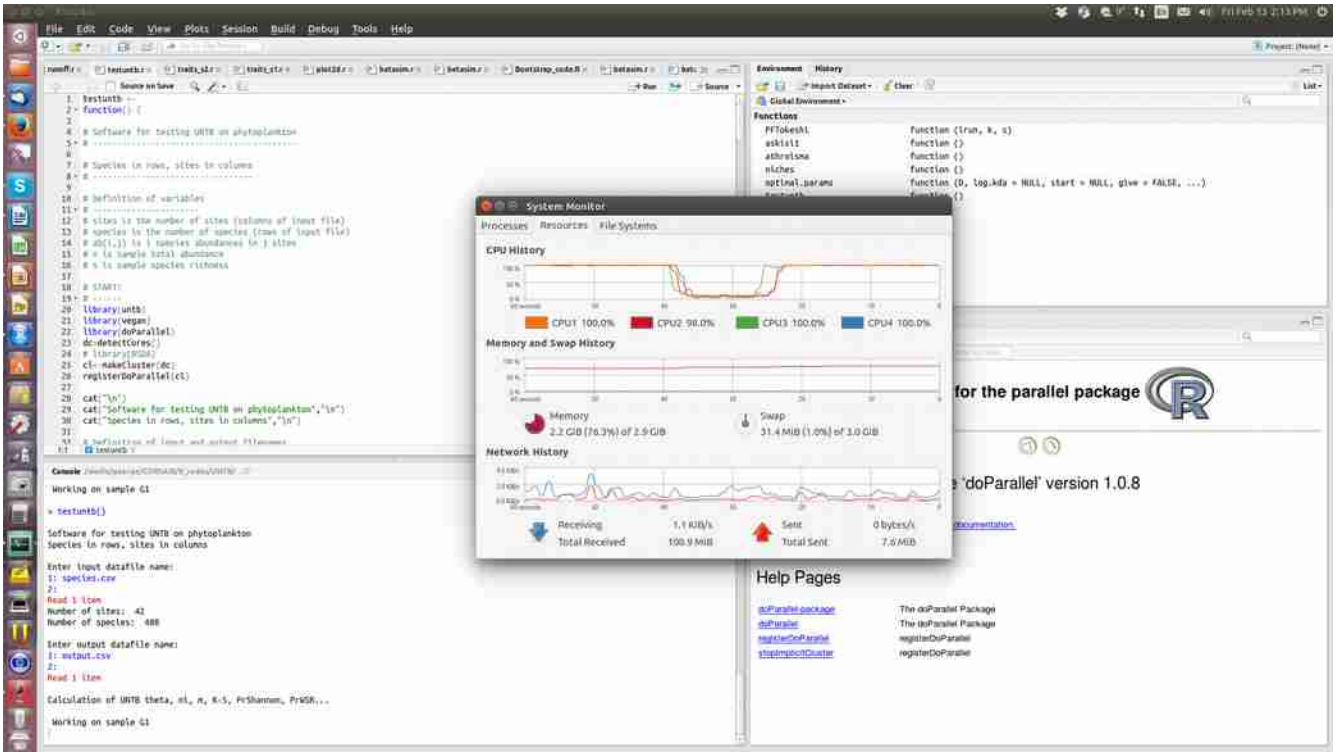
Εικόνα Z8. *Συνήθης εκτέλεση προγράμματος της R με την χρήση ενός πυρήνα σε ΗΥ τεσσάρων πυρήνων.*

Μία συνήθης περίπτωση όπου η χρήση παράλληλου προγραμματισμού ενδείκνυται είναι όταν το πρόγραμμα περιέχει δομή ή δομές πολλών επαναλήψεων με μεγάλο υπολογιστικό φόρτο σε κάθε επανάληψη. Την περίπτωση αυτή θα καλύψει το παράδειγμα που ακολουθεί. Αρχικά για να είναι δυνατή η χρήση παράλληλων εντολών πρέπει να εγκατασταθεί το κατάλληλο πακέτο της R. Υπάρχουν αρκετά διαθέσιμα, ένα εκ των οποίων είναι το πακέτο doParallel. Η εγκατάσταση του πακέτου γίνεται με τον τρόπο που αναφέρθηκε για όλα τα πακέτα στην προηγούμενη ενότητα και η ενεργοποίηση με την εκτέλεση της εντολής library(doParallel) στην γραμμή εντολών ή κατά την εκτέλεση του προγράμματος της R. Μετά την εντολή library(doParallel) μέσα στο πρόγραμμα πρέπει να ακολουθήσουν οι εντολές:

```
dc=detectCores()

cl=makeCluster(dc)

registerDoParallel(cl)
```

Με την πρώτη το πρόγραμμα βρίσκει πόσους πυρήνες έχει ο επεξεργαστής του χρησιμοποιούμενου

ΗΥ. Με την δεύτερη φτιάχνεται ένα cluster (εικονική ομάδα συνεργαζόμενων υπολογιστών) τόσων υπολογιστών, όσοι οι πυρήνες που βρέθηκαν με την προηγούμενη εντολή. Εδώ υπάρχει η δυνατότητα να αφιερώσουμε στην R όχι το σύνολο των πυρήνων αλλά π.χ. έναν λιγότερο, τροποποιώντας την δεύτερη εντολή και γράφοντας cl=makeCluster(dc-1). Με τον τρόπο αυτό κρατάμε έναν (ή και περισσότερους) πυρήνα διαθέσιμο για άλλες εργασίες που ίσως εκτελούμε ταυτόχρονα ή για τις ανάγκες του λειτουργικού συστήματος. Τέλος η τρίτη εντολή εγγράφει το cluster που δημιουργήθηκε στην διάθεση των συναρτήσεων του πακέτου doParallel.



Εικόνα Z9. *Παράλληλη εκτέλεση του προγράμματος της εικόνας Z4 με την εκμετάλλευση του συνόλου της επεξεργαστικής ισχύος του ΗΥ.*

Όπως προαναφέρθηκε η συνηθέστερη χρήση παράλληλου κώδικα γίνεται σε δομές πολλών επαναλήψεων με μεγάλο υπολογιστικό φόρτο στην κάθε επανάληψη. Για να εκτελεστεί παράλληλα μία δομή επανάληψης, το πακέτο doParallel διαθέτει την εντολή foreach που αντικαθιστά τη γνωστή εντολή επανάληψης for. Δίνεται στη συνέχεια ένα παράδειγμα προγράμματος σε R στο οποίο υπολογίζονται τα τετράγωνα ακέραιων αριθμών από 1 έως n, αθροίζονται και στη συνέχεια βρίσκεται ο μέσος όρος τους. Το πρόγραμμα σε μη παράλληλο κώδικα της R είναι το εξής:

```
partest<-function() {
  print("Define n")
  n=scan()
  x=vector(length=n)
  for(i in 1:n) {
    x[i]=i^2
  }
  s=sum(x)
  print(s/n)
}
```

Το αντίστοιχο παράλληλο πρόγραμμα έχει ως εξής:

```
partest<-function() {
  library(doParallel)
  dc=detectCores()
  cl=makeCluster(dc)
  registerDoParallel(cl)
  print("Define n")
  n=scan()
  x=vector(length=n)
  x=foreach(i=1:n, .combine=c) %dopar% {
     x[i]=i^2
  }
s=sum(x)
print(s/n)
```

```
stopCluster(cl)

}
```

Η εντολή foreach αντικαθιστά την for και έχει διαφορετική σύνταξη. Ο δείκτης της επανάληψης i παίρνει τιμές γράφοντας i=1,n στην foreach αντί i in 1:n στη for. Τα αποτελέσματα της επανάληψης αποθηκεύονται σε μία μεταβλητή (τη μεταβλητή x στο παράδειγμα) που έχει κανονικά τη μορφή λίστας (list) της R. Γράφοντας όμως στα ορίσματα της foreach, .combine=c όπως στο παράδειγμα, η μεταβλητή x ορίζεται ως διάνυσμα (vector). Η παράμετρος %dopar% της foreach ορίζει ότι η εκτέλεση είναι παράλληλη. Αν αντί %dopar%, γραφεί %do%, η δομή επανάληψης εκτελείται κανονικά και όχι παράλληλα. Τέλος το εικονικό cluster που σχηματίστηκε με την εντολή makeCluster στην αρχή του προγράμματος, καταργείται με την εντολή stopCluster.

Βιβλιογραφία-Σύνδεσμοι

http://manuals.bioinformatics.ucr.edu/home/programming-in-r

http://cran.r-project.org/