

1 Introducere in R

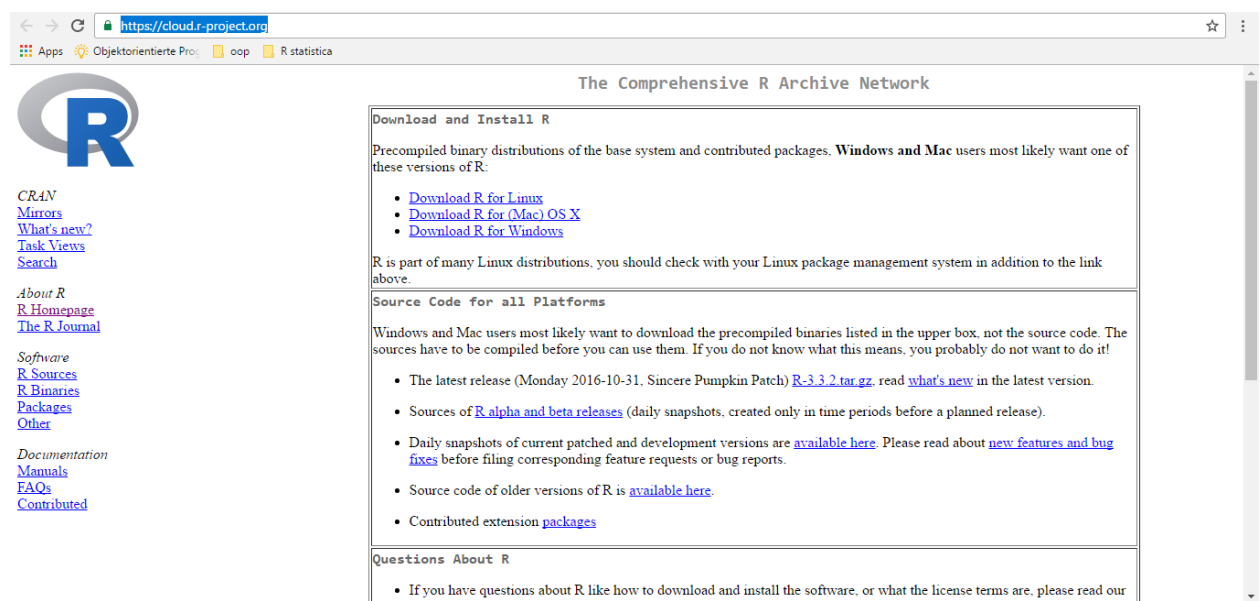
R este un soft pentru prelucrări statistice a datelor foarte performant. Este asemănător ca și performanțe cu SPSS doar că este un mediu care seamănă mai mult cu mediul linux lucrându-se mult din linie de comandă. Limbajul R este un limbaj orientat obiect. De fapt toate variabilele cu care vom lucra sunt obiecte.

1.1 Instalarea

Informații referitoare la proiectul R pentru statistică computațională se pot accesa la adresa de web <https://www.r-project.org/>. Pentru a downloada softul trebuie accesat un server CRAN (Comprehensive R Archive Network) de la adresa <https://cran.r-project.org/mirrors.html>.

Ultima versiune a limbajului R este versiunea R-3.3.2 pentru 32-bit și 64-bit. În mod implicit se instalează versiunea pe 32 de biți pentru Windows, dar se poate alege versiunea pe 64 de biți. Evident ca fanii sistemelor de operare Linux sau MacOX au la dispoziție variantele corespunzătoare.

Adresa de download de la care am instalat softul este <https://cloud.r-project.org/>

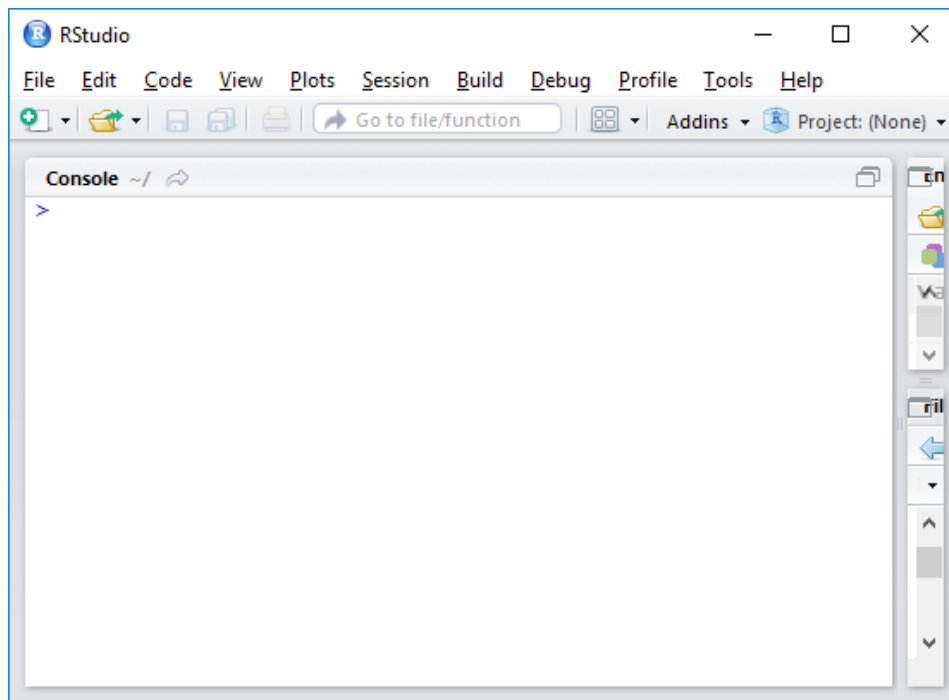


The screenshot shows the CRAN website in a web browser. The address bar displays <https://cloud.r-project.org/>. The page title is "The Comprehensive R Archive Network". On the left, there is a sidebar with the R logo and a list of links: CRAN, Mirrors, What's new?, Task Views, Search, About R, R Homepage, The R Journal, Software, R Sources, R Binaries, Packages, Other, Documentation, Manuals, FAQs, and Contributed. The main content area is titled "Download and Install R" and contains the following text: "Precompiled binary distributions of the base system and contributed packages. **Windows and Mac** users most likely want one of these versions of R." Below this, there are three bullet points: "Download R for Linux", "Download R for (Mac) OS X", and "Download R for Windows". Further down, it says "R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above." and "Source Code for all Platforms". It then states "Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!" and lists four bullet points: "The latest release (Monday 2016-10-31, Sincere Pumpkin Patch) [R-3.3.2.tar.gz](#), read [what's new](#) in the latest version.", "Sources of [R alpha and beta releases](#) (daily snapshots, created only in time periods before a planned release).", "Daily snapshots of current patched and development versions are [available here](#). Please read about [new features and bug fixes](#) before filing corresponding feature requests or bug reports.", and "Source code of older versions of R is [available here](#)". The last bullet point is "Contributed extension [packages](#)". At the bottom, there is a section titled "Questions About R" with one bullet point: "If you have questions about R like how to download and install the software, or what the license terms are, please read our [FAQ](#)".

După instalarea unei versiuni de R, pentru a simplifica modul de lucru vom instala o interfață grafică (tot gratuită) oferită de R-Studio de la adresa de web: <https://www.rstudio.com/products/rstudio/download/>

1.2 Lucrul în mediul R

La pornirea RStudio interfața pornește afișând un prompter „>” semn care indică faptul că mediul așteaptă comenzi:



Comenzile se lansează apăsând tasta ENTER. În cazul în care în loc de < apare + înseamnă că instrucțiunea se completează pe rândul următor:

Exercițiu 1.1

Stergeți conținutul consolei.

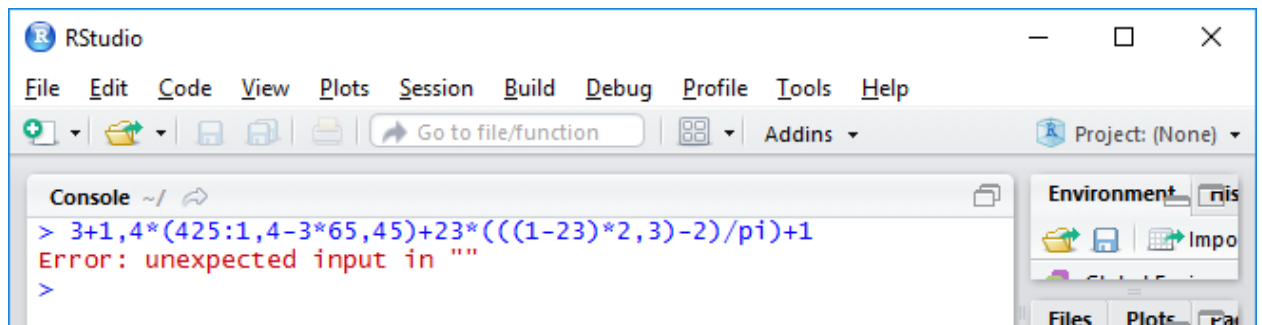
Indicație: *Ctrl+I*

Ex: 1.2

Calculați expresia $3+1,4*(425:1,4-3*65,45)+23*(((1-23)*2,3)-2)/\pi)+1$

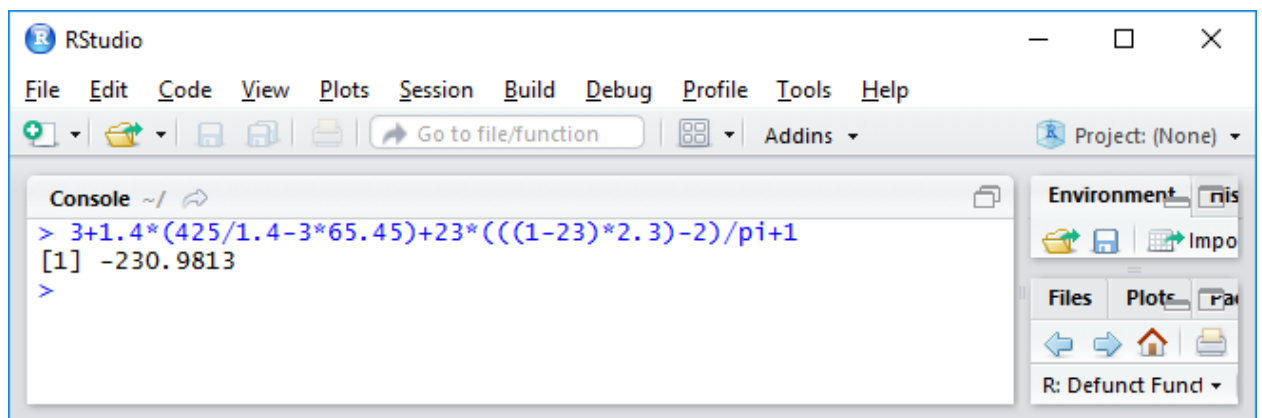
Indicație: *Ctrl+I*

Atenție la utilizarea separatorului zecimal și la semnul împărțirii altfel veți obține un mesaj de eroare:



The screenshot shows the RStudio interface. In the console, the following command was entered: `> 3+1,4*(425:1,4-3*65,45)+23*(((1-23)*2,3)-2)/pi)+1`. The command is highlighted in blue. Below it, an error message is displayed in red: `Error: unexpected input in ""`. The console prompt `>` is visible at the bottom.

Corect este:



The screenshot shows the RStudio interface. In the console, the following command was entered: `> 3+1.4*(425/1.4-3*65.45)+23*(((1-23)*2.3)-2)/pi+1`. The command is highlighted in blue. Below it, the output is displayed: `[1] -230.9813`. The console prompt `>` is visible at the bottom.

Exercițiu 1.3

Testați până la câte paranteze imbricate poate lucra limbajul R?

În linia de comandă se pot scrie mai multe instrucțiuni care se vor separa prin ”;”

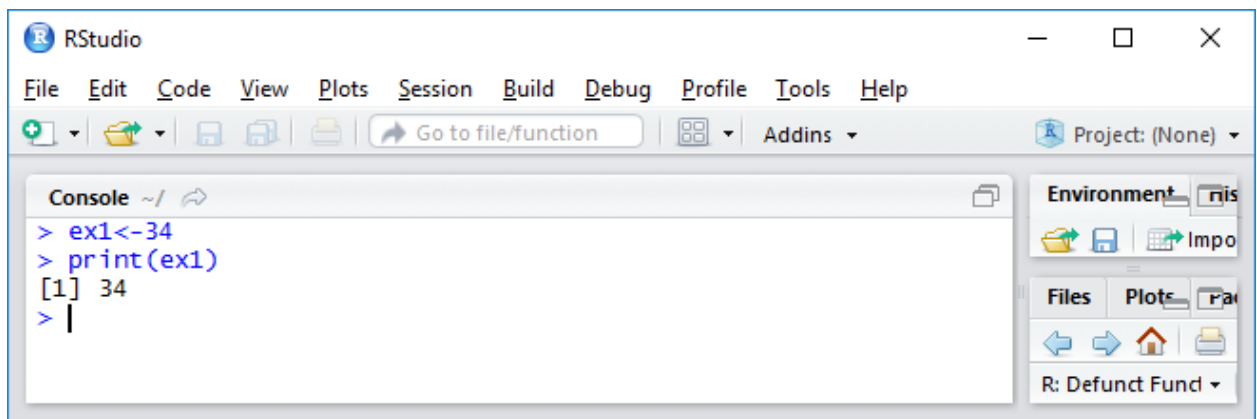
Pentru a vizualiza comenzi anterioare se folosește tasta ”săgeată sus”. Pentru a ieși din mediul de dezvoltare R se folosește funcția `q()`. La ieșire avem opțiunea de a salva istoricul instrucțiunilor scrise în consolă.

1.3 Variabile

În limbajul R variabilele sunt definite cu ajutorul caracterelor literale. Atenție la numele variabilelor deoarece în R numele este ”case sensitive”. În R toate variabilele sunt de fapt obiecte.

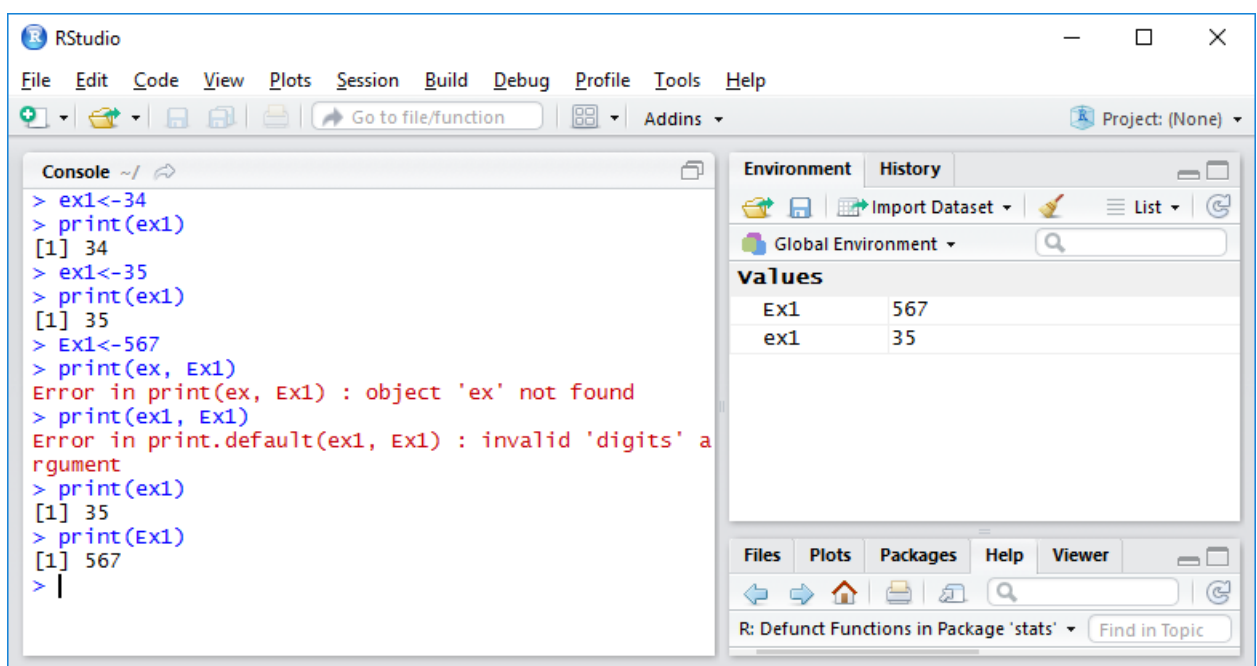
Atribuirea unei valori unui obiect se face cu "<=".

Exemplu:

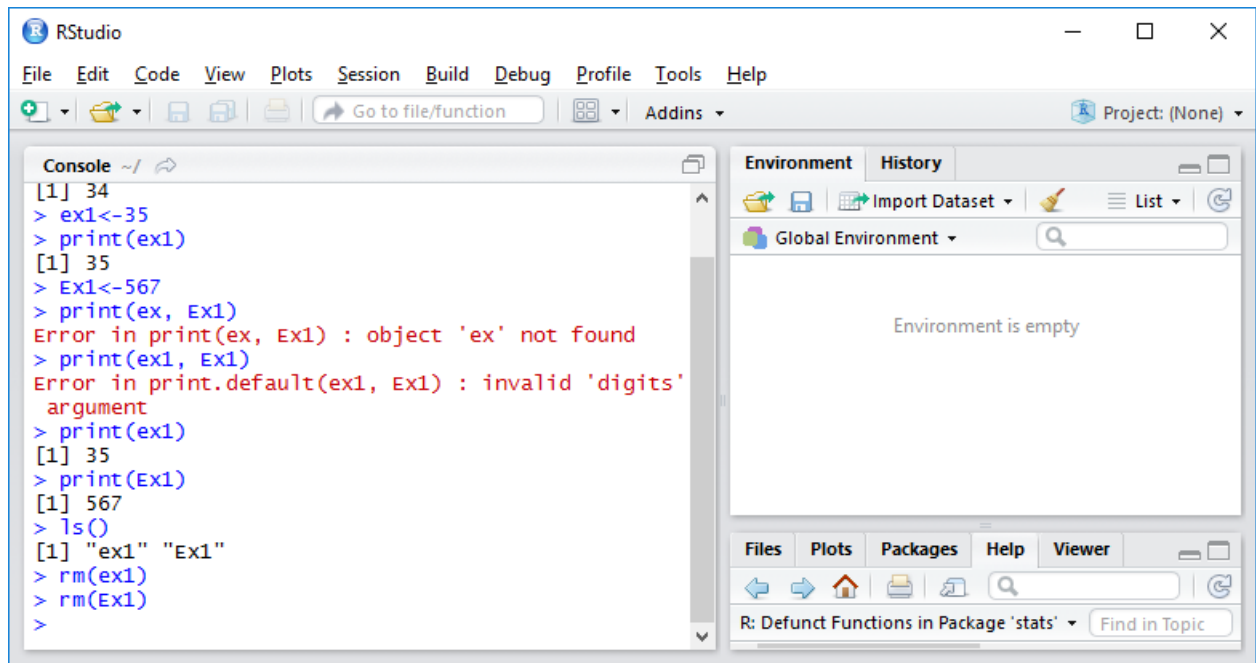


În exemplul de mai sus variabilei ex1 i s-a atribuit valoarea 34. Apelând funcția print cu numele obiectului ca parametru se afișează valoarea variabilei.

Exemplu:



Observăm că în exemplul de mai sus valoarea inițială a variabilei `ex1` a fost suprascrisă și este alta ca cea a variabilei `Ex1`. În panelul de "Environment" se afișează valorile curente ale variabilelor utilizate. Listarea variabilelor utilizate se face apelând funcția `ls()`. Pentru a distruge variabilele utilizate se folosește funcția `rm()` care primește ca parametru numele variabilei de șters.



1.4 Tipuri de date în R

În continuare vom prezenta cele mai utilizate tipuri de date în limbajul R

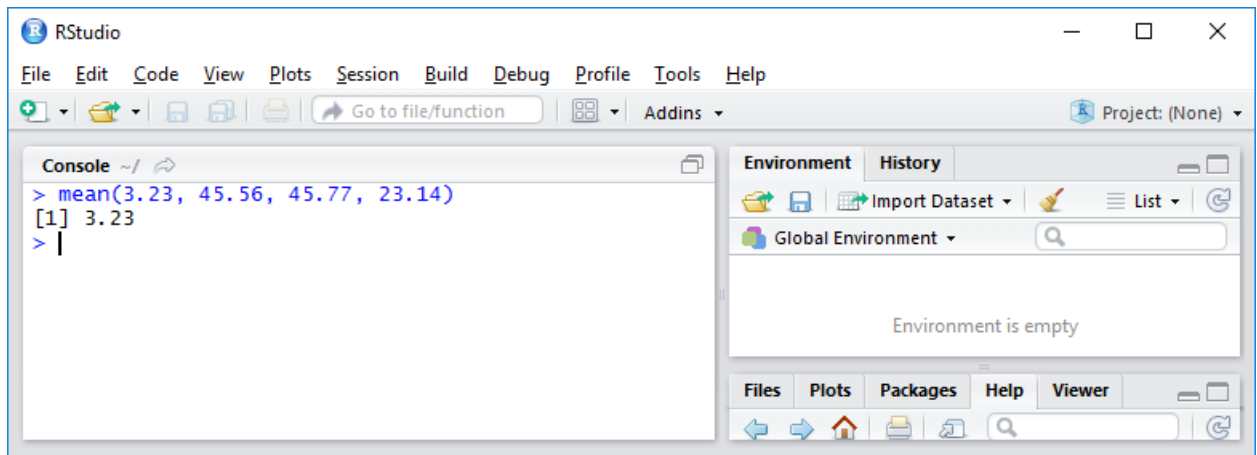
1.4.1 Vectori

Exercițiu 1.4

Să se calculeze media aritmetică a valorilor 3.23 45.56 45.77 23.14.

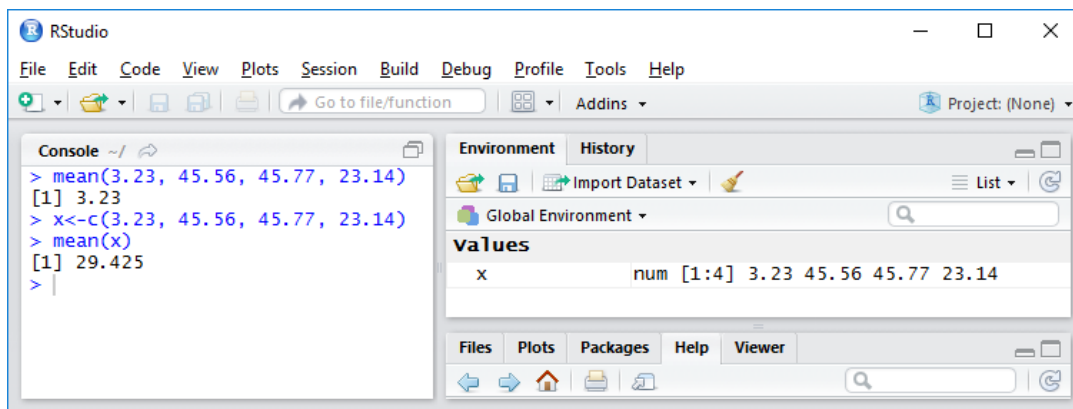
Soluția 1: utilizarea formulei pentru medie.

Soluția 2: utilizarea funcției `mean()`.



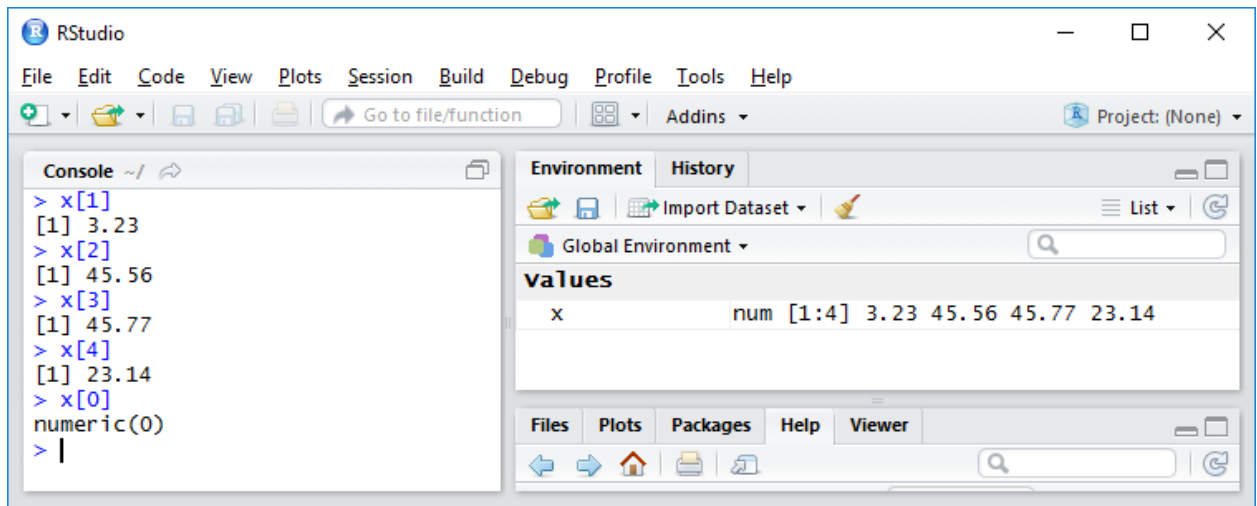
Observăm că funcția `mean()` afișează ca rezultat prima valoare care am trimis-o ca parametru. Celelalte valori sunt ignorate. Pentru a putea calcula media corect trebuie să atribuim valorile prezentate mai sus unui vector și apoi să trimitem acest vector ca parametru funcției `mean()`.

Practic trebuie să ”concatenăm” valorile utilizând funcția `c()`. (`c` este prescurtarea de la ”concatenate”)



Să înțelegem ce s-a întâmplat:

Făcând atribuirea `x<-c(3.23, 45.56, 45.77, 23.14)` variabila `x` a devenit vector cu 4 valori. Pentru a le accesa evident că putem folosi notația clasică `x[i]`.

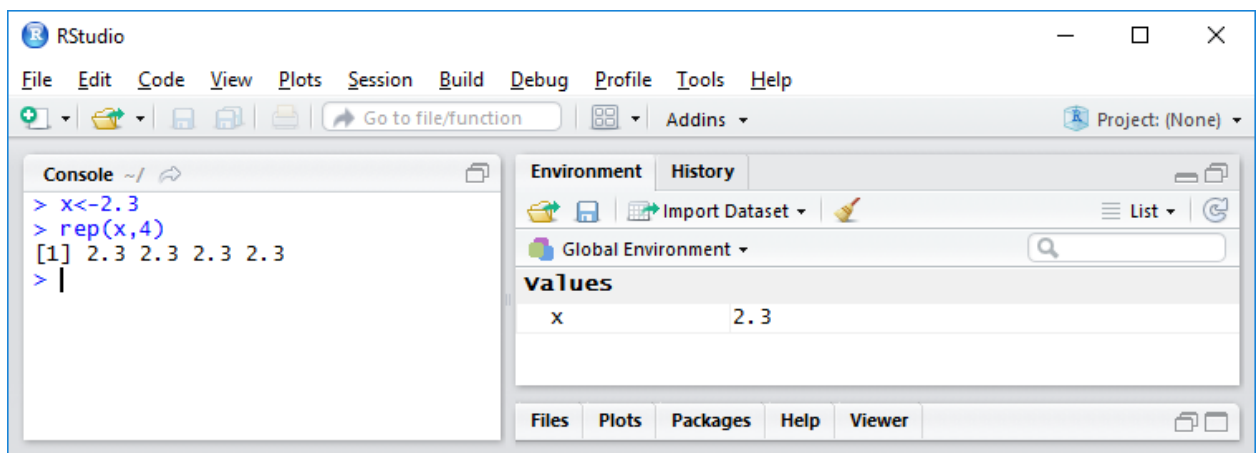


Observăm că în R indecșii unui vector încep de la 1. Acest lucru se poate observa și în panelul de valori în care variabila x conține valori numerice de la indexul 1 la 4.

1.4.2 Generarea de vectori

1.4.2.1 Funcția `rep()` :

Sintaxa: `rep(x, times=n)` repetă obiectul x de n ori



Observație:

În exemplul de mai sus observăm că în consolă valoarea 2.3 a fost repetată de 4 ori dar valoarea obiectului x nu s-a modificat dacă verificăm valoarea lui x afișată în "global environment".

Exercițiu 1.5

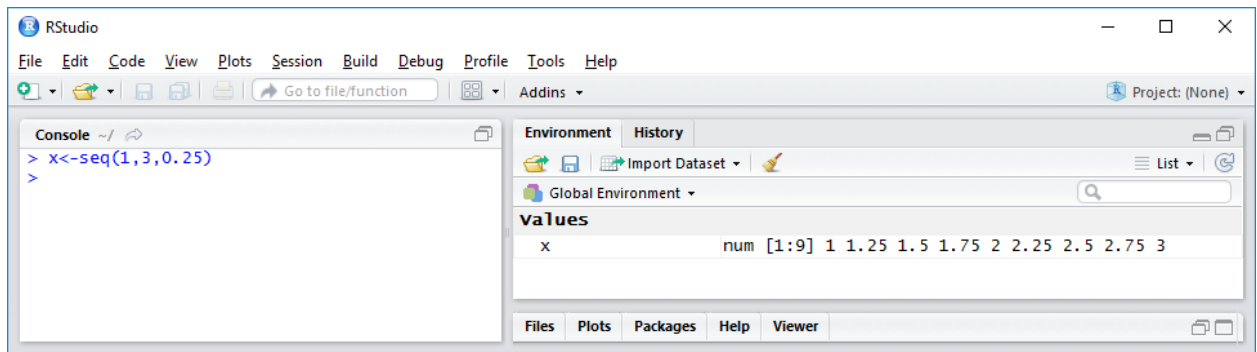
Atribuiți obiectului x de 5 ori valoarea 3.24 utilizând doar funcția `rep()` .

Observație

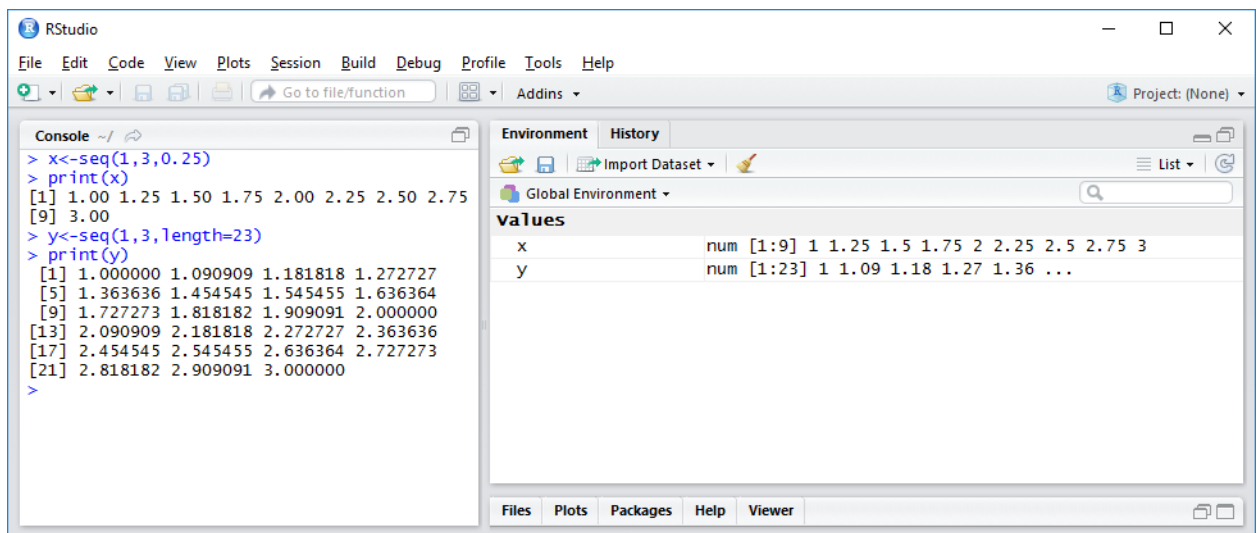
`rep(1, 4)` este identic cu `c(1, 1, 1, 1)`

1.4.2.2 Funcția seq():

Sintaxa: `seq(from=, to=, by=, length=)`



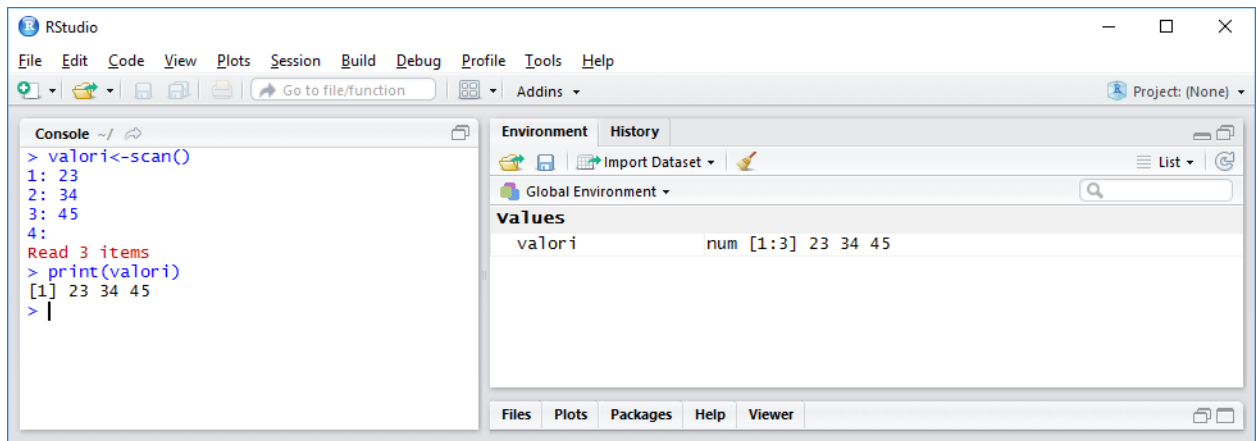
În exemplul de mai sus am omis parametrul `length` dacă îl utilizăm atunci vom obține un vector de valori de lungime `length` și lăsăm mediul R să calculeze pasul.



1.4.2.3 Funcția scan()

Citește de la tastatură sau dintr-un fișier câte un rând.

Exemplu citire de la tastatură:



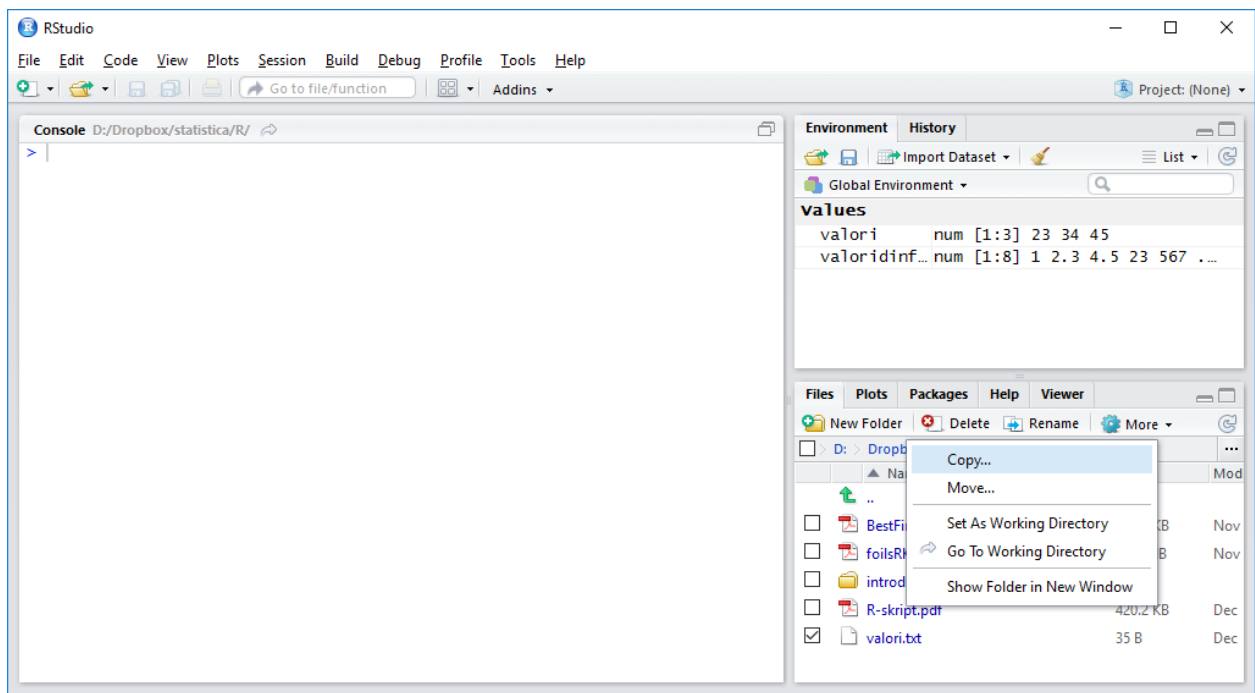
Observăm că citirea se încheie dacă la un pas apăsăm doar ENTER fără să mai adăugăm o valoare.

Exercițiu 1.6

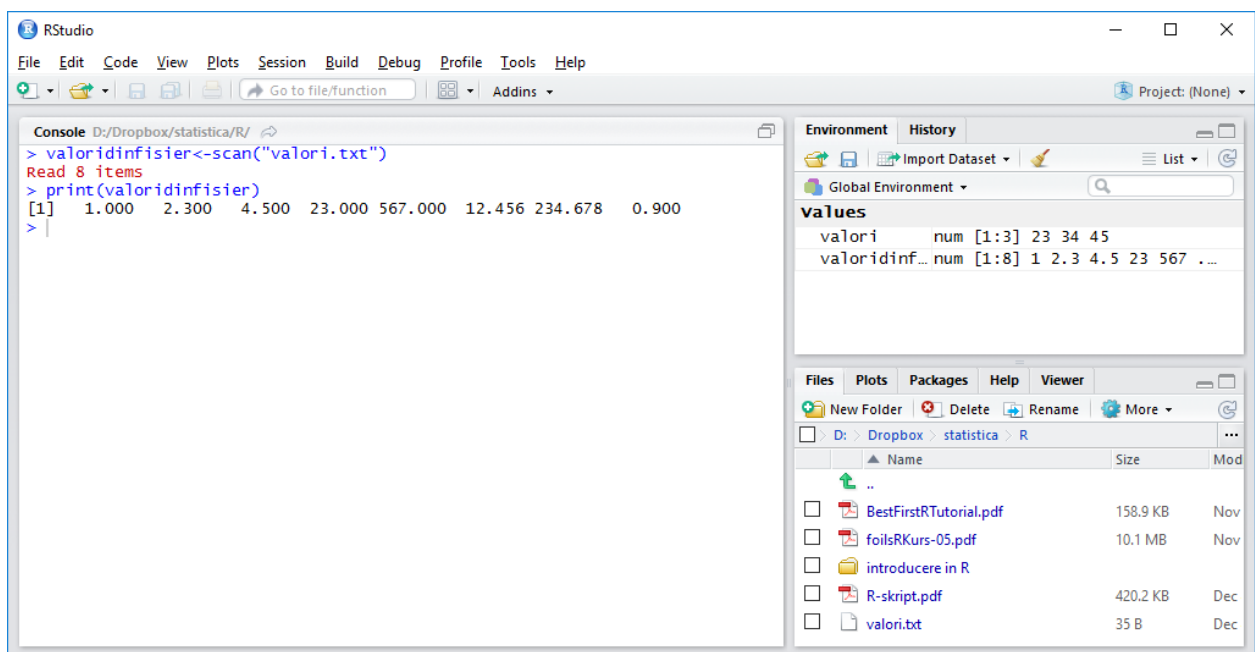
Care este o altă modalitate de a obține vectorul "valori"?

Citirea dintr-un fișier cu ajutorul funcției `scan()`

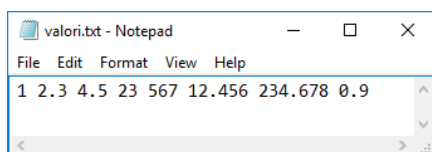
Exemplu citire dintr-un fișier „valori.txt”. Numele fișierului trebuie dat ca parametru între ghilimele duble funcției `scan("valori.txt")`. Aici apare problema locației fișierului. Prima varianta ar fi să folosim calea absolută spre fișier sau să stabilim un director de lucru și să copiem fișierul în acel director. Stabilirea directorului de lucru se poate face dând click pe meniul MORE din panelul de FILES și alegând opțiunea "Set as working directory" directorul curent.



După stabilirea directorului de lucru (a se vedea bara consolei):



Datele din fisier



Exerciții

1. În obiectele `alfa`, `beta`, `gama1` și `gama2` atribuiți 20 de valori utilizând funcțiile `c()`, `seq()`, `scan()` și `scan("fisier.txt")` și apoi calculați media aritmetică pentru fiecare obiect.

2 Creați vectorii:

(a) $(1, 2, 3, \dots, 19, 21)$

(b) $(30, 29, \dots, 2)$

(c) $(1, 2, 3, \dots, 15, 20, 19, 18, \dots, 2, 1)$

(d) atribuiți obiectului `baburiba` valorile $(5, 9, 1)$.

(e) $(5, 9, 1, 5, 9, 1, \dots, 5, 9, 1)$ unde 5 apare de 10 ori.

(f) $(5, 9, 1, 5, 9, 1, \dots, 5, 9, 1, 5)$ unde 5 apare de 11 ori, 9 de 10 ori și 1 de 10 ori.

(g) $(5, 5, \dots, 5, 9, 9, \dots, 9, 1, 1, \dots, 1)$ unde 5 apare de 10 ori, 9 apare de 20 de ori și 1 apare de 30 ori.

3. Creați un vector de valori $e^x \cos(x)$ pentru $x = 3, 3.1, 3.2, \dots, 6$.

4. Calculați suma elementelor vectorului `baburiba`

5. Adunați valoarea 1 tuturor elementelor vectorului `baburiba`.

1.4.3 Calculul cu vectori

1.4.3.1 Operații aritmetice simple

Calculul pe vectori se efectuează pe fiecare componentă a vectorului în parte. Aplicarea de funcții simple asupra unui vector se face tot respectând același principiu.

Explicați exemplul de mai jos:

```
> x<-1:4
> y<-rep(2,4)
> x
[1] 1 2 3 4
> y
[1] 2 2 2 2
> x+y
[1] 3 4 5 6
> x*y
[1] 2 4 6 8
> sqrt(x)
[1] 1.000000 1.414214 1.732051 2.000000
> x^2
[1] 1 4 9 16
```

În cazul în care vectorii sunt de lungimi diferite se repetă valorile din vectorul mai “scurt” de atâtea ori până când ajunge la lungimea celui mai lung. Apoi se efectuează operațiile aritmetice de bază.

De exemplu:

```
> x<-1:3
> y<-rep(10,5)
> x
[1] 1 2 3
> y
[1] 10 10 10 10 10
> x+y
[1] 11 12 13 11 12
warning message:
In x + y : longer object length is not a multiple of shorter object length
> y+x
[1] 11 12 13 11 12
warning message:
In y + x : longer object length is not a multiple of shorter object length
```

Se observă faptul că valorile din vectorul x au fost repetate până când a ajuns la lungimea 5, apoi a fost efectuată operația de adunare componentă cu componentă. Evident că adunarea este comutativă.

Exercițiu

1. Atribuiți în vectorul x, 5 valori, și în vectorul z 4 valori.

Calculați:

$x*z$; $x+z$; x^z

2. Atribuiți atâtea valori lui x și y astfel încât să nu obținem mesajul de avertizare la operațiile pe vectori.

1.4.3.2 Aplicarea de funcții pe vectori

`length()` – determină lungimea vectorului

`t()` – calculează vectorul transpus

`sort()` – sortează un vector crescător

`sum()` – însumează valorile unui vector

`min()` – valoarea minimă a vectorului

`max()` – valoarea maximă a vectorului

```
> x
[1] 2 3 4 5
> y
[1] 1 2 3 4 5 6 7 8
> length(x)
[1] 4
> length(y)
[1] 8
> t(x)
     [,1] [,2] [,3] [,4]
[1,] 2    3    4    5
> t(y)
     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
[1,] 1    2    3    4    5    6    7    8
```

Observație: rezultatul transpunerii unui vector este o matrice. A se vedea secțiunea 1.4.4

```
> c<-c(2, -1,3.4,-34,11,45,-2.4)
> c
[1] 2.0 -1.0 3.4 -34.0 11.0 45.0 -2.4
> sort(c)
[1] -34.0 -2.4 -1.0 2.0 3.4 11.0 45.0
> sort(c, decreasing = TRUE)
[1] 45.0 11.0 3.4 2.0 -1.0 -2.4 -34.0
```

```
> sum(c)
```

```
[1] 24
```

```
> min(c)
[1] -34
> max(c)
[1] 45
```

1.4.3.3 Extragerea de subvectori

Pentru a extrage valorile unui subvector dintr-un vector dat se pot alege diverse modalități pentru a alege indecșii din vectorul curent

a. stabilirea unui interval

```
> c<-c(2, -1,3.4,-34,11,45,-2.4)
> c
[1] 2.0 -1.0 3.4 -34.0 11.0 45.0 -2.4
```

Pentru alegerea primelor 3 valori stabilim intervalul 1:3

```
> c[1:3]
[1] 2.0 -1.0 3.4
```

b. alegerea unor indecși cu funcția c()

```
> c[c(1,3,5)]
[1] 2.0 3.4 11.0
```

c. alegerea unor valori din vectori prin utilizarea unor expresii logice

```
> c[c>2]
[1] 3.4 11.0 45.0
```

Sau utilizând expresii compuse

```
> c[c>2 & c^2<100]
[1] 3.4
```

Operatorii logici care pot fi utilizați:

<	mai mic ca
>	mai mare ca

<=	mai mic sau egal ca
>=	mai mare sau egal ca
==	egal
!=	diferit
	sau pe fiecare element
	sau
!	not
&	și pe fiecare element
&&	și
xor(a,b)	sau exclusiv

Atenție:

```
> c[c>2 && c^2<100]
numeric(0)
```

Există și funcția `subset()` care în principiu face același lucru ea putând fi utilizată și la alte tipuri de obiecte.

```
> subset(c, c>2 & c^2<100)
[1] 3.4
```

1.4.3.4 Vectori nenumerici

În afară de vectorii numerici există și alte tipuri de vectori cum ar fi vectorii de tip string și vectori care conțin expresii logice.

Acești vectori se declară și se accesează exact ca și vectorii numerici:

```
> mama<-c("Cretulescu", "Elena","TRUE")
> tata<-c("cretulescu","vasile", "gheorghe")
> mama
[1] "Cretulescu" "Elena"
> tata
[1] "cretulescu" "vasile"      "gheorghe"
```

```
> L = c(FALSE, TRUE, FALSE, TRUE, FALSE)
```

Dacă dorim să alegem acum valori din vectorul tata bazându-ne pe vectorul de valori logice L

```
> L=c(TRUE,FALSE,TRUE)
> tata
[1] "cretulescu" "vasile"      "gheorghe"
> tata[L]
[1] "cretulescu" "gheorghe"
```

Denumirea valorilor vectorului;

```
> names(tata)<-c("Nume","Prenume0","Prenume 1")
> tata
      Nume      Prenume0      Prenume 1
"cretulescu" "vasile"    "gheorghe"
```

Evident acum putem accesa valoarea vectorului prin numele cheii

```
> tata["Nume"]
      Nume
"cretulescu"
```

1.4.4 Array-uri

Array-urile sunt vectori cu dimensiune dată. Atribuirea valorilor într-un array se face apelând funcția `array(data=, dim=)`

```
> s<-array(c(1:24),24)
> s
[1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
24
```

Exercițiu:

Generați doua array-uri de dimensiune 10 și 12 și adunați aceste două array-uri.

1.4.5 Matrici

Sintaxa:

```
matrix(data, nrow=, ncol=, byrow=FALSE)
```

```
> t<-matrix(0,2,3)
> t
      [,1] [,2] [,3]
[1,]    0    0    0
[2,]    0    0    0
```


Pentru a declara o matrice se pot utiliza și denumirile explicite ale parametrilor:

```
> s<-matrix(1,nrow=12,ncol=3)
> s
      [,1] [,2] [,3]
[1,]    1    1    1
[2,]    1    1    1
[3,]    1    1    1
[4,]    1    1    1
[5,]    1    1    1
[6,]    1    1    1
[7,]    1    1    1
[8,]    1    1    1
[9,]    1    1    1
[10,]   1    1    1
[11,]   1    1    1
[12,]   1    1    1
```

Pentru a introduce in matrice valori le citim prima într-un vector apoi folosim acel vector ca vector de date ca prim parametru în funcția matrix().

```
> valori<-c(1:24)
> mm<-matrix(valori,nrow=4,ncol=3)
> mm
      [,1] [,2] [,3]
[1,]    1    5    9
[2,]    2    6   10
[3,]    3    7   11
[4,]    4    8   12
```

Valorile se pot introduce prin completarea rândurilor utilizând parametrul byrow=TRUE:

```
> mm<-matrix(valori,ncol=3, byrow = TRUE)
> mm
      [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
[4,]   10   11   12
[5,]   13   14   15
[6,]   16   17   18
[7,]   19   20   21
[8,]   22   23   24
```

Citirea unei matrice dintr-un fișier:

```
> ma<-matrix(scan("fismeu.txt"), ncol=2, byrow=TRUE)
Read 8 items
> ma
      [,1] [,2]
[1,]    1    2
[2,]    4    2
[3,]   56   12
[4,]  234    0
```

Pentru simplificarea adresării unei linii sau coloane ca și în cazul vectorilor coloanele sau liniile pot fi denumite.

```
colnames(x) <- value
rownames(x) <- value
```

value este un vector cu aceeași dimensiune ca și dimensiunea coloanelor sau a rândurilor

```
> colnames(ma)
NULL
> colnames(ma)<-c("mere", "pere")
> ma
      mere pere
[1,]     1    2
[2,]     4    2
[3,]    56   12
[4,]   234    0
```

Accesarea unui element din matrice se poate face acum:

```
> ma[1,"mere"]
mere
      1
> ma[, "mere"]
[1]  1  4 56 234
```

Exerciții:

1. Să se creeze următorii vectori:

a.) $s = (0.1^3 \cdot 0.2^1, 0.1^6 \cdot 0.2^4, \dots, 0.1^{36} \cdot 0.2^{34})$

b.) $m = \left(2, \frac{2^2}{2}, \frac{2^3}{3}, \dots, \frac{2^{25}}{25} \right)$

2. Să se calculeze:

a.) $\sum_{i=10}^{100} (i^3 + 4i^2)$

b.) $\sum_{i=1}^{25} \left(\frac{2^i}{i} + \frac{3^i}{i^2} \right)$

1.4.5.1 Calculul cu matrici

Pentru a accesa un rând sau o coloană dintr-o matrice se utilizează notația [nr_rând ,] sau [, nr_coloană]

```
> mm
      [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
[4,]   10   11   12
[5,]   13   14   15
[6,]   16   17   18
[7,]   19   20   21
[8,]   22   23   24
>
> mm[,1]
[1]  1  4  7 10 13 16 19 22
> mm[2,]
[1]  4  5  6
```

1.4.5.1.1 Înmulțirea matricilor componentă cu componentă

Se folosește operatorul classic *:

```
> val1<-c(1:4)
> val2<-c(1,0,0,1)
> a<-matrix(val1,2,2)
> b<-matrix(val2,2, byrow = TRUE)
> a
      [,1] [,2]
[1,]    1    3
[2,]    2    4

> b
      [,1] [,2]
[1,]    1    0
[2,]    0    1

> a*b
      [,1] [,2]
[1,]    1    0
[2,]    0    4
```

1.4.5.1.2 Înmulțirea matricilor clasică

Se folosește operatorul %*%:

```
> val1<-c(1:4)
> val2<-c(1,0,0,1)
> a<-matrix(val1,2,2)
> b<-matrix(val2,2, byrow = TRUE)
```

```
> a
      [,1] [,2]
[1,]    1    3
[2,]    2    4

> b
      [,1] [,2]
[1,]    1    0
[2,]    0    1

> a%%b
      [,1] [,2]
[1,]    1    3
[2,]    2    4
```

1.4.5.2 Funcții aplicate matricilor

Adăugarea unei linii sau coloane

```
rbind(vector1, vector2,...)
cbind(vector1, vector2,...)
```

Evident în cazul adăugării unui rând cu ajutorul funcției `rbind()` numărul de valori din vectorul de adăugat trebuie să fie identic cu numărul coloanelor matricei.

```
> ma
      mere pere
[1,]    1    2
[2,]    4    2
[3,]   56   12
[4,]  234    0

> rbind(ma, c(0,0))
      mere pere
[1,]    1    2
[2,]    4    2
[3,]   56   12
[4,]  234    0
[5,]    0    0

> cbind(ma, c(1,1,1,1,1))
      mere pere
[1,]    1    2  1
[2,]    4    2  1
[3,]   56   12  1
[4,]  234    0  1
Warning message:
In cbind(ma, c(1, 1, 1, 1, 1)) :
  number of rows of result is not a multiple of vector length (arg 2)

> cbind(ma, c(1,1,1,1))
      mere pere
[1,]    1    2  1
[2,]    4    2  1
[3,]   56   12  1
[4,]  234    0  1
```

În cazul în care numărul valorilor din vectorul pe care îl adăugăm este diferit atunci apare un mesaj de avertizare cum se poate vedea din exemplul de mai sus.

Transpunerea unei matrici

Se efectuează cu ajutorul funcției `t ()`

```
> ma
      mere pere
[1,]      1    2
[2,]      4    2
[3,]     56   12
[4,]    234    0
> t(ma)
      [,1] [,2] [,3] [,4]
mere      1    4   56  234
pere      2    2   12    0
```

>

Calculul inversei unei matrici

Din calculul matricial se cunoaște relația:

$$A \cdot A^{-1} = I \text{ unde } I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

Pentru a calcula inversa A^{-1} a matricii A vom utiliza funcția `solve()`. Această funcție rezolvă ecuații de forma

`a %*% x = b` unde a și b pot fi și matrici

```
> matricea<-matrix(c(1,2,3,5), ncol=2, byrow = TRUE)
> matricea
      [,1] [,2]
[1,]      1    2
[2,]      3    5
> b<-matrix(c(1,0,0,1), ncol=2, byrow=TRUE)
> b
      [,1] [,2]
[1,]      1    0
[2,]      0    1

> matricea_inv2<-solve(matricea,b)
> matricea_inv2
      [,1] [,2]
[1,]     -5    2
[2,]      3   -1
```

Verificați!

1.4.6 Liste

În R o listă este în esență un vector de vectori care poate să conțină tipuri de date diferite.

Pentru a crea o listă se apelează funcția `list()`.

```
> lista1<-list(valori=c(1,2,3), caractere=c("Ana", "Livia", "Radu"))
> lista1
$valori
[1] 1 2 3

$caractere
[1] "Ana" "Livia" "Radu"
```

Pentru a accesa elemente din listă se poate utiliza caracterul `$`

```
> lista1$valori
[1] 1 2 3
> lista1$valori[1]
1
> lista1$valori[2]
[1] 2

> lista1$caractere[1]
[1] "Ana"
```

>

1.4.7 Data Frames

1.4.7.1 Crearea Data Frames

Un `data.frame` este o combinație de listă și vector. Este o listă în care vectorii au aceeași lungime dar care conțin ca elemente obiecte diferite. Un `data.frame` poate fi de asemenea considerat ca fiind o generalizare a unei matrici. Este, probabil, cel mai important tip de date în R, deoarece datele de analizat se prezintă în astfel de structuri de date.

Crearea unui `data.frame` se poate face prin apelul următoarelor funcții:

```
data.frame(obiect1, obiect 2,...)
```

```
> x <- data.frame("masa"=c(65,75),"inaltime"=c(168,175),"gen"=c("m","f"))
> x
  masa inaltime gen
1   65     168   m
2   75     175   f
```

sau citind datele dintr-un fișier salvate sub formă tabelară de exemplu de tip csv

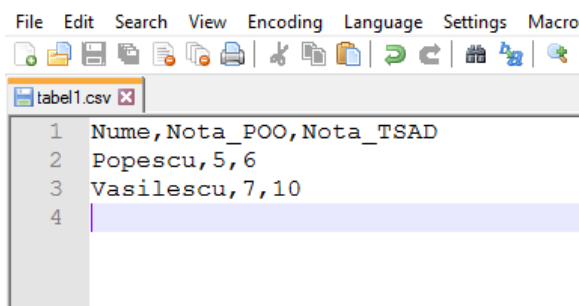
```
> tabel<-read.table("tabel1.csv")
> tabel
           V1
1 Nume,Nota_POO,Nota_TSAD
2      Popescu,5,6
```

```
3         vasilescu,7,10
```

În exemplul de mai sus au fost citite toate rândurile din fișierul csv antetul de tabel fiind considerat set de date. În exemplul următor s-au citit datele utilizând primul rând ca denumire a coloanelor.

```
> tabel<-read.table("tabel1.csv", header=TRUE, sep=",")
> tabel
      Nume Nota_POO Nota_TSAD
1  Popescu         5         6
2 Vasilescu         7        10
```

Fișierul csv folosește ca separatori semnul „, ”



Exemple de acces a datelor din tabel și utilizarea acestora

```
> tabel
      Nume Nota_POO Nota_TSAD
1  Popescu         5         6
2 Vasilescu         7        10
> Nota_grafica<-c(8,8)
> tabel_modificat<-cbind(tabel,Nota_grafica)
> tabel_modificat
      Nume Nota_POO Nota_TSAD Nota_grafica
1  Popescu         5         6           8
2 Vasilescu         7        10           8
> tabel_modificat$Nota_POO
[1] 5 7
> mean(tabel_modificat$Nota_POO)
[1] 6
```

Scrierea datelor obținute în fișiere.

```
> write.table(tabel_modificat, file= "tabel_modificat1.txt", sep=",")
> write.csv(tabel_modificat,file="tabel_modificat2", sep=",")
```

Exerciții:

1. Adăugați în obiectul tabel_modificat încă 4 rânduri
2. Afișați toate rândurile care au Nota_POO > 3

3. Afișați toate rândurile care au $\text{Nota_POO} > 7$ și $\text{Nota_TSAD} > 9$

1.5 Funcții

1.5.1 Funcții matematice predefinite

Limbajul **R** conține o serie de funcții matematice predefinite. Aceste funcții se aplică atât valorilor discrete cât și vectorilor/matricilor (pe fiecare componentă). În continuare vom prezenta doar o mica parte a acestor funcții

<code>exp()</code>	Exponențială
<code>log()</code>	Logaritm în baza e
<code>logb(x, base=2)</code>	Logaritm în baza base
<code>sqrt()</code>	Rădăcină pătratică
<code>sin()</code>	Sinus
<code>cos()</code>	Cosinus
<code>tan()</code>	Tangentă

Funcții pe vectori/matrici

<code>min(), max()</code>	Minim, maxim
<code>sort()</code>	Sortare
<code>sum()</code>	Suma elementelor
<code>prod()</code>	Produsul elementelor
<code>diff()</code>	Diferența dintre elementele vectorului $x[i]-x[i-1]$
<code>range()</code>	Cea mai mică și cea mai mare valoare

Alte funcții utile:

Calculul integrale definite cu ajutorul funcției `integrate()`

```
> integrate(sin,-pi/2,pi/2)
0 with absolute error < 2.2e-14
> integrate(sin,-pi/2,pi/2)$value
[1] 0
```

Funcția `summary()`

```
> p<-1:20
> summary(p)
  Min. 1st Qu.  Median    Mean 3rd Qu.   Max.
  1.00   5.75   10.50   10.50   15.25   20.00
```

1.5.2 Definirea de funcții noi

Funcțiile definite în R respectă principiul apelului prin valoare

```
> f<-function(x){
+ x<-x+2
+ return(x^2)
```



```
+ }
> x<-2
> f(x)
[1] 16
```

Alt exemplu în care putem să dăm parametric cu valori implicite

```
> radical<-function(p, n=2){
+ sol<-p^(1/n)
+ return(sol)
+ }
> radical(4)
[1] 2
> radical(27,3)
[1] 3
```

Funcții care primesc ca paramteri liste:

Definim funcția functiaMea

```
> functiaMea <- function(x,n){
+ d<-x-n
+ m<-x*n
+ rezultat<-list(diferenta=d,produsul=p,x=x,n=n)
+ return(rezultat)
+ }
> functiaMea(2,4)
Error in functiaMea(2, 4) : object 'p' not found
```

Apare o problemă deoarece atribuirea `produsul=p` nu este posibilă deoarece nu am folosit în funcție variabila `p` ci `m`. Pentru a modifica funcția apelăm funcția `fix(ume funcție)` și putem edita apoi funcția noastră.

```
> fix(functiaMea)
> functiaMea(2,4)
$diferenta
[1] -2

$produsul
[1] 8

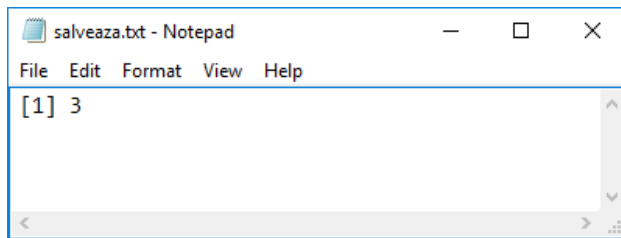
$x
[1] 2

$n
[1] 4
```

1.5.3 Scrierea în fișiere

Funcția `sink()` redirecționează outputul de la consolă în fișier.

```
> sink("salveaza.txt")
> 1+2
> sink()
```



2 Distribuții

Generarea de numere distribuite conform unei distribuții cunoscute se face utilizând funcțiile:

distribuție() Densitate
p distribuție() Funcția de probabilitate a distribuției
q distribuție() Quantile
r distribuție() Numere aleatoare

unde valorile pentru *distribuție*: norm, unif, exp, pois, binom, t, f, chisq etc.

De exemplu pentru generarea unui set de 100 de numere uniform distribuite cu media aritmetică =2 și deviația standard=0.1 folosim funcția `rnorm`

```
> numere<-rnorm(100,mean=2, sd=0.1)
> numere
 [1] 1.996208 1.924471 1.995623 1.955869 1.934958 1.985129 1.951143 1.984
515 2.115971 2.064433
 [11] 2.028890 1.996545 2.087308 1.880035 1.955719 2.151568 1.935364 2.190
617 1.886136 2.048633
 [21] 2.027011 1.922421 2.180685 2.045044 1.869331 2.064747 1.952784 1.939
662 1.906126 1.897287
 [31] 2.073637 2.040782 2.010664 2.084951 2.018086 2.138636 1.975646 2.135
113 2.088036 1.936401
 [41] 1.976402 1.982991 1.853243 2.080560 1.936877 2.086589 1.909468 2.013
439 1.926444 1.834750
 [51] 1.785049 2.144798 1.922648 1.949003 1.970682 2.015880 1.984176 1.892
246 1.991920 1.944621
 [61] 1.979957 1.974352 2.110393 2.038215 2.270807 1.998156 1.880609 1.984
951 1.887727 1.877901
 [71] 1.925445 1.839687 2.092676 1.958135 1.885875 2.042546 2.083896 2.077
297 2.023613 1.729689
 [81] 1.944807 1.944213 1.856021 1.875772 2.100604 1.940077 1.829855 2.075
180 2.122540 1.861886
 [91] 2.062329 2.204146 1.904063 2.165706 1.997327 1.987452 1.879956 2.094
658 2.033153 2.133163
```

Exerciții

1. Generați 50 de numere cu distribuție normală și media -1 iar abaterea standard =3
2. Generați 20 de numere cu distribuția binomială (Atenție la argumentele necesare funcției `rbinom`)

3 Histograme

3.1 Tipărirea de histograme

Dorim să tipărim histograma pentru 1000 de valori cu distribuție binomială unde numărul de încercări $n=10$ iar probabilitatea $p=0.4$

```
dbi<-rbinom(1000, 10, 0.4)
> par(mfrow=c(1,3))
> hist(dbi,freq=FALSE,main="Histograma cu hist")
> hist(dbi,freq=TRUE,main="Histograma cu hist")
```

Utilizând al doile parametru cu TRUE se obține histograma pe frecvențe iar în celălalt caz se obțin frecvențe relative (densitate)

