Instalare R și R studio

Exemplele de la această disciplină vor fi realizate în limbajul de programare R. Putem descărca R de la adresa https://cloud.r-project.org, unde este găzduit proiectul CRAN – the **c**omprehensive **R** **a**rchive **n**etwork ce permite distribuția limbajului dar și a pachetelor suplimentare.

RStudio este un mediu de programare (IDE) pentru limbajul R. Se va downloada și instala separat de la adresa http://www.rstudio.com/download.

Familiarizare limbaj R

Consola va fi folosită pentru a executa câte o comandă, însă dacă dorim să executăm mai multe deodată vom crea un fișier cu extensia .R cu click pe pictograma Run. Pentru a executa o comandă din scriptul R, ne poziționăm pe comanda respectivă și tastăm Crtl+Enter. Se recomandă ca o sesiune de lucru să fie salvată intr-un script – pentru ca toate comenzile executate să rămână disponibile, si comenzile să fie rulate in consolă cu comanda Crtl+Enter.

Executați următoarele comenzi în consolă (explicațiile sunt date sub formă de comentarii – precedate de semnul #). Rezultatele nu sunt afișate. Veți observa că majoritatea rezultatelor sunt afișate după cifra 1 încadrată de paranteze []. Aceasta arată că rezultatele afișate încep de la primul element (și valorile simple precum numere, caractere etc sunt văzute sub forma unui vectori de dimensiune 1).

```
#adauga valoarea 12 la variabila A și 17 la B; pentru a crea o variabila ne folosim de semnele <- sau =
A<-12
B=17
#observav faptul ca R este un limbaj case sensitive. Prin urmare, A si a pot desemna doua variabile diferite

#pentru afișarea în consolă a valorii unei variabile e suficientă numirea variabilei
A

#funcția class() ne arată tipul unei variabile
#variabila A va avea tipul numeric
class(A)

#tipul integer este un subtip al tipului numeric, pentru a indica tipul integer la o varibilă adăugăm litera L la final; suprascriem valoarea B
B<-12L

#operatorii aritmetici: +,-,*,/,^ (ridicare la putere), %/%(câtul împărțirii), %%(restul împărțirii)
2^3
13%/%2
13%%2
#spre deosebire de alte limbaje de programare în R există reprezentare pentru conceptul Infinit
a=5/0
c = Inf
exp(a) #Inf
exp(-a) # 0

#putem genera o secvență regulate de numere întregi de ex de la 1 la 10- un vector de numere [ntregi de la 1 la 10
# elementele unui vector întotdeauna sunt de același tip
a<-1:10

#operatorul : nu trebuie confundat cu impartirea;
1:10-1  #1 2 3 4 5 6 7 8 9

#pentru secvențe putem folosi și funcția seq() ce ne permite specificarea unui pas
seq(1,5,0.5) #[1] 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0
```

#putem crea secvențe de numere repetitive
rep(1,10) #repetă 1 de 10 ori

Secvențele generate vor avea un rol important în generarea datelor de test.

#putem verifica apartenența la un tip cu funcția is.*tip()*
is.numeric(A)
is.integer(A)

#putem schimba tipul variabilei cu funcția as.*tip()*
as.character(A)

variabilele create se acumulează într-o zonă de memorie numită workspace; putem să vizualizăm conținutul acesteia în panoul
#Global Environment (dreapta sus) fie folosind funcția ls(), care enumereaza variabilele disponibile
ls()

#comanda ls() returnează doar numele obiectelor din memorie; folosim ls.str() pentru mai multe detalii precum tip și valoare
ls.str()

#zona de memorie poate ocupa o dimensiune considerabilă în special când se lucrează cu seturi mari de date
#este nevoie uneori să ștergem o variabilă din workspace folosind funcția rm(variabila)
rm(B)
#putem șterge toată memoria în felul următor
rm(list=ls())

Formule

În R formulele au rolul de a exprima un model statistic ce se stabilește între o variabilă dependentă și alte variabile independente. Putem defini o formulă între variabila Y (variabila dependentă) și variabila X (variabilă independentă): observați operatorul ~ (tildă) ce separă cele două variabile.

#atribuim formule variabilelor c și d
c<- Y~X
d<-Y~2*X+e

Formulele au avantajul că pot fi definite folosind variabile care încă nu au fost inițializate (atât variabila Y cât și X nu există încă și nu sunt create prin folosirea lor în formule). Acest lucru este deosebit de important deoarece ne permite ca apelul funcțiilor să conțină referințe la variabile fără ca acestea să fie evaluate.

Vectori

#pentru a crea un vector în R folosim și de functia c();
#creăm un vector studenti cu cateva nume
studenti<-c("Ana", "Cristina", "Bob")

#indexarea elementelor dintr-un vector incepe de la elementul 1, de exemplu:
studenti[1]

#accesarea primului și celui de-al treilea element din vector printr-o enumerare a indecsilor
studenti[c(1,3)]
studenti[c(3,1)]

#creăm un vector cu note
note<-c(10,6,7,8,6)

#notele, în felul curent, sunt lipsite de înțeles, le putem adăuga niște etichete
#etichetele le avem într-un alt vector ce reprezintă disciplinele, poate fi de lungime diferita
discipline<-c("informatica","matematica", "engleza","contabilitate", "finante")

#adăugăm etichetele la vectorul note prin functia names(); obținem un *named vector.* l
names(note)<-discipline

#putem crea un vector cu etichete intr-un singur pas; nu mai este neapărat nevoie să folosim ghilimenle la cheile atasate
altenote<-c(informatica=9, matematica=5, engleza=8, contabilitate=7)

#accesăm valoarea unei note folosindu-ne de eticheta sau cheia atașată; e nevoie de ghilimele la specificarea cheii
note["informatica"]

#valorile pentru informatica și engleza
note[]

#accesăm valorile de la elementul 2 la elementul 3
note[2:3]

#testăm dacă variabile create sunt vectori; chiar și variabila X este un vector de dimensiune 1
is.vector(studenti)
is.vector(note)
is.vector(A)
length(A)

#operațiile cu vectori se aplică element cu element (element-wise); creștem notele cu un pct
note+1

Matrice

Pot conține doar aceleași tipuri de valori. O metodă de a crea matrice este funcția **matrix()** ce preia ca intrare un *vector* cu valorile ce vor forma conținutul matricei și cel puțin o dimensiune (numarul de linii prin argumentul nrow sau numarul de coloane prin argumentul ncol; daca se specifica doar unul din aceste argumente, R va deduce automat pe celalalt). Implicit valorile in matrice sunt completate pe coloane:

#crează o matrice de 2x3 cu elemente de la 1 la 6. Observăm că valorile sunt completate implicit pe coloane.
d <- matrix(1:6, nrow=2)

#prin byrow specificăm că dorim completarea valorilor pe linie
matrix(1:6, nrow=2, byrow=TRUE)

completare ciclica a elementelor matricei: se refolosesc elementele de intrare pentru a completa toate valorile din matrice, folosindu-se ordinea ce completare implita (pe coloane) sau cea specificata
matrix(1:3, nrow=2, ncol=3)
#apare un mesaj de avertizare doar daca nu sunt refolosite din nou toate elementele de intrare
matrix(1:4, nrow=2, ncol=3)

Alte metode de a crea matrice sunt funcțiile **rbind()** și **cbind()** ce unesc vectorii sub forma liniilor dintr-o matrice respectiv a coloanelor. Dacă intrarile nu sunt de dimensiune egala se vor refolosi elementele din vectorul mai mic pentru a-l aduce la aceeași dimensiune cu celălalt.

#crează o matrice din unirea a două coloane cu elementele de la 1 la 3; respective din unirea a două linii
cbind(1:3, 1:3)
rbind(1:3,1:3)

Funcțiile rbind() si cbind() pot fi folosite si pentru a adăuga linii sau coloane unei matrice existente:

#creăm o matrice m 2x3
m<-matrix(1:6, nrow=2)
#adăugăm o coloană nouă cu numerele 7 si 8
m <- cbind(m, 7:8)

În cazul vectorilor foloseam funcția names() pentru a adăuga etichete elementelor din vector. Pentru matrice avem funcțiile rownames() și colnames()

#adăugăm denumiri la linii
rownames(m)<-c("x1","x2")
#adăugăm denumiri la coloane
colnames(m)<-c("y1","y2", "y3", "y4")

Selecția elementelor din matrice

extragem ultimul element al matricei m – 6
m[2,3]
#prin omiterea indexului de linie sau coloana extragem vectori - extragem ultima linie din matricea m – 2 4 6 8
m[2,]
#extragem prima coloana din m – 1 2
m[,1]
#daca transmitem doar un singur număr x, R va număra elementele din matrice pe coloană si va returna al x-lea număr
m[5]

Putem specifica elemente multiple in selecția noastră enumerându-le printr-un vector

#extrage de pe randul doi, al doilea si al treilea element
m[2,c(2,3)]
m[2, 2:3]

OBS. nu putem selecta elemente din matrice ce nu au in comun vreun index de linie sau coloana; de ex primul element de pe prima linie si ultimul element de pe ultima linie:

#se returnează o submatrice formată din liniile specificate prin primul vector si coloanele specificate de al doilea vector
m[1:2,1:2]

Selecția multiplă a elementelor mai poate fi făcută și cu ajutorul vectorilor cu valori logice, aplicând aceleași reguli:

#selecție de pe linia 2 a elementelor de pe a 2 si a treia coloana, specificăm sub forma a doi vectori corespunzători dimensiunilor m și n
m[c(FALSE,TRUE),c(FALSE,TRUE,TRUE)]

Selecția elementelor din matrice poate fi făcută folosind numele atribuite liniilor și coloanelor în același mod în care am folosit indecșii dar adăugând ghilimele.

Operații cu matrice

#adunarea coloanelor într-un vector ce contine sumele pe coloane
colSums(m) #ne va da vectorul 3 7 11 15
#adunarea liniilor într-un vector ce contine sumele pe linii
rowSums(m) # ne va da vectorul 16 si 20

Operațiile matematice pe matrice sunt asemănătoare cu cele realizate pe vectori – element cu element.

#dublarea elementelor din matricea m
m*2

#operațiile cu vectori- se aplică de asemenea element cu element pe coloana; in cazul in care dimensiunea vectorului este mai mica decât a matricei se realizează reciclarea
#scăderea nr 1 din primul element, nr 2 din al doilea element de pe prima coloana etc.
m-c(1,2)

#operațiile cu alte matrici – de asemenea element cu element
#se va inmulti elementul m[1,1] cu elementul m[1,1], elementul m[1,2] cu elementul m[1,2] si asa mai departe
m*m

Funcția dim() ne dă dimensiunea matricei:

Factori

Factorii sunt acele structuri de date de tip vectori care sunt specializate în păstrarea variabilelor de tip categorie (variabile cu un număr limitat de valori diferite) sau variabile discrete. Sunt create cu ajutorul funcției factor() pornind de la un vector.

```
#fie vectorul următor
days<-c("Luni", "Marti","Luni", "Miercuri", "Joi", "Vineri", "Marti", "Miercuri", "Joi", "Sambata", "Duminica")
#creăm factorul
days_factor<-factor(days)
```

R realizează 2 operații la apelul funcției factor() asupra unui vector de caractere:

1) identifică toate categoriile care există – în cazul nostru sunt date de denumirea unică a zilelor din săptămână (în Levels) și le sortează alfabetic;

2) convertește vectorul de caractere într-un vector cu valori întregi pe care îl va folosi la afișare

```
#avem un factor cu 7 nivele, categorii: Duminica e primul element și va fi codat cu 1, urmează Joi codat cu 2 samd.; acest lucru se vede inspectând structura factorului cu funcția str()
str(days_factor)
```

Dacă dorim să avem o altă ordonare decât cea implicită (alfabetică) putem să o specificăm folosind argumentul levels în cadrul funcției factor()

```
#crearea unui factor cu categorii ordonate
days_factor2<-factor(days, levels=c("Luni","Marti","Miercuri", "Joi", "Vineri", "Sambata", "Duminica"))
```

Numele de categorii sunt derivate din denumirile elementelor din vectorul inițial însă putem să avem denumiri proprii pe care le specificăm prin argumentul labels

```
#următorul exemplu crează factorul cu categorii intr-o anumită ordine și în plus redenumește denumirile categoriilor, noile denumiri vor fi folosite și în afișarea elementelor
days_factor3<-factor(days, levels=c("Luni","Marti","Miercuri", "Joi", "Vineri", "Sambata", "Duminica"), labels=c("L","Ma", "Mi", "J", "V", "S", "D"))
```

Dacă nu am folosit la crearea factorului argumentul labels prin care să atribuim propriile denumiri putem să le schimbăm și ulterior prin funcția levels(), într-un mod asemănător cum am făcut cu funcția names() prin care adăugam etichete unor vectori.

```
#adăugăm denumiri scurte la primul factor creat
levels(days_factor)<-c("D","J", "L", "Ma", "Mi", "S", "V")
#funcția levels afișează categoriile din factor
levels(days_factor)
```

În cazul variabilelor categorice se face deosebirea între cele ordonate și cele neordonate. Exemplul anterior nu exprima în mod explicit o relație de mărime între categorii, cu alte cuvinte nu aveam o variabila ordonată (cum ar putea fi de exemplu dacă am avea categoriile Small, Medium, Large și am dori să exprimăm că Small<Medium<Large).

```
#Dacă am dori să realizăm o comparație cu elementele unui factor neordonat vom primi un mesaj de avertizare
days_factor[1]<days_factor[2]
#recreăm factorul cu argumentul ordered și apoi realizăm din nou comparația
days_factor4<-factor(days, ordered=TRUE, labels=c("L","Ma", "Mi", "J", "V", "S", "D"))
days_factor4[1]<days_factor4[2]
```

Liste

Vectorii sau matricile pastreaza elemente de acelasi tip. Spre deosebire de acestea, listele în R pot să păstreze elemente de tipuri diferite. Listele pot să cuprindă orice obiecte R de la valori simple, vectori, matrice sau alte structuri complexe, precum alte liste, dataframes. Se folosește funcția list() în care transmitem valorile ce vrem să le păstrăm. La fel ca la vectori putem să le atribuim și nume/etichete

#creăm o listă în care vrem să păstrăm informații despre filmul Avatar, precum regizorul (string), anul (int), bugetul (double)
Avatar<-list("James Cameron",2009,237)
#atribuim nume elementelor din listă folosindu-ne de un vector de caractere
names(Avatar)<-c("regizor", "an", "buget")
#la fel ca și în cazul vectorilor putem să specificăm etichetele în momentul creării listelor
Passengers<-list(regizor="Morten Tyldum", an=2016, budget=150)
#deoarece afișarea listelor este destul de voluminoasă, ne putem folosi de funcția str()
str(Avatar)
#creăm o listă pornind de la listele anterioare, adăugând și valoarea "USA" etichetată cu denumirea „origine"
filmeUSA <-list(Avatar, Passengers, origine="USA")

Selecția elementelor din listă prin [] ne returnează alte liste, chiar dacă ar trebui să avem doar o valoare simplă. Folosim [[]] pentru a extrage doar elementele simple.

Extragerea elementelor din listă folosindu-ne de indecși

l<-filmeUSA[3] # aici ne-ar trebui doar valoarea string USA
#verificăm cu funcția is.lst() faptul că se returnează o listă
is.list(l)
#selecția cu paranteze duble ne returnează doar valoarea USA
filmeUSA[[3]]
#folosind paranteze simple se extrage o listă de dimensiune 1
length(filmeUSA[1])
#folosind paranteze duble se extrage sublista, va avea deci dimensiunea egală cu cea a sublistei
length(filmeUSA[[1]])

#pentru a extrage numele regizorului de la primul film, întâi extragem sublista corespunzătoare primului film și apoi primul element din ea
filmeUSA[[1]][[1]]
#același lucru folosindu-ne de un vector cu elementele de selecție
filmeUSA[[c(1,1)]] # numele regizorului din primul film
filmeUSA[[c(1,3)]] # bugetul primului film
#extragerea multiplă de elemente cu paranteze duble dintr-o listă formată doar de elemente simple generează eroare
Avatar[[c(1,3)]] # ar fi echivalent cu Avatar[[1]][[3]]
#selecție multiplă de elemente dintr-o listă
filmeUSA[c(1,3)]

Extragerea elementelor din listă folosindu-ne de nume

#asemănător cu selecția prin index însă ne folosim de numele specificate între ghilimele, pot fi folosite ambele variante de extragere, atât cu paranteze simple cât și duble
filmeUSA["origine"] #rezultatul este o listă
filmeUSA[["origine"]] #rezultatul este stringul "USA"

O modalitate nouă de a extrage elemente din listă este prin folosirea semnului $. Aceasta seamănă cu extragerea folosind paranteze duble însă este posibilă doar pe listele etichetate.

#extragerea valorii USA
filmeUSA$origine

Adăugarea elementelor in lista.

#folosind semnul $
filmeUSA$subtitrare<-"Engleza"

#e echivalent cu următoarea linie de cod
filmeUSA[[“subtitrare”]]<-“Engleza”
#putem să adăugăm fără să folosim o etichetă
filmeUSA[[5]]<-“Sci-Fi”

Ștergerea elementelor din listă se face atribuind valoarea NULL; în cazul în care se șterge un element de pe o poziție frontală restul elementelor ce urmează în listă sunt repoziționate

filmeUSA[[5]]<-NULL

Putem să afișăm elemente din listă cu excepția unora (deci fără să fie șterse propriu zis din listă) dacă transmitem sub forma argumentelor negative cele care dorim să nu fie incluse

#afișăm tot conținutul din listă fără primul element
filmeUSA[-1]
#fără primele primul și al doilea element
filmeUSA[c(1,2)]

Extragerea elementelor din listă folosindu-ne de valorile logice este posibilă doar pentru varianta în care ne folosim de paranteze simple.

#afișăm tot conținutul din listă fără primul element
filmeUSA[c(FALSE,TRUE,TRUE)]
#fără primele primul și al doilea element
filmeUSA[c(FALSE,FALSE,TRUE)]

Data Frames

Sunt structuri de date tabelare specializate pentru a păstra date despre observații (rânduri) și caracteristicile pentru anumite variabile (coloanele), de tipuri diferite.

De obicei nu se vor crea data frames în R ci acestea ajung să fie importate dintr-o altă sursă de date, de ex fișiere csv, excel, un tabel dintr-o bază de date relațională etc.

Dacă dorim să creăm manual structuri de tip data frame în R putem folosi funcția data.frame() căruia îi transmitem ca argumente vectori de aceeași dimensiune ce corespund variabilelor /coloanelor.

#creăm întâi trei vectori ce reprezintă caracteristicile pentru fiecare variabilă/coloană din data frame, cu aceeași dimensiune, altfel avem eroare
nume<-c("Ana","Bob", "Andrei", "Lucia", "Alex")
varsta<-c(30,32,28,27,29)
fumator<-c(FALSE,FALSE,FALSE,TRUE,TRUE)
df<-data.frame(nume,varsta,fumator) #numele coloanelor din setul de date va fi preluat din numele acestor variabile
#pentru a specifica alte nume putem folosi aceeași tehnica de la vectori și liste
#folosind funcția names()
names(df)<-c(“Nume”, “Varsta”, “Fumator”)
#trasnmițând numele în argumente
df<-data.frame(Nume=nume, Varsta=varsta, Fumator=fumator)

Selecția elementelor din data frames

Pentru a selecta valoarea unui singur element din data frame, de pe o anumită linie și coloană putem folosi aceeași sintaxă care o foloseam la extragerea elementelor din matrice: paranteze simple pătrate cu doi indecși în interior

#extragem varsta lui Andrei, ce se află pe linia 3 și coloana 2
df[3,2]

#putem să ne folosim și de numele coloanelor când le referim, același exemplu folosind denumirea coloanei
df[3, "Varsta"]

La fel ca în cazul matricelor putem să omitem unul din indecși pentru a obține o linie sau o coloană întreagă

#extragem toate informațiile despre Andrei
df[3,]
rezultatul e un data frame cu un singur rând/ observație
is.data.frame(df[3,]) #TRUE
#rezultatul poate fi un vector daca se returnează date de același tip
is.data.frame(df[,2]) #FALSE
is.vector(df[,2]) #TRUE
Dacă se specifică un singur index acesta va fi considerat corespunzător coloanelor

#afișează doar coloana Nume
df[1]
#la fel ca în cazul matricei putem să transmitem doi vectori corespunzători dimensiunilor m și n în care să specificăm care linii și care coloane să fie extrase, de pe liniile 3 și 5 si coloanele Varsta și Fumator
df[c(3,5),c("Varsta", "Fumator")]

Elementele din data frame pot fi selectate și cu sintaxă preluată de la liste, respectiv [], [[]], sau $. A se face diferenta între structurile de date returnate de sintaxa [] (structuri de tip data frame) față de cele cu [[]] sau semnul $ (vectori).

#extragerea elementelor de pe coloana Varstă, următoarele comenzi returnează același rezultat, un vector cu elementele Varsta
df$Varsta
df[[""Varsta"]]
df[[2]]

Extinderea unei structuri data frame: adăugare de linii sau coloane noi.

La fel ca în cazul listelor putem folosi semnul $ sau [[]]

#adăugarea coloanei greutate
greutate<-c(63,97,87,60,70)
df$Greutate <-greutate
df[["Greutate"]]<-greutate

Putem prelua sintaxa de la matrice pentru adăugarea unei coloane noi, respectiv a unei linii noi, cu funcțiile cbind() și rbind()

#adăugarea coloanei înălțime
Înălțime<-c(173,197,189,160,170) #folositi litera mare I la cuvântul Înălțime pentru a se potrivi cu restul numelor de coloană
cbind(df,Inaltime) #rezultatul e afișat pe ecran cu noua coloană însă nu este adăugat la setul nostru de date
df<- **cbind(df,Inaltime)** #se adaugă și coloana Inaltime, numele coloanei este preluat de la numele variabilei
#adăugarea unei linii noi
ted<-data.frame(Nume="Ted", Varsta=67,Fumator=TRUE,Greutate=66,Inaltime=175) # e necesara folosirea structurii data frame pt ca avem date de tipuri diferite, de asemenea este necesară specificarea denumirilor de coloane
df<-rbind(df,ted)

Putem modifica și elementele existente deja în data frame

#schimbăm vârsta pentru Andrei
df[3,2] <-32

Sortarea datelor în data frame

Folosim funcția sort() pentru sortări date din data frame crescător sau descrescător, sau funcția order() însă acestea nu vor fi realizate peste datele din dataframe. Pentru sortare în ordine decrescătoare adăugăm argumentul decreasing=TRUE

```
#sortare alfabetică după nume
sort(df$Nume)
#sortare crescătoare după Vârstă
sort(df$Vârstă)
#sortare decrescătoare după Greutate
sort(df$Greutate, decreasing=TRUE)
```

Funcția order() ne returnează un vector ce conține pozițiile ordonate a fiecărui element. Aceste poziții vor putea fi folosite pentru a extrage elementele din setul de date într-o anumită ordine: cu alte cuvinte ordonate după un anumit criteriu

```
#ordonăm după Varsta și păstrăm indecsii poziiilor în variabila poz
poz<-order(df$Varsta)
#afișăm datele ordonate după vârstă
df[poz,]
#ordonare după mai multe criterii, întâi ordonează crescător după vârstă și în caz de aceeași valoare ordonează crescător după Nume
order(df$Varsta, df$Nume)
```

Tibbles

Tibbles sunt structuri de date tabelare ce sunt create prin librăria tidyverse. Alte librării din R folosesc structuri asemănătoare de tip data frame. Putem să realizăm conversia folosind funcția as_tibble().

Tibbles sunt create prin alăturarea unor vectori de dimensiune egală sau de dimensiune 1 care sunt extinși prin replicare la aceeași dimensiune cu restul vectorilor folosiți în compunere. Folosim funcția **tibble**() ce preia ca argumente vectorii componenți; denumirea vectorilor va apărea sub forma capului de tabel; nu e obligatoriu să se folosească vectori numiți, asta înseamnă că putem avea denumiri create de R pornind forma de construcție a acestora (a se vedea primul exemplu de creare tibble)

```
#se incarca libraria tidyverse
library(tidyverse)

#se creează un tabel cu 5 randuri (date de dimensiunea vectorilor); prima coloană e completată cu numere de la 2 până la 6, a doua coloană cu numărul 1 peste tot iar a treia coloană cu 5 numere extrase aleator  din numerele cuprinse între 1 și 100 (a se observa funcția sample()). Pentru generare de date mai putem folosi funcțiile runif(n) generează n numere aleatoare dupa o distributie uniforma; rnorm(n) generează numere aleatoare dupa o distribuție normală
tibble(2:6, 1, sample(1:100,5)

#putem combina vectori de tipuri diferite într-un tibble
tibble(2:6, "Ana", 1.3)

#creare tibble din vectori numiți
x=1:7
y=1
z=x+y
tibble(x,y,z)
#putem crea tibble cu nume de coloane atribuite de noi, singura condiție e ca numele să fie încadrate de ghilimele
t<- tibble(
+':)'="smile",
+ ``='space',
+ "21"=21)
```

O altă modalitate de a crea tibbles este de a ne folosi de funcția **tribble**() prescurtarea de la *transposed tibble*, ce este specializată pentru transmiterea valorilor în cod: capurile de coloane sunt formule iar valorile sunt separate prin virgulă. Acest lucru face posibil să aranjăm date într-o formă ușor de citit (spre deosebire de tibble unde valorile pentru vectorii de intrare erau pe linie, aici valorile vor fi citite pe coloana)

```
tt<-tribble(
~nume, ~disciplina, ~nota,
#--/--/--
“ana”, “matematica”, 10,
“bob”, “romana”, 10)
```

În comparație cu structurile de date create cu data.frame există diferențe legate de modul de afișare și de operațiile de extragere elemente.

Tibbles afișează datele optimizat, adică limitează numărul de variabile (de coloane) ce sunt afișate la dimensiunea ecranului, precum și numărul de linii la 10, ceea ce facilitează lucrul cu seturi mari de date. În plus sub fiecare coloană este afișat și tipul de dată. Totuși dacă se dorește vizualizarea de date suplimentare există opțiuni ce pot fi folosite.

Lista completă a opțiunilor care pot fi folosite poate fi văzută cu comanda

```
package?tibble
```

In RStudio se poate folosi pentru vizualizare funcția View() ce va deschide tabelul complet într-o fereastră separată

```
View(tt)
```

Pentru a extrage date dintr-un tabel ne putem folosi fie de semnul $ fie de paranteze pătrate duble [[ ]]

```
#extragerea datelor de pe coloana nume din tabelul cu tt
```