PARTE XI: Introduzione all'ambiente R

Piccola digressione (2)

La sintassi delle istruzioni in R è "case-sensitive":
- **R distingue tra MAIUSCOLE e MINUSCOLE, quindi il comando DGET non corrisponde a dget**
- **R non fa caso agli spazi che lasciamo purché non alterino i nomi degli oggetti o delle funzioni**

Tipi di dati in R

- Le tipologie di dati che R gestisce sono

1. Dati di tipo numerico, es x<-3.56;
2. Stringhe di caratteri, es x<- “nome”;
3. Dati di tipo logico, che assumono valori “TRUE”e ”FALSE” a seconda del verificarsi di una data condizione, es.x<-3.56; f <- x<4; f <-TRUE;

Gli oggetti in R

- In ambiente R, le allocazioni di memoria sono associate ad oggetti, di cui si distinguono 5 tipologie, a seconda della struttura dei dati cui sono associati:

1. vettori , (numerici, logici odi caratteri);
2. matrici;
3. Liste (oggetti lista);
4. Array;
5. data.frames
6. Mille altri tipi in realtà…

Vettori e Matrici (parentele)

5x1 Ikea Expedit "Vector"

4x4 Ikea Expedit "Matrix"

Array e Liste (parentele)

K x w! Ikea Expedit ”List"

Il workspace

- Gli oggetti creati durante la sessione di lavoro possono essere visualizzati attraverso il comando ls() e vengono salvati nel workspace attraverso il comando save.image().
- Per conoscere la directory di lavoro si utilizza il comando getwd().
- Per cambiarla si utilizza setwd("percorso").
- Il comando rm() serve ad eliminare gli oggetti dal workspace.
- Specificando il nome dell'oggetto all'interno delle parentesi si possono eliminare uno o più elementi dal workspace.

Sintassi funzionale di base

- La "chiamata" di una qualsiasi funzione segue sempre lo schema:
- Nome_fun(arg_1,arg_2,.. arg_n,op_1, op_2,..op_n)
- L'indicizzazione di un qualsiasi oggetto segue sempre gli schemi:
- Oggetto[n]
- Matrice[r,c]
- Lista[[n]]

I Vettori

- In ambiente R, indipendentemente dalla natura dei dati, è possibile creare dei vettori attraverso la funzione c() di concatenazione orizzontale, ed etichettare la corrispondente allocazione di memoria attraverso <-, operatore di assegnazione; è possibile ottenere lo stesso risultato utilizzando la funzione assign(). Esempi:

- x<-c(2,5, 7.5, 9)

- assign("x",c(2, 5, 7.5, 9))

- mese<-c ("gennaio", "febbraio","marzo")

- assign("mese",c("gennaio", "febbraio","marzo"));

I Vettori numerici

- I vettori contenenti valori numerici comprendono anche gli scalari: in ambiente R uno scalare è un vettore unidimensionale.
- Le operazioni applicabili ai vettori di tipo numerico sono:
- Operazioni elementari +, -, /, *, ^
- Funzioni aritmetiche quali log(), exp(), sin(), cos(), tan(), sqrt()
- Funzioni vettoriali 'classiche' quali min(), max(), range(), sum(), prod(), length(), mean(), var(), sort()

Estrazione/indicizzazione di oggetti

- Indipendentemente dal tipo di vettore, per selezionarne elementi basta richiamare il nome del vettore seguito dall'indice di posizione degli elementi che si vogliono selezionare in [ ].
- Esempi:
- Se a<-c(2,5,8,10,18), a[1] restituisce 2;
- a[2:4] restituisce 5, 8 e 10;
- a[-3] restituisce 2, 5, 10 e 18;
- a[c(1,3,5)] restituisce 2, 8 e 18;

Creazione di sequenze…

- In R è possibile creare delle sequenze in due modi:

1. Operatore ':', es: 1:10 produce unasequenza di passo 1 da 1 a 10;

2. La funzione seq(from, to, by/ length), in cui è possibile definire le caratteristiche della sequenza che si vuole generare; in particolare,è possibile stabilire il punto di partenza, di arrivo, il passo e la lunghezza. E' chiaro che, settando tre dei quattro parametri, la sequenza è univocamente determinata.

Creazione di sequenze…

- In R è possibile creare delle sequenze in due modi:

3. seq(1,20,3) restituisce 1 4 7 10 1316 19
4. definiti i parametri from, to e by seq(1,20, length=5) restituisce 1 4.88.6 12.4 16.2 20
5. definiti i parametri from, to e length seq(1,7,5) restituisce 18 1522 29
6. definiti i parametri from, by e length seq(by=3, to=20,from=1) restituisce 14 710 13 16 19

Vettori logici

- Altra classe di vettori che R gestisce è associata, ai dati di tipo logico: elementi di tale tipo di vettori sono 'TRUE', 'FALSE' e 'NA' (not available).
- Generare dei vettori logici coincide con il verificare una data condizione su un insieme di dati.
- Esempio: sia x <- c(2,3,5,7.5,9) possiamo verificare la condizione x > 7;
- quindi l'istruzione veclog <- x > 7, associa all'etichetta veclog il vettore logico [FALSE FALSE FALSE TRUE TRUE]

Vettori logici

- NA è il valore assegnato ai dati mancanti: in questo caso non viene applicato alcun tipo di operazione sulle celle contenenti NA.
- La funzione is.na() applicata ad un vettore, restituisce un vettore logico di uguale lunghezza con valori TRUE in caso di NA o NaN (not a number, dato da inf-inf oppure 0/0).
- La function is.nan() opera esclusivamente su NaN.

Operatori relazionali

- La condizione esprime una relazione tra variabili, gli operatori relazionali definiscono il tipo di relazione

Operatore	Simbolo
Uguale	==
Minore	<
Minore o uguale	<=
Maggiore	>
Maggiore o uguale	>=
Diverso	!=

- E' possibile combinare diverse condizioni, o espressioni logiche, ed ottenerne di composte, attraverso gli operatori logici.

Operatori logici

- Date due condizioni elementari p1 e p2 che possono risultare TRUE oppure FALSE, gli operatori logici sono

Operatore	Simbolo
AND	&
OR	|
Negazione	!

Vettori di caratteri (stringhe)

- Una stringa di caratteri è costituita da una sequenza di caratteri delimitata da (“”).
- E’ possibile ottenere dei vettori di caratteri utilizzando la funzione di concatenazione c().
- Altra interessante funzione per ottenere vettori di caratteri è paste(): tale funzione trasforma i vettori in input in caratteri, indipendentemente dalla loro natura numerica, logica o di caratteri, e ne effettua una concatenazione orizzontale elemento per elemento, utilizzando il separatore sep= “nome”.

Vettori di caratteri (stringhe)

- Utilizzo e finalità di tale funzione sono di semplice intuizione. Considerando il seguente esempio:
- Squadra<- c (“Lazio”, “Roma”)
- p=<- c(47, 40)
- classifica<- paste(Squadra,p,sep= “ punti ”)

Factors

- Un particolare tipo di oggetto vettore presente in R è costituito dal **factor**: esso definisce una classificazione degli elementi di un vettore, raggruppando gli elementi uguali, siano essi valori numerici, stringhe di caratteri o logici (TRUE, FALSE,NA) nei corrispondenti **levels**(livelli) del factor;

- Il comando factor importa in ambiente R qualunque tipo di vettore, trattandolo come qualitativo.

Gli oggetti in R

- In R, quasi tutte le cose (funzioni, datasets, risultati, ecc., tranne - quasi sempre – I grafici) sono OGGETTi
- Gli Script possono essere pensati come modi per manipolare oggetti
- Il vostro obiettivo è scrivere (o usare!) codici che producano gli oggetti da voi desiderati

Gli oggetti in R

- Gli oggetti sono classificati con 2 criteri:
- MODE: come gli oggetti sono archiviati in R (character, numeric, logical, factor, list, & function)
- CLASS: come gli oggetti sono trattati dalle funzioni (vector, matrix, array, data.frame, ecc.)
- Potete sempre ISPEZIONARE un oggetto con I comandi mode() e class()
- Generalmente MODE e CLASS coincidono… ma presto scoprirò che non è sempre così!!!

Matrici ed operazioni algebriche

- Gli operatori aritmetici definiti per i vettori numerici vengono applicati elemento per elemento
- Se Y<-c(10, 8, 16), allora Y*2 = 20, 16, 32
- Moltiplicando Y per un vettore di uguali dimensioni, ad esempio y stesso, si ottiene
- YY<- Y*Y = 100, 64, 256
- Ossia il prodotto elemento per elemento del vettore y con se stesso……

Matrici ed operazioni algebriche

- …. Per ottenere il prodotto vettoriale (Y' x Y) è necessario YY<- t(Y)%*% Y, il cui risultato è 501
- funzione `t()` effettua la trasposizione del vettore.
- Analogamente possiamo ottenere (Y x Y') digitando YY<-Y %*% t(Y) che restituisce la matrice

```
     [,1]  [,2]  [,3]  [,4]
[1,]  100   80   160   90
[2,]  80    64   128   72
[3,]  160  128   256  144
[4,]  90   72    144   81
```

Matrici ed operazioni algebriche

- E' possibile definire matrici ex-novo, attraverso il comando matrix(data,nr,nc) con:
- •data che rappresenta i valori da inserire all'interno della matrice;
- •nr che rappresenta il numero di righe della matrice;
- •nc che rappresenta il numero di colonne della matrice

Matrici ed operazioni algebriche

- La matrice A è costruita inserendo i valori in input 'per colonna',per default.
- E' possibile ottenere che i valori siano inseriti per riga nella matrice aggiungendo in matrix l'opzione `byrow=true` alla sintassi di chiamata
- La funzione di trasposizione `t()` si applica come sui vettori ed effettua la trasposta della matrice T(A)=A'

Ma l'help di R è davvero così criptico??

L'help di R: un primo assaggio…

- Dato un qualsiasi comando, è possibile accedere all'help di R digitando il comando senza argomento e preceduto da ?
- Esempio: ?matrix invoca l'help del comando matrix

L'help di R: un primo assaggio…

matrix {base}

Description

`matrix` creates a matrix from the given set of values.

`as.matrix` attempts to turn its argument into a matrix.

`is.matrix` tests if its argument is a (strict) matrix.

Usage

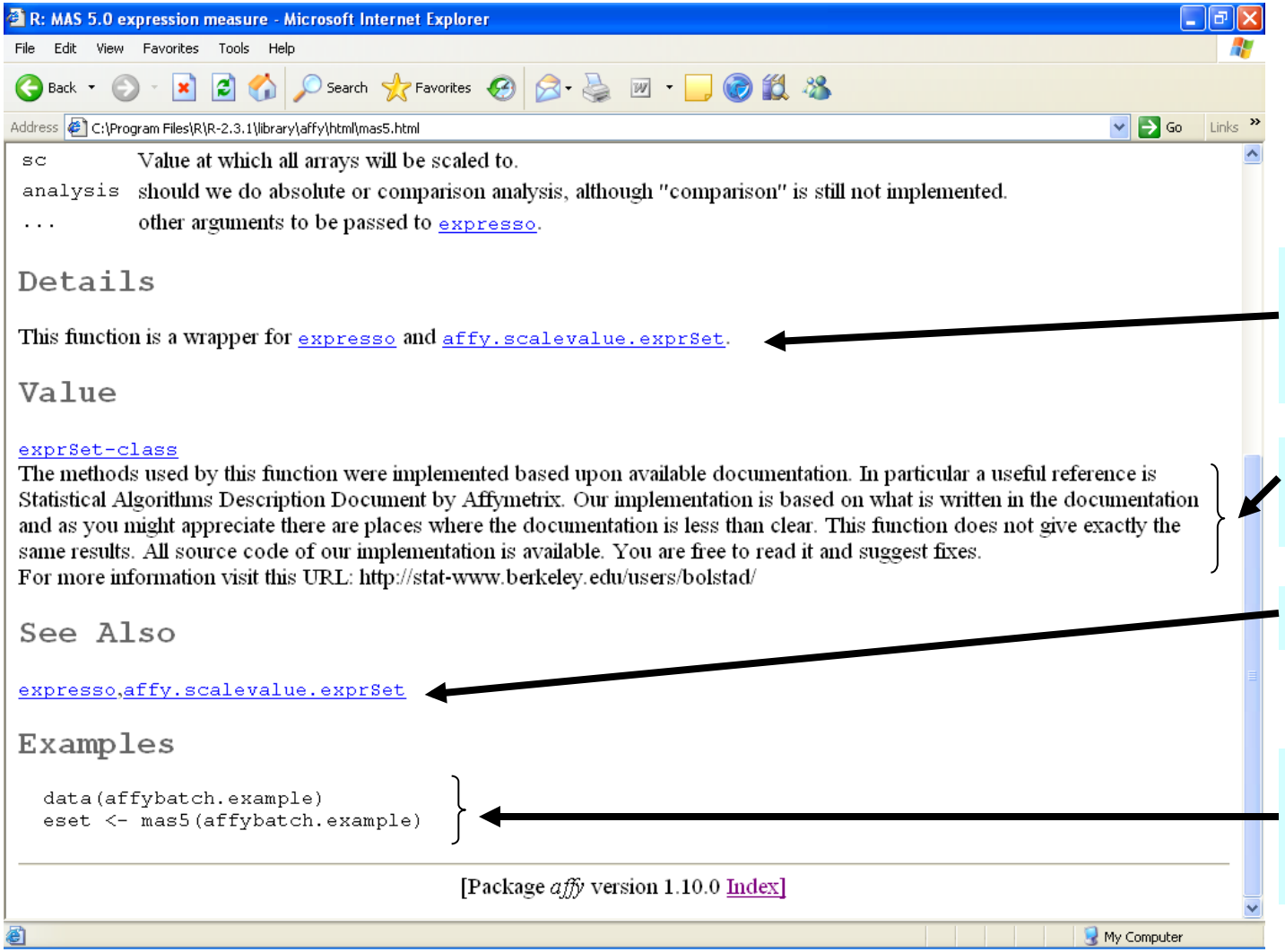
```
matrix(data = NA, nrow = 1, ncol = 1, byrow = FALSE,
       dimnames = NULL)

as.matrix(x, ...)
## S3 method for class 'data.frame'
as.matrix(x, rownames.force = NA, ...)
```

Function {package}

General description

Command and it's argument

Detailed description of arguments



Description of how function actually works

What function returns

Related functions

Examples, can be run from R by: `example(mas5)`

Matrici ed operazioni algebriche

- Come per i vettori, la sintassi per la selezione degli elementi di una matrice è data dal nome della matrice seguito dalla posizione degli elementi da selezionare in [].
- Vale quanto detto per i vettori numerici: gli operatori aritmetici vengono applicati elemento per elemento alle matrici, mentre per ottenere ad esempio il prodotto tra matrici è necessario usare %*%

Il comando which

- Attraverso il comando which(mat,condizione) è possibile selezionare gli elementi della matrice che soddisfino una data condizione.

Le liste

Il vettore list è un 'contenitore' di oggetti nel senso che accoglie oggetti di diversa natura (numerici, logici,stringhe).

Definendo ad esempio degli oggetti di diverso tipo

```
stringa <- c("giugno","luglio", "agosto")
logico<- c(TRUE,TRUE,FALSE, FALSE, TRUE)
mat<-matrix(1,4,2)
```

È possibile includerli in un'unica list

```
Lista=list(stringa,logico,mat)
```

Cosa contiene una lista?

Una descrizione deglielementi di una lista si ottiene con il comando str()

Str(lalista) restituisce la seguente descrizione
List of 3
$ mesi: chr[1:3] "giugno" "luglio" "agosto"
$ visita: logi[1:5] TRUE TRUEFALSE FALSE TRUE
$ matrice:num [1:4,1:2] 11 11 1 11 1
In cui vengono indicati rispettivamente nome, tipologia, dimensioni e valori per ciascuno degli oggetti in lista.

Selezione di elementi da una lista

Per selezionare elementi contenuti negli oggetti di una lista occorre richiamare il nome della lista ne seguente modo

nomelista$ nomeoggetto [posizione]

Ad esempio il comando lalista$ mesi[2]

richiama

[1] "luglio"

Alternativamente è possibile richiamare un intero oggetto nella lista attraverso la sintassi

lalista[[3]]

I Dataframe

u	X	Y	Z	W
Unità statistica	Stato civile	Grado di istruzione	Numero di figli	Peso in kg
1	N	L	0	72.5o
2	S	O	1	54.28
3	V	A	3	50.02
4	V	O	4	88.88
5	C	L	1	62.30
6	N	S	1	45.21
7	C	S	0	57.50
8	C	O	2	78.40
9	V	L	3	75.13
10	N	O	0	58.00

Il Dataframe

Per generare un dataframe si usa la funzione

data.frame

Dati<-data.frame("etich.variabile1"=var1, "etich.variabile2"=var2,….)

Accedere agli oggetti di un Dataframe

Un modo per accedere ai dati contenuti in un data frame è utilizzare la sintassi nome_dataframe $ nome_variabile.

Volendo richiamare la variabile stato civile da datamat

datamat$ Stato.civile.

Un modo più diretto per accedere agli elementi di un vettore si basa sulla coppia di comandi attach() e detach().

Le funzioni in R

- Lo schema generale per la definizione di una funzione `e il seguente:
- nome_funzione <- function(<arg>, <opz>) {

  <corpo_della_funzione>

  return(<valore_restituito_dalla_funzione>)

  }
- Il <nome_funzione> può essere un qualsiasi nome, contenente caratteri alfanumerici, compreso il “.”
- NB: Se già esiste una funzione con lo stesso nome, questa ultima verrà sostituita dalla nuova funzione definita

Le funzioni in R

- Il <corpo_della_funzione> contiene tutte le espressioni (comandi) in R che definiscono ciò che la funzione effettivamente esegue.

- Gli argomenti e le variabili che sono create all'interno della funzione sono variabili locali, nel senso che al termine della funzione il loro valore non viene mantenuto.

- Nell'esecuzione dei comandi all'interno della funzione le variabili locali hanno precedenza sulle variabili globali definite a livello di workspace, mentre tutte le variabili non definite localmente devono essere definite a livello di workspace altrimenti si produrrà un errore.

Importare Dati in R

Il comando per importare dati da file esterni è read.table, la cui sintassi di chiamata è

read.table("nomefile.estensione", header=T,row.names=numero,sep="separatore utilizzato")

L'opz header specifa se la prima riga del file di dati contiene le etichette. Se vogliamo le etichette di riga,queste vanno indicata con l'opzione rownames.

Mi converrà sempre importare dati da software come Excel salvandoli prima in csv e poi caricandoli in R con read.table

I pacchetti di R

- I package forniscono uno strumento semplice ed efficient e per gestire collezioni di funzioni e di dati (librerie) e la re lativa documentazione
- I package R forniscono librerie liberamente disponibili scri tte da sviluppatori esperti in diversi domini applicativi
- I moduli di R sono organizzati in un apposito sito detto CRAN (http://cran.r-project.org/) (CRAN = Comprehensive R Archive Network), in analogia al CTAN [è l'acronimo di Comprehensive TEX Archive Network] e CPAN [l'acronimo di Comprehensive Perl Archive Network])

I pacchetti di R

- Caratteristiche dei package R:
- •Caricamento dinamico in memoria: vengono caricati in memoria quando necessario
- Facili da installare ed aggiornare: le funzioni, dati e documentazione sono installati con un singolo comando che può essere eseguito dall' interno o dall' esterno di R.
- Estendibili ed adattabili alle esigenze degli utenti: gli utenti possono creare propri package (si veda il manuale disponibile on line "Writing R extensions")

I pacchetti di R

- I package possono essere installati:
- – direttamente dal sito CRAN: cran.r-project.org.
- selezionando dal menu Packages della finestra RGui l'opzione Install packages from CRAN
- da file locali compressi tramite il programma zip (file "zippati") selezionando dal menu Packages della finestra RGui l'opzione Install packages from local zip files
- Analogamente l' aggiornamento di package già installati può essere effettuato selezionando dal menu Packages l'opzione Update packages from CRAN.

I pacchetti di R

- Le funzioni ed i dati dei package devono essere caria cati in memoria perchè possano essere utilizzabili.
- Per caricare un package si può scegliere uno dei mod i seguenti:
- – Dal menu Packages selezionare l'opzione Load package
- – Dal prompt digitare: library(nome_package)

I pacchetti di R

- Per avere accesso alla documentazione sui package install ati nel sistema si può selezionare dal menu Help l'opzione HT ML help.
- Selezionando Packages dalla pagina del browser si può sce gliere dalla lista dei package il package desiderato ed accedere alle informazioni dettagliate in formato HTML.
- Dal prompt si può anche digitare help(package=xxx) per av ere informazioni sul package xxx

I pacchetti di R

- Spesso i package dispongono della medesima documentazione in formato pdf
- Ogni funzione, insieme di dati, classe ed in generale ogni oggetto disponibile nei package dispone di una propria documentazione specifica.

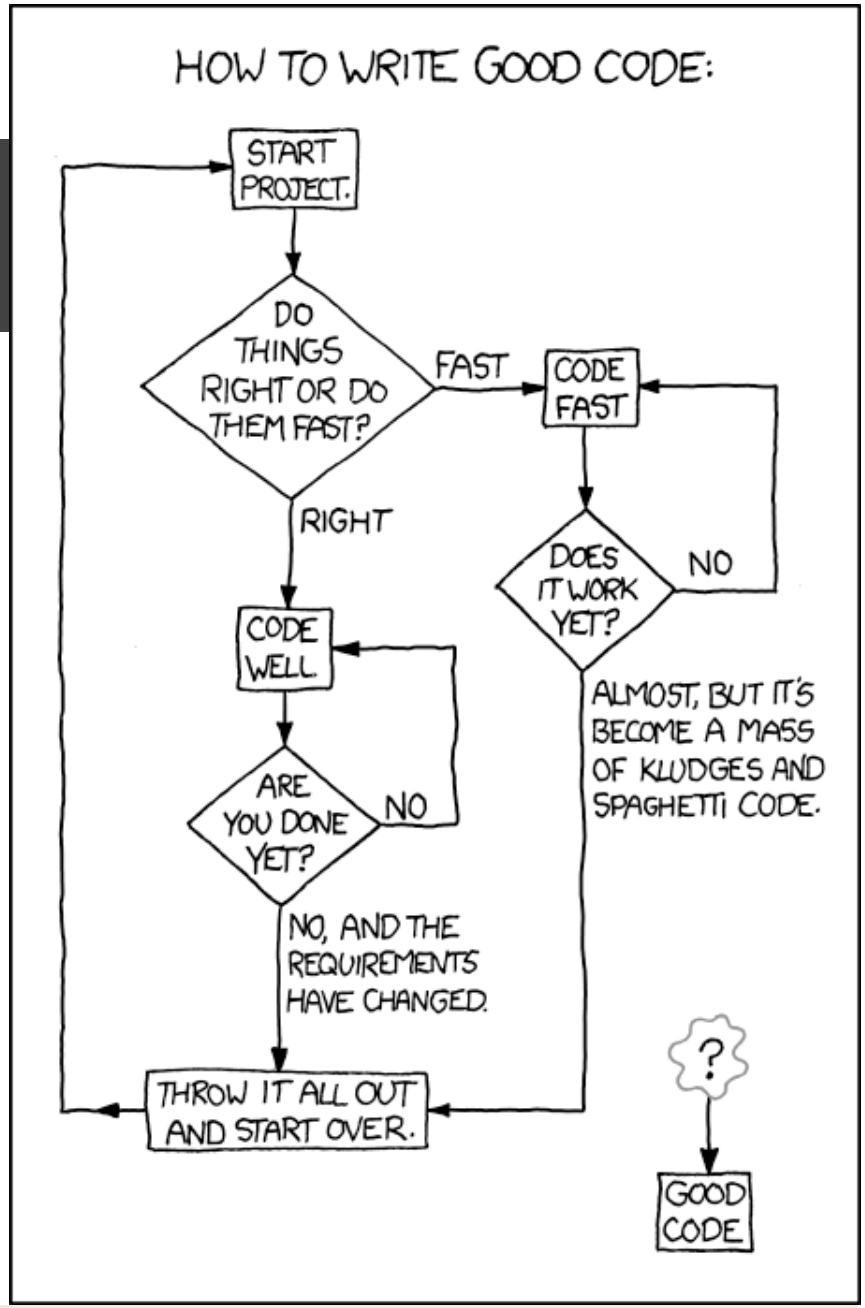
I pacchetti di R

- Indipendentemente dal formato della documentazione (HTML, pdf, etc), nella pagina di documentazione di una funzione si trovano solitamente le seguente sezioni:
- Nome della funzione e package nella quale è contenuta
- Description: descrizione sintetica delle caratteristiche generali della funzione
- Usage: sintassi (formato della chiamata della funzione)

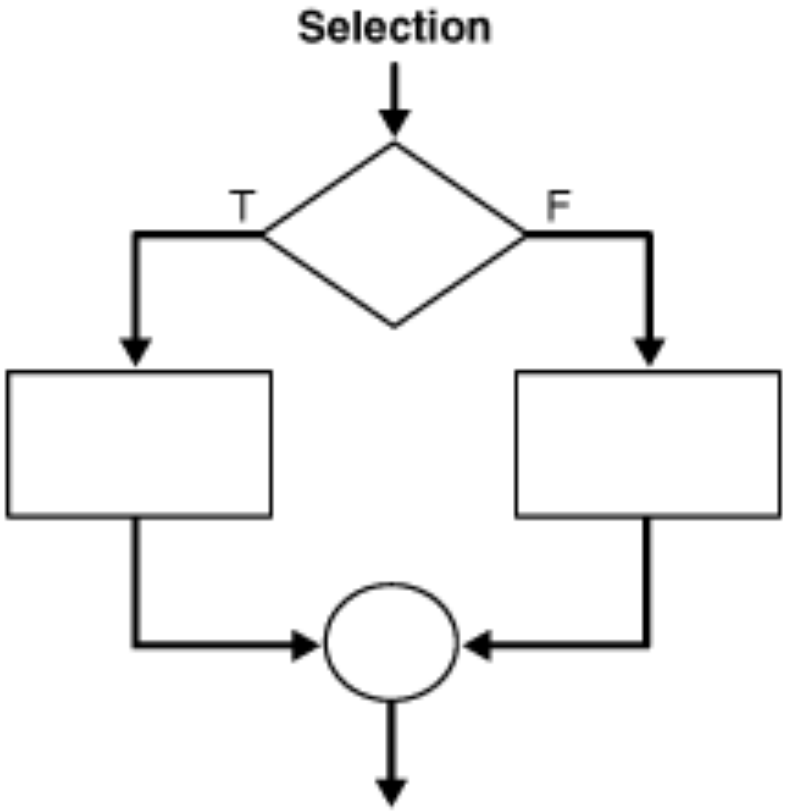
I pacchetti di R

- Arguments: elenco e descrizione degli argomenti della funzione
- Value: valore ritornato dalla funzione
- Details: descrizione più dettagliata della funzione (se necessaria)
- References: riferimenti bibliografici utili
- See also: link ad altre funzioni, oggetti correlati alla funzione
- Examples: alcuni esempi di utilizzo della funzione

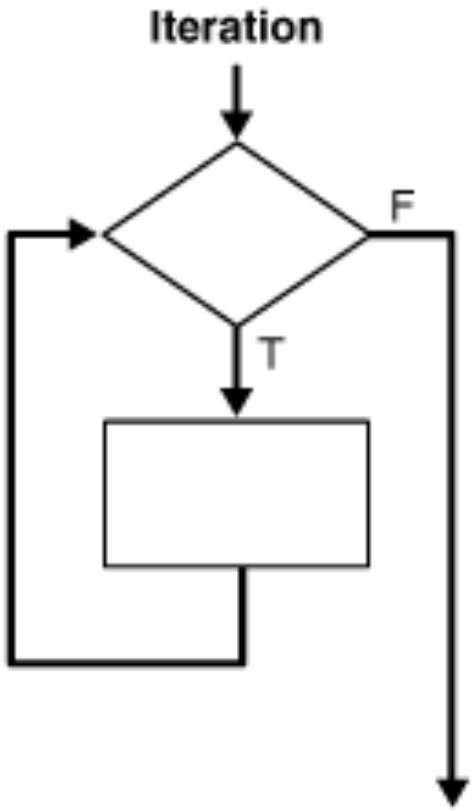
Programmare in R



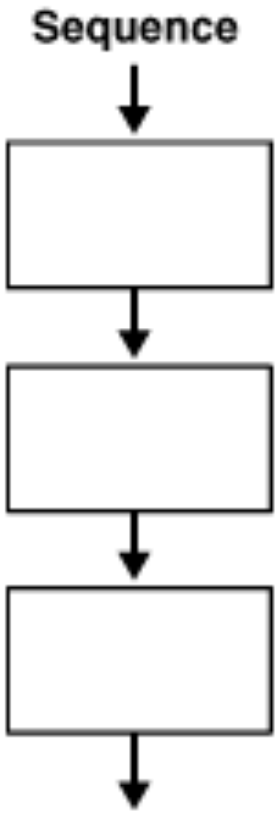
Programmare in R



if ... then ... else



for ...
while ...



Ciclo FOR

- **Repeats a line or lines (known as a block) of code until:**
  - **(for loop) a count limit is reached**

```
for (i in 1:10) {
  ... code ...
}
```

- **the above loop runs 10 times**
- **'{' and '}' enclose code looped**
- **the variable i updated in the loop to values in the sequence 1:10**

Ciclo WHILE

- **Repeats a line or lines (known as a block) of code until:**
  - **(while loop) a logical condition is reached**

```
while (stop != TRUE) {
  ... code ...
}
```

- **the above loop runs until code sets**
- **stop = TRUE**

- **warning: if not properly written while loops can run infinitely**

Ciclo WHILE

```
# Numbers from our analysis
v1 = c(21, 22, 53, 74, 85, 96, 97, 58, 49, 30, 85)

# Iterator
i = 1

# Look for first instance of 85
while(v1[i] != 85) {
        i = i + 1
}

# Print out where we found it
print(paste('v1[',i,'] = 85',sep=''))
```

Risorse e Riferimenti:

- Il materiale di questa lezione è stato assemblato utilizzzando le seguenti risorse disponili online:
  - http://psych.colorado.edu/wiki/lib/exe/fetch.php?media=courses:keller:5541:lecture1_2015.ppt