```
### R code from vignette source 'RIntro.Rnw'

###################################################
### code chunk number 1: "House keeping"
###################################################
options(width=81,continue=" ")


###################################################
### code chunk number 2: "assigning variables"
###################################################
y <- 5
y
assign("e",2.7183)
e
s = sqrt(2)
s
r <- rnorm(n=2)
r


###################################################
### code chunk number 3: "creating a vector and matrix and data.frame"
###################################################
x <- c(3.1416,2.7183)
m <- matrix(rnorm(9),nrow=3)
tab <- data.frame(store=c("downtown","eastside","airport"),sales=c(32,17,24))
cities <- c("Seattle","Portland","San Francisco")
ls()


###################################################
### code chunk number 4: "house keeping"
###################################################
options(width=40)


###################################################
### code chunk number 5: "type of an object"
###################################################
x
typeof(x)
cities
typeof(cities)


###################################################
### code chunk number 6: "house keeping"
###################################################
options(width=81)


###################################################
### code chunk number 7: "class of an object"
###################################################
m
class(m)
tab
class(tab)
```

```
####################################################
### code chunk number 8: "vectorized operations"
####################################################
constants <- c(3.1416,2.7183,1.4142,1.6180)
names(constants) <- c("pi","euler","sqrt2","golden")
constants
constants^2
10*constants


####################################################
### code chunk number 9: "vector indexing"
####################################################
constants[c(1,3,4)]
constants[c(-1,-2)]
constants[c("pi","golden")]
constants > 2
constants[constants > 2]


####################################################
### code chunk number 10: "illustrating recycling rule"
####################################################
constants
constants*2
constants*c(0,1)
constants*c(0,1,2)


####################################################
### code chunk number 11: "seq function arguments"
####################################################
args(seq.default)


####################################################
### code chunk number 12: "creating sequences"
####################################################
1:5
-5:5
seq(from=0,to=1,len=5)
seq(from=0,to=20,by=2.5)


####################################################
### code chunk number 13: "passing arguments to a function"
####################################################
seq(0,10,2)
seq(by=2,0,10)
seq(0,10,len=5)
seq(0,10)


####################################################
### code chunk number 14: "plot function arguments"
####################################################
args(plot.default)


####################################################
```

```
### code chunk number 15: "rep function examples"
####################################################
rep(0,10)   # initialize a vector
rep(1:4, 2) # repeat pattern 2 times
rep(1:4, each = 2)  # repeat each element 2 times
rep(1:4, c(2,1,2,1))
rep(1:4, each = 2, len = 10)   # 8 integers plus two recycled 1's.
rep(1:4, each = 2, times = 3)  # length 24, 3 complete replications


####################################################
### code chunk number 16: "methods for the generic function plot"
####################################################
methods(plot)[1:15]


####################################################
### code chunk number 17: "loading a library and loading data"
####################################################
args(library)
args(data)
library(nutshell)
data(top.bacon.searching.cities)
top.bacon.searching.cities[1,]


####################################################
### code chunk number 18: "install.packages function arguments"
####################################################
args(install.packages)


####################################################
### code chunk number 19: "installing packages" (eval = FALSE)
####################################################
## install.packages("nutshell")
## # or if repository needs to be specified
## install.packages("nutshell", repos="http://cran.fhcrc.org")


####################################################
### code chunk number 20: RIntro.Rnw:749-755
####################################################
percentChange <- function(x)
{
  100*(x[-1]/x[-length(x)]-1)
}
sales <- c(100,105,110,105,100)
percentChange(sales)


####################################################
### code chunk number 21: "creating lists"
####################################################
myList <- list(pi=3.1416,euler=2.7183,golden=1.6180)
class(myList)
length(myList)
myList
diverseList <- list(magic=myList,random=matrix(rnorm(4),ncol=2),
  state=c("WA","OR"))
```

```
####################################################
### code chunk number 22: "extracting elements from a list"
####################################################
myList[2]
myList[[2]]
myList[["pi"]]
myList$golden
diverseList[[3]][2]


####################################################
### code chunk number 23: "exploring a data.frame object"
####################################################
data(batting.2008)
class(batting.2008)
dim(batting.2008)
batting.2008[1:2,1:4]
class(batting.2008[,2])
class(batting.2008[,3])
class(batting.2008[,4])


####################################################
### code chunk number 24: "head and tail functions"
####################################################
args(getS3method("head","data.frame"))
data(dow30)
head(dow30)
tail(dow30,3)


####################################################
### code chunk number 25: "extracting elements from a data.frame"
####################################################
dow30[1:4,c("symbol","Date","Close")]
head(dow30[-1,c(1,2,6)],3)
dow30[dow30[,"Volume"]>1.5e9,
  c("symbol","Date","Close","Volume")]


####################################################
### code chunk number 26: RIntro.Rnw:927-932
####################################################
pet.str <- c("dog","cat","cat","dog","fish","dog","rabbit")
pets <- as.factor(pet.str)
pets
as.numeric(pets)
levels(pets)


####################################################
### code chunk number 27: "getting and setting the working directory"
####################################################
(my.wd <- getwd())
setwd(R.home())
getwd()
setwd(my.wd)
getwd()
```

```
#####################################################
### code chunk number 28: "read.table function arguments"
#####################################################
args(read.table)


#####################################################
### code chunk number 29: "write.table function example"
#####################################################
intc <- dow30[dow30[,"symbol"]=="INTC",-1]
write.table(x=intc,file="INTC.CSV",quote=F,sep=",",col.names=T,row.names=F)
msft <- dow30[dow30[,"symbol"]=="MSFT",-1]
write.table(x=msft,file="MSFT.CSV",quote=F,sep=",",col.names=T,row.names=F)


#####################################################
### code chunk number 30: "read.table function example"
#####################################################
dat <- read.table("intc.csv",header=TRUE,sep=",",as.is=TRUE)
dat[1:5,]


#####################################################
### code chunk number 31: "write.table function arguments"
#####################################################
args(write.table)
args(write)


#####################################################
### code chunk number 32: "write.table function example"
#####################################################
write(x=constants,file="vector.dat",sep="\t")
write.table(x=m,file="matrix.dat",sep="\t",row.names=F,col.names=F)
file.info(list.files(pattern="[.][d][a][t]",full.names=T))[,c("size","mtime")]


#####################################################
### code chunk number 33: "paste function example"
#####################################################
args(paste)
a <- 2; b <- 2
paste("We know that: ", a, " + ", b, " = ", a+b, sep ="")
paste("variable",1:5,sep="")


#####################################################
### code chunk number 34: "apply function example"
#####################################################
args(apply)
set.seed(1)
(m <- matrix(sample(9),ncol=3))
apply(m,2,sum)


#####################################################
### code chunk number 35: "opening R help system"
#####################################################
help.start()
```

```
####################################################
### code chunk number 36: "getting help for a function"
####################################################
help(read.table)


####################################################
### code chunk number 37: "alternative method for getting help for a function"
####################################################
??predict


####################################################
### code chunk number 38: "plot function arguments"
####################################################
args(plot.default)


####################################################
### code chunk number 39: plotPoints
####################################################
library(Ecdat)
data(Capm)
plot(Capm[,"rf"])


####################################################
### code chunk number 40: plotLine
####################################################
plot(Capm[,"rf"],type="l")


####################################################
### code chunk number 41: plotBars
####################################################
plot(Capm[,"rmrf"],type="h")


####################################################
### code chunk number 42: plotXY
####################################################
plot(Capm[,"rmrf"],Capm[,"rcon"])


####################################################
### code chunk number 43: "points function arguments"
####################################################
args(points.default)


####################################################
### code chunk number 44: "lines function arguments"
####################################################
args(lines.default)


####################################################
### code chunk number 45: "text function arguments"
####################################################
args(text.default)
```

```
####################################################
### code chunk number 46: plotEmptyFrame
####################################################
plot(0,xlim=c(-20,20),ylim=c(-20,20),type="n",
  xlab="market return",ylab="construction return")


####################################################
### code chunk number 47: plotAllComponentsSeperately
####################################################
plot(0,xlim=c(-20,20),ylim=c(-20,20),type="n",
  xlab="market return",ylab="construction return")
points(x=Capm[,"rmrf"],y=Capm[,"rcon"],col="gray")
lines(x=-20:20,y=-20:20,lwd=2,col="darkred")
text(20,20,labels="slope = 1",pos=2)


####################################################
### code chunk number 48: "segments function arguments"
####################################################
args(segments)


####################################################
### code chunk number 49: "curve function arguments"
####################################################
args(curve)


####################################################
### code chunk number 50: "abline function arguments"
####################################################
args(abline)


####################################################
### code chunk number 51: "matplot function arguments"
####################################################
args(matplot)


####################################################
### code chunk number 52: "par function parameters"
####################################################
names(par())


####################################################
### code chunk number 53: "legend function arguments"
####################################################
args(legend)


####################################################
### code chunk number 54: "barplot function arguments"
####################################################
args(barplot.default)


####################################################
```

```
### code chunk number 55: plotPDF
####################################################
args(dnorm)
x <- seq(from = -5, to = 5, by = 0.01)
x[1:10]
y <- dnorm(x)
y[1:5]
par(mar = par()$mar + c(0,1,0,0))
plot(x=x,y=y,type="l",col="seagreen",lwd=2,
  xlab="x",ylab="density\ny = dnorm(x)")
grid(col="darkgrey",lwd=2)
title(main="Probability Density Function (PDF)")


####################################################
### code chunk number 56: plotCDF
####################################################
args(pnorm)
args(qnorm)
y <- pnorm(x)
par(mar = par()$mar + c(0,1,0,0))
plot(x=x,y=y,type="l",col="seagreen",lwd=2, xlab="x = qnorm(y)",
   ylab="probability\ny = pnorm(x)")  ; grid(col="darkgrey",lwd=2)
title(main="Cumulative Distribution Function (CDF)")


####################################################
### code chunk number 57: "house keeping"
####################################################
set.seed(1)


####################################################
### code chunk number 58: "rnorm function argument"
####################################################
args(rnorm)
x <- rnorm(150)
x[1:5]
y <- rnorm(50,sd=3)
y[1:5]


####################################################
### code chunk number 59: "hist function arguments"
####################################################
args(hist.default)


####################################################
### code chunk number 60: plotDefaultHistogram
####################################################
hist(x,col="seagreen")


####################################################
### code chunk number 61: plotDensityHistogram
####################################################
hist(c(x,y),prob=T,breaks="FD",col="seagreen")


####################################################
```

```
### code chunk number 62: plotHistogramWithManyBreaks
####################################################
hist(c(x,y),prob=T,breaks=50,col="seagreen")


####################################################
### code chunk number 63: "create zoo object"
####################################################
library(zoo)
msft.df <- read.table("MSFT.CSV", header = TRUE, sep = ",", as.is = TRUE)
head(msft.df,2)
args(zoo)
msft.z <- zoo(x=msft.df[,"Close"],order.by=as.Date(msft.df[,"Date"]))
head(msft.z)


####################################################
### code chunk number 64: "inspect zoo object"
####################################################
class(msft.z)
start(msft.z)
end(msft.z)
frequency(msft.z)
class(coredata(msft.z))
class(time(msft.z))


####################################################
### code chunk number 65: plotZooTimeSeries
####################################################
ticks <- as.Date(unique(as.yearmon(index(msft.z))))
plot(msft.z, xaxt='n',xlab="",ylab="$",main="Microsoft Stock Price")
axis(side=1, at=ticks, lab=format(ticks,"%b-%y"))


####################################################
### code chunk number 66: "read zoo object"
####################################################
args(read.zoo)
soft <- read.zoo(file="MSFT.CSV",header=TRUE,sep=",")
head(soft,2)
class(soft)
class(coredata(soft))
class(index(soft))


####################################################
### code chunk number 67: "variable scoping example"
####################################################
f <- function(x) {
  y <- 2*x
  print(x) # formal parameter
  print(y) # local variable
  print(z) # free variable
}


####################################################
### code chunk number 68: "check search path"
####################################################
search()
```

```
####################################################
### code chunk number 69: "variable scoping example 1 and 2"
####################################################
# example 1
a <- 10
x <- 5
f <- function (x) x + a
f(2)
# example 2
f<- function (x)
{
  a<-5
  g(x)
}
g <- function(y) y + a
f(2)


####################################################
### code chunk number 70: "variable scoping example 3 and 4
####################################################
# example 3
f <- function (x) {
  a<-5
  g <- function (y) y + a
  g(x)
}
f(2)
# example 4
f <- function (x) {
  x + mean(rivers) # rivers is defined in the dataset package
}
f(2)
```