**R مقدماتی**

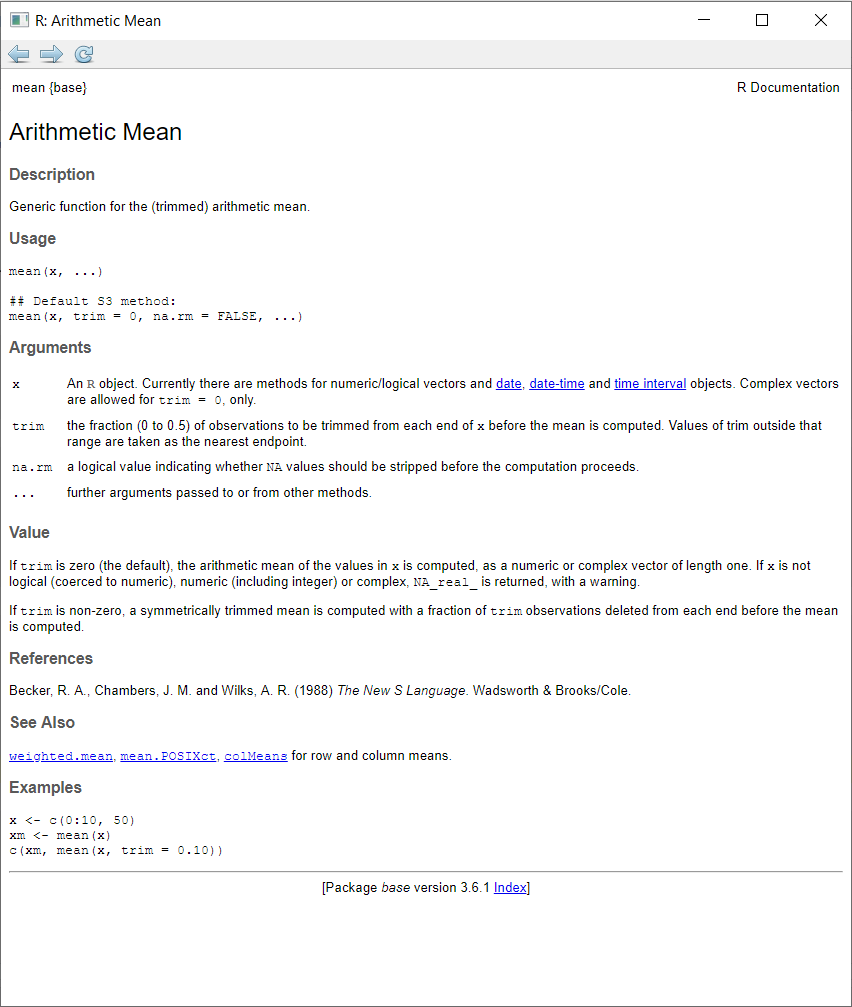
|  |
| --- |
| **استفاده از Help** |

**دسترسی به فایل‏های کمکی**

سه روش کلی برای استفاده از Help برنامه R وجود دارد:

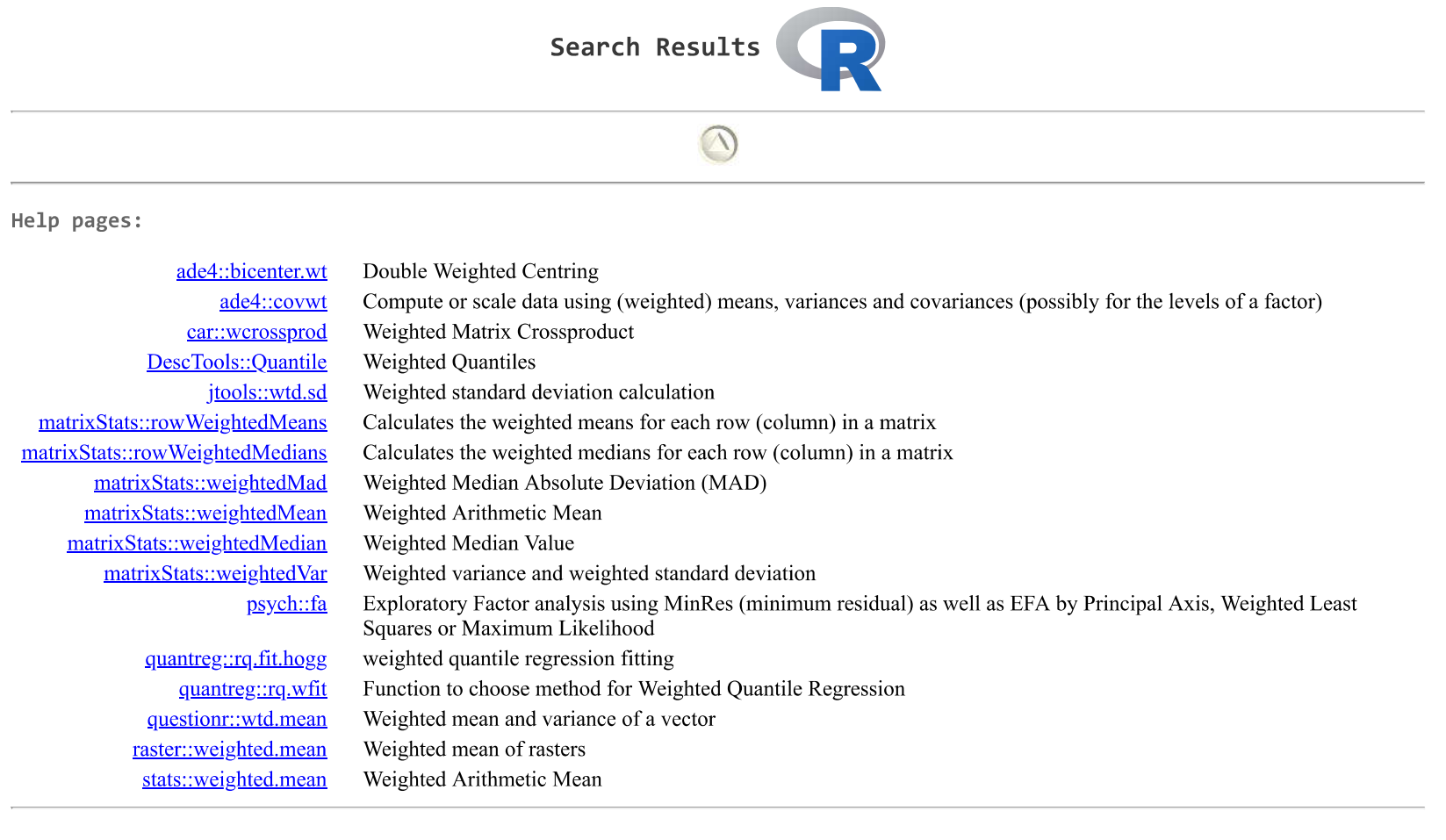
* **نیاز به کمک در مورد عملکرد تابعی خاص:** به طور مثال نیاز است در مورد ساختار نوشتن تابع mean و آرگمان‏های ورودی آن اطلاعاتی بدست آید.

?mean



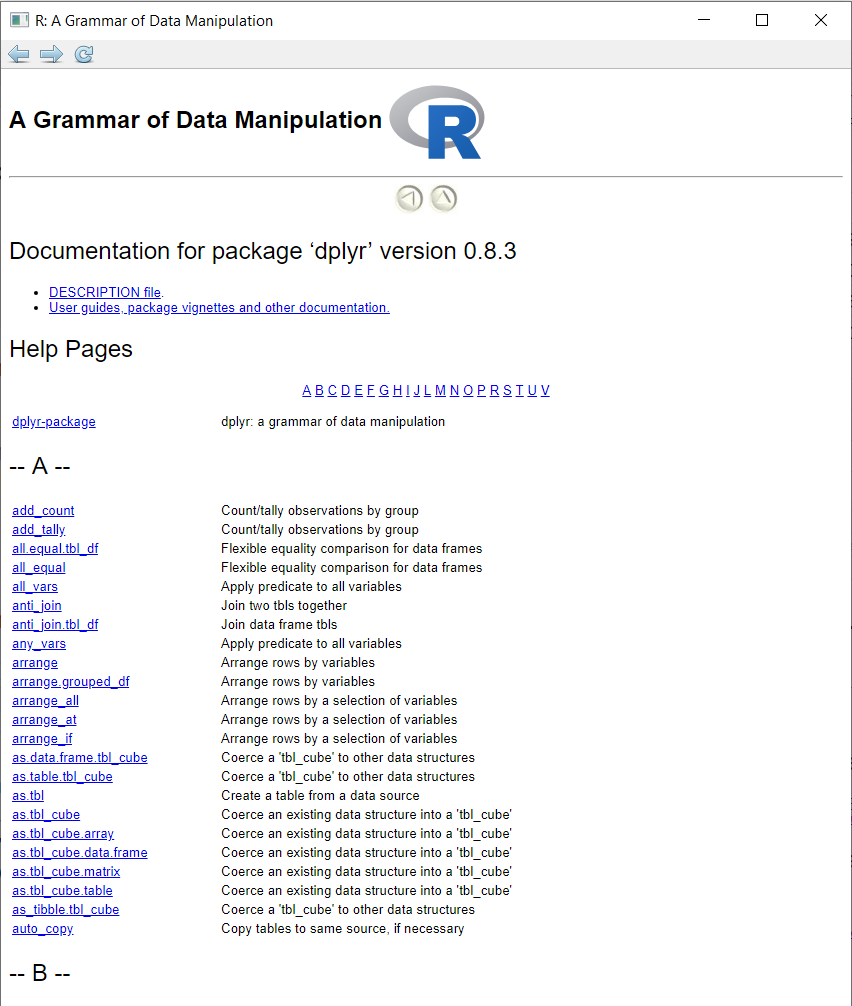
* **جستجو برای یک کلمه یا عبارتی خاص:** به طور مثال نیاز است در مورد چگونگی استفاده از میانگین وزنی در R اطلاعاتی بدست آید.

help.search(pattern = "weighted mean")



* **نیاز به راهنمایی در مورد یک بسته (Package):** به طور مثال نیاز است در مورد بسته dplyr اطلاعاتی کسب شود.

**help**(package = "dplyr")



**گرفتن اطلاعات بیشتر در مورد یک شیء‏ (Object)**

* **دریافت خلاصه‏ای از ساختار یک شیء:** به طور مثال نیاز است خلاصه‏ای از دیتاست **iris** ارائه شود.

data("iris")  
str(object = iris)

## 'data.frame': 150 obs. of 5 variables:  
## $ Sepal.Length: num 5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...  
## $ Sepal.Width: num 3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...  
## $ Petal.Length: num 1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...  
## $ Petal.Width: num 0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...  
## $ Species: Factor w/ 3 levels "setosa", "versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...

* **پیدا کردن کلاس یک شیء:** به طور مثال نیاز است کلاس دیتاست **iris** مشخص شود.

class(x = iris)

## "data.frame"

|  |
| --- |
| **استفاده از بسته‏ها** |

بیش از ۵,۰۰۰ بسته آماده برای زبان **R** وجود دارد که می‏توان آن‏ها را دانلود و از آن‏ها استفاده کرد.

* **دانلود و نصب یک بسته از CRAN[[1]](#footnote-1):** در مثال زیر نحوه نصب بسته **dplyr** نشان داده شده است.

install.packages (pkgs = "dplyr")

* **فراخوانی یک بسته و استفاده از تمام توابع آن:** در مثال زیر بسته **dplyr** فراخوانی می‏شود.

library(package = dplyr)

* **استفاده از یک تابع خاص درون یک بسته، بدون فراخوانی بقیه توابع آن بسته:** در مثال زیر تابع **select** از بسته **dplyr** فراخوانی می‏شود.

dplyr::select(.data = iris, Species)

* **بارگزاری دیتاست‏های از پیش ساخته شده به درون محیط کاری R:** در مثال زیر دیتاست **iris** از بسته **datasets** به درون محیط کاری **R** فراخوانی می‏شود.

data(iris)

|  |
| --- |
| **دایرکتوری جاری[[2]](#footnote-2)** |

دایرکتوری جاری، فولدر پیش فرضی است که خواندن و نوشتن داده‏ها از داخل آن انجام می‏شود.

* **نمایش دایرکتوری جاری:** با اجرای دستور زیر، محل دایرکتوری جاری برنامه R نشان داده می‏شود.

**getwd()**

## "C:/Users/Pooya/OneDrive/Academic/Books/StatisticalTests"

* تغییر دایرکتوری جاری: با اجرای دستور زیر محل دایرکتوری جاری به آدرس “C:/Users/Pooya” تغییر پیدا می‏کند.

setwd(dir = "C:/Users/Pooya")

|  |
| --- |
| **بردارها[[3]](#footnote-3)** |

یک بردار آرایه‏ای یک بعدی از شیء‏ها با نوع یکسان است. یک بردار فقط دارای یک سطر یا ستون است. طول یک بردار برابر تعداد داده‏های موجود در آن می‏باشد. هر داده دارای یک ایندکس مخصوص است. با استفاده از ایندکس می‏توان به آن داده خاص در بردار ارجاع نمود. برای تعریف یک بردار می‏توان از تابع c() استفاده نمود.

**ایجاد یک بردار**

* **ایجاد یک بردار با استفاده از تابع c():** با اجرای دستور زیر یک بردار با مقادیر 1، 2، 10 و 20 در محیط کاری برنامه R ایجاد می‏شود.

**c**(1, 2, 10, 20)

## 1 2 10 20

* **ایجاد یک بردار با استفاده از تابع c() که هر عنصر دارای نام باشد:** در مثال زیر هدف ایجاد یک بردار با اطلاعات زیر می‏باشد.

Age = 30, ID = 12, Year = 2020

c("Age" = 30, "ID" = 12, "Year" = 2020)

## Age ID Year  
## 30 12 2020

* **ایجاد یک بردار از توالی اعداد با گام 1:** با استفاده از عملگر “:” و اجرای دستور زیر یک بردار با مقادیر 2 تا 8 با گام 1 ایجاد می‏شود.

2**:**8

## 2 3 4 5 6 7 8

* **ایجاد یک بردار از توالی اعداد با گام‏های مختلف با استفاده از تابع seq:** در مثال زیر هدف ایجاد برداری بین اعداد 2 تا 4 با گام 5/0 با استفاده از تابع seq می‏باشد.

seq(from = 2, to = 4, by = 0.5)

## 2.0 2.5 3.0 3.5 4.0

* **تکرار یک بردار پشت سر هم، به تعداد مورد نیاز با استفاده از تابع rep:** در مثال زیر بردار 1:2، سه بار تکرار می‏شود و برداری جدید با شش عضو ایجاد می‏شود.

rep(x = 1:2, 3)

## 1 2 1 2 1 2

* **تکرار عناصر یک بردار پشت سر هم، به تعداد مورد نیاز با استفاده از تابع rep:** در مثال زیر هر عنصر بردار 1:2، سه بار تکرار می‏شود و برداری جدید با شش عضو ایجاد می‏شود.

**rep**(x = 1**:**2, each = 3)

## 1 1 1 2 2 2

**توابع بردارها**

* **مرتب کردن یک بردار با استفاده از تابع sort:** با استفاده از تابع sort می‏توان عناصر بردار c(1, 5, 2, 0) را از کوچک به بزرگ مرتب کرد. برای اینکه عناصر از بزرگ به کوچک مرتب شوند باید مقدار آرگمان decreasing تابع sort، مساوی با TRUE قرار گیرد.

sort(x = c(1, 5, 2, 0))

## 0 1 2 5

sort(x = c(1, 5, 2, 0), decreasing = TRUE)

## 5 2 1 0

* **معکوس کردن یک بردار با استفاده از تابع rev:** در مثال زیر عناصر بردار c(2, 0, 5) با استفاده از تابع rev معکوس، و برداری جدید ایجاد می‏شود.

rev(x = c(1, 5, 2, 0))

## 0 2 5 1

* **استخراج عناصر منحصر به فرد یک بردار با استفاده از تابع unique:** در مثال زیر تابع unique مقادیر تکراری بردار c(1, 2, 1, 0, 5, 5, 7, 0) را حذف و یک بردار با مقادیر منحصر به فرد بر می‏گرداند.

unique(x = c(1, 2, 1, 0, 5, 5, 7, 0))

## 1 2 0 5 7

* **شمارش تعداد عناصر منحصر به فرد در یک بردار با استفاده از تابع table:** در مثال زیر تابع table تعداد هر عنصر منحصر به فرد در بردار c(1, 2, 1, 0, 5, 5, 7, 0) را به صورت یک جدول بر می‏گرداند.

table(x = c(1, 2, 1, 0, 5, 5, 7, 0))

## x  
## 0 1 2 5 7   
## 2 2 1 2 1

**انتخاب عناصر یک بردار**

می‏توان عناصر موجود در یک بردار را با استفاده از [ ] و شماره ایندکس عناصر انتخاب کرد.

* **انتخاب یک عنصر خاص:** در مثال زیر هدف انتخاب عنصر چهارم از بردار x می‏باشد.

x = c(1, 5, 0, 7, 1, 1, 2)  
x[4]

## 7

* **انتخاب همه به غیر از یک عنصر خاص:** در مثال زیر هدف انتخاب همه عناصر به غیر از عنصر چهارم از بردار x می‏باشد.

x = c(1, 5, 0, 7, 1, 1, 2)  
x[-4]

## 1 5 0 1 1 2

* **انتخاب چندین عنصر پشت سر هم:** در مثال زیر هدف انتخاب عناصر دوم تا چهارم از بردار x می‏باشد.

x = c(1, 5, 0, 7, 1, 1, 2)  
x[2:4]

## 5 0 7

* **انتخاب همه به غیر از چند عنصر خاص پشت سر هم:** در مثال زیر هدف انتخاب همه عناصر به غیر از عناصر دوم تا چهارم از بردار x می‏باشد.

x = c(1, 5, 0, 7, 1, 1, 2)  
x[-(2:4)]

## 1 1 1 2

* **انتخاب چندین عنصر همزمان:** در مثال زیر هدف انتخاب عناصر اول و پنجم از بردار x می‏باشد.

x = c(1, 5, 0, 7, 1, 1, 2)  
x[c (1, 5)]

## 1 1

می‏توان عناصر موجود در یک بردار را با استفاده از مقادیرشان نیز انتخاب کرد.

* **انتخاب عناصری از یک بردار که با یک مقدار خاص برابر هستند:** در مثال زیر هدف انتخاب عناصری از بردار x است که مقادیرشان برابر با 1 می‏باشد.

x = c(1, 5, 0, 7, 1, 1, 2)  
x[x == 1]

## 1 1 1

* **انتخاب عناصری از یک بردار با اعمال شرط روی مقادیر آن:** در مثال زیر هدف انتخاب عناصری از بردار x است که مقادیرشان بزرگتر از 1 می‏باشد.

x = c(1, 5, 0, 7, 1, 1, 2)  
x[x > 1]

## 5 7 2

* **انتخاب عناصری از یک بردار که با بردار دوم مشترک هستند:** در مثال زیر هدف انتخاب عناصری از بردار x است که مقادیرشان با مقادیر بردار c(1, 7) برابر می‏باشد.

x = c(1, 5, 0, 7, 1, 1, 2)  
x[x %in% c(1, 7)]

## 1 7 1 1

می‏توان عناصر موجود در یک بردار را با استفاده اسم آن‏ها نیز انتخاب کرد.

* **انتخاب یک عنصر از یک بردار با استفاده از اسم آن عنصر:** در مثال زیر هدف انتخاب عنصری از بردار x است که اسم آن برابر Year می‏باشد.

x = c("Age" = 30, "ID" = 12, "Year" = 2020)  
x["Year"]

## Year  
## 2020

* **انتخاب چند عنصر از یک بردار با استفاده از اسم آن عناصر:** در مثال زیر هدف انتخاب عناصری از بردار x است که اسم آن‏ها برابر با Age و Year می‏باشد.

x = c("Age" = 30, "ID" = 12, "Year" = 2020)  
x[c("Age", "Year")]

## Age Year  
## 30 2020

|  |
| --- |
| **برنامه‏نویسی** |

برتری زبان R نسبت به نرم‏افزارهای آماری مشابه خود این است که ویژگی‏های یک زبان برنامه‏نویسی در آن مستتر است و مانند سایر زبان‏های برنامه‏نویسی دارای ساختارهای کنترلی می‏باشد.

|  |
| --- |
| **دستورات شرطی** |

ساختارهای تصمیم‏گیری مستلزم این است که برنامه‏نویس یک یا چند شرط را مشخص کند که باید توسط برنامه برآورد یا تست شود، اگر این شرط درست و برقرار باشد، دستور یا دستوراتی باید اجرا شوند، و به صورت دلخواه می‏توان دستوراتی را نوشت که در صورت برقرار نبودن آن شروط اجرا شوند. در حالت کلی ساختار دستورات شرطی در زبان برنامه‏نویسی R به صورت زیر می‏باشد:

|  |
| --- |
| if (condition) {  Do Somethings  } else {  Do Somethings  } |

علاوه بر این اگر دستورات قسمت if و else به صورت تک دستوری باشند، می‏توان از تابع ifelse نیز استفاده نمود. ساختار تابع ifelse به صورت زیر می‏باشد:

|  |
| --- |
| ifelse(test = condition, yes = Do Something, no = Do Something) |

* **ساختارهای شرطی if:** در مثال زیر مقدار متغیر score برابر 15 است. در ساختار دستورات شرطی اگر مقدار score بزرگتر مساوی 10 باشد کلمه pass چاپ خواهد شد، در غیر اینصورت کلمه fail چاپ خواهد شد.

score = 15  
  
if (score >= 10) {  
 print(x = "Pass")  
} else {  
 print(x = "Fail")  
}

## "Pass"

score = 15  
  
ifelse(test = score >= 10, yes = print(x = "Pass"), no = print(x = "Fail"))

## "Pass"

|  |
| --- |
| **حلقه‏ها** |

حلقه‏های for و while در R برای تکرار شدن بخشی از کد استفاده می‏شود. تعداد تکرار با ایجاد شرط برای تابع while و یا تعیین مقدار مشخص یا مقدار متغیر برای تابع for، مشخص می‏شود.

**حلقه for**

حلقه for هنگامی مورد استفاده قرار می‏گیرد که تعداد تکرار مشخص باشد. در حالت کلی ساختار حلقه‏های for در زبان برنامه‏نویسی R به صورت زیر می‏باشد:

|  |
| --- |
| for (variable in sequence) {  Do Somethings  } |

که در آن variable، یک متغییر شمارنده، و sequence می‏تواند یک بردار باشد. دستورات درون بدنه حلقه به تعداد درایه‏های بردار sequence تکرار و اجرا می‏شود.

* **حلقه‏های for:** با اجرای دستورات زیر، دستور print چهار بار و به ازای هر i اجرا خواهد شد.

for (i in 1:4) {  
 print(i)  
}

## 1  
## 2  
## 3  
## 4

**حلقه while**

حلقه while تا زمانی ادامه می‏یابد که شرط برقرار باشد. در حالت کلی ساختار حلقه‏های while در زبان برنامه‏نویسی R به صورت زیر می‏باشد:

|  |
| --- |
| while (condition) {  Do Somethings  } |

که در آن condition، شرط ورود به حلقه است، و تا زمانی که این شرط برقرار باشد حلقه تکرار می‏شود.

* **حلقه‏های while:** با اجرای دستورات زیر، تا موقعی که مقدار i کوچکتر از 5 باشد، دستور print اجرا خواهد شد.

i = 1  
  
while (i < 5) {  
 print(i)  
 i = i + 1  
}

## 1  
## 2  
## 3  
## 4

|  |
| --- |
| **توابع** |

تابع یک بخشی از برنامه است که وظیفه خاصی را انجام می‏دهد. در حالت کلی ساختار توابع در زبان برنامه‏نویسی R به صورت زیر می‏باشد:

|  |
| --- |
| function\_name = function (variables) {  Do Somethings  return(new\_variable)  } |

که در آن function\_name، نامی است که برای تابع انتخاب می‏شود، variables آرگمان‏های ورودی تابع و new\_variable متغیری است که از تابع به عنوان نتیجه، خارج می‏شود.

* **توابع:** با اجرای دستورات زیر، تابعی با عنوان square، که آرگمان x را به عنوان ورودی گرفته و توان دوم آن را به عنوان خروجی تحویل می‏دهد، در محیط کاری برنامه R ایجاد می‏شود. با فراخوانی نام تابع و دادن ورودی به آن، خروجی مورد نظر نمایش داده می‏شود.

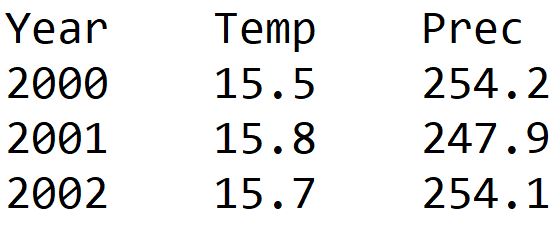
square <- **function**(x) {  
 z = x **\*** x  
 **return**(z)  
}  
  
**square**(x = 5)

## 25

|  |
| --- |
| **خواندن و نوشتن داده‏ها** |

**خواندن و نوشتن داده‏هایی که با یک کارکتر خاص از همدیگر جدا شده‏اند**

فرض کنید داده‏های میانگین دما و مجموع بارش سالانه برای سه سال متوالی در فایلی به نام data.txt ذخیره شده باشد. مقادیر هر سطر با فاصله یک تب از هم جدا شده‏اند.



* **تابع read.table:** با اجرای دستورات زیر، محتوای فایل data.txt خوانده شده و در متغیر df ذخیره می‏گردد.

df = read.table(file = "C:/Users/Pooya/Desktop/data.txt",  
 header = TRUE,  
 sep = "\t")  
  
print(df)

## Year Temp Prec  
## 1 2000 15.5 254.2  
## 2 2001 15.8 247.9  
## 3 2002 15.7 254.1

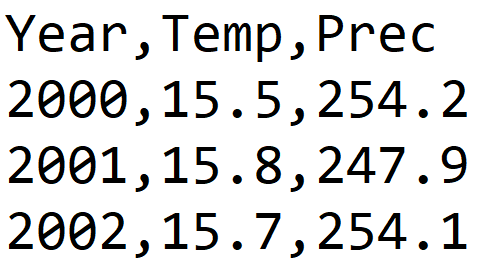
که آرگومان file برای مشخص کردن آدرس فایل data.txt، و آرگومان sep برای مشخص کردن اینکه داده‏ها با چه کارکتری از هم دیگر جدا شده‏اند، استفاده شده است.

* **تابع write.table:** با اجرای دستورات زیر، محتوای متغیر df در فایل newData.txt ذخیره می‏گردد. در فایل جدید داده‏ها با فاصله تب از همدیگر جدا می‏شوند.

write.table(x = df,  
 file = "C:/Users/Pooya/Desktop/newData.txt",  
 sep = "\t")

**خواندن و نوشتن داده‏هایی که با کاما از همدیگر جدا شده‏اند**

این یک حالت خاص از توابع read.table و write.table می‏باشد. فرض کنید داده‏های میانگین دما و مجموع بارش سالانه برای سه سال متوالی در فایلی به نام data.csv ذخیره شده باشد. مقادیر هر سطر با یک کاما از همدیگر جدا شده‏اند.



* **تابع read.csv:** با اجرای دستورات زیر، محتوای فایل data.csv خوانده شده و در متغیر df ذخیره می‏گردد.

df = read.csv(file = "C:/Users/Pooya/Desktop/data.csv",  
 header = TRUE)  
  
print(df)

## Year Temp Prec  
## 1 2000 15.5 254.2  
## 2 2001 15.8 247.9  
## 3 2002 15.7 254.1

* **تابع write.csv:** با اجرای دستورات زیر، محتوای متغیر df در فایل newData.csv ذخیره می‏گردد. در فایل جدید داده‏ها با یک کاما از همدیگر جدا می‏شوند.

write.csv (x = df,  
 file = "C:/Users/Pooya/Desktop/newData.csv")

|  |
| --- |
| **عملگرهای منطقی** |

عملگرهای منطقی برای شرطی کردن اجرای دستورات استفاده می‏شوند. خروجی این عملگرها در صورت برقرار بودن شرط TRUE و در صورتیکه شرط برقرار نباشد FALSE می‏باشد.

|  |  |
| --- | --- |
| **عملگر** | **شرح** |
| > | بزرگتر |
| < | کوچکتر |
| >= | بزرگتر یا مساوی |
| <= | کوچکتر یا مساوی |
| == | مساوی |
| != | نامساوی |
| ! | نقیض |
| & | "و" منطقی |
| | | "یا" منطقی |

* **عملگرهای منطقی:** با اجرای دستورات زیر، با توجه به نوع عملگر، خروجی TRUE یا FALSE نمایش داده می‏شود.

5 > 4  
5 < 4  
5 >= 4  
5 <= 4  
5 == 4  
5 != 4  
!4  
(5 **<** 4) **&** (4 **<** 10)  
(5 **<** 4) **|** (4 **<** 10)

## TRUE  
## FALSE  
## TRUE  
## FALSE  
## FALSE  
## TRUE  
## FALSE  
## FALSE  
## TRUE

|  |
| --- |
| **نوع داده‏ها در R** |

در حالت کلی می‏توان داده‏ها را در برنامه R به پنج زیر مجموعه زیر تقسیم‏بندی کرد:

|  |  |
| --- | --- |
| نوع داده | مثال |
| منطقی (logical) | x = TRUE print(class(x = x))  ## "logical" |
| عددی (numeric) | x = 12.5 print(class(x = x))  ## "numeric" |
| صحیح (integer) | x = 2L print(class(x = x))  ## "integer" |
| مختلط (complex) | x = 3 + 2i print(class(x = x))  ## "complex" |
| متن (character) | x = "Hello" print(class(x = x))  ## "character" |

|  |
| --- |
| **عملگر تخصیص** |

برای ذخیره یک ساختار داده باید به آن یک نام اختصاص داد. بدین منظور از عملگرهای تخصیص “=” و “<-” استفاده می‏شود.

* **عملگرتخصیص:** با اجرای دستورات زیر، مقدار 5 در متغیر x و مقدار TRUE در متغیر y ذخیره می‏گردد.

x = 5  
y <- TRUE  
  
print(x)  
print(x)

## 5  
## TRUE

|  |
| --- |
| **مدیریت متغیرها** |

* **مشاهده همه متغیرهای محیط کاری R با استفاده از تابع ls:** با اجرای دستورات زیر، متغیرهای موجود در محیط کاری نمایش داده می‏شوند.

x = 2  
y = TRUE  
z = "Hello"  
  
ls()

## "x" "y" "z"

* **حذف یک متغیر از محیط کاری R با استفاده از تابع rm:** با اجرای دستورات زیر، متغیر x حذف خواهد شد.

x = 2  
y = TRUE  
z = "Hello"  
  
ls()  
**rm**(x)  
**ls**()

## "x" "y" "z"  
## "y" "z"

* **حذف همه متغیرها از محیط کاری R با استفاده از تابع rm:** با اجرای دستورات زیر، همه متغیرها حذف خواهند شد.

x = 2  
y = TRUE  
z = "Hello"  
  
**rm**(list = **ls**())  
**ls**()

## character(0)

|  |
| --- |
| **توابع ریاضی در R** |

* **لگاریتم طبیعی با استفاده از تابع log:** با اجرای دستورات زیر، لگاریتم طبیعی همه عناصر موجود در بردار x محاسبه می‏شود.Inf، به معنی بینهایت و NaN، به معنی وجود نداشتن مقداری است.

x = **c**(1, 25, 0, 4, 1, 100, -1, 44, 100, -1)  
**log**(x = x)

## 0.0 3.2 -Inf 1.4 0.0 4.6 NaN 3.8 4.6 NaN

* **لگاریتم بر پایه 10 با استفاده از تابع log10:** با اجرای دستورات زیر، لگاریتم بر پایه 10 همه عناصر موجود در بردار x محاسبه می‏شود.Inf، به معنی بینهایت و NaN، به معنی وجود نداشتن مقداری است.

x = **c**(1, 25, 0, 4, 1, 100, -1, 44, 100, -1)  
**log10**(x = x)

## 0.0 1.4 -Inf 0.6 0.0 2.0 NaN 1.6 2.0 NaN

* **تابع نمایی () با استفاده از تابع exp:** با اجرای دستورات زیر، تابع نمایی عناصر موجود در بردار x محاسبه می‏شود.

x = **c**(1, 25, 0, 4, 1, 100, -1, 44, 100, -1)  
**exp**(x = x)

## 2.7e+00 7.2e+10 1.0e+00 5.5e+01 2.7e+00 2.7e+43 3.7e-01 1.3e+19 2.7e+43 3.7e-01

* **پیدا کردن مقدار بیشینه عناصر یک بردار با استفاده از تابع max:** با اجرای دستورات زیر، بزرگترین عنصر موجود در بردار x نمایش داده می‏شود.

x = **c**(1, 25, 0, 4, 1, 100, -1, 44, 100, -1)  
**max**(x)

## 100

* **پیدا کردن مقدار کمینه عناصر یک بردار با استفاده از تابع min:** با اجرای دستورات زیر، کوچکترین عنصر موجود در بردار x نمایش داده می‏شود.

x = **c**(1, 25, 0, 4, 1, 100, -1, 44, 100, -1)  
**min**(x)

## -1

* **پیدا کردن مجموع همه عناصر یک بردار با استفاده از تابع sum:** با اجرای دستورات زیر، مجموع همه عناصر موجود در بردار x محاسبه و نمایش داده می‏شود.

x = **c**(1, 25, 0, 4, 1, 100, -1, 44, 100, -1)  
**sum**(x)

## 273

* **پیدا کردن میانگین همه عناصر یک بردار با استفاده از تابع mean:** با اجرای دستورات زیر، میانگین همه عناصر موجود در بردار x محاسبه و نمایش داده می‏شود.

x = **c**(1, 25, 0, 4, 1, 100, -1, 44, 100, -1)  
**mean**(x)

## 27

* **پیدا کردن میانه عناصر یک بردار با استفاده از تابع median:** با اجرای دستورات زیر، میانه عناصر موجود در بردار x محاسبه و نمایش داده می‏شود.

x = **c**(1, 25, 0, 4, 1, 100, -1, 44, 100, -1)  
**median**(x = x)

## 2.5

* **پیدا کردن واریانس همه عناصر یک بردار با استفاده از تابع var:** با اجرای دستورات زیر، واریانس همه عناصر موجود در بردار x محاسبه و نمایش داده می‏شود.

x = **c**(1, 25, 0, 4, 1, 100, -1, 44, 100, -1)  
**var**(x = x)

## 1681

* **پیدا کردن انحراف معیار همه عناصر یک بردار با استفاده از تابع sd:** با اجرای دستورات زیر، انحراف معیار همه عناصر موجود در بردار x محاسبه و نمایش داده می‏شود.

x = **c**(1, 25, 0, 4, 1, 100, -1, 44, 100, -1)  
**sd**(x = x)

## 41

* **محاسبه ضریب همبستگی بین دو بردار با استفاده از تابع cor:** با اجرای دستورات زیر، ضریب همبستگی بین دو بردار x و y محاسبه و نمایش داده می‏شود.

x = **c**(2, 8, 4, 5, 1)  
y = **c**(1, 7, 4, 5, 3)  
**cor**(x = x, y = y)

## 0.9

* **پیدا کردن رتبه عناصر یک بردار با استفاده از تابع rank:** با اجرای دستورات زیر، رتبه هر یک از عناصر موجود در بردار x محاسبه و نمایش داده می‏شود.

x = **c**(1, 25, 0, 4, 1, 100, -1, 44, 100, -1)  
**rank**(x = x)

## 4.5 7.0 3.0 6.0 4.5 9.5 1.5 8.0 9.5 1.5

* **گرد کردن اعداد با استفاده از تابع round:** با اجرای دستورات زیر، همه عناصر بردار x تا دو رقم اعشار گرد می‏شوند.

x = **c**(1.254, 100.24, 5.8975)  
**round**(x = x, digits = 2)

## 1.25 100.24 5.90

|  |
| --- |
| **ماتریس‏ها** |

ماتریس‏ها یکی از شی‏ء‏های زبان برنامه‏نویسی R می‏باشند که در آن، عناصر در یک طرح مستطیلی دو بعدی مرتب می‏شوند. عناصر ماتریس‏ها می‏توانند شامل نوع داده‏های مختلف مانند منطقی، عددی، صحیح، مختلط و متنی باشند. نکته مهم در اینجا این است که همه عناصر موجود در یک ماتریس باید از یک نوع داده باشند.

* **ایجاد یک ماتریس با استفاده از تابع matrix:** با اجرای دستورات زیر، ماتریسی با عناصر بردار c(1, 2, 3, 4, 5, 6) تشکیل می‏شود. این ماتریس دارای 2 سطر و 3 ستون بوده و در صورتیکه مقدار آرگمان byrow برابر با TRUE باشد مقادیر بردار، به صورت سطر به سطر ماتریس را تشکیل می‏دهد. در صورتیکه مقدار آرگمان byrow برابر با FALSE باشد ماتریس به صورت ستون به ستون با استفاده از مقادیر بردار مورد نظر تشکیل خواهد شد.

x = matrix(data = c(1, 2, 3, 4, 5, 6),  
 nrow = 2,  
 ncol = 3,  
 byrow = TRUE)  
  
print(x)

## [,1] [,2] [,3]  
## [1,] 1 2 3  
## [2,] 4 5 6

x = matrix(data = c(1, 2, 3, 4, 5, 6),  
 nrow = 2,  
 ncol = 3,  
 byrow = FALSE)  
  
print(x)

## [,1] [,2] [,3]  
## [1,] 1 3 5  
## [2,] 2 4 6

* **انتخاب یک سطر از ماتریس:** با اجرای دستورات زیر، تمام عناصر موجود در سطر دوم ماتریس x انتخاب و نمایش داده می‏شود.

x = matrix(data = 1:12,  
 nrow = 4,  
 ncol = 3,  
 byrow = TRUE)  
  
x[2, ]

## 4 5 6

* **انتخاب چند سطر از ماتریس:** با اجرای دستورات زیر، تمام عناصر موجود در سطرهای دوم و چهارم ماتریس x انتخاب و نمایش داده می‏شود.

x = matrix(data = 1:12,  
 nrow = 4,  
 ncol = 3,  
 byrow = TRUE)  
  
x[**c**(2, 4), ]

## [,1] [,2] [,3]  
## [1,] 4 5 6  
## [2,] 10 11 12

* **انتخاب یک ستون از ماتریس:** با اجرای دستورات زیر، تمام عناصر موجود در ستون دوم ماتریس x انتخاب و نمایش داده می‏شود.

x = matrix(data = 1:12,  
 nrow = 4,  
 ncol = 3,  
 byrow = TRUE)  
  
x[ , 2]

## 2 5 8 11

* **انتخاب چند ستون از ماتریس:** با اجرای دستورات زیر، تمام عناصر موجود در ستون‏های دوم و سوم ماتریس x انتخاب و نمایش داده می‏شود.

x = matrix(data = 1:12,  
 nrow = 4,  
 ncol = 3,  
 byrow = TRUE)  
  
x[ , 2:3]

## [,1] [,2]  
## [1,] 2 3  
## [2,] 5 6  
## [3,] 8 9  
## [4,] 11 12

* **انتخاب یک عنصر خاص از ماتریس:** با اجرای دستورات زیر، عنصر موجود در سطر سوم و ستون دو ماتریس x انتخاب و نمایش داده می‏شود.

x = matrix(data = 1:12,  
 nrow = 4,  
 ncol = 3,  
 byrow = TRUE)  
  
x[3, 2]

## 8

|  |
| --- |
| **لیست‏ها** |

در زبان برنامه‏نویسی R، لیست‏ها مجموعه‏ای منظم از عناصر می‏باشند که ضرورتی ندارد که مولفه‏های آن از یک نوع داده و ابعاد آن با هم مساوی باشد. به عنوان مثال یک لیست می‏تواند شامل بردارهای منطقی، عددی، صحیح، اعداد مختلط و متنی باشد.

* **ایجاد یک لیست با استفاده از تابع list:** با اجرای دستورات زیر، یک لیست با عنوان l تشکیل می‏شود. این لیست شامل دو مقدار است. x، که برداری از نوع عددی و مقادیر 1 تا 5 و y، که یک بردار متنی شامل کارکترهای a و b.

l = list(x = 1:5,  
 y = c("a", "b"))  
  
print(l)

## $x  
## 1 2 3 4 5  
## $y  
## "a" "b"

* **انتخاب یک عنصر خاص از یک لیست با شماره ایندکس آن (خروجی یک لیست باشد):** با اجرای دستورات زیر، عنصر‏ دوم (y) انتخاب و نمایش داده می‏شود. خروجی در این حالت یک لیست می‏باشد.

l = list(x = 1:5,  
 y = c("a", "b"))  
  
l[2]

## $y  
## "a" "b"

* **انتخاب یک عنصر خاص از یک لیست با شماره ایندکس آن (خروجی بستگی به ماهیت آن عنصر دارد):** با اجرای دستورات زیر، عنصر‏ دوم (y) انتخاب و نمایش داده می‏شود. خروجی در این حالت یک بردار از کارکترها می‏باشد.

l = list(x = 1:5,  
 y = c("a", "b"))  
  
l[[2]]

## "a" "b"

* **انتخاب یک عنصر خاص از یک لیست با استفاده از نام آن (خروجی یک لیست باشد):** با اجرای دستورات زیر، عنصر‏ اول (x) انتخاب و نمایش داده می‏شود. خروجی در این حالت یک لیست می‏باشد.

l = list(x = 1:5,  
 y = c("a", "b"))  
  
l["x"]

## $x  
## 1 2 3 4 5

* **انتخاب یک عنصر خاص از یک لیست با استفاده از نام آن (خروجی بستگی به ماهیت آن عنصر دارد):** با اجرای دستورات زیر، عنصر‏ اول (x) انتخاب و نمایش داده می‏شود. خروجی در این حالت یک بردار از نوع عددی می‏باشد.

l = list(x = 1:5,  
 y = c("a", "b"))  
  
l$x

## 1 2 3 4 5

|  |
| --- |
| **دیتا فریم‏ها** |

دیتا فریم‏ها از نظر ساختاری زیر مجموعه لیست‏ها حساب می‏شوند ولی شباهت بسیار زیادی به ماتریس‏ها دارند. یک دیتا فریم را می‏توان ماتریسی در نظر گرفت که ستون‏های آن بردارهایی با ساختارهای مختلف داده است. از سوی دیگر می‏توان گفت که دیتا فریم لیستی از بردارهای هم اندازه و با ساختارهای مختلف است.

* **ایجاد یک دیتا فریم با استفاده از تابع data.frame:** با اجرای دستورات زیر، یک دیتا فریم با عنوان df تشکیل می‏شود. این دیتا فریم شامل دو عنصر یا ستون با سه عضو یا سطر می‏باشد. x، که برداری از نوع عددی و مقادیر 1 تا 3 و y، که یک بردار متنی شامل کارکترهای a، b و c می‏باشد.

df = data.frame(x = 1:3,  
 y = c("a", "b", "c"))  
  
print(df)

## x y  
## 1 1 a  
## 2 2 b  
## 3 3 c

* **انتخاب یک ستون خاص از یک دیتا فریم** **با شماره ایندکس آن (خروجی بستگی به ماهیت آن ستون دارد):** با اجرای دستورات زیر، ستون‏ دوم دیتا فریم (y) انتخاب و نمایش داده می‏شود. خروجی در این حالت یک بردار متنی شامل کارکترهای a، b و c می‏باشد.

df = data.frame(x = 1:3,  
 y = c("a", "b", "c"))  
  
df[, 2]

## "a" "b" "c"

* **انتخاب یک ستون خاص از یک دیتا فریم با استفاده از نام آن ستون (خروجی بستگی به ماهیت آن ستون دارد):** با اجرای دستورات زیر، ستون‏ دوم دیتا فریم (y) انتخاب و نمایش داده می‏شود. خروجی در این حالت یک بردار متنی شامل کارکترهای a، b و c می‏باشد.

df = data.frame(x = 1:3,  
 y = c("a", "b", "c"))  
  
df$y

## "a" "b" "c"

* **انتخاب یک ستون خاص از یک دیتا فریم با استفاده از نام آن ستون (خروجی یک دیتا فرم است):** با اجرای دستورات زیر، ستون‏ دوم دیتا فریم (y) انتخاب و نمایش داده می‏شود. خروجی در این حالت یک دیتا فریم با سه سطر و یک ستون می‏باشد.

df = data.frame(x = 1:3,  
 y = c("a", "b", "c"))  
  
df["y"]

## y  
## 1 a  
## 2 b  
## 3 c

* **انتخاب یک سطر خاص از یک دیتا فریم با استفاده شماره ایندکس آن سطر (خروجی یک دیتا فرم است):** با اجرای دستورات زیر، سطر‏ دوم دیتا فریم انتخاب و نمایش داده می‏شود. خروجی در این حالت یک دیتا فریم با یک سطر و دو ستون می‏باشد.

df = data.frame(x = 1:3,  
 y = c("a", "b", "c"))  
  
df[2, ]

## x y  
## 1 2 b

* **انتخاب یک عنصر خاص از یک دیتا فریم با استفاده شماره ایندکس سطر و ستون آن (خروجی بستگی به ماهیت آن عنصر دارد):** با اجرای دستورات زیر، سطر‏ دوم از ستون دوم دیتا فریم انتخاب و نمایش داده می‏شود. خروجی در این حالت یک کارکتر می‏باشد.

df = data.frame(x = 1:3,  
 y = c("a", "b", "c"))  
  
df[2, 2]

## b

* **مشاهده ابتدا و انتهای یک دیتا فریم با استفاده از توابع head و tail:** با اجرای دستورات زیر، سه سطر اول و دو سطر آخر دیتا فرم df نمایش داده می‏شود.

df = data.frame(x = 1:5,  
 y = LETTERS[1:5],  
 z = c(TRUE, TRUE, FALSE, TRUE, FALSE))  
  
head(x = df, n = 3)

## x y z  
## 1 1 A TRUE  
## 2 2 B TRUE  
## 3 3 C FALSE

df = data.frame(x = 1:5,  
 y = LETTERS[1:5],  
 z = c(TRUE, TRUE, FALSE, TRUE, FALSE))  
  
tail(x = df, n = 2)

## x y z  
## 4 4 D TRUE  
## 5 5 E FALSE

* **مشاهده تعداد سطرها، تعداد ستون‏ها و ابعاد یک دیتا فریم به ترتیب با استفاده از توابع nrow، ncol و dim:** با اجرای دستورات زیر، تعداد سطرها، تعداد ستون‏ها و ابعاد دیتا فریم df نمایش داده می‏شود.

df = data.frame(x = 1:5,  
 y = LETTERS[1:5],  
 z = c(TRUE, TRUE, FALSE, TRUE, FALSE))  
  
nrow(x = df)  
  
**ncol**(x = df)  
  
**dim**(x = df)

## 5  
## 3  
## 5 3

* **اضافه کردن ستون به یک دیتا فریم با استفاده از تابع cbind:** با اجرای دستورات زیر، بردار w به عنوان ستون جدید به دیتا فریم df اضافه می‏گردد. تعداد درایه‏های بردار w باید با تعداد سطرهای دیتا فریم df برابر باشد.

df = data.frame(x = 1:5,  
 y = LETTERS[1:5],  
 z = c(TRUE, TRUE, FALSE, TRUE, FALSE))  
  
print(x = df)

## x y z  
## 1 1 A TRUE  
## 2 2 B TRUE  
## 3 3 C FALSE  
## 4 4 D TRUE  
## 5 5 E FALSE

w = **c**(10L, 20L, 30L, 40L, 50L)  
  
df = **cbind**(df, w)  
  
**print**(df)

## x y z w  
## 1 1 A TRUE 10  
## 2 2 B TRUE 20  
## 3 3 C FALSE 30  
## 4 4 D TRUE 40  
## 5 5 E FALSE 50

* **اضافه کردن سطر به یک دیتا فریم با استفاده از تابع rbind:** با اجرای دستورات زیر، دیتا فریم w به عنوان سطر جدید به انتهای دیتا فریم df اضافه می‏گردد. تعداد ستون‏های دیتا فریم w و نوع داده هر ستون باید با تعداد ستون‏ها و نوع داده‏های دیتا فریم df برابر باشد.

df = data.frame(x = 1:5,  
 y = LETTERS[1:5],  
 z = c(TRUE, TRUE, FALSE, TRUE, FALSE))  
  
w = data.frame(x = 6, y = "F", z = FALSE)  
  
df = rbind(df, w)  
  
print(df)

## x y z  
## 1 1 A TRUE  
## 2 2 B TRUE  
## 3 3 C FALSE  
## 4 4 D TRUE  
## 5 5 E FALSE  
## 6 6 F FALSE

|  |
| --- |
| **توزیع‏های احتمالی** |

در زبان برنامه نویسی R بیشتر توابع احتمالی وجود دارد. هر تابع دارای چهار شکل متفاوت است:

* دستور dfunc(x, …): عرض تابع را در نقطه x نشان می‏دهد (Density Function).
* دستور pfunc(x, …): مقدار احتمال تجمعی را تا نقطه x نشان می‏دهد (Cumulative Dist.).
* دستور qfunc(p, …): مقدار چندک تابع را به ازای 0 < p < 1 نشان می‏دهد (Quantile).
* دستور rfunc(x, …): نمونه تصادفی از تابع را شبیه‏سازی می‏کند (Random Variates).

در جدول زیر، funcهای مختلف موجود در زبان برنامه‏نویسی R نشان داده شده است:

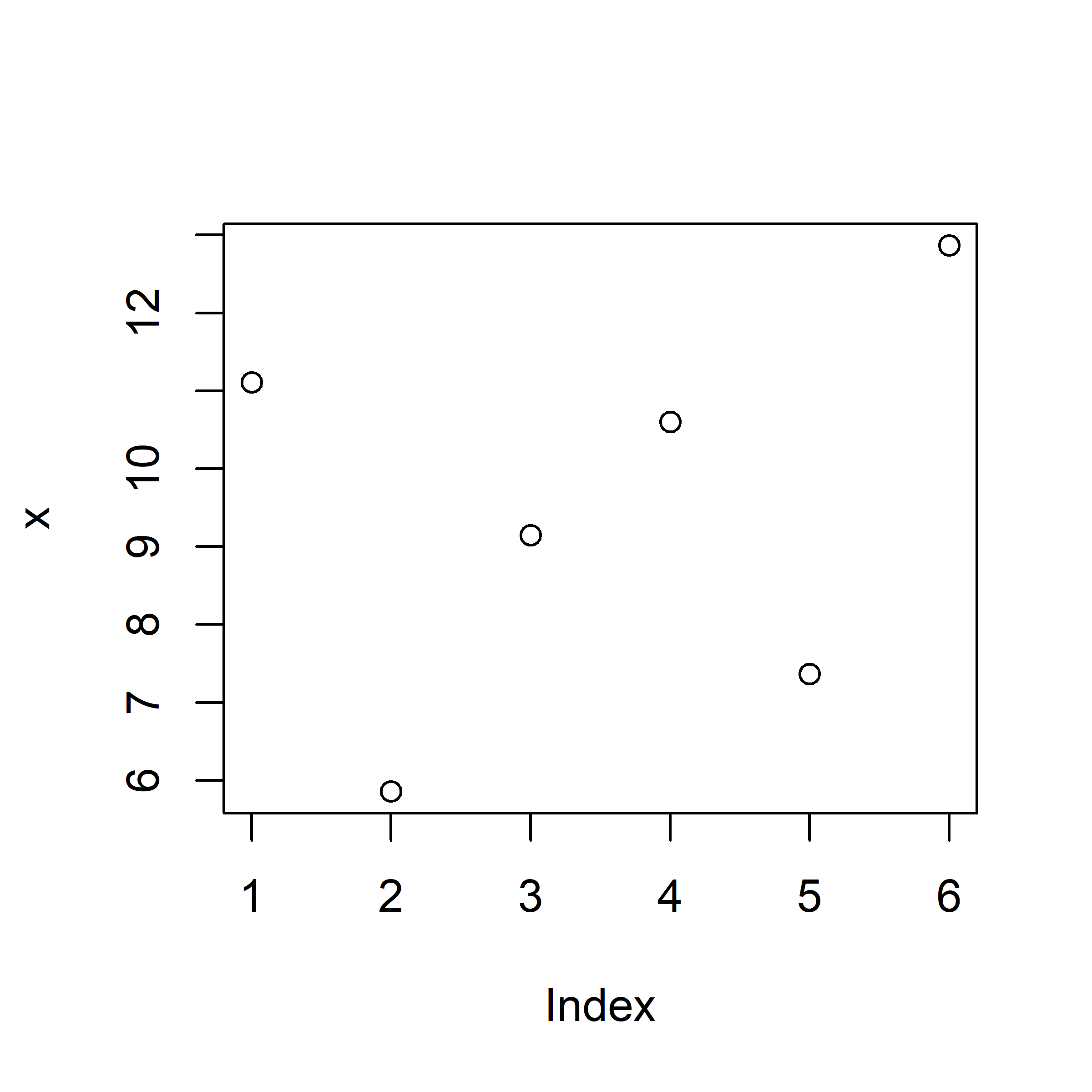
|  |  |  |
| --- | --- | --- |
| **آرگمان‏های ورودی** | **func** | **توزیع** |
| shape1, shape2 | beta | Beta |
| size, prob | binom | Binomial |
| df, ncp | chisq | Chi Square |
| df1, df2, ncp | f | F |
| shapem, rate, scale | gamma | Gamma |
| location, scale | logis | Logistic |
| meanlog, sdlog | lnorm | Log Normal |
| mean, sd | norm | Normal |
| lambda | pois | Poisson |
| df, ncp | t | t Student |
| shape, scale | Weibull | Weibull |
| min, max | unif | Uniform |

|  |
| --- |
| **نمودارها** |

در زبان برنامه‏نویسی R می‏توان نمودارها متفاوت و مختلفی را رسم نمود.

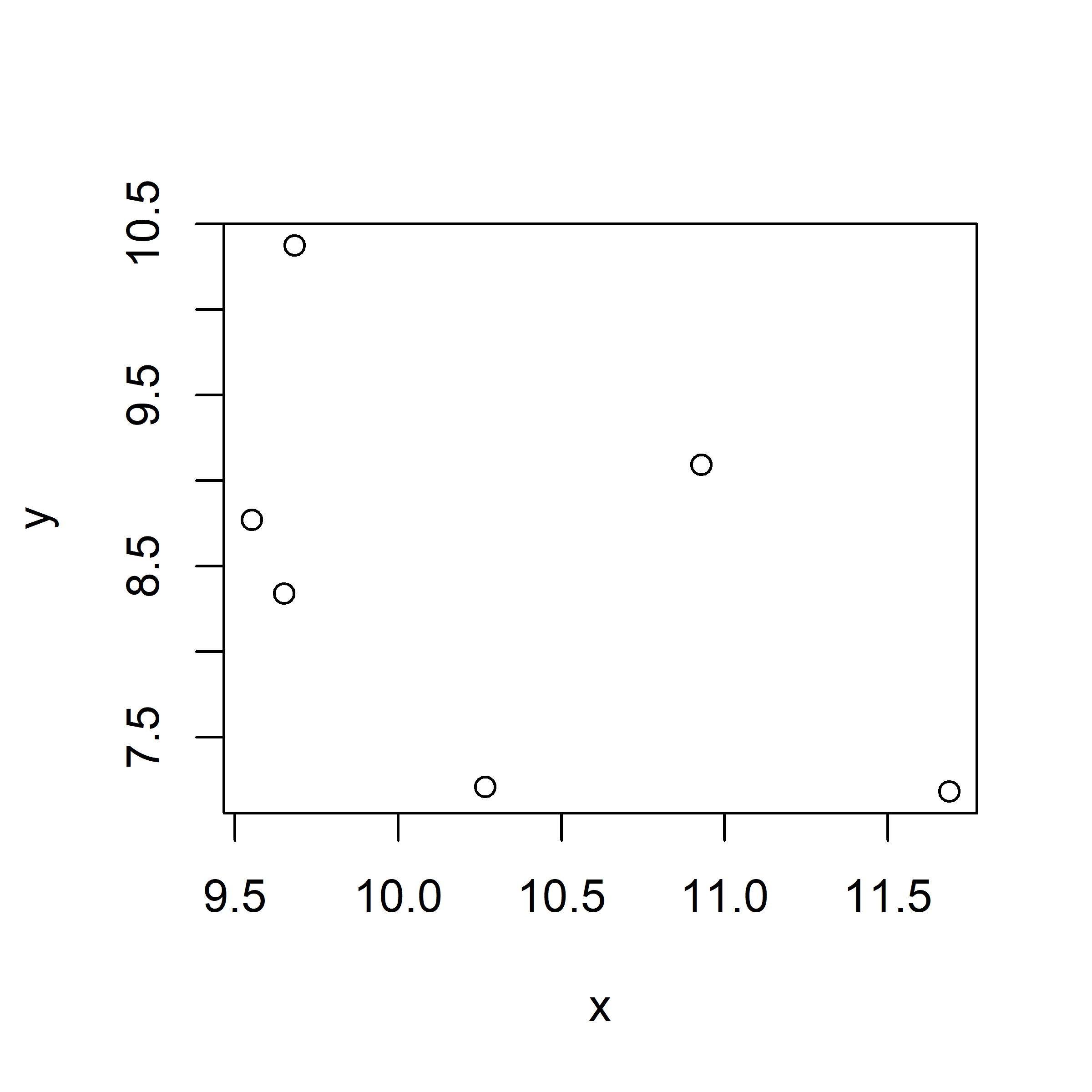
* **رسم نمودار بردار x با استفاده از تابع plot:** با اجرای دستورات زیر، نمودار پراکندگی بردار x نسبت به شماره ایندکس آن‏ها رسم می‏گردد.

x = rnorm(n = 6, mean = 10, sd = 2)  
  
plot(x = x)



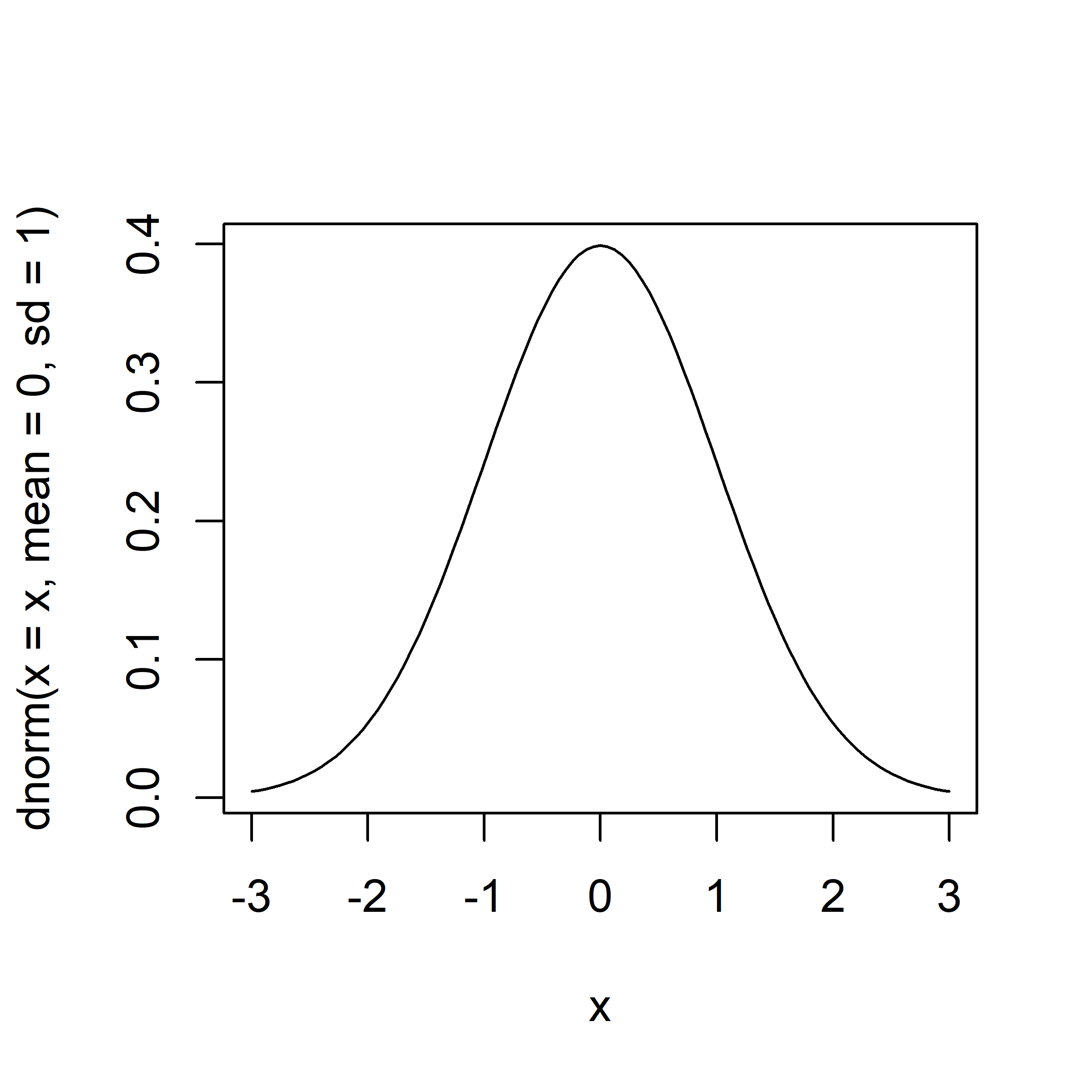
* **رسم نمودار پراکندگی بردار x در مقابل بردار y با استفاده از تابع plot:** با اجرای دستورات زیر، نمودار پراکندگی بردار x در مقابل بردار y رسم می‏گردد.

x = rnorm(n = 6, mean = 10, sd = 2)  
y = rnorm(n = 6, mean = 10, sd = 2)  
  
plot(x = x, y = y)



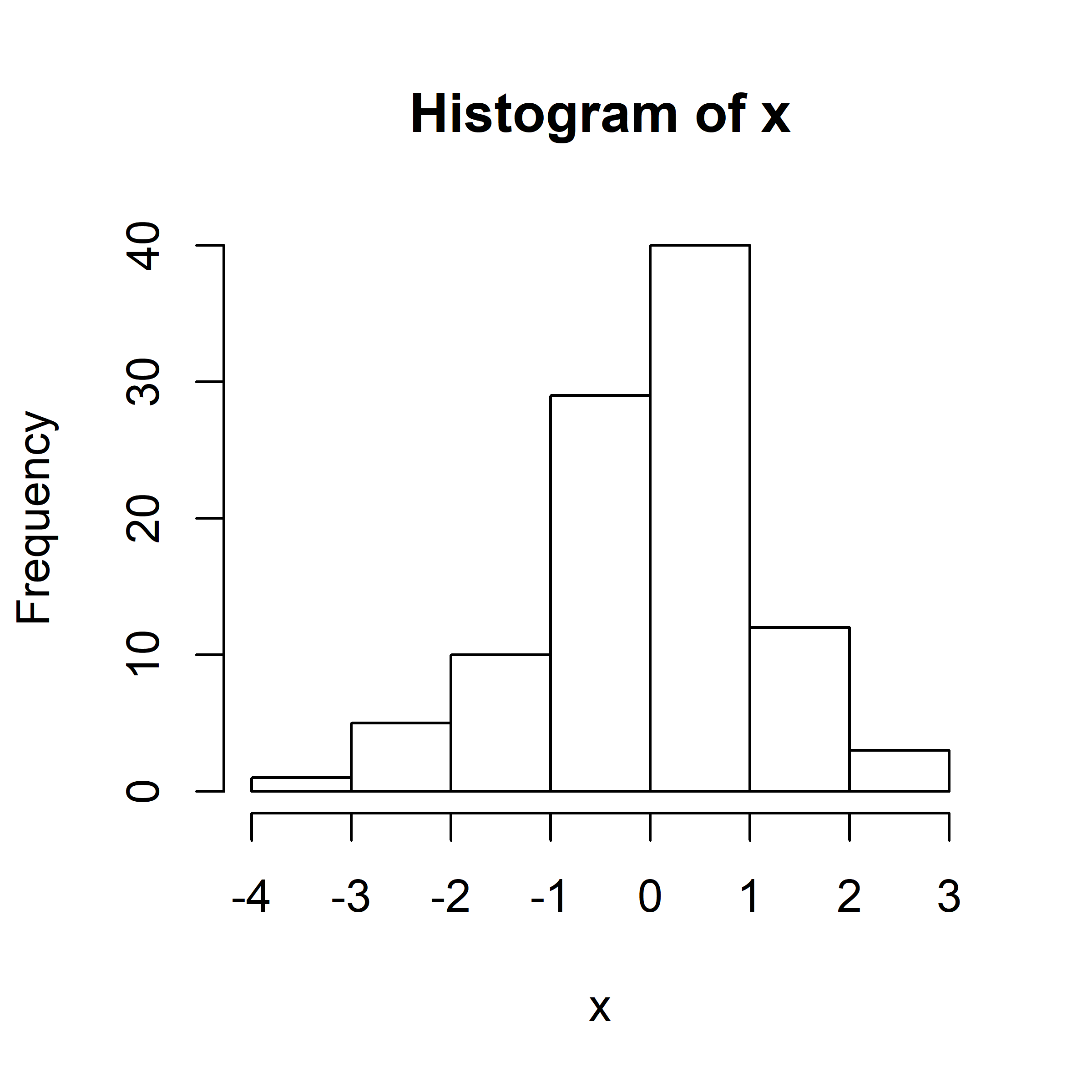
* **رسم منحنی یک تابع در محدوده تعیین شده با استفاده از تابع curve:** با اجرای دستورات زیر، منحنی توزیع آماری نرمال در فاصله 3- تا 3 رسم می‏گردد..

curve(expr = dnorm(x = x, mean = 0, sd = 1), from = -3, to = 3)



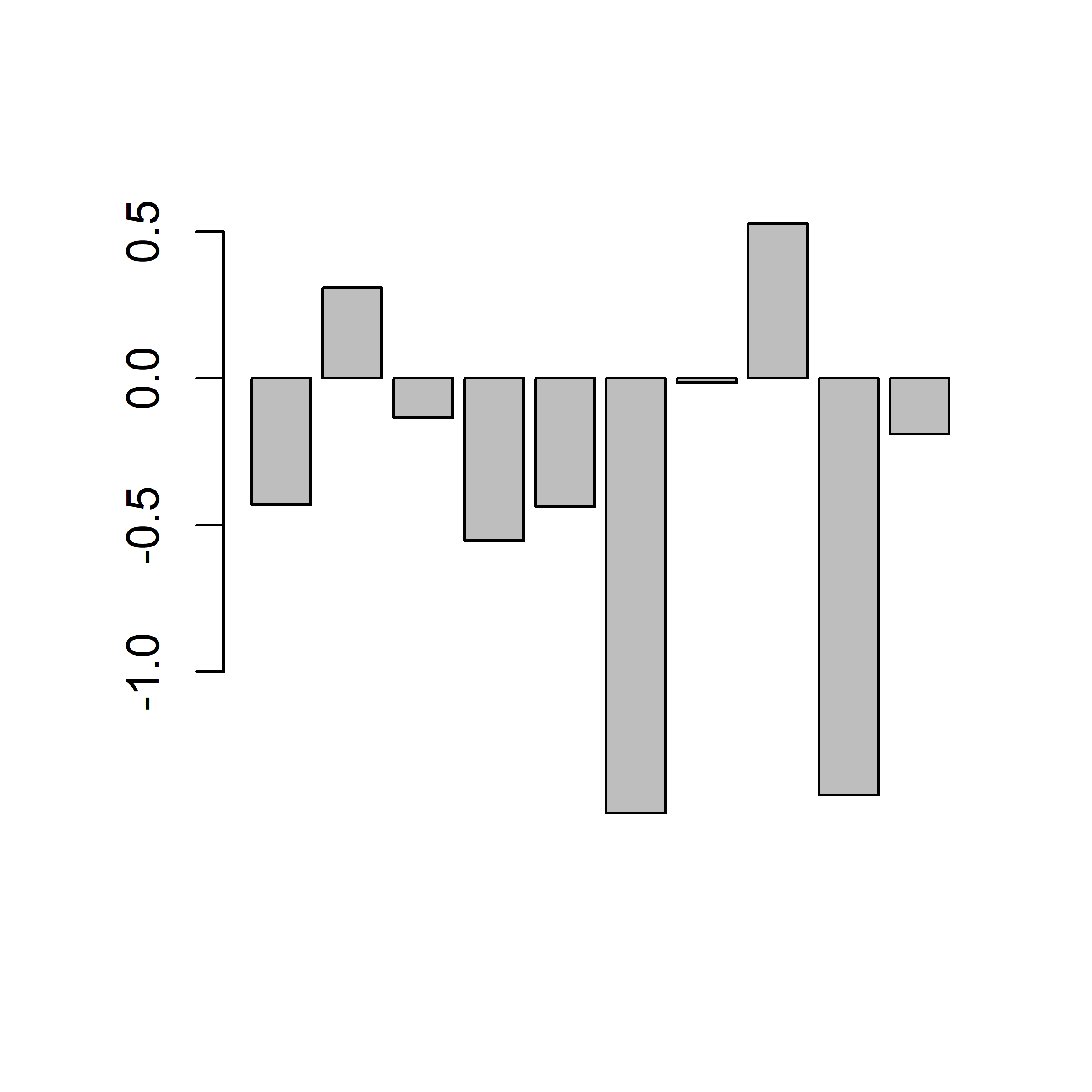
* **رسم هیستوگرام یک بردار با استفاده از تابع hist:** با اجرای دستورات زیر، هیستوگرام بردار x رسم خواهد شد.

x = rnorm(n = 100, mean = 0, sd = 1)  
hist(x = x)



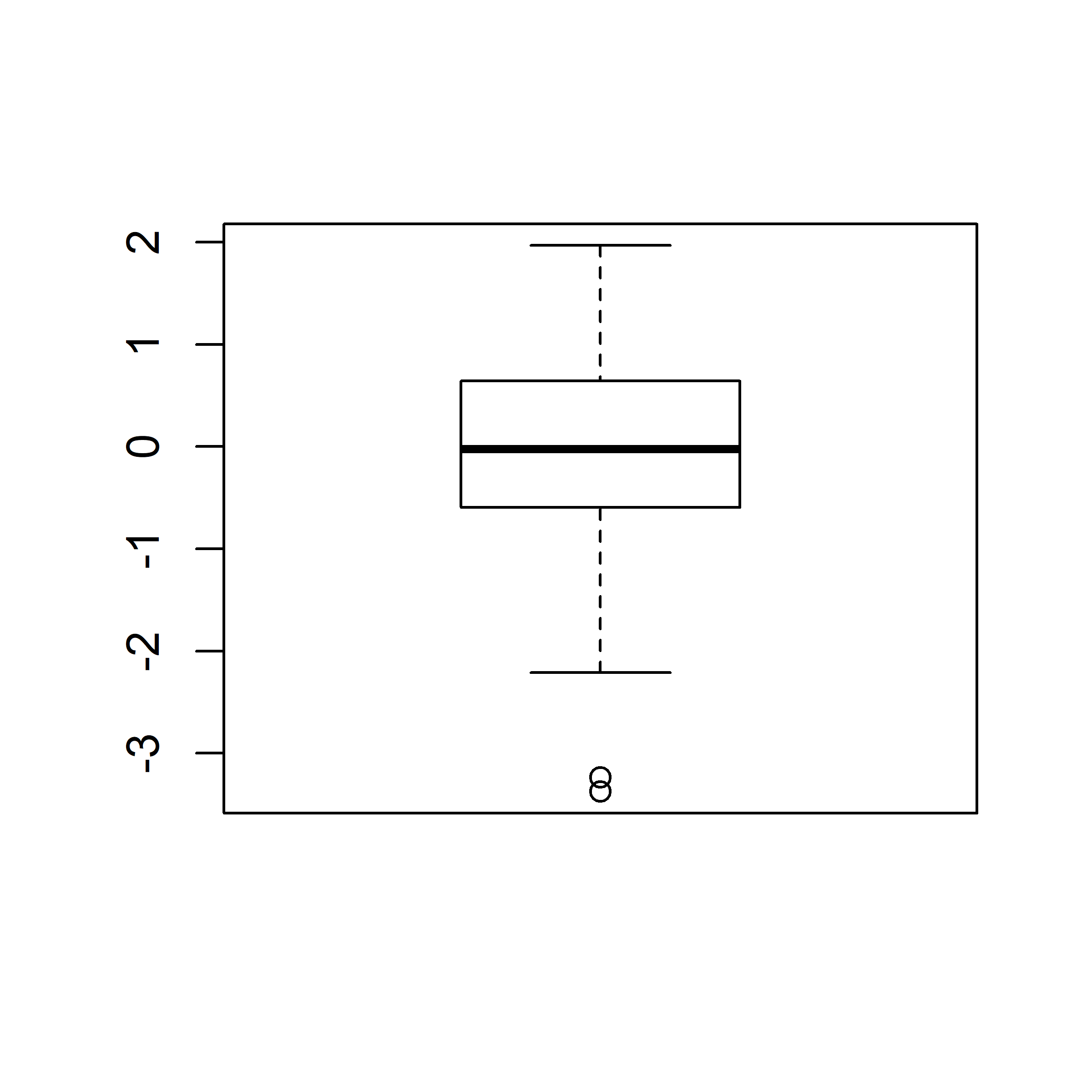
* **رسم نمودار میله‏ای یک بردار با استفاده از تابع barplot:** با اجرای دستورات زیر، نمودار میله‏ای بردار x رسم خواهد شد.

x = rnorm(n = 10, mean = 0, sd = 1)  
barplot(height = x)



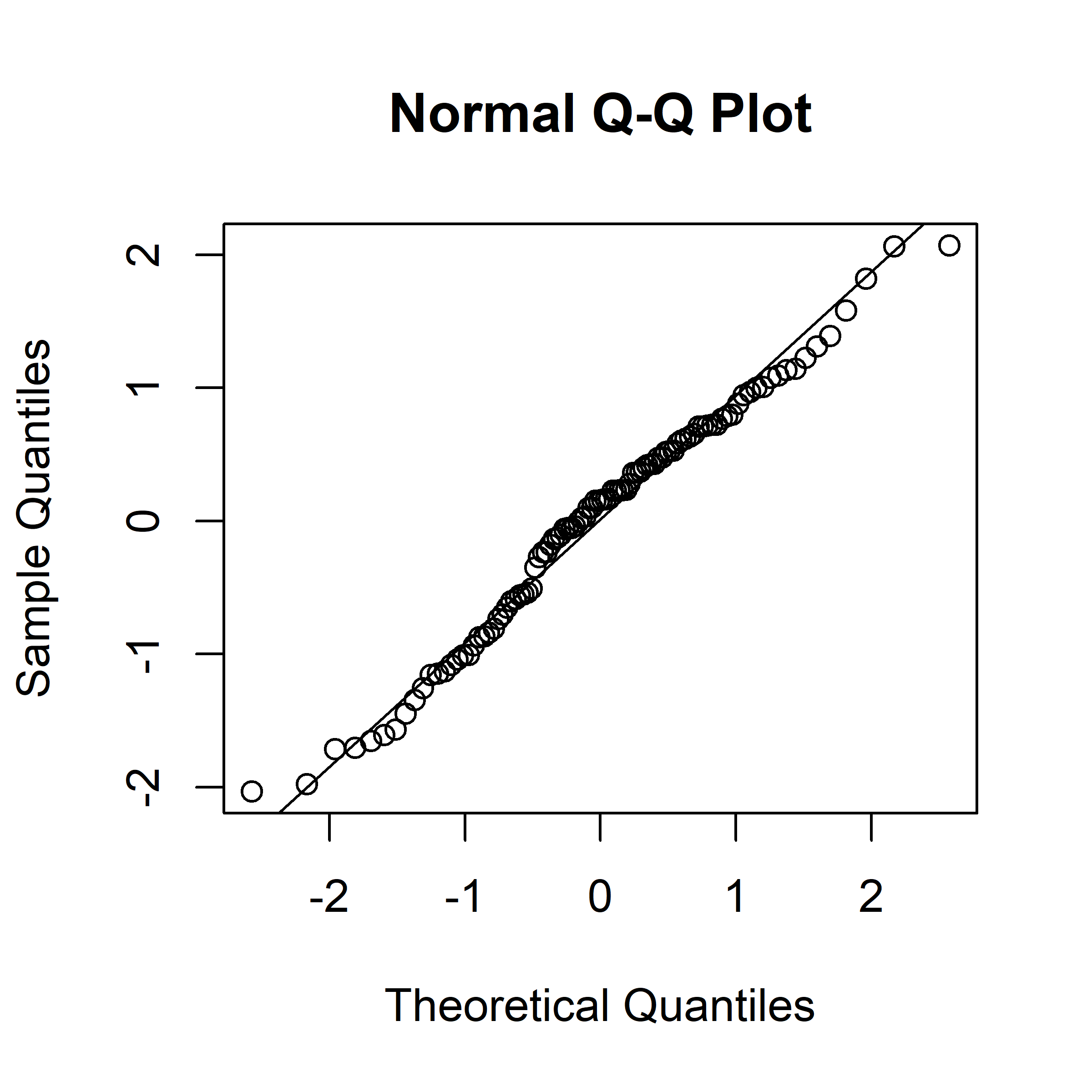
* **رسم نمودار جعبه‏ای یک بردار با استفاده از تابع barplot:** با اجرای دستورات زیر، نمودار جعبه‏ای بردار x رسم خواهد شد.

x = rnorm(n = 100, mean = 0, sd = 1)  
boxplot(x = x)



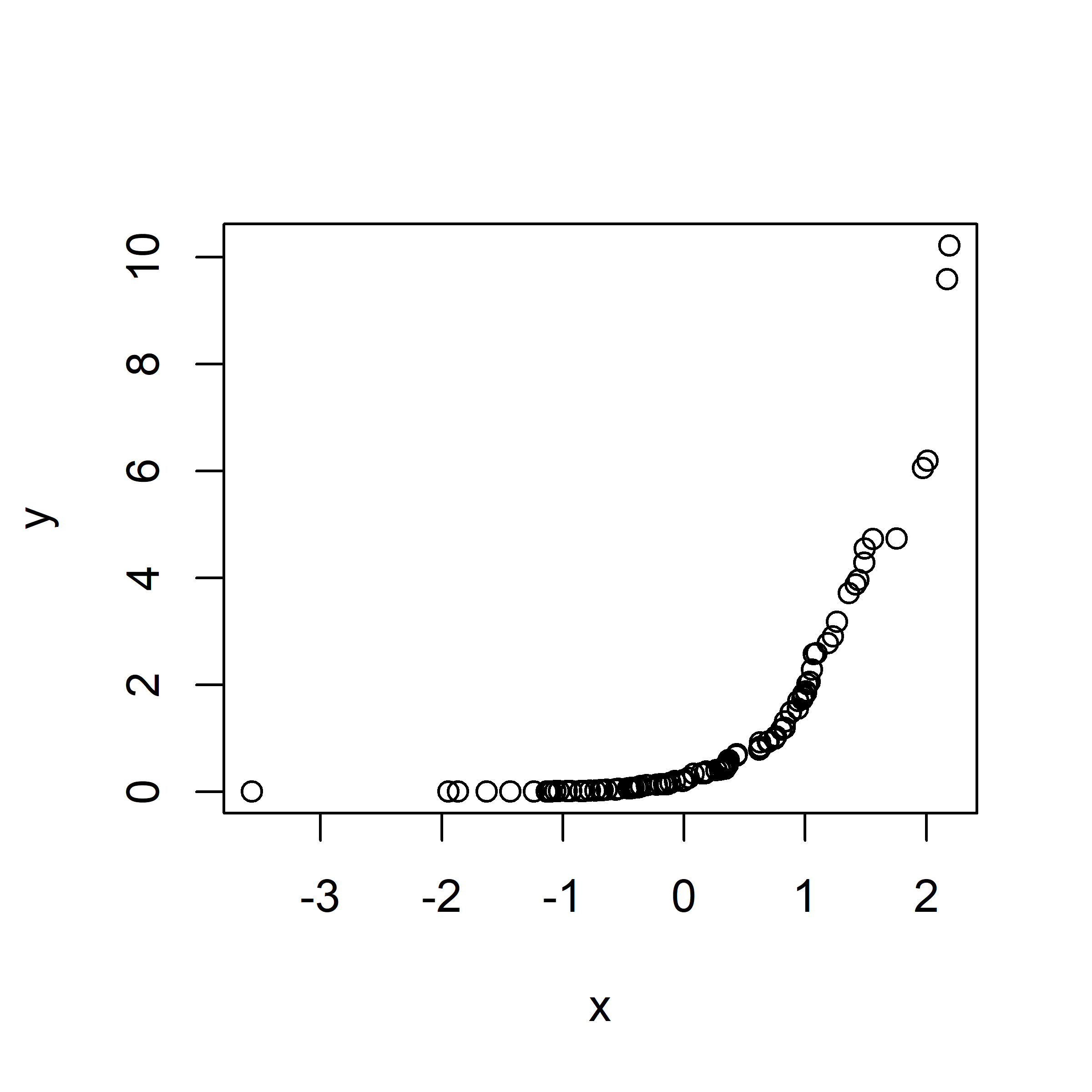
* **رسم نمودار برای بررسی مقایسه توزیع‏ها با استفاده از توابع qqnorm و qqline:** با اجرای دستورات زیر، می‏توان نرمال بودن توزیع داده‏های بردار x را با تابع qqnorm و رسم خط نرمال روی نمودار را با تابع qqline بررسی کرد.

x = rnorm(n = 100, mean = 0, sd = 1)  
qqnorm(y = x)  
qqline(y = x)



* **رسم نمودار برای مقایسه توزیع‏ دو متغیر با استفاده از تابع qqplot:** با اجرای دستورات زیر، می‏توان توزیع مقادیر دو بردار x و y را با هم مقایسه و هم توزیع بودن آن‏ها را بررسی کرد.

x = rnorm(n = 100, mean = 0, sd = 1)  
y = rf(n = 100, df1 = 1, df2 = 20)  
  
qqplot(x = x, y = y)



1. Comprehensive R Archive Network [↑](#footnote-ref-1)
2. Working Directory [↑](#footnote-ref-2)
3. Vectors [↑](#footnote-ref-3)