# 第 08 天

## 函數

對 R 語言的使用者而言，函數會是我們最得力的助手，事實上我們早就開始使用函數了，不管是建立向量時我們使用的 c() 或者是檢查變數類型的 class()，都是函數。使用方法是在小括號中放入所謂的輸入（input），然後呼叫函數就可以得到所謂的輸出（output）。接著我們來探討一些針對數值（numeric）、文字（character）與描述統計（Descriptive Statistics）的常用函數。

### 數值函數

我們一起來練習幾個常用的數值函數，要注意這些都是針對數值處理的函數，如果輸入文字是會出現錯誤訊息的：

#### abs() 函數

abs() 是能夠將我們輸入的數值轉換為絕對值的函數，它的作用跟

相同：

> abs(-5)  
[1] 5  
> abs(-5:-1)  
[1] 5 4 3 2 1  
> abs("Hello")  
Error in abs("Hello") : non-numeric argument to mathematical function

#### sqrt() 函數

sqrt() 是能夠將我們輸入的數值開根號的函數：

> sqrt(2)  
[1] 1.414214  
> sqrt(1:10)  
 [1] 1.000000 1.414214 1.732051 2.000000 2.236068 2.449490 2.645751  
 [8] 2.828427 3.000000 3.162278

#### ceiling() 函數

ceiling() 是能夠將輸入的數值無條件進位的函數：

> ceiling(9.527)  
[1] 10  
> nums <- sqrt(c(7, 17, 27, 37, 47))  
> nums  
[1] 2.645751 4.123106 5.196152 6.082763 6.855655  
> ceiling(nums)  
[1] 3 5 6 7 7

#### floor() 函數

floor() 是能夠將輸入的數值無條件捨去的函數：

> floor(9.527)  
[1] 9  
> nums <- sqrt(c(7, 17, 27, 37, 47))  
> nums  
[1] 2.645751 4.123106 5.196152 6.082763 6.855655  
> floor(nums)  
[1] 2 4 5 6 6

#### round() 函數

round() 是能夠指定保留幾個小數位數的函數，以四捨五入的規則決定，使用的時候可以多給一個參數 digits，預設是 digits = 0，就是取為整數：

> round(9.527)  
[1] 10  
> round(9.527, digits = 0)  
[1] 10  
> round(9.527, digits = 1)  
[1] 9.5  
> round(9.527, digits = 2)  
[1] 9.53

#### exp() 函數

exp() 是能夠將輸入的數值 x 轉換為

的函數（其中

= 2.718282）。

> exp(1)  
[1] 2.718282  
> exp(2)  
[1] 7.389056

#### log() 函數

log() 是能夠將輸入的數值取自然對數（Natural logarithm）的函數，它的作用跟

相同：

> log(exp(1))  
[1] 1  
> log(exp(2))  
[1] 2  
> log(exp(3))  
[1] 3

#### log10() 函數

log10() 是能夠將輸入的數值取 10 為底對數（Logarithm）的函數，它的作用跟

相同：

> log10(10)  
[1] 1  
> log10(10^2)  
[1] 2  
> log10(10^3)  
[1] 3

### 文字函數

接著我們一起來練習幾個常用的文字函數：

#### toupper() 函數

toupper() 是能夠將輸入的文字全部轉換為大寫的函數：

> toupper("Learn R the easy way")  
[1] "LEARN R THE EASY WAY"

#### tolower() 函數

tolower() 是能夠將輸入的文字全部轉換為小寫的函數：

> tolower("Learn R the easy way")  
[1] "learn r the easy way"

#### substr() 函數

substr() 是能夠將輸入的文字擷取部分出來的函數，使用的時候可以搭配兩個參數，start 指定要從哪一個位置開始擷取，stop 指定要擷取到哪一個位置停下來：

> my\_char <- "Learn R the easy way"  
> substr(my\_char, start = 1, stop = 5)  
[1] "Learn"  
> substr(my\_char, start = 7, stop = 7)  
[1] "R"  
> substr(my\_char, start = 9, stop = 11)  
[1] "the"  
> substr(my\_char, start = 13, stop = 16)  
[1] "easy"  
> substr(my\_char, start = 18, stop = 20)  
[1] "way"

#### grep() 函數

grep() 是能夠在文字向量中搜尋某種樣式並將符合樣式的索引值回傳的函數，使用 pattern 參數指定樣式，如果搜尋後沒有符合樣式的，則會回傳 integer(0)：

> my\_char <- c("Learn", "R", "the", "easy", "way")  
> grep(pattern = "Learn", my\_char)  
[1] 1  
> grep(pattern = "R", my\_char)  
[1] 2  
> grep(pattern = "the", my\_char)  
[1] 3  
> grep(pattern = "easy", my\_char)  
[1] 4  
> grep(pattern = "way", my\_char)  
[1] 5  
> grep(pattern = "xyz", my\_char)  
integer(0)

grep() 可以指定參數 ignore.case 為 TRUE，這樣在搜尋樣式的時候會忽略文字的大小寫（預設為 FALSE，不忽略大小寫。）以這個例子來說明的話，原本尋找 "learn" 由於 l 是小寫，所以回傳 integer(0)；但是如果加入參數 ignore.case = TRUE 則可以成功搜尋到。

> my\_char <- c("Learn", "R", "the", "easy", "way")  
> grep(pattern = "learn", my\_char)  
integer(0)  
> grep(pattern = "learn", my\_char, ignore.case = TRUE)  
[1] 1

#### sub() 函數

sub() 除了可以像 grep() 搜尋樣式，搜尋到以後還可以指定要替換為其他的文字，使用 pattern 參數指定樣式，使用 replacement 參數指定要替換而成的文字，以這個例子來說明的話，我們將 "easy" 替換為 "effortless"：

> my\_char <- c("Learn", "R", "the", "easy", "way")  
> sub(pattern = "easy", replacement = "effortless", my\_char)  
[1] "Learn" "R" "the" "effortless" "way"

sub() 同樣可以指定參數 ignore.case 為 TRUE，在搜尋樣式的時候忽略文字的大小寫（預設為 FALSE，不忽略大小寫。）

> my\_char <- c("Learn", "R", "the", "easy", "way")  
> sub(pattern = "EASY", replacement = "effortless", my\_char)  
[1] "Learn" "R" "the" "easy" "way"   
> sub(pattern = "EASY", replacement = "effortless", my\_char, ignore.case = TRUE)  
[1] "Learn" "R" "the" "effortless" "way"

#### strsplit() 函數

strsplit() 是能夠將一個文字切割成向量的函數，使用 split 參數可以指定要根據什麼分割符號來進行文字的切割，值得注意的是，這個函數的輸出是以清單的資料結構儲存：

> my\_char <- "Learn R the easy way"  
> strsplit(my\_char, split = " ") # 使用一個空格作為切割的根據  
[[1]]  
[1] "Learn" "R" "the" "easy" "way"  
> my\_char <- "Learn,R,the,easy,way"  
> strsplit(my\_char, split = ",") # 使用一個逗號作為切割的根據  
[[1]]  
[1] "Learn" "R" "the" "easy" "way"

#### paste() 函數

paste() 是能夠將文字連結起來的函數，使用 sep 參數可以指定連結起來的時候要使用什麼樣的分隔符號，預設是以一個空格來連結：

> char\_1 <- "Learn"  
> char\_2 <- "R"  
> char\_3 <- "the"  
> char\_4 <- "easy"  
> char\_5 <- "way"  
>   
> paste(char\_1, char\_2, char\_3, char\_4, char\_5)  
[1] "Learn R the easy way"  
> paste(char\_1, char\_2, char\_3, char\_4, char\_5, sep = ",")  
[1] "Learn,R,the,easy,way"  
> paste(char\_1, char\_2, char\_3, char\_4, char\_5, sep = "")  
[1] "LearnRtheeasyway"

### 描述統計函數

接著我們一起來練習幾個常用的描述統計函數，這些描述統計函數都有一個共同的參數 na.rm 能夠將輸入中的遺漏值排除，如果輸入帶有遺漏值，必須要指定 na.rm = TRUE 才能夠回傳答案（na.rm 預設為 FALSE）：

#### mean() 函數

mean() 是能夠將輸入數值向量的平均值回傳的函數：

> my\_seq <- 1:5  
> mean(my\_seq)  
[1] 3  
> my\_seq <- c(my\_seq, NA) # 加入一個 NA  
> mean(my\_seq) # 輸出為 NA  
[1] NA  
> mean(my\_seq, na.rm = TRUE) # 排除遺漏值  
[1] 3

#### sd() 函數

sd() 是能夠將輸入數值向量的標準差回傳的函數：

> my\_seq <- 1:5  
> sd(my\_seq)  
[1] 1.581139  
> my\_seq <- c(my\_seq, NA) # 加入一個 NA  
> sd(my\_seq)  
[1] NA  
> sd(my\_seq, na.rm = TRUE) # 排除遺漏值  
[1] 1.581139

#### median() 函數

median() 是能夠將輸入數值向量的中位數回傳的函數：

> my\_seq <- 1:5  
> median(my\_seq)  
[1] 3  
> my\_seq <- c(my\_seq, NA) # 加入一個 NA  
> median(my\_seq)  
[1] NA  
> median(my\_seq, na.rm = TRUE) # 排除遺漏值  
[1] 3

#### range() 函數

range() 是能夠將輸入數值向量的最大值與最小值回傳的函數：

> my\_seq <- 1:5  
> range(my\_seq)  
[1] 1 5  
> my\_seq <- c(my\_seq, NA) # 加入一個 NA  
> range(my\_seq)  
[1] NA NA  
> range(my\_seq, na.rm = TRUE) # 排除遺漏值  
[1] 1 5

#### sum() 函數

sum() 是能夠將輸入數值向量所有數值加總回傳的函數：

> my\_seq <- 1:5  
> sum(my\_seq)  
[1] 15  
> my\_seq <- c(my\_seq, NA) # 加入一個 NA  
> sum(my\_seq)  
[1] NA  
> sum(my\_seq, na.rm = TRUE) # 排除遺漏值  
[1] 15

#### max() 函數

max() 是能夠將輸入數值向量的最大值回傳的函數：

> my\_seq <- 1:5  
> max(my\_seq)  
[1] 5  
> my\_seq <- c(my\_seq, NA) # 加入一個 NA  
> max(my\_seq)  
[1] NA  
> max(my\_seq, na.rm = TRUE) # 排除遺漏值  
[1] 5

#### min() 函數

min() 是能夠將輸入數值向量的最小值回傳的函數：

> my\_seq <- 1:5  
> min(my\_seq)  
[1] 1  
> my\_seq <- c(my\_seq, NA) # 加入一個 NA  
> min(my\_seq)  
[1] NA  
> min(my\_seq, na.rm = TRUE) # 排除遺漏值  
[1] 1

### 小結

好啦！第八天的內容就是這麼多，我們練習了 R 語言中常見的數值函數、文字函數與描述統計函數。

### 練習題

###### 1. R 語言有內建圓周率（），只要輸入 pi 就可以使用它，請練習使用 ceiling()、floor() 與 round() 函數將它轉換為 4、3 與 3.14

pi  
ceiling(\_\_\_)  
floor(\_\_\_)  
round(\_\_\_, digits = \_\_\_)

###### 2. 請在 R 語言的命令列（console）用一段文字輸出今天的系統日期

today\_char <- "Today is:"  
sys\_date <- Sys.Date()  
paste(\_\_\_, \_\_\_)

###### 3. 我們有一個數值向量 num\_vector ，請使用描述性統計幫我們計算它的總和與中位數

num\_vector <- c(11:14, NA)  
sum(\_\_\_, na.rm = \_\_\_)  
median(\_\_\_, na.rm = \_\_\_)

### 延伸資訊

* [Built-in Functions](http://www.statmethods.net/management/functions.html)