成為初級資料分析師 I R 程式設計與資料科學應用

流程控制： `for` 迴圈

郭耀仁

When you've given the same in-person advice 3 times, write a blog post.

David Robinson

大綱

- `for` 迴圈
- `while` 或 `for` 迴圈?

for 迴圈

迴圈是用來解決需要反覆執行、大量手動複製貼上程式碼的任務

`for` 迴圈的 Code Block

- 保留字 `for` 與 `in`
- ITERATOR 迭代子
- ITERABLE 可迭代的物件
- 重複任務

```
1 for (ITERATOR in ITERABLE) {
2   # do something iteratively until ITERATOR hits the end of ITERABLE
3 }
```

ITERABLE 指的是可迭代的物件

- 向量
- 其他資料結構

常見的迴圈任務

- `print()`
- 加總（Summation）
- 計數（Counter）
- 合併（Combine）

將介於 1 至 100 的偶數印出

In [ ]:
```
for (i in seq(2, to = 100, by = 2)) {
    print(i)
}
```

計算 1 到 100 之間的偶數總和：[The Story of Gauss (https://www.nctm.org/Publications/Teaching-Children-Mathematics/Blog/The-Story-of-Gauss/)](https://www.nctm.org/Publications/Teaching-Children-Mathematics/Blog/The-Story-of-Gauss/)

In [ ]:
```
summation <- 0
for (i in seq(2, to = 100, by = 2)) {
    summation <- summation + i
}
summation
```

計算 x 到 y 之間的偶數個數（包含 x 或 y 如果它們為偶數）

In [ ]:

```
x <- 55
y <- 66
even_counter <- 0

for (i in x:y) {
    even_counter <- even_counter + 1
}
even_counter
```

因數分解：x 的因數有哪些

In [ ]:

```
x <- 56
divisors <- c()
for (i in 1:x) {
    if (x %% i == 0) {
        divisors <- c(divisors, i)
    }
}
divisors
```

隨堂練習：判斷質數

在超過 1 的正整數中，除了 1 和該數自身外，無法被其他正整數整除的數字

In [1]:
```
x <- 89
```

In [3]:
```
msg
```

'89 是質數'

In [4]:
```
x <- 56
```

In [6]:
```
msg
```

'56 不是質數'

同樣可以搭配保留字

- `break`
- `next`

In [ ]:

```r
divisors <- c()
for (i in 1:x) {
    print(sprintf("檢查第 %s 次", i))
    if (x %% i == 0) {
        divisors <- c(divisors, i)
    }
}
if (length(divisors) == 2) {
    msg <- sprintf("%s 是質數", x)
} else {
    msg <- sprintf("%s 不是質數", x)
}
msg
```

In [ ]:

```
divisors <- c()
for (i in 1:x) {
    print(sprintf("檢查第 %s 次", i))
    if (x %% i == 0) {
        divisors <- c(divisors, i)
    }
    if (length(divisors) > 2) {
        break
    }
}
if (length(divisors) == 2) {
    msg <- sprintf("%s 是質數", x)
} else {
    msg <- sprintf("%s 不是質數", x)
}
msg
```

樓層的忌諱

In [ ]:

```
for (i in 1:10) {
    if (i == 4) {
        next
    } else {
        print(sprintf("%s 樓", i))
    }
}
```

隨堂練習：判斷介於 x 與 y 之間的質數（包含 x 與 y 如果他們也是質數）

In [7]:
```
x <- 1
y <- 5
primes <- c()
```

In [9]:
```
length(primes)
primes
```

3

2 3 5

In [10]:
```
x <- 5
y <- 19
primes <- c()
```

In [12]:
```
length(primes)
primes
```

6

5 7 11 13 17 19

while 或 for 迴圈?

所有的 `for` 都可以用 `while` 重現，但反之不然

- 確定重複運行次數的情境：可優先採用 `for`（使用 `while` 亦無不可）
- 不確定重複運行次數的情境：僅能採用 `while`

確定重複運行次數：從 random_numbers 中找出奇數

In [ ]:
```
set.seed(87)
random_numbers <- sample(1:1000, size = 100, replace = FALSE)
```

In [ ]:

```r
# for
odds <- c()
for (i in random_numbers) {
    if (i %% 2 == 1) {
        odds <- c(odds, i)
    }
}
odds
```

In [ ]:

```
# while
odds <- c()
i <- 1
while (i <= length(random_numbers)) {
    random_num <- random_numbers[i]
    if (random_num %% 2 == 1) {
        odds <- c(odds, random_num)
    }
    i <- i + 1
}
odds
```

確定重複運行次數：fizz-buzz

- 印出介於 1 到 100 之間的整數
- 如果遇到可以被 3 整除的，印出 `"Fizz"`
- 如果遇到可以被 5 整除的，印出 `"Buzz"`
- 如果遇到可以被 15 整除的，印出 `"Fizz Buzz"`
- 其餘狀況印出整數本身

In [ ]:

```
# for
for (i in 1:100) {
    if (i %% 15 == 0) {
        ans <- "Fizz Buzz"
    } else if (i %% 3 == 0) {
        ans <- "Fizz"
    } else if (i %% 5 == 0) {
        ans <- "Buzz"
    } else {
        ans <- i
    }
    print(ans)
}
```

In [ ]:

```
# while
i <- 1
while (i <= 100) {
    if (i %% 15 == 0) {
        ans <- "Fizz Buzz"
    } else if (i %% 3 == 0) {
        ans <- "Fizz"
    } else if (i %% 5 == 0) {
        ans <- "Buzz"
    } else {
        ans <- i
    }
    print(ans)
    i <- i + 1
}
```

不確定重複運行次數：從介於 1 到 1000 的正整數中隨機抽樣，直到抽出 56 的倍數

In [ ]:

```
sampled <- c()
while (TRUE) {
    sampled_num <- sample(1:1000, size = 1)
    sampled <- c(sampled, sampled_num)
    if (sampled_num %% 56 == 0) {
        break
    }
}
length(sampled)
sampled
```

隨堂練習：投擲一個公正骰子，總共要投幾次才能得到三次 6 點?

In [13]:
```
dice <- 1:6
dice_rolls <- c()
```

In [15]:
```
length(dice_rolls)
dice_rolls
```

17

4 2 1 1 5 2 4 6 4 3 3 1 2 1 3 6 6

隨堂練習：投擲一個公正骰子，總共要投幾次才能「連續」得到三次 6 點?

In [16]:
```
dice <- 1:6
dice_rolls <- c()
```

In [18]:
```
length(dice_rolls)
dice_rolls[(length(dice_rolls) - 2):length(dice_rolls)]
```

782

6 6 6

隨堂練習：製作一個撲克牌向量

In [19]:
```
suits <- c("Spade", "Heart", "Diamond", "Club")
ranks <- c("Ace", 2:10, "Jack", "Queen", "King")
poker_card <- c()
```

In [21]:
```
poker_card
```

'Spade Ace' 'Spade 2' 'Spade 3' 'Spade 4' 'Spade 5' 'Spade 6' 'Spade 7'
'Spade 8' 'Spade 9' 'Spade 10' 'Spade Jack' 'Spade Queen' 'Spade King'
'Heart Ace' 'Heart 2' 'Heart 3' 'Heart 4' 'Heart 5' 'Heart 6' 'Heart 7' 'Heart 8'
'Heart 9' 'Heart 10' 'Heart Jack' 'Heart Queen' 'Heart King' 'Diamond Ace'
'Diamond 2' 'Diamond 3' 'Diamond 4' 'Diamond 5' 'Diamond 6' 'Diamond 7'
'Diamond 8' 'Diamond 9' 'Diamond 10' 'Diamond Jack' 'Diamond Queen'
'Diamond King' 'Club Ace' 'Club 2' 'Club 3' 'Club 4' 'Club 5' 'Club 6' 'Club 7'
'Club 8' 'Club 9' 'Club 10' 'Club Jack' 'Club Queen' 'Club King'