Creating for loops in R

Amir Djalovski

2019-01-08

Generally speaking, building loops in R is condemned and in some case might even be considered as blasphemy!. That been said, knowing and writing loops might be in some cases more efficient or useful than vectorizing. For example, (fortunately/unfortunately) R uses the computer's RAM during calculations, thus in some cases memory issues might cause a function to crush/stop during "simple" vectorizing operation on big objects.

When creating loops we need to supply 4 arguments:

1. iterator - index to iterate over
2. start - where would we start?
3. stop - when would we stop?
4. what to do

Let's try to understand more on how to build a loop from the flowing example:

for (i in 1:5) {  
   
 print(i)  
  
}

In this example, i is our iterator, 1 and 5 are our start and stop, and we print out our iterator. If we'd like to put it into words we can say: "for every i in 1 to 5 print i".

Now, let's run our loop and see what is happing:

i <- 1  
for (i in 1:5) {  
   
 print(i)  
  
}

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

Great! We managed to print our iterator correctly. Now, we can try to make things even more interesting. Using loops we can also iterate over vectors, data.frame, lists, etc.

We can build a vector of fruit names:

myVec <- c("apple", "banana", "orange")

Now, using a new loop, we can print each fruit:

i <- 1  
for (i in 1:length(myVec)) {  
   
 print(myVec[i])  
  
}

## [1] "apple"  
## [1] "banana"  
## [1] "orange"

Here we have a few new things, first, the function length tells us how long is myVec. In our case 3, because we have 3 fruits.

length(myVec)

## [1] 3

In addition, we call the index ith of our vector. Which is same as doing below for each and every i.

myVec[1]

## [1] "apple"

If we'll put our loop into words it will read like: "for every i in 1 to the length of myVec, print the value in each i index".

The same can be applied for every type of indexed object in R. The only thing that we need to note is how the object is indexed. For example, we can also iterate over a data.frame. Example data.frame:

df <- data.frame(name = c("James", "Anne", "James II", "Mary II"),   
 dob = c(1566, 1665, 1633, 1662),   
 gender = c("male", "female", "male", "female"))

We can loop over the names in our data.frame (Please note the use of dim over length as length will give us the number of columns in a data.frame):

i <- 1  
for (i in 1:dim(df)[1]) {  
   
 print(df[i, "name"])  
  
}

## [1] James  
## Levels: Anne James James II Mary II  
## [1] Anne  
## Levels: Anne James James II Mary II  
## [1] James II  
## Levels: Anne James James II Mary II  
## [1] Mary II  
## Levels: Anne James James II Mary II

The date of birth:

i <- 1  
for (i in 1:dim(df)[1]) {  
   
 print(df[i, "dob"])  
  
}

## [1] 1566  
## [1] 1665  
## [1] 1633  
## [1] 1662

Lastly, we can build sentences using paste and print and loop over our data.frame

i <- 1  
for (i in 1:dim(df)[1]) {  
 print(  
 paste("The ruler was", df[i, "name"],   
 "the date of birth was", df[i, "dob"],   
 "the gender was", df[i, "gender"]))  
}

## [1] "The ruler was James the date of birth was 1566 the gender was male"  
## [1] "The ruler was Anne the date of birth was 1665 the gender was female"  
## [1] "The ruler was James II the date of birth was 1633 the gender was male"  
## [1] "The ruler was Mary II the date of birth was 1662 the gender was female"

Thank you for reading!