Sequence of Numbers

Raphael Carvalho

09/05/2019

Sequence of numbers

The simplest way to create a sequence of numbers in R is by using the : operator. Type 1:20 to see how it works.

```
1:20
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
```

That gave us every integer between (and including) 1 and 20. We could also use it to create a sequence of real numbers. For example, try pi:10

```
pi:10
```

```
## [1] 3.141593 4.141593 5.141593 6.141593 7.141593 8.141593 9.141593
```

What happens if we do 15:1? Give it a try to find out.

```
15:1
```

```
##  [1] 15 14 13 12 11 10  9  8  7  6  5  4  3  2  1
```

Often, we'll desire more control over a sequence we're creating than what the : operator gives us. The seq() function serves this purpose. The most basic use of seq() does exactly the same thing as the : operator. Try seq(1, 20) to see this.

```
seq(1, 20)
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
```

This gives us the same output as 1:20. However, let's say that instead we want a vector of numbers ranging from 0 to 10, incremented by 0.5. seq(0, 10, by=0.5) does just that. Try it out.

```
seq(1, 10, by = 0.5)
```

```
##  [1]  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5  7.0  7.5
## [15]  8.0  8.5  9.0  9.5 10.0
```

Or maybe we don't care what the increment is and we just want a sequence of 30 numbers between 5 and 10. seq(5, 10, length=30) does the trick. Give it a shot now and store the result in a new variable called my_seq.

```
seq(5, 10, length = 30)
```

```
##  [1]  5.000000  5.172414  5.344828  5.517241  5.689655  5.862069  6.034483
##  [8]  6.206897  6.379310  6.551724  6.724138  6.896552  7.068966  7.241379
## [15]  7.413793  7.586207  7.758621  7.931034  8.103448  8.275862  8.448276
## [22]  8.620690  8.793103  8.965517  9.137931  9.310345  9.482759  9.655172
## [29]  9.827586 10.000000
```

You're using the same function here, but changing its arguments for different results. Be sure to store the result in a new variable called my_seq, like this: my_seq <- seq(5, 10, length=30).

```
my_seq <- seq(5, 10, length = 30)
```

To confirm that my_seq has length 30, we can use the length() function. Try it now.

```
length(my_seq)
```

```
## [1] 30
```

Let's pretend we don't know the length of my_seq, but we want to generate a sequence of integers from 1 to N, where N represents the length of the my_seq vector. In other words, we want a new vector (1, 2, 3, ...) that is the same length as my_seq.here are several ways we could do this. One possibility is to combine the : operator and the length() function like this: 1:length(my_seq). Give that a try.

```
1:length(my_seq)
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
## [24] 24 25 26 27 28 29 30
```

Another option is to use seq(along.with = my_seq). Give that a try.

```
seq(along.with = my_seq)
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
## [24] 24 25 26 27 28 29 30
```

However, as is the case with many common tasks, R has a separate built-in function for this purpose called seq_along(). Type seq_along(my_seq) to see it in action.

```
seq_along(my_seq)
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
## [24] 24 25 26 27 28 29 30
```

One more function related to creating sequences of numbers is rep(), which stands for 'replicate'. Let's look at a few uses. If we're interested in creating a vector that contains 40 zeros, we can use rep(0, times = 40). Try it out.

```
rep(0, times = 40)
```

```
##  [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [36] 0 0 0 0 0
```

If instead we want our vector to contain 10 repetitions of the vector (0, 1, 2), we can do rep(c(0, 1, 2), times = 10). Go ahead.

```
rep(c(0, 1, 2), times = 10)
```

```
##  [1] 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2
```

Finally, let's say that rather than repeating the vector (0, 1, 2) over and over again, we want our vector to contain 10 zeros, then 10 ones, then 10 twos. We can do this with the `each` argument. Try rep(c(0, 1, 2), each = 10).

```
rep(c(0, 1, 2), each = 10)
```

```
##  [1] 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2
```