R Functions

Andy Hsu

All About Functions in R

Functions are the way we get stuff done in R. We call a function to read data, compute, plot, and do just about anything in R.

R makes writing our own function accessible, but it is important to understand the fundamentals and write a functioning snippet of code before diving into function creation.

Starting With a Snippet

To start, we will grade a class of student assignments. The first exercise will be with a small sample of 3 students.

```r
# Example input vectors to start with
student1 <- c(100, 100, 100, 100, 100, 100, 100, 90)
student2 <- c(100, NA, 90, 90, 90, 90, 97, 80)
student3 <- c(90, NA, NA, NA, NA, NA, NA, NA)
```

To properly calculate the grade, we want to drop the students' lowest scores and calculate the average score afterward. We can do that with the following code.

```r
mean(student1[-which.min(student1)])
```

```
[1] 100
```

To deal with Na values, we can use the argument `trim=` to check for Na values and set them to 0.

```r
noNaTemp <- student2
noNaTemp[is.na(noNaTemp)] <- 0
```

Putting it all together, the final snippet of code is as follows. Note that the filtering of Na values needs to occur before dropping the lowest score.

```
noNaTemp <- student3
noNaTemp[is.na(noNaTemp)] <- 0
mean(noNaTemp[-which.min(noNaTemp)])
```

```
[1] 12.85714
```

Creating a Function

To package this into a function with its proper arguments, we can use `function(x)`, where x is the argument.

```
grade <- function(x) {
  # Set NA values to 0
  x[is.na(x)] <- 0
  # Drop the lowest score and take the mean
  mean(x[-which.min(x)])
}
```

Now, when we call the function with the argument of a scoreset that we want graded, R returns us the proper grade.

```
grade(student1)
```

```
[1] 100
```

```
grade(student2)
```

```
[1] 91
```

```
grade(student3)
```

```
[1] 12.85714
```

Using the `apply()` Function

Now, we want to use this function to grade multiple students at once. We first obtain the data frame of scores.

```r
gradebook <- read.csv("https://tinyurl.com/gradeinput",row.names=1)
gradebook
```

```
           hw1 hw2 hw3 hw4 hw5
student-1  100  73 100  88  79
student-2   85  64  78  89  78
student-3   83  69  77 100  77
student-4   88  NA  73 100  76
student-5   88 100  75  86  79
student-6   89  78 100  89  77
student-7   89 100  74  87 100
student-8   89 100  76  86 100
student-9   86 100  77  88  77
student-10  89  72  79  NA  76
student-11  82  66  78  84 100
student-12 100  70  75  92 100
student-13  89 100  76 100  80
student-14  85 100  77  89  76
student-15  85  65  76  89  NA
student-16  92 100  74  89  77
student-17  88  63 100  86  78
student-18  91  NA 100  87 100
student-19  91  68  75  86  79
student-20  91  68  76  88  76
```

To allow the function to read multiple students at once from a data frame, we use the `apply()` function. The arguments for this function are: `X`, which specifies a dataset, `MARGIN`, which specifies how the function should be applied (i.e. by row, by column), and `FUN`, which specifies which function to be applied.

```r
apply(gradebook,1,grade)
```

```
 student-1  student-2  student-3  student-4  student-5  student-6  student-7 
     91.75      82.50      84.25      84.25      88.25      89.00      94.00 
 student-8  student-9 student-10 student-11 student-12 student-13 student-14 
     93.75      87.75      79.00      86.00      91.75      92.25      87.75 
student-15 student-16 student-17 student-18 student-19 student-20 
     78.75      89.50      88.00      94.50      82.75      82.75 
```

To find the highest scoring student in the class, we can take the `which.max()` of the result.

```r
which.max(apply(gradebook,1,grade))
```

```
student-18 
        18 
```

Looking at averages across each assignment, we can see that the hardest assignment appears to be HW2.

```r
y <- gradebook
y[is.na(y)] <- 0
apply(y,2,mean)
```

```
  hw1   hw2   hw3   hw4   hw5 
89.00 72.80 80.80 85.15 79.25 
```

To find which assignment was best correlated with score, we can call the `cor()` function within an `apply()`.

```r
grades <- apply(gradebook,1,grade)
y <- gradebook
y[is.na(y)] <- 0
apply(y,2,cor,grades)
```

```
      hw1       hw2       hw3       hw4       hw5 
0.4250204 0.1767780 0.3042561 0.3810884 0.6325982 
```

Reading the highest correlation coefficient, we find that HW5 was the most indicative of a student's score.

And those were some of the basics of writing your own functions in R!