Basic Course on **R**: The apply family of functions

David Nieuwenhuijse

May 18th - 24th, 2017

Part D: Using the apply family of functions

1. Use apply() to turn the following code into something shorter:

```
#This function determines if a number is a prime number
isPrime <- function(num){
  if (num == 2) {
    return(TRUE)
  }
  if(num > 1) {
    for(i in 2:(num-1)) {
      if ((num %% i) == 0) {
        return(FALSE)
      }
    }
  } else {
    return(FALSE)
  }

  return(TRUE)
}

#The matrix with numbers to be checked:
mat <- matrix(1:100, nrow=10)
#The matrix with answers (TRUE/FALSE)
answer <- matrix(rep(x = TRUE,100), nrow=10)

for (x in 1:10) {
  for (y in 1:10) {
    answer[x,y] <- isPrime(mat[x,y])
  }
}

#The resulting prime numbers:
mat[answer]
```

```
##  [1]  2  3  5  7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83
## [24] 89 97
```

Answer:

```
#This function determines if a number is a prime number
isPrime <- function(num){
  if (num == 2) {
    return(TRUE)
  }
  if(num > 1) {
    for(i in 2:(num-1)) {
      if ((num %% i) == 0) {
```

```
        return(FALSE)
      }
    }
  } else {
    return(FALSE)
  }

  return(TRUE)
}

#The matrix with numbers to be checked:
mat <- matrix(1:100, nrow=10)

#With apply we do not need to create an answer matrix anymore
answer <- apply(mat, c(1,2), isPrime)

#The resulting prime numbers:
mat[answer]
```

```
##  [1]  2  3  5  7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83
## [24] 89 97
```

2. Answer question 2.3 again using the lapply() function.

```
source("kennedys.txt")

lapply(Kennedys, length)
```

```
## $JosephJr
## [1] 0
##
## $John
## [1] 3
##
## $Rosemary
## [1] 0
##
## $Kathleen
## [1] 0
##
## $Eunice
## [1] 5
##
## $Patricia
## [1] 4
##
## $Robert
## [1] 11
##
## $Jean
## [1] 4
##
## $Edward
## [1] 3
```

3. Answer question 2.3 again using the sapply() function. What is the class of the output?

```
sapply(Kennedys, length)
```

```
## JosephJr     John Rosemary Kathleen   Eunice Patricia   Robert     Jean
##        0        3        0        0        5        4       11        4
##   Edward
##        3
```

```
class(sapply(Kennedys, length))
```

```
## [1] "integer"
```

4. Read in the diamonds.txt dataset using read.table, make sure the headers are correctly loaded. Calculate the average price of diamonds by color and clarity using the tapply function.

```
diamonds <- read.table("diamonds.txt", header = TRUE)

average_price <- tapply(diamonds$Price, list(diamonds$Color, diamonds$Clarity), mean)

print(average_price)
```

```
##         IF      VS1      VS2      VVS1     VVS2
## D 8035.000 3635.857 1947.500 12839.000 7319.500
## E 1813.600 4488.833 5128.571  6553.000 4273.222
## F 3079.875 4038.800 4808.000  3649.389 3553.882
## G 1499.818 3457.083 4799.778  3713.500 4196.095
## H 1993.909 4421.818 5068.714  3991.300 4315.467
## I 2025.375 4451.750 4444.000  5344.600 4884.333
```