Assignment 2: Coding Basics

Clara Fast

OVERVIEW

This exercise accompanies the lessons in Environmental Data Analytics on coding basics.

Directions

1. Change "Student Name" on line 3 (above) with your name.
2. Work through the steps, **creating code and output** that fulfill each instruction.
3. Be sure to **answer the questions** in this assignment document.
4. When you have completed the assignment, **Knit** the text and code into a single PDF file.
5. After Knitting, submit the completed exercise (PDF file) to the dropbox in Sakai. Add your first and last name into the file name (e.g., "FirstLast_A02_CodingBasics.Rmd") prior to submission.

Basics Day 1

1. Generate a sequence of numbers from one to 100, increasing by fours. Assign this sequence a name.
2. Compute the mean and median of this sequence.
3. Ask R to determine whether the mean is greater than the median.
4. Insert comments in your code to describe what you are doing.

```
#1.
#Generate a sequence of numbers from 1 to 100, increasing by 4. Assign this to
#sequenceby4
sequenceby4<-seq(1,100,4)

#2.
#Calculate mean and median of sequence generated previously
mean(sequenceby4)
```

```
## [1] 49
```

```
median(sequenceby4)
```

```
## [1] 49
```

```
#3.
#Ask R whether mean of sequence is larger than the median by using the greater than
#symbol. Will generate either TRUE or FALSE
mean(sequenceby4)>median(sequenceby4)
```

```
## [1] FALSE
```

Basics Day 2

5. Create a series of vectors, each with four components, consisting of (a) names of students, (b) test scores out of a total 100 points, and (c) whether or not they have passed the test (TRUE or FALSE) with a passing grade of 50.
6. Label each vector with a comment on what type of vector it is.
7. Combine each of the vectors into a data frame. Assign the data frame an informative name.
8. Label the columns of your data frame with informative titles.

```
#5. & #6.
names<-c("Ben", "Ariel", "Luna", "Hans") #Vector type = character
testscores<-c(40,80,70,65) #Vector type = numeric
teststatus<-c(FALSE,TRUE,TRUE,TRUE) #Vector type = logical

#7. & #8.
dfstudenttests<-cbind("Student Name"=names, "Test Scores"=testscores, "Pass"=teststatus)
```

9. QUESTION: How is this data frame different from a matrix?

   Answer: A matrix can only have one value type, whereas a data frame can have values of different modes, or nature. That is, whereas a matrix for example could only store integers, a data frame would be able to include integers, characters, and numeric data (and more). In this case, the data frame consists of characters, numeric data, and logical data.

10. Create a function with an if/else statement. Your function should determine whether a test score is a passing grade of 50 or above (TRUE or FALSE). You will need to choose either the `if` and `else` statements or the `ifelse` statement. Hint: Use `print`, not `return`. The name of your function should be informative.
11. Apply your function to the vector with test scores that you created in number 5.

```
#10.
#Using if & else
passinggrade <- function(x) {
  if(x >= 50) {
    x=TRUE
  }
  else {
    x=FALSE
  }
}

#Example input
print(passinggrade(60))
```

```
## [1] TRUE
```

```r
#Using ifelse function
passinggrade2 <- function(x){
  ifelse(x>=50, "Pass", "Fail") #log_exp, if TRUE, if FALSE
}

#Example input
print(passinggrade2(50))
```

```
## [1] "Pass"
```

```r
#11.
#Using ifelse function with original vector of test scores
passinggrade_originalvector <- function(x){
  ifelse(testscores>=50,"Pass","Fail") #log_exp, if TRUE, if FALSE
}
print(passinggrade_originalvector(testscores))
```

```
## [1] "Fail" "Pass" "Pass" "Pass"
```

12. QUESTION: Which option of `if` and `else` vs. `ifelse` worked? Why?

    Answer: 'if' and 'else' worked for question 10 where I had to create a general function with the logical values of TRUE and FALSE. I could not use this same language when using the 'ifelse' function as it already has 'if TRUE' and 'if FALSE' arguments built into the function. I had to therefore input what the ifelse function should generate if the test scores were over or equal to 50, namely if it was TRUE, the student would "Pass" and if the student got below 50, whereby it was FALSE, the student would "Fail".
    I used the ifelse function for question 11 where I had to use the vector I generated earlier, once again inputting "Pass" and "Fail". This worked well with a vector, and was much more efficient to use.