Assignment 2: Coding Basics

Maggie O'Shea

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. Creating the sequence and assigning it the name "fours"
fours<-seq(1, 100, 4)

#2. Finding the mean and median of "fours"
mean(fours)
```

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

```
median(fours)
```

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

```
#3. Creating a function that states that if the median is greater then print "median", and if the mean

greatest <- function(x) {
  if(mean(x) < median(x)) {
    "median"
  }
  else if (mean(x) > median(x)) {
```

```
    "mean"
  }
  else {
    "equal"
  }
}
greatest.mct <- greatest(fours); greatest.mct
```

```
## [1] "equal"
```

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.

```
#Names vector is a character vector
names <- c("xander", "julia", "sally", "frederick")
class(names)
```

```
## [1] "character"
```

```
#Scores vector is numeric
scores<- c(49, 90, 34, 100)
class(scores)
```

```
## [1] "numeric"
```

```
#Passing vector is logical
passing <- c(FALSE, TRUE, FALSE, TRUE)
class(passing)
```

```
## [1] "logical"
```

```
#Combining these vectors into a dataframe. The column labels are the names of the associated vectors.
testresults.df <- as.data.frame(cbind(names, scores, passing))
testresults.df
```

```
##       names scores passing
## 1    xander     49   FALSE
## 2     julia     90    TRUE
## 3     sally     34   FALSE
## 4 frederick    100    TRUE
```

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

Answer: A matrix can only store data that is all of the same type, for example all numbers. In contrast, this dataframe has character, logical, and numeric data all stored toegther.

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.

```
passfail <- function(x) {
  ifelse(x < 50, "fail", "pass")
}
```

11. Apply your function to the vector with test scores that you created in number 5.

```
pf.test<-passfail(scores)
print(pf.test)
```

```
## [1] "fail" "pass" "fail" "pass"
```

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

Answer: `ifelse` worked because ifelse can apply the function to an entire vector and return a vector. When attempting to use just `if` and `else`, only the first element in the vector was used because it could not apply the function to the whole vector.