Basic Course on R: Objects and Functions Practical Answers

Karl Brand* and Elizabeth Ribble†

18-24 May 2017

Contents

1 Objects and Functions **2**
- 1.1 Create the following objects: 2
- 1.2 Create a list called `myFirstList` that has elements `lettervec`, `numbersmat`, and `pets` (from Q1.1 above). 3
- 1.3 Create a data frame called `myDF` from the data generated in Q1.1 with the column names: 3
- 1.4 Use R as a calculator to calculate the following values: 4
- 1.5 Use the operators `%%` and `%/%` to do the following: 4
- 1.6 Do the last calculation from Q1.4 in another way, like this: 4
- 1.7 Do the following to practice saving and opening files in R. 5

2 Document License **6**

*brandk@gmail.com
†emcclel3@msudenver.edu

1 Objects and Functions

1.1 Create the following objects:

(a) A vector of letters called `lettervec`, of class `character`, from 'a' to 'd', using the function `c()`.

Explicitly using `c()`:

```
lettervec <- c("a", "b", "c", "d")
```

(b) A matrix of numbers called `numbersmat`, using the function `matrix()`, that looks like this when printed:

5 9 13
6 10 14
7 11 15
8 12 16

```
numbersmat <- matrix(5:16, nrow = 4)
numbersmat
```

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

(c) A vector (using `c()` then `factor()`) of class `factor` called `pets`, that looks approximately like this when printed:

cat dog dog NA dog cat

We can do this explicitly using `c()` and `factor()`:

```
pets <- c("cat", "dog", NA, "cat")
pets <- factor(pets)
pets
```

```
## [1] cat  dog  <NA> cat
## Levels: cat dog
```

Tricker, but maybe quicker way:

```
pets <- factor(c(1, 2, NA, 1),
               labels = c("cat", "dog"))
pets
```

```
## [1] cat  dog  <NA> cat
## Levels: cat dog
```

1.2 Create a list called `myFirstList` that has elements `lettervec`, `numbersmat`, and `pets` (from Q1.1 above).

```
myFirstList <- list(lettervec, numbersmat, pets)
myFirstList
```

```
## [[1]]
## [1] "a" "b" "c" "d"
##
## [[2]]
##      [,1] [,2] [,3]
## [1,]    5    9   13
## [2,]    6   10   14
## [3,]    7   11   15
## [4,]    8   12   16
##
## [[3]]
## [1] cat  dog  <NA> cat
## Levels: cat dog
```

1.3 Create a data frame called `myDF` from the data generated in Q1.1 with the column names:

(a) `lettervec` and

(b) `pets`.

```
myDF <- data.frame(lettervec, pets)
myDF
```

```
##   lettervec pets
## 1         a  cat
## 2         b  dog
```

```
## 3         c <NA>
## 4         d  cat
```

1.4 Use R as a calculator to calculate the following values:

$$17^4, \quad 45 - 2 \cdot 3, \quad (45 - 2) \cdot 3$$

```
17^4
```

```
## [1] 83521
```

```
45 - 2 * 3
```

```
## [1] 39
```

```
(45 - 2) * 3
```

```
## [1] 129
```

1.5 Use the operators %% and %/% to do the following:

(a) Calculate the remainder after dividing 29,079 into 184,277,809.

```
184277809 %% 29079
```

```
## [1] 4186
```

(b) How many times does 29,079 go into 184,277,809 (i.e. what's the "integer divide" value)?

```
184277809 %/% 29079
```

```
## [1] 6337
```

1.6 Do the last calculation from Q1.4 in another way, like this:

```
a <- 45
b <- 2
c <- 3
d <- (a - b) * c
```

Now check what `a`, `b`, `c`, and `d` are. You can just type the variable name (e.g. `a`) and hit 'Control' then 'Return' or use the command `print(a)`.

```r
a <- 45
b <- 2
c <- 3
d <- (a - b) * c
a
```

```
## [1] 45
```

```r
b
```

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

```r
c
```

```
## [1] 3
```

```r
d
```

```
## [1] 129
```

1.7 Do the following to practice saving and opening files in R.

(a) Look at the variables (or other objects) that are stored in your Workspace by typing either `objects()` or `ls()`.

Answers will vary.

(b) Check your working directory by typing `getwd()`. Now change it to a different directory - preferably your own flash drive - by using the function `setwd()`, for example:

```r
setwd("C:/Users/Elizabeth/My Documents/R Course")
```

Answers will vary.

(c) Use the function "save.image()" to save your R session to a file called `YourLastName_practical1.RData` (replace YourLastName with your last name). Note that this will save a .RData file that contains only those objects you see when you run `ls()`. It does not save any code you typed into the console or into the source pane.

```r
save.image(file = "Ribble_practical1.RData")
```

(d) Use the RStudio "File" drop-down menu to save your R source code to a file called `YourLastName_practical1.R` (replace YourLastName with your last name). Note that this will only save the text you've typed into the source pane. It does not save any objects or anything typed into or ran through the console.

```
savehistory(file = "Ribble_practical1.Rhistory")
```

You may get this error: `'Error in savehistory(file) : no history available to save'` This means no commands have been directly in entered into your *console*, but only from your text editor, which is of course it's own permanent record, assuming you save it!

(e) Use the function "save()" to save only the objects `myFirstList` and `pets` to a file called `YourLastName_objects.RData` (replace YourLastName with your last name).

```
save(myFirstList, pets, file = "Ribble_objects.RData")
```

(f) Now close out RStudio entirely, select "Save" or "Yes" in any dialog boxes that pop up, and then reopen RStudio. Is your source code still there?

It should be; the default in RStudio is to keep source code open!

(g) Run `ls()`; are your objects still there?

The objects available in your previous session are still available and RStudio reloads them for the new session.

(h) You can change these kinds of options by going to "Tools - Global Options". Go there and deselect "Restore .RData into workspace at startup". Then close RStudio and choose to save the .RData file.

(i) Reopen RStudio; your environment should be empty. Load your objects back in using `load()` (e.g. `load("Ribble_practical1.RData")`) and then run ls() again. Do you see your objects now?

They should appear after you've loaded in the file that contained your previously created objects.

(j) Check what the working directory is by again running `getwd()` - has it been reset?

It should have been reset to the default RStudio working directory. This means everytime you open RStudio you should specify your desired working directory!

2 Document License