R Scripting

Exercises for unit 2 - Data structures (II)

Marcus Wurzer

Please solve the following problems!

1. Try to answer quiz questions 4. and 5. at the beginning of the "Data structures" chapter of the coursebook: http://adv-r.had.co.nz/Data-structures.html
2. Try to answer quiz questions 1. to 5. at the beginning of the "Subsetting" chapter of the coursebook: http://adv-r.had.co.nz/Subsetting.html
3. Coursebook exercises:
   a. What are the six types of atomic vector? How does a list differ from an atomic vector?
   b. Test your knowledge of vector coercion rules by predicting the output of the following uses of `c()`:
   ```
   c(1, FALSE)
   c("a", 1)
   c(list(1), "a")
   c(TRUE, 1L)
   ```
   c. Why is the default missing value, `NA`, a logical vector? What's special about logical vectors? (Hint: think about `c(FALSE, NA_character_)`.)
   d. What does `dim()` return when applied to a vector?
   e. How would you describe the following three objects? What makes them different to `1:5`?
   ```
   x1 <- array(1:5, c(1, 1, 5))
   x2 <- array(1:5, c(1, 5, 1))
   x3 <- array(1:5, c(5, 1, 1))
   ```
   f. What does `as.matrix()` do when applied to a data frame with columns of different types?
   g. Fix each of the following common data frame subsetting errors:
   ```
   mtcars[mtcars$cyl = 4, ]
   mtcars[-1:4, ]
   mtcars[mtcars$cyl <= 5]
   mtcars[mtcars$cyl == 4 | 6, ]
   ```
   h. Why does `mtcars[1:20]` return an error? How does it differ from the similar `mtcars[1:20, ]`?
   i. What does `df[is.na(df)] <- 0` do? How does it work?
   j. How would you randomly permute the columns of a data frame? (This is an important technique in random forests.) Can you simultaneously permute the rows and columns in one step?
   k. How could you put the columns in a data frame in alphabetical order?
4. The following table is given:

	name	figure	height	weight	drinks	job
1	Susi	chubby	1.97	98	TRUE	student
2	Tim	chubby	1.67	89	n/a	student
3	Christine	beefy	1.90	71	TRUE	student
4	Mathias	skinny	1.81	86	TRUE	employed

(n/a = not available)

a. Generate the variables

name as character string
job as factor with the following categories: *student, employed, self-employed*
figure as ordered factor (using function `ordered()`) with the following categories: *skinny, lean, slim, normal, chubby, beefy*
height as numeric (continuous) variable
weight as numeric (discrete) variable
drinks as logical (binary) variable

Pay attention to the correct specification of missing values!

b. Generate a data frame `friends` from these variables that looks similar to the table given above and print it!

5. Dataset `chickwts` is included in R:

```
data(chickwts)
?chickwts
```

It is a data frame that contains two variables: The weight of chickens contingent upon various feed supplements (`weight` and `feed`).

a. Print the first 6 rows of the dataset.
b. Extract all factor levels of the variable `feed`.
c. Extract all rows of the data frame for chickens that are fed with **meatmeal** (Hint: `subset()`).
d. Extract the weight of the chickens that are fed with **casein** *or* **horsebean** (Hint: Operator for logical or: `|`).
e. Compute the mean chicken weight separately for each feed supplement (Hint: `aggregate()`).

6. Use dataset `chickwts` again.

a. Compute the mean weight of all chickens (Hint: function `mean()`).
b. Use `weight` to generate a categorical variable `weight_cat` with the three categories `light`, `medium`, and `heavy`. All categories should approximately contain the same number of chickens (Hint: `cut()`).)
c. Subsequently, combine the `light` and `medium` categories to a new category `standard`, and rename the `heavy` category to `premium`.
d. How many "premium" chickens are there that have been fed with **meatmeal**? (Do not count manually, but use R functions!)

7. A digitized black and white image can be represented by a binary matrix, with "0" symbolizing color value "white" and "1" color value "black".

a. Create an 800x600 matrix and randomly fill it with black and white values.
b. Compute the grey value (= mean color value) of the whole matrix.
c. Compute the grey value for a certain image section, namely the lower right quadrant of the picture (Hint: Use horizontal and vertical splitting to get the four equally sized image sections).
d. Invert the picture to get a negative (0 becomes 1 and vice versa). Compute the grey value again.

8. Use the following R-code to generate a matrix of temperature measurements (in °C) for the 31 daily average high temperatures of July in two cities:

```r
set.seed(1)
temps <- matrix(c(round(rnorm(31, 26, 5), 1), round(rnorm(31, 22, 4), 1)),
                nrow = 2, byrow = TRUE)
rownames(temps) <- c("City A", "City B")
colnames(temps) <- paste("July", 1:31)
```

Unfortunately, many of the entries have been lost when reading the data. . . :

```r
temps[sample(1:length(temps), 20)] <- NA
```

a. Compute mean, minimum, and maximum for all entries and separately for both cities.
b. Compute the maximum temperature for the second half of the month (starting with the 16^th of July)
c. Save the observational units in a list having two components (one for each city). The list should only contain the actual measurements and reference dates (i.e., without missings).