G54DMA - Lab 2: Advanced R

Question Sheet

This question sheet has a series of exercises designed to help you deepen your knowledge of R. We will focus on creating functions, and reading and writing from different types of files. You will also practice how to implement conditional and iterative structures, like *if* and *for* loops.

A. Creating Functions

In this section, you will create simple R functions with conditional clauses in some cases. You will then execute them. Remember to choose sensible names (all in lower case) for your functions.

Given *M*, *N* and *O* such as:

$$M=\begin{pmatrix} 1 & 3 & 4 \\ 9 & 6 & 2 \\ 52 & 17 & 1 \end{pmatrix} \quad N=\begin{pmatrix} 1 & 1 & 3 \\ 9 & 8 & 5 \\ 2 & 34 & 9 \end{pmatrix} \quad O=\begin{pmatrix} 1 & 3 \\ 9 & 5 \\ 10 & 2 \end{pmatrix}$$

1. Create a function that given a matrix *A*, returns a matrix *B* that is the double of *A* plus 3.
   a) What is the result of calling your function with *N*?
   b) And with *M*?

2. Create a function that given a matrix *A* and a matrix *B*, returns the matrix with the largest number of rows or a message saying “They have the same number of rows” if they do.
   a) Call your function with *M*, *N*.
   b) Create *P* a matrix obtained by concatenating by rows *M* and *N*. Call your function with *P* and *O*.

3. Write a function that given a matrix *A* of logical values, returns the number of True values and the number of False values on a vector v = (True_values False_values).

4. Create a function that given two matrices *A* and *B*, returns a message saying if those two matrices can be multiplied in the order *A*B* or if they cannot. If the multiplication can be done, also return the result of the operation.
   What is the solution of the following calls to the function?
   a) canmutiply(M,N)
   b) canmultiply(N,O)

5. Create a function that will solve quadratic equations in the form $ax^2 + bx + c = 0$. The function will take three parameters (*a*, *b* and *c)* and return a vector *x* with both solutions to the equation.
   Remember that:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

   What is the solution of the following equations?
   a) $x^2 - 5x - 3 = 0$
   b) $x^2 - 4 = 0$
   c) $2x^2 + 4x - 4 = 0$
   d) $5x^2 + 2x - 1 = 0$
   e) $6x^2 + 11x - 35 = 0$

B. Control Structures

In this section, we will combine the creation of functions with iterative structures. Remember to test your functions with different parameters, especially borderline cases (empty matrices, matrices with different dimensions, 0 cases, etc.). Just because a function works with one set of parameters, it does not mean it is correct!

1. Create a function that given a vector *v*, returns a vector *w* which is the inverse of *v*.
2. Write a function that given a matrix *A,* returns the maximum value in A. **You cannot use the function *max()*.**
3. Create a function that given a matrix *A* and an index *i,* returns the minimum value in the *ith* row of A. **You cannot use the function *min()*.**
4. Write a function that given an N-dimensional point *x* and an *M*x*N*-dimensional matrix *A*, returns the closest point to *x* in *A* using the Euclidean distance.
5. Create a function that given two vectors *v* and *u*, returns one vector containing the values present in both vectors.
6. Create a function that given three matrices, *A, B* and *C*, returns the matrix with the highest determinant. If the highest determinant is shared by two or more of the matrices, the function should return either and print how many matrices had the same determinant.
7. Create a function that given two lists *L* and *P*, returns a list *C* with the concatenation of both lists and prints the number of elements of *C*.
8. Write a function that, given a numerical list *L* of variable length, returns the mean of each of the "components". (**Hint:** Remember the functions *apply()*, *lapply()* and *sapply()*)
9. Write a function that, given a numerical one-dimentional list *L,* returns which values inside that list are prime numbers. (**Hint:** Remember *schoolmath*)
10. Create a function called *generateMenu* that given four lists *Food, Nutrition*, *Calories,* and *Prices,* returns a data frame called *menu* with the information.

    Food: Toast, Pancakes, Eggs Benedict, Chocolate Muffin, Bacon, Coffee, Tea, Croissant, Oatmeal, Banana, Apple, Sausage, Orange Juice

    Nutrition: Vegan, Vegetarian, Vegetarian, Vegetarian, Meat, Vegan, Vegan, Vegetarian, Vegetarian, Vegan, Vegan, Meat, Vegan

    Calories: 150, 500, 250, 300, 100, 1,1, 260, 2290, 50, 10, 125, 70

    Price: 2.5, 5.75, 8, 4.5, 5, 2, 3.25, 3.75, 3, 1.5, 1.5, 2.75, 2.5

Use the *menu* dataframe to answer the following questions:

a. Which food items have lower than 250 calories?
b. Which of the food items that are Vegetarian are cheaper than £6?
c. What is the most expensive Vegan item on the menu?
d. What are the average calories of each of the Nutrition groups?
e. What is the Nutrition group with less food items?
f. Given a dataframe *order* such as:

```
order = data.frame(items = c(….), quantity = c(…))
```

Create a function *overall_order(order)* that, given an *order,* returns its overall price and caloric intake. Call your function with:

- Order 1: 1 orange juice, 1 tea, 1 eggs benedict and bacon
- Order 2: 2 coffees, 2 sausages and 1 croissant.

C. Reading/writing data from files

In this section, we will combine everything you have worked on until now with reading and writing data from files.

1. Modify your *generateMenu* function so it saves your data frame *menu* into a TXT file.
   a. Write a function that, given a *Nutrition* preference, prints in a file which items in the menu comply with it and how much they cost.
   b. Write a function that, given a data frame called *order* which has a variable number of food items and the quantity of these, saves the price of ordering all of the items in the data frame on a file and prints it on the screen.
   c. Write a function that given a maximum number of calories, prints a list of the food items that have those calories or less and their nutrition information on a CSV file.
   d. Write a function that given a variable called *budget* which is the maximum amount of money you can spend, prints the three food items closest in price to *budget.*

2. We have collected the marks obtained in five different modules (*Biology, Maths, Chemistry, Physics* and *Programming*) from six different students:

John: 32, 52, 50, 44, 50
Mary: 88, 67, 59, 70, 70
Mark: 78, 77, 68, 67, 80
June: 89, 90, 81, 89, 87
Claire: 61, 65, 50, 78, 50
Anthony: 67, 68, 65, 40, 66

We also know their gender and their age:

John: M, 18
Mary: F, 19
Mark: M, 19
June: F, 19
Claire: F, 18
Anthony: M, 18

a) Create a matrix with the marks. What is the average mark for each person? And the standard deviation in their grades?
b) What is the minimum mark for each subject?
c) Write a function that creates a single data frame compiling all of this information, including module name, student name and gender. Make sure that all rows and columns are named correctly. This function should also save the data frame in a file called "class_marks.csv".
d) Read this files into a variable called *class2018* and edit it to add a new module: *Statistics*.

John: 89
Mary: 99
Mark: 76
June: 85
Claire: 96
Anthony: 65

Use the data frame *class2018* for the following exercises.

e) Write a function that, given a name, returns the highest score that person has obtained and the name of the module.
f) Write a function that, given a name
    a. If the name is present in the data frame, it returns the average score of that person and the average score of the people with the same age.
    b. If the name is not present, it prints an error message.
g) Write a function that, given a module and a threshold mark, it saves on a TXT the name of people who have obtained less than that mark in the module.
h) Write a function that, given a module, prints the name of the students in descending order according to their marks on the screen.
i) Write a function that, given a module, returns the average score participants according to age.

3. Download the file called "Team.csv" from Moodle. Read it into a data frame.

a) How many Teams are there? What are their names?
b) What is the age and team of the highest paid forward?
c) Of all of the female players who are between the ages of 20 and 25, who is the fastest one? Print all the information related to this payer (name, surname, age, height, …).
d) How many of the selected players are form Dragon Island?
e) Write a function that, given a *team*, a *position* and a *salary*, saves in a TXT the name of all the players of that team that play in that position with a salary less an equal to *salary*.
f) Write a function that given a *team* and a *position*, saves the members of *team* that play in *position* in a TXT file in decreasing order according to Salary.

g) Write a function that, given a Body Max Index *bmi*, saves in a CSV file the name, surname and team of the players whose BMI index is strictly lower than *bmi*.
Remember that the BMI is calculated as:

$$\text{Weight in kg} / (\text{Height in m})^2$$