EXERCISES SESSION #02

BASIC OPERATIONS

1. Create a vector x containing the elements 0,1,2,3,4,5,6. HINT: c()
2. Write an if-else statement to check the first element of x. If it is zero, output “This is good”, else output “This is bad”. HINT: print() and logical operator ==
3. Write a for-loop that iterates over each element of vector x using print()
4. Write a for-loop that iterates over each element of vector x and print the cube of each value. HINT: 2^3 is to produce the cube of 2, 8
5. Write a for-loop that makes a right angle triangle pattern where each row is a number x repeating itself x times. HINT: rep()

The pattern is:

1

22

333

4444

1. Write a for-loop that iterates over the inbuilt iris dataset that evaluates the number of characters in the column name. The output should be like: Sepal.Length (12). HINT: paste0() and nchar()
2. Write a for-loop that iterates over the inbuilt iris dataset ad outputs for each flower/row as output: Flower 1 is a setosa and has a Sepal.Length of 5.1
3. Write a for-loop that iterates over each element of vector y and only outputs the positive numbers, i.e. ignoring the negative numbers. HINT: next()
4. Write a for-loop that iterates over each element of vector y, till it reaches value 10. If it reaches value 10, stop the for-loop. HINT: break()
5. Use the inbuilt rivers dataset and write a for-loop that checks each value. If the value is lower than 500, output “short river”. If the river is more than 2000, output “long river”. Everything else, output the original value.

READR & DPLYR

1. Import readr and dplyr packages
2. Import dataset tidy.csv and store it in an object called “mydata”. HINT: read\_csv()
3. Check out mydata, how many rows and colums are there?
4. Genotype, treatment and gender should be categorical variables with some order (instead of the default alphabetical order). Transform these into order “WT”,”GM” for genotype, “CTR\_X”,”TRT\_X”,”TRT\_Y”,”TRT\_Z” for treatment and “M”,”F” for gender. HINT: mutate() and factor(…, levels =c(…)).

Example: factor(class, levels = c(“level\_1”,”level\_2”,”level\_3”))

1. Variables are unnormalized. Create three new columns with the variables multiplied by the norm column. Name the new columns var\_x\_norm, var\_y\_norm and var\_z\_norm. Store in an object called “mydata\_t”. HINT: mutate()
2. Continue from mydata\_t. Filter values that only contain male (gender == “M”) samples, store this in an object called “mymales”. Continue working with mymales untill question 20. HINT: filter()
3. The summarise() function is very useful for collapsing a large dataset into a single observation. Summarise the mymales dataset to find the following columns / values: min\_value\_x\_norm (minimum value for var\_x\_norm) and mean\_value\_x\_norm (mean value for value\_x\_norm). HINT: min() and mean()
4. In combination with group\_by() function, you can summarise the value of interest by each group. For example, grouping according to genotype, you can find the values of question 17 for each genotype. HINT: %>% makes the line of execution more readable.

Example:

data %>% group\_by(x) %>% summarise(…)

is equivalent to (but more readable than)

summarise(group\_by(data, x), …)

1. Grouping can be performed on multiple levels. Try to find values of question 17 when grouping for genotype and then treatment. Summaries can also be stored in an object. Store this in object called “mysummary”
2. Export your result with write\_csv(). Call it “myfirstRoutput.csv”

GGPLOT2

1. Work with mydata\_t from the previous segment. Create a boxplot with on the y-axis var\_x\_norm and on the x-axis genotype. How does your plot look like? What can you conclude from this? HINT: ggplot(data, aes(x = x, y = y) + geom\_boxplot()
2. The previous graph takes a general view of var\_x\_norm according to genotype. Add within aesthetics (aes) that treatment is defined by fill. HINT: fill = …
3. Re-use the code from question 22 and adjust it slightly so you have violinplots instead of boxplots.
4. Re-use the code from question 22 and adjust it slightly so you have stacked bars instead of boxplots. HINT: geom\_col()
5. Re-use the code from question 24 and adjust it slightly to have side-by-side bars instead of stacked bars. HINT: position = “dodge” inside geom\_col()
6. Re-use the code from question 24 and adjust it slightly to have stacked bars of equal height, instead of stacked bars with different height. HINT: position = “fill”
7. Create a scatter plot with on the y-axis var\_x\_norm and on the x-axis genotype. How does your plot look like? HINT: geom\_point()
8. All points are only segmented according to genotype, but no difference is made between treatments. Add within aesthetics (aes) that treatment is defined by color. HINT: color = …
9. Treatments are defined with different colors but are all on top of each other. To make more clear. Add inside geom\_point() that position has to be changed. This can be done with the statement *position = position\_dodge(1)*. The number (0-1) determines how much space there is between the respective columns within genotype. Try out different values to see how it makes the graph more readable.
10. Besides colors and fills, shapes can also be used as an aesthetic to define categorical groups. Add shape = gender within aesthetics to see what happens. What can you see? How can the visibility be improved? See facet\_wrap() or facet\_grid() and try out.