Bash Exercises

These exercises should be completed in the terminal.

Exercise 0: Preparing for scripting

Using the terminal, make a new directory inside day1 called “scripts”.

Exercise 1: Your first bash script

Using nano, write an executable shell script (called *myfirst.sh*) that prints the statement “I am alive”.

Tip 1: Remember the shebang line.

Tip 2: Bash uses echo to print lines.

Exercise 2: Variables

Define four variables named: **NAME1**, **NAME2**, **NAME3**, and **EVENT**.

Using **echo**, print the sentences below in the *myfirst.sh* script. The variables are not written correctly and are missing the $ character.

"NAME1, NAME2, and NAME3 are on a trip, but NAME1 and NAME2 are fighting over EVENT. NAME3 stays quiet because they caused EVENT and don't want them to know."

Exercise 3: Positional arguments

Create a script called *icecream.sh*. Use positional arguments and echo to print the following sentence:
“Why do people like X when Y is superior?”

Here, X and Y represent the first and second positional arguments passed to the script.

Exercise 4: Count sequences with variables

Create a script called *count_fasta.sh* that counts the number of sequences in a given FASTA file. Choose a FASTA file from P_aeruginosa/assemblies and define the chosen FASTA file as a variable.

Tip: Use grep, ‘>’ and pipe (‘|’)

Exercise 5: Count sequences with positional arguments

Update *count_fasta.sh* to use positional arguments instead of using variable names.
Run *count_fasta.sh* with the necessary arguments.

Exercise 6: If statements

Create a script called *greater_than.sh* that uses an if-statement and positional arguments to check if the input number is greater than 60.

Hint: Use echo for the command that triggers if TRUE.

Exercise 7: Else statements

Modify *greater_than.sh* by adding an elif-statement to check if the input value is exactly 60. Include an else-statement to handle cases where neither condition is met.

Extra exercises

Exercise 8:

Create a script called *check_fasta.sh*.
Make a for loop that iterates over all FASTA files from P_aeruginosa/assemblies and print the file names.

Exercise 9:

Create a script called *array.sh*.
Make an array containing toppings that should be on a pizza. Print all elements from the array using a for-loop.

Exercise 10:

Modify the check_fasta.sh script to incorporate the word count function from count_fasta.sh. The script should loop through all FASTA files in the P_aeruginosa/assemblies directory and, for each file, print the number of sequences along with its name.