Command-line exercises

All exercises today will be performed in the terminal. Detailed descriptions of every command can be found in the Handout. We have provided the commands for the first exercise to help you get started. As the course progresses, you will need to refer to the handout to find the appropriate commands and adapt them to your specific tasks.

Exercise 0: Setting the stage

The first step of any analysis performed in the terminal is to set up your workspace. Please follow the instructions below:

- Open the terminal and make sure you are in "home" by typing the following command and pressing enter (cd without argument will always take you to /home):

  ```
  cd
  ```

- Get a sense of which files and subdirectories are already inside your current working directory by listing its contents with the following command in the terminal:

  ```
  ls
  ```

- From this directory, we need to move into the working environment for today.
  Tip: You can use the "Tab" key for autocomplete—if you type the first part of a folder or file name, it will complete the rest, as demonstrated in the talk.

  ```
  cd BTG/intro/day1
  ```

- List the content of your working directory.

  ```
  ls
  ```

- Here you see a compressed file called BacterialData. Decompress the file (extracting the contents) by using the command below:

  ```
  tar -zxvf BacterialData.tar.gz
  ```

- List the content again and **notice the difference in color**. The file has changed from a zip file to a directory you can enter. To enter this directory through the terminal, run the following command:

  ```
  cd BacterialData
  ```

- The stage is now set, let's get on with the exercises.

Exercise 1: Exploration of the data

You are presented with a disorganized directory for a bacteria project and tasked with organizing it to enable proper analysis. However, let's first investigate the contents of the directory. In these exercises, we will explore the BacterialData directory to understand how the data is stored.

- Exercise 1A: How many subdirectories are present within the directory called "E_coli"?
- Exercise 1B: In which subdirectory can the file "E_coli_CP127297.fasta" be found?
- Exercise 1C: How many files are in the "Virus" subdirectory?
- Exercise 1D: What is the full path for the file named "R1_E_coli_02.fq"?

Exercise 2: Cleaning it up!

Now, it's time to organize the messy directory! In this process, we will learn how to create new directories, move files, and rename files.

- Exercise 2A: Move the assembly file "E_coli_WWcol315.fasta" into the "assemblies" sub-directory within the "E_coli" directory.
- Exercise 2B: Move the read files "R1_E_coli_01.fq" and "R2_E_coli_01.fq" into the "reads" sub-directory within the "E_coli" directory.

We will now create a directory named "P_aeruginosa" to organize the reads and assemblies from the messy directory. Then, we will place all the assemblies in a subdirectory called "assemblies" and all the reads in a subdirectory called "reads".

- Exercise 2C: In the "BacterialData" directory, create a subdirectory named "P_aeruginosa".
- Exercise 2D: In the "P_aeruginosa" directory, make a subdirectory called "reads".
- Exercise 2E: In the "P_aeruginosa" directory, make another subdirectory called "assemblies".

All files containing reads start with the letter "R" and have a ".fq" extension. All files containing assemblies have a ".fasta" extension.

- Exercise 2F: Move all read files into P_aeruginosa/reads using only a single command.

You may notice that one of the assemblies has a misspelling that needs to be corrected.

- Exercise 2G: Rename "P_oeruginosa_PPF1.fasta" to "P_aeruginosa_PPF1.fasta"
- Exercise 2H: Move all assembly files into P_aeruginosa/assemblies. Try to do it with a single command.

Exercise 3: Removing redundant content

In addition to moving files, you should delete any files not belonging to this bacteria project. Let's get rid of the noise!

- Exercise 3A: Remove the "Credit_cards.txt" file from "BacteriaData" (nothing to see here, we promise, delete!).
- Exercise 3B: Remove the "Virus" directory and everything in it. As the folder insinuates, we are only working with bacteria.

Exercise 4: Inspection

It is time to inspect the files in the "P_aeruginosa/assemblies" directory and ensure that everything is in order before we begin the analysis.

- Exercise 4A: Inspect the assembly file "P_aeruginosa_TOprJ3-positive_part1.fasta" using the **cat** command.
- Exercise 4B: Inspect the assembly file "P_aeruginosa_TOprJ3-positive_part2.fasta" using the **less** command. Observe the difference between **cat** and **less**. Note: press "q" to exit **less**.
- Exercise 4C: When would you use cat and when would you use less? Discuss with your partner

Opening and navigating through an entire file is sometimes unnecessary when you are only interested in the beginning or end of a text file, particularly if the file is immense.

- Exercise 4D: Get the first three lines of "P_aeruginosa_TOprJ3-positive_part2.fasta" using the **head** command.
- Exercise 4E: Get the last five lines of "P_aeruginosa_TOprJ3-positive_part2.fasta" using the **tail** command.

Exercise 5: Combining files

For some odd reason, the reference is divided into three separate files. We need to combine these back into a single file.

- Exercise 5A: Using **cat,** concatenate the two files called "P_aeruginosa_TOprJ3-positive_**part1**.fasta" and "P_aeruginosa_TOprJ3-positive_**part2**.fasta" into a new file called "P_aeruginosa_TOprJ3-positive.fasta" in the P_aeruginosa/assembly folder.
- Exercise 5B: Using **cat**, append the file "P_aeruginosa_TOprJ3-positive_**part3**.fasta" to the newly created "P_aeruginosa_TOprJ3-positive.fasta".

NOTE: You could have combined the files in one step using the solution from 5a, but for learning purposes, this was a great way to introduce the difference between writing to a file and appending to a file. Understanding this distinction is important to avoid accidentally overwriting important data files. Making mistakes is free in this course, but it might not be free in real life.

Exercise 6: Searching within files

There are many powerful tools in Unix! You have already encountered several (i.e., cat, less), and now it's time to learn about grep. A handy tool for searching within files, grep is your friend when exploring files.

For this exercise, you will work with the file "P_aeruginosa_TOprJ3-positive.fasta" that you created in Exercise 5.

- Exercise 2A: Use **grep** to extract any line containing "terA".
- Exercise 2B: Count the total number of sequences; keep in mind that FASTA headers begin with the ">" symbol. Hint: remember proper quotations around ">".

Exercise 7: Word count and piping results between programs

Sometimes, you need to count. It can be counting lines, how often a word appears or even how many files you have. And while you can do it yourself, you can also let the computer do it for you, so that you can have a coffee break. Let's familiarize you with the ***wc*** *(word count) command, so you can take that break.*

- Exercise 3A: Begin by counting the number of lines in "P_aeruginosa_TOprJ3-positive.fasta".
- Exercise 3B: Determine how many words are in the "README.md" file.

It's time to count the number of files in your directory using the ***wc*** *command! This might seem unnecessary if there are only a few files, but imagine you've just used a new command to copy a large number of files from one directory to another, and you want to make sure they're all there without manually counting them.*

- Exercise 3C: Combine the **ls** and **wc** commands using a pipe to count the files in the P_aeruginosa/assemblies directory.

Extra exercises

Extra Exercises 1: More fun with grep!

When using grep, we have encouraged you to search using quotation marks " ". While this is more of a best practice than a strict requirement, it helps avoid potential issues. In this exercise, we will explore a scenario that can occur if it is not used. You will be working with the file "P_aeruginosa_TOprJ3-positive.fasta".

- Extra 1A: Use **grep** to save the header names in a file called "header_names.txt". View the file using **less**.
- Extra 1B: Let's discover what happens when you don't use proper quotations with **grep**. Display the content of header_names.txt using less. Run the following command:

  ```
  grep > header_names.txt
  ```

  Then display the content of header_names.txt again.
  What happened and why? Discuss with your partner.

Extra exercise 2: Options to ls!

*You can customize the ls command by adding options. When dealing with many files, the default **ls** output can be cluttered. Try using the **-l** option for a long listing format and **-lh** to make file sizes human-readable. Observe the difference in output between these options and the default **ls**.*

- Extra 2A: What is the largest file in P_aeruginosa/assemblies?

Extra Exercises 3: Save space – use symbolic links

You need to run ResFinder on the assemblies. To do so, you must copy the assembly files into a new directory called "Resfinder".

- Extra 3A: Make a new subdirectory called "Resfinder" inside "BacterialData".
- Extra 3B: Copy the "E_coli_ASM584v2_reference.fasta" assembly file from E_coli/assemblies and place the copy in the "Resfinder" directory.

Some programs require running on an entire directory, necessitating specialized directories for such analysis. Having multiple copies of the same files is inefficient and costly. Instead, use symbolic links to these files, which are space-efficient and avoid data duplication. More details can be found in the handout!

- Extra 3C: Create a symbolic link to "E_coli_ASM584v2_reference.fasta" from the E_coli/assemblies directory in the "Resfinder" directory.
- Extra 3D: Create symbolic links to all assembly files in the "E_coli" and "P_aeruginosa" directories within the "Resfinder" directory.
- Extra 3E: List the contents of the "Resfinder" directory and observe the visual differences between symbolic links and actual file copies.

Extra Exercises 4: Files for a Colleague – Compressing files

A colleague also needs to run ResFinder and would like a copy of your files. Let's be helpful and provide her with that.

- Extra 4A: Compress the "Resfinder" directory using the **tar** command and name the resulting gzipped file "ResFinder.tar.gz". Using the tar command, ensure that tar zips the real files through the symbolic links, not just the symbolic links themselves. Make sure you're not inside the "Resfinder" directory when you do this.
- Extra 4B: Retrieve the path for the "ResFinder.tar.gz" file so you can share it with your colleague.