Introduction to the Command Line, R, and Jupyter

National Aeronautics and Space Administration

GL4U: Introduction 2024
Lecture 3 of 4

Amanda M. Saravia-Butler, Ph.D.

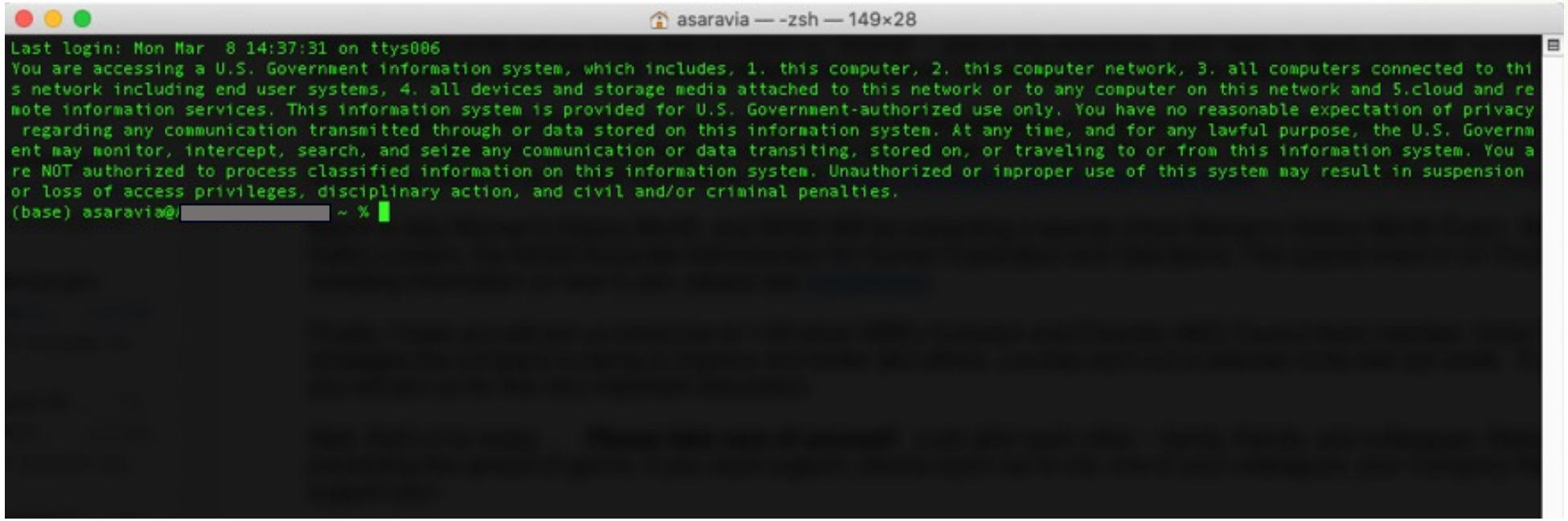
NASA GeneLab Science Lead

Contractor: KBR

Biological & **P**hysical **S**ciences

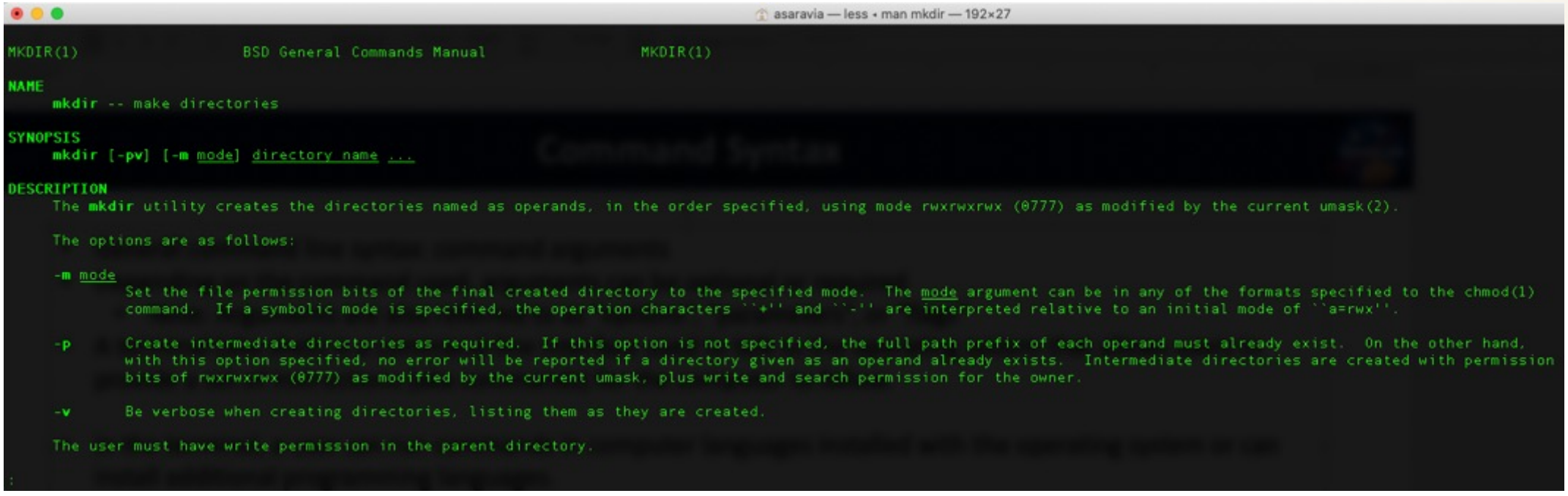
What is a Terminal?

- A terminal window is a command-line interface used to interact with your computer (without graphics)
- The terminal window displays a blank text window, or “command line”, where you can type commands to communicate with your computer
- A terminal uses a specific language interpreter, or “shell”, to interpret commands typed into the command line
- Unix is a computer operating system written in the C programming language
  - By default, terminals on Linux and Mac computers run a Unix shell
  - Unix shells come in a variety of flavors including zsh, csh, tcsh, and bash, each having slightly different features and functionality
  - We will be using a Unix shell simulator, bash, in the GL4U modules



Command Syntax

- General command line syntax: command arguments
- Depending on the command used, arguments can be optional or required
  - Note: Arguments are also referred to as “options”, “parameters”, or “flags”
- All available arguments for many commands can be found by looking at the command’s manual or help page, which can be opened in the terminal.
  - To view a command’s manual, we use the `man` command in Unix followed by the command of interest
    - Example: man <command_of_interest>
  - To view a command’s help page, we use the `--help` or `-h` flag for the command of interest
    - Example: <command_of_interest> --help
- Below is the manual of a common Unix command, mkdir, which was displayed in a terminal by running the command: man mkdir



Running Unix Commands

- The SYNOPSIS section of the mkdir manual shows how the command should be run:

```
SYNOPSIS
     mkdir [-pv] [-m mode] directory name ...
```

- For the mkdir command, `mkdir` is written first, followed by the flags indicated in the brackets, and ending with the directory name.
  - Note: Anything in brackets [] is optional and any part of the command not in brackets is required.
  - For the mkdir command, which parts are optional and which are required?
- Do you notice how the different parts of the command are separated? **With spaces!**
- On the command line, spaces are special
  - Spaces are used to know how to properly separate distinct parts of a command
  - For this reason, it's not ideal for file or directory (aka folder) names to contain spaces. Thus, it's better to use dashes (-) or underscores (_) instead of spaces when naming files and directories.
    - Example: "example_test1.txt" is preferred over "example test1.txt"

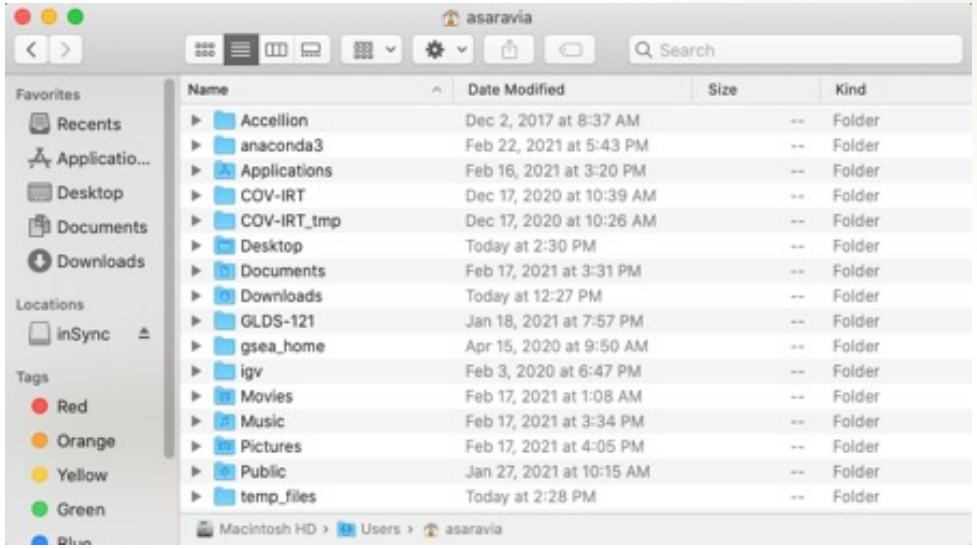
Running Unix Commands

- Before we run the mkdir command to make a new directory, we first need to decide where to make the new directory.
- On computers, files are organized into directories (aka folders).
- When you open a terminal, it opens to a particular directory on your system, referred to as your “working (aka current) directory”.
- You can see what your working directory is by executing the `pwd` command, which stands for “print working directory”
- In the example below, you’ll see the output of the `pwd` command:

```
(base) asaravia@ ~ % pwd
/Users/asaravia
```

- What is the current (aka working) directory?
- You can list what’s in your working directory with the `ls` command as shown on the left below (note: `ls` stands for “list”) :

```
(base) asaravia@ ~ % ls
Accellion       Documents       Music           igv
Applications    Downloads       Pictures        temp_files
COV-IRT         GLDS-121        Public
COV-IRT_tmp     Library         anaconda3
Desktop         Movies          gsea_home
```

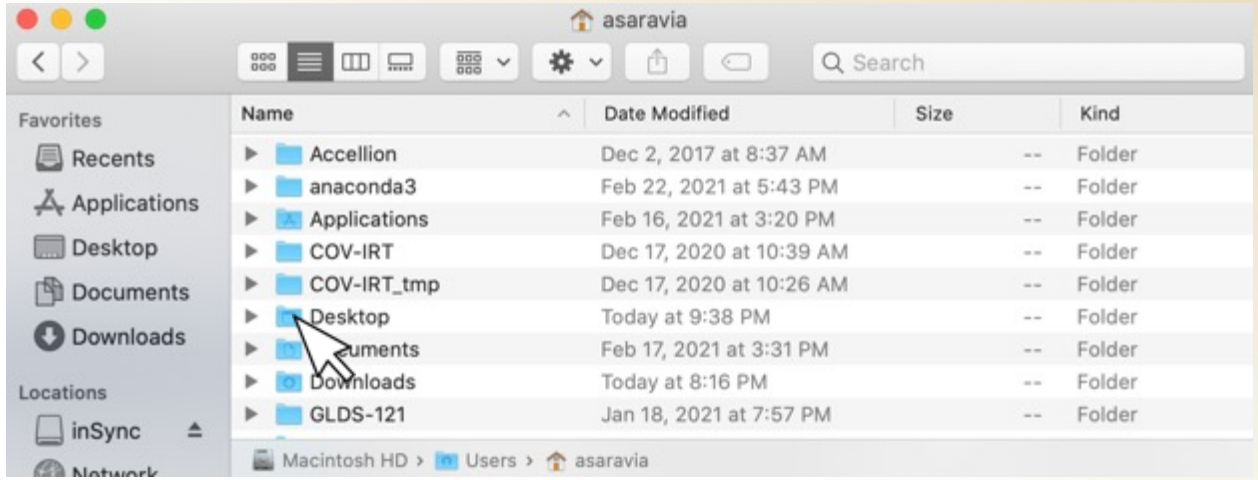


- We can also navigate to the /Users/asaravia directory using the Finder window as shown on the right above.
  - Do you notice any similarities between the files shown in /Users/asaravia on the command line versus the Finder window?

Running Unix Commands

- Similar to how you can navigate to a different directory (aka folder) within your current directory by double clicking on the folder icon in the Finder window, you can use the `cd` command to change directories in your terminal (note: `cd` stands for “change directory”).
- Below, we will navigate to the Desktop using the command line (left) and in Finder (right):

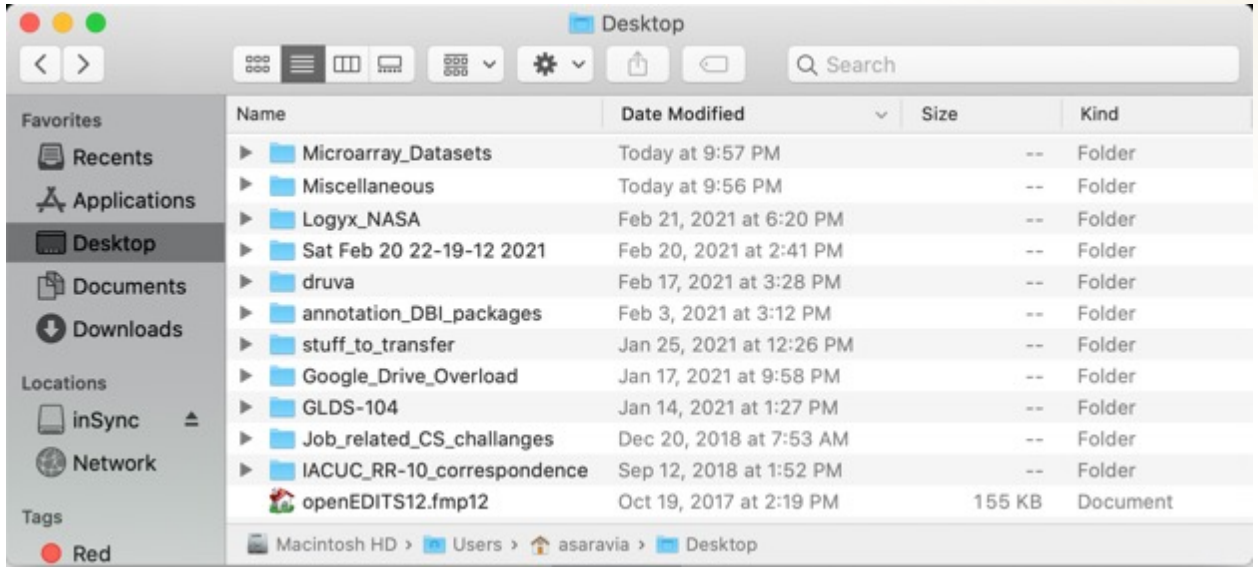
```
(base) asaravia@[redacted] ~ % cd Desktop
(base) asaravia@[redacted] Desktop % pwd
/Users/asaravia/Desktop
```



- What is the working directory now?
- How are directories distinguished on the command line?

- Let’s see what’s on the Desktop using the `ls` command in the terminal below:

```
(base) asaravia@[redacted] Desktop % ls
GLDS-104                    Miscellaneous
Google_Drive_Overload       Sat Feb 20 22-19-12 2021
IACUC_RR-10_correspondence  annotation_DBI_packages
Job_related_CS_challanges   druva
Logyx_NASA                  openEDITS12.fmp12
Microarray_Datasets         stuff_to_transfer
```



- Compare the content on the Desktop in the terminal with what is shown in Finder

Running Unix Commands

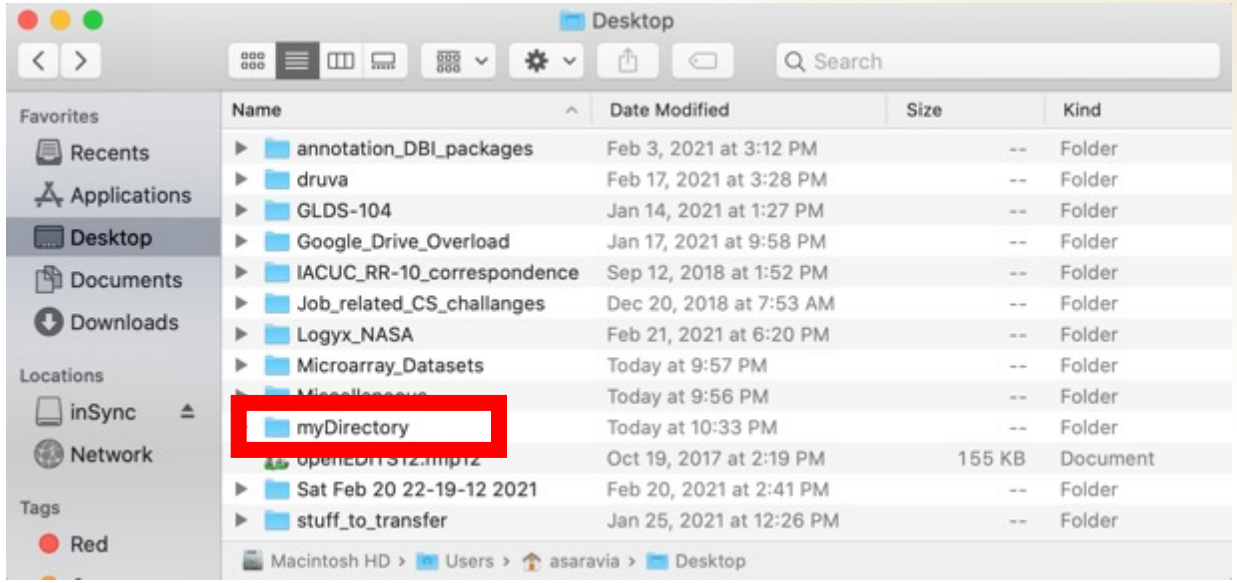
- Now that we've set the Desktop as our working directory, let's use the `mkdir` command to create a new folder on the Desktop.
- Recall the SYNOPSIS section of the mkdir manual, which shows how the command should be run:

```
SYNOPSIS
     mkdir [-pv] [-m mode] directory name ...
```

- In the example below, we'll make a directory titled "myDirectory" on the Desktop using the command line then view what's on the Desktop using both the command line and Finder:

```
(base) asaravia@[redacted] Desktop % mkdir myDirectory
```

```
(base) asaravia@[redacted] Desktop % ls
GLDS-104                        Sat Feb 20 22-19-12 2021
Google_Drive_Overload           annotation_DBI_packages
IACUC_RR-10_correspondence      druva
Job_related_CS_challanges       myDirectory
Logyx_NASA                      openEDI1512.fmp12
Microarray_Datasets             stuff_to_transfer
Miscellaneous
```



Note: The `mkdir` command is the equivalent of creating a new folder in Finder.

Creating Variables

- There will be times when we need to set our working directory to a folder that is deep within a directory structure. Rather than navigating one directory at a time, like you would in Finder, you can navigate directly to your folder of interest with a single `cd` command in the terminal by writing the explicit “path” to that directory as shown:

```
(base) asaravia@[redacted] ~ % pwd
/Users/asaravia
(base) asaravia@[redacted] ~ % cd /Users/asaravia/Desktop/myDirectory
(base) asaravia@[redacted] myDirectory % pwd
/Users/asaravia/Desktop/myDirectory
```

- The “path” is like the address of where a folder or file is located in the computer.
- If there is a location that we need to visit often, rather than writing out the explicit path every time, we can create a variable containing the explicit path. In Unix, you can do this using an equal sign to assign the path to a variable name as shown:

```
(base) asaravia@[redacted] ~ % pwd
/Users/asaravia
(base) asaravia@[redacted] ~ % myDir=/Users/asaravia/Desktop/myDirectory
(base) asaravia@[redacted] ~ % echo $myDir
/Users/asaravia/Desktop/myDirectory
(base) asaravia@[redacted] ~ % cd $myDir
(base) asaravia@[redacted] myDirectory % pwd
/Users/asaravia/Desktop/myDirectory
```

- What command was introduced in the example above? **echo**
- What was added to the variable name when using the variable in Unix? **$**

Creating Variable Names

- When creating variable names there are a few import things to keep in mind:
  - Do not name your variable after a common command, function, or a variable built-in to the coding language being used
  - Avoid using spaces and any characters other than letters, numbers, dashes, and underscores – Why?
  - Choose informative, memorable, and short variable names

- Example: You want to create a variable to store gene expression data from Drosophila melanogaster (aka fruit flies) that were grown at 10°C, 20°C, or 30°C. Which of the names below would be good variable names to use?

1. GeneExpressionData_Drosophilamelanogaster_DifferentTemperatures
2. Dmel_GE_dT
3. FruitFlies_GE_dT
4. FruitFlies_GE_10°C_ 20°C_ 30°C
5. echo
6. myData
7. DmelGEdT
8. Dmelanogaster GE Data

Some Common Unix Commands (we will practice these together)

date

- prints out the date and time

head

- prints the first 10 lines by default of a file

pwd

- stands for "print working directory", will print the path to your current working directory

ls

- stands for ”list", lists all files in the current working directory, or in the path provided as a positional argument at the end of the command

mkdir

- creates a new folder (aka directory)

cd

- stands for "change directory”, allows you to navigate to different directories on the command line

cp

- stands for "copy”, allows you to copy files from one location to another (or in the same location as the original)

mv

- stands for "move”, used to either rename a file/folder and/or move a file/folder to another location

cat

- stands for "concatenate”, prints the entire contents of a file and can be used to combine files

Some Common R Functions (we will practice these together)

R command format: In R, you pass arguments to functions by putting them between parentheses

help()

- provides a summary of how to use the R function provided inside the parenthesis

read.csv()

- allows you to read information in your .CSV file into R

head()

- prints the first 6 lines by default of a data object

dim()

- prints the dimensions of a data object loaded in R

summary()

- prints a summary of the data object provided inside the parenthesis

list.files()

- lists all the files in the location indicated in the parenthesis (if no location is indicated, it will list the files in the current or working directory)

getwd()

- prints the current working directory

setwd()

- changes the current working directory to the location provided by a file path string

write.csv(data, 'filename.csv')

- creates a csv file containing the data held in the variable indicated to the left of the comma with the file name indicated to the right of the comma

Some Common Data Frame Manipulations in R (we will practice these together)

DF + 1

- add 1 to all cells of a data frame (represented by DF above)

log(DF)

- calculates the natural log for all cells of a data frame

ceiling(DF)

- rounds decimal values in each cell of a data frame up to the nearest integer

DF['column1']

- extracts only the indicated column from a data frame

DF['row1',]

- extracts only the indicated row from a data frame

DF[rowSums(DF)>0 ,]

- filters the data in a data frame to only contain the rows whose sum is greater than zero (the mathematical operator and filtering value can be changed as needed)

DF['column4'] <- c(1,2,3,4,5,6,7,8,9,10)

- adds a new column (column 4) to the data frame with the values indicated (note: the number of values must match the number of rows in the data frame the column is added to)

cbind(DF, DF2)

- combines the columns in data frame, DF, with the columns in another data frame, DF2

Jupyter Notebook

- The GL4U modules utilize Jupyter Lab and Jupyter Notebooks (JN) to run commands.
- Jupyter Notebook (JN) was created as part of Project Jupyter (https://jupyter.org/) and is a web application that allows you to create documents that have both narrative text and live code; meaning you can run commands through a web browser (instead of a terminal).
- Additional advantages of JNs:
  - Facilitates sharing code and results between groups
  - Ease of publishing your code
  - Allows for reproducibility

4c. Volcano Plot

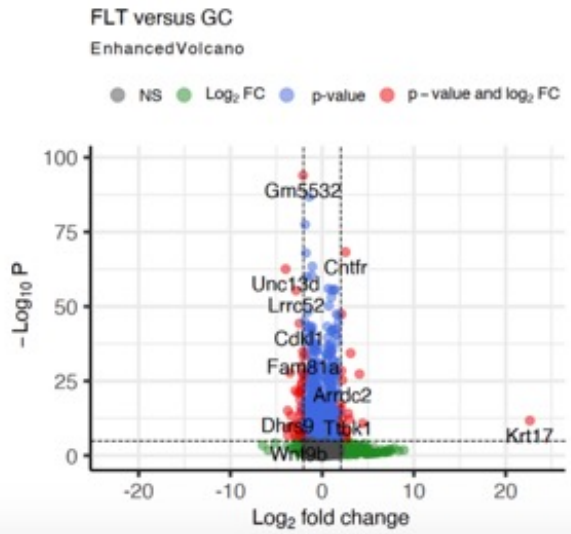
Finally, let's make a volcano plot to identify a few interesting genes. A volcano plot is a scatterplot which shows the relationship of the adjusted p-value significant by adjusted p-value are labeled.

First, we'll use the default settings from the `EnhancedVolcano()` function: log2 Fold Change cutoff > |2|, and the adjusted p-value cutoff is < 10e-6.

You can read about `EnhancedVolcano()` and see some examples by clicking here

```
# Volcano plot showing genes differentially expressed in FLT vs GC
EnhancedVolcano(DGE_output_table,
    lab = DGE_output_table$SYMBOL,
    x = 'Log2fc_(FLT)v(GC)',
    y = 'Adj.p.value_(FLT)v(GC)',
    title = 'FLT versus GC',
    pCutoff = 10e-6,
    FCcutoff = 2,
    pointSize = 3.0,
    labSize = 6.0,
    colAlpha=0.5)

## Save your volcano plot
ggsave(file.path(DGE_plots,'GLDS-104_volcano_DGE.png'), width = 6, height = 8.5, dpi = 300)
```



Additional Training Resources

Below are some basic coding training resources (in addition to the GL4U: Introduction module set):

Unix-like Resources:

- Happy Belly Bioinformatics: Unix Introduction
- HBC Training: Intro to the command line interface (shell/bash/Unix/Linux)
- Babraham Bioinformatics: An Introduction to Unix

R Resources:

- Happy Belly Bioinformatics: R Introduction
- HBC Training: Introduction to R (Online)
- Babraham Bioinformatics: Various R Courses
- Hadley Wickham, et al.: R for Data Science
- RStudio Education: Hands-On Programming with R

END
GL4U: Introduction
Lecture 3 of 4