Part 1: Review Questions

General Concepts

1.) What is TCGA and why is it important?

TCGA or *The Cancer Genome Atlas* is a publically available data set of cancer data. This data includes genomic, transcriptomic, proteomic, and epigenomic data that has been matched to a normal sample and has data that spans 33 cancer types. This publically available data has been used in many bioinformatic studies as it is stringently collected and TCGA data has been used to find biomarkers, treatments, subtypes, and many other types of information about cancer.

2.) What are some strengths and weaknesses of TCGA?

Some strengths of TCGA data include its wide spread of cancers (including rarer ones) and the inclusion of multi-omics data in its analysis. 33 different cancer types are present in TCGA data and they are studied at the genomic, transcriptomic, epigenomic, and proteomic levels. These cancers include cancers such as ACC and Thymoma that are not usually studied due to their rarity and can give valuable insight into cancers that we don't know much about. In addition to the omic data, TCGA also provides clinical data that can be used to further study. This data also follows a standard procedure so that the data can be collated into a homogenous format. However, even with its wide range of data, some of the cancer types still have small sizes and not all data is recorded for every patient. Groups like ACC only have 79 samples as the cancer itself is really rare. TCGA data also commonly don't have certain clinical data available so the data set is not entirely complete. The processes that the TCGA uses to collect data are also not as strict as projects such as BRCA so the TCGA data is a little bit less reliable. Even so, TCGA data is incredibly informative for even rare cancers with small sample sizes and has been a massive driver of bioinformatics research.

Coding Skills

1.) What commands are used to save a file to your GitHub repository?

After cd into the correct directory, you can use:

git add To add the file/directory you want to save into the index of files/directories to add to github on your computer.
git commit -m "message" To attach a message to the files that you have git added that describes their purpose and/or contents as well as saving changes made to the index by git add.
git push Which saves the files to your github repository with the attached message.

2.) What command(s) must be run in order to use a package in R?

The general form of an R package install is:

install.packages("package name")
library (package name)

The install.packages() command installs a package from CRAN which can then be used inside of the project you are working on by using the library() function.

3.) What command(s) must be run in order to use a Bioconductor package in R?

To use a Bioconductor package in R, you must first download Bioconductor and load it in:

install.packages("BiocManager")
library(BiocManager)

And then once Bioconductor is ready to be used, you use the function:

BiocManager::install("package name")

4.) What is boolean indexing? What are some applications of it?

Boolean indexing is the creation of a vector full of boolean values that can be used to segment data based on certain values that you select for through the use of masking. You will create a vector full of boolean values (true or false) called a mask and then apply that mask to a data frame to subset that data with true values being data that you keep. You can use boolean masking to subset data using specific clinical variables or cleaning data of any NA or other not informative values.

5.) 5. Draw a mock up (just a few rows and columns) of a sample dataframe.

df

	1	**2**	**3**
A	2.2	4	x
B	1	6	y
C	6	7	x

DataFrame with name df with column names 1, 2, 3 and row names A, B, C

5a.) Show an example of an ifelse() statement and explain what the code does.

ifelse(df$1 > 2 , "greater_than_2", "less_than_2)

This ifelse() statement will look at column 1 of our dataframe (df) and check the values of the column to see if they are greater than 2. The code will generate a vector based on the conditions with values above 2 being labelled as greater_than_2 and values less than 2 being labelled as less_than_2. The ifelse() statement should return the following vector:

“greater_than_2”, “less_than_2”, “greater_than_2”

5b.) Show an example of boolean indexing and explain what the code does.

```
x_mask <- ifelse(df$3 == “x”, T, F)
df <- df[x_mask, ]
```

The first statement creates a boolean vector called x_mask which returns the boolean value TRUE for all x values in the 3 column and FALSE for any other value in the 3 column. This vector can then be applied to the same dataframe using the second statement removing all rows who have a value in the 3 column that isn’t x. In effect, we are creating a new cleaned dataframe that would look like this:

	1	**2**	**3**
A	2.2	4	x
C	6	7	x

As the only row that didn’t have an x value in the 3rd column was row B.