Week 11 Activity: Independent CPTAC Data Analysis!

When writing a Python script, keep in mind the following good practices:

1. Always write your `main()` function and `if __name__ == "__main__"` statement. The code you want to be executed should *always* be in your `main()` function -- NEVER have code floating around outside of `main()`:
2. Import your packages at the top of your script.
3. Write functions!
4. Always use **relative paths** -- if you share the script with someone else, their computer files won't be set up the same way as yours!

The Assignment:

Rather than analyzing the correlation between the expression of transcriptomics/proteomics, we'll compare the expression of individual *genes*.

Creating the Heatmap

Let's say I want to investigate the following 7 genes:

```
genes = ["KRAS", "BRAF", "TP53", "CD3E", "TPI1", "GAPDH", "PDHA1"]
```

First, review the code we ran to create `protein_shared` and `rna_shared`, which are the data frames containing the patients and genes that have proteomics and transcriptomics data. Then, we want to create TWO 7x7 heatmaps:

1. The first heat map compares how the proteomic expression data. For example, the first cell calculates the correlation of KRAS expression to KRAS, the next cell compare KRAS to BRAF, and so on.
2. The second heat map compares the transcriptomic expression data.

If you're confused, refer to the slides!

To do so, write the following functions:

1. `intersectDataFrames()`. Tales in two parameters, (1) the protein data frame and (2) the RNA data frame. It should return two data frames, where the returned datawframes have only the rows/column names that are shared between the two input data frames (i.e. the same thing we did last week).
2. `createCorrelationMatrix()`. Takes in two parameters, (1) a list of genes and (2) a data frame of expression data, and returns a correlation matrix.
3. `createCorrHeatmap()`. Takes in four parameters, (1) a file path, (2) a list of names to title the plots (i.e. for your purposes you'll probably call the function wtih `["Protein", "RNA"]` as its argument), and (3, 4) two correlation matrices. It doesn't return anything, but saves a plot directly to a file path (first argument).

Your code structure should look like the following. Note that when you're done, main will only have about 6 lines of code in it -- this is good! Keep main short.

```
# import libraries here

# write your two functions here
```

```
def main():
    genes = ["KRAS", "BRAF", "TP53", "CD3E", "TPI1", "GAPDH", "PDHA1"]
    # TODO: create two correlation matrices (one for protein, one for RNA)
    # TODO: use these two correlation matrices to create a heatmap using your heatmap
function
    # remove the TODO comments when you're done!


if __name__ == "__main__":
    main()
```

Requirements:

- Write the three functions specified above and the `main()` function.
- `main()` in theory should just create the `genes` list, load in the data (2 lines), call `intersectDataFrames()`, call `createCorrelationMatrix()` twice, and `createCorrHeatmap()` once (so it's possible to do this in 6 lines).
- Write `if __name__ == "__main__":` line and only call `main()` after it.
- In general, keep the code style clean and consistent.

This script will be checked off before you leave.

Extra Hint

The code to generate the scatterplot will be a good reference for the kind of analysis you'll be doing (but you might notice that it's quite repetitive -- this is a strong hint to write a function separates out the work!):

```
fig, ax = plt.subplots(1, 2, constrained_layout=True)

sns.scatterplot(
    x = prot_shared["KRAS"],
    y = prot_shared["BRAF"],
    ax = ax[0]
)

sns.scatterplot(
    x = rna_shared["KRAS"],
    y = rna_shared["BRAF"],
    ax = ax[1]
)

corr_prot, p_prot = stats.spearmanr(prot_shared.KRAS, prot_shared.BRAF, nan_policy =
"omit")
corr_rna, p_rna = stats.spearmanr(rna_shared.KRAS, rna_shared.BRAF, nan_policy =
"omit")
ax[0].set(title = f"Protein rho = {round(corr_prot, 3)}")
ax[1].set(title = f"RNA rho = {round(corr_rna, 3)}")


plt.savefig("scatter_cors.png")
```