Honors Class04 Activity: Working with Data Tables

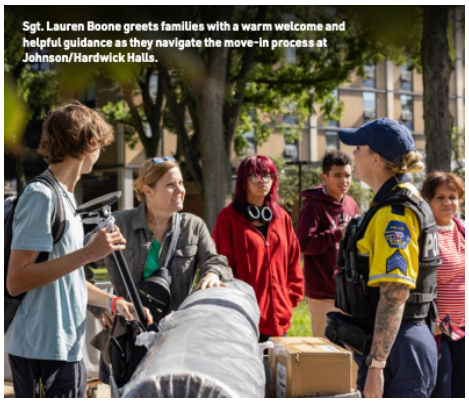
Team Members: _____

Learning Objectives

- Introduction to the datascience module
- Learn how to create and manipulate data tables.

Data Tables

Tabular data is common in all fields, including science, so we need ways to work with data table in Python. We will be using the *datascience* module in this course.



Our first table we will build from scratch. In Lab03 you will save typing by reading data from a file into a table. The data comes from Temple's 2025 Annual Security and Fire Safety Report. To keep the table small, we will focus on just a small subset of the data, just crime in the residence halls and crime categories with non-zero values. For example, there was no embezzlement, vagrancy, or driving under the influence (surprise!) in the dorms, so those rows have been omitted.

Open a fresh Jupyter notebook and enter the following in a code cell.

```
# Load all of the functions in the datascience module
from datascience import *

# Data for each column goes in a list or array.
categories = [
    "Simple Assault",
    "Harassment",
    "Fraud",
    "Theft",
    "Vandalism",
    "Disorderly Conduct",
    "Sex Assault (Other)",
    "Drunkenness",
```

```
    "All Other Offenses",
]

counts_2024 = [13, 44, 6, 30, 13, 1, 1, 8, 0]
counts_2023 = [0, 61, 9, 24, 17, 6, 0, 7, 1]
counts_2022 = [3, 32, 9, 23, 9, 0, 1, 5, 6]

# Create the table
res_hall = Table().with_columns(
    "Category",
    categories,
    "2024",
    counts_2024,
    "2023",
    counts_2023,
    "2022",
    counts_2022,
)

res_hall
```

Category	2024	2023	2022
Simple Assault	13	0	3
Harassment	44	61	32
Fraud	6	9	9
Theft	30	24	23
Vandalism	13	17	9
Disorderly Conduct	1	6	0
Sex Assault (Other)	1	0	1
Drunkenness	8	7	5
All Other Offenses	0	1	6

It is import to understand what just happened. A data table is comprised of columns. The columns are arrays of data. Different columns can hold different types of data, but all of the rows in a given column must be the same data type. Here we have one column of strings (the type of crime) and three numeric columns for the three years of data. Each column in the table has a label, the column header.

So to build this table, we first created a list for each column. Then we called `Table().with_columns()` to buld the table. The arguments to this function were label, array, label, array, …
for at many columns a needed. Note: all of the columns must have the same number number of rows.

Working with Data Tables

Once the data is in a datascience table, you have have many properties and methods available to operate on the data.

Table Properties

Table properties and simple that, properties of the table. They do not involve manipulating the data. Here are a couple of examples for you to try. **Type them into your notebook.**

```
# The number of rows is a table property
res_hall.num_rows
```

```
9
```

```
# The number of columns is a table property
res_hall.num_columns
```

```
4
```

```
# The collection of column headers is a table property
res_hall.labels
```

```
('Category', '2024', '2023', '2022')
```

Table Methods

Unlike properties, table methods are Python functions built into data tables that operate on the table data. **Type the following examples into your notebook and see if you can figure out what they do.**

```
res_hall.select('Category', '2024')
```

Category	2024
Simple Assault	13
Harassment	44
Fraud	6
Theft	30
Vandalism	13
Disorderly Conduct	1
Sex Assault (Other)	1
Drunkenness	8
All Other Offenses	0

The `select()` method does what?

```
res_hall.column('2024')
```

```
array([13, 44,  6, 30, 13,  1,  1,  8,  0])
```

The `column()` method does what?

```
res_hall.take(3)
```

Category	2024	2023	2022
Theft	30	24	23

The `take()` method does what?

```
res_hall.sort('2024')
```

Category	2024	2023	2022
All Other Offenses	0	1	6
Disorderly Conduct	1	6	0
Sex Assault (Other)	1	0	1
Fraud	6	9	9
Drunkenness	8	7	5
Simple Assault	13	0	3
Vandalism	13	17	9
Theft	30	24	23
Harassment	44	61	32

The `sort()` method does what?

Filtering Tables

In Lab03 you will get practice with the `where()` method, which returns the rows of a table meet some condition specified in the where *predicate*. Here are some examples. You pass the `where()` function the label for the column you want to filter and the condition. Here we return a new table with just the rows where the `2024` column has values greater than 25.

```
res_hall.where('2024', are.above(25))
```

Category	2024	2023	2022
Harassment	44	61	32
Theft	30	24	23

```
res_hall.where("Category", are.equal_to("Fraud"))
```

Category	2024	2023	2022
Fraud	6	9	9

(One wonders what sort of fraud occurs in residence halls.)

```
res_hall.where("2022", are.between(5, 10))
```

Category	2024	2023	2022
Fraud	6	9	9
Vandalism	13	17	9
Drunkenness	8	7	5
All Other Offenses	0	1	6

Suppose we wanted to know the total number of incidents in the table for 2024. One way to do this is to extract the data from that column and sum it.

```
incidents_2024 = res_hall.column("2024")
sum(incidents_2024)
```

```
116
```

How many total incidents were there in 2022? _____

For a small table such as this one, it might be easier to use a calculator, but image a table with millions of rows!

Discussion Quesions

The activity emphasizes that columns are arrays.

Why do you think the designers of the datascience module emphasize column-wise thinking instead of row-wise thinking? How does this choice influence the kinds of questions that are easy vs. hard to ask?

The where() method filters rows based on conditions.

Is filtering a neutral operation? How could filtering choices unintentionally bias an analysis?

Suppose sum(res_hall.column("2024")) returned a value that surprised you. **What would be your first step in diagnosing the issue? Which intermediate checks would you perform before assuming the data were wrong?**

The activity computes total incidents by summing a column. **What assumptions are required for this total to be meaningful? Are all incidents equally "countable" in the same way?**