Master Python Programming with ChatGPT as Your Personal Intelligent Tutor

Part III

Minder Chen

1. **Generating datasets**
2. **Manipulating data**
3. **Save data in an Excel file**
4. **Visualize the data: Plot**
5. **Create an assignment**
6. **Solve the assignment**

My conversation with ChatGPT has been documented using the following format. The content may be edited for clarity. Some key terms and concepts may be highlighted by the author of this document for the reader.

[Task type]

MI: Here is my instruction or question submitted to ChatGPT

Response: Here is the response including code or dataset generated by ChatGPT

Note: Pygment, a Python syntax highlighter, has been used to highlight the code in colors. Pygment is free and can be found at https://pygments.org/demo/

[Generating datasets]

MI: Create a dataset contain students' midterm, final, and project score the score can be 0 to 100. I have 5 students in my class: Mary, John, Bill, July, and Paul. The weight for midterm, final and project weights are: 25%, 35%, and 40%.

Response:

Here is an example of a dataset that contains the midterm, final, and project scores for five students, as well as the weighting for each component of the final grade:

Student	Midterm Score	Final Score	Project Score
Mary	0	100	0
John	50	60	70
Bill	80	85	80
July	90	80	90
Paul	75	70	75

And the weights for each component: Midterm: 25% Final: 35% Project: 40%

The final grade for each student can be calculated by:

final_grade = (midterm_score * 0.25) + (final_score * 0.35) + (project_score * 0.4)

[Manipulating data]

MI: Create a Python Program to calculate their final score.

Response:

Here is a Python program that calculates the final score for each student based on their midterm, final, and project scores, as well as the weighting for each component:

```
# Data of students scores
```

```python
students = [
    {"name": "Mary", "midterm": 0, "final": 100, "project": 0},
    {"name": "John", "midterm": 50, "final": 60, "project": 70},
    {"name": "Bill", "midterm": 80, "final": 85, "project": 80},
    {"name": "July", "midterm": 90, "final": 80, "project": 90},
    {"name": "Paul", "midterm": 75, "final": 70, "project": 75},
]

# Weights
midterm_weight = 0.25
final_weight = 0.35
project_weight = 0.4

# Calculate the final score for each student
for student in students:
    midterm_score = student["midterm"] * midterm_weight
    final_score = student["final"] * final_weight
    project_score = student["project"] * project_weight
    final_grade = midterm_score + final_score + project_score
    student["final_grade"] = final_grade
    print(f'{student["name"]} : {final_grade}')
```

This program first creates a list of dictionaries called "students", where each dictionary contains the name, midterm score, final score, and project score for a single student. Then, the program defines the weighting for each component, midterm_weight, final_weight and project_weight. Then, it iterates through the list of students and calculates the final grade for each student by multiplying their midterm score by the midterm weight, their final score by the final weight, and their project score by the project weight, then adding those three values together. Finally it creates a new key in each student dictionary called 'final_grade' which stores the final score of each student. and print out the final score of each student.

Please note that the program and data is for demonstration purpose only and you can change it to fit your need.

[Save data in an Excel file]

MI: Generate test dataset and store it in an Excel file.

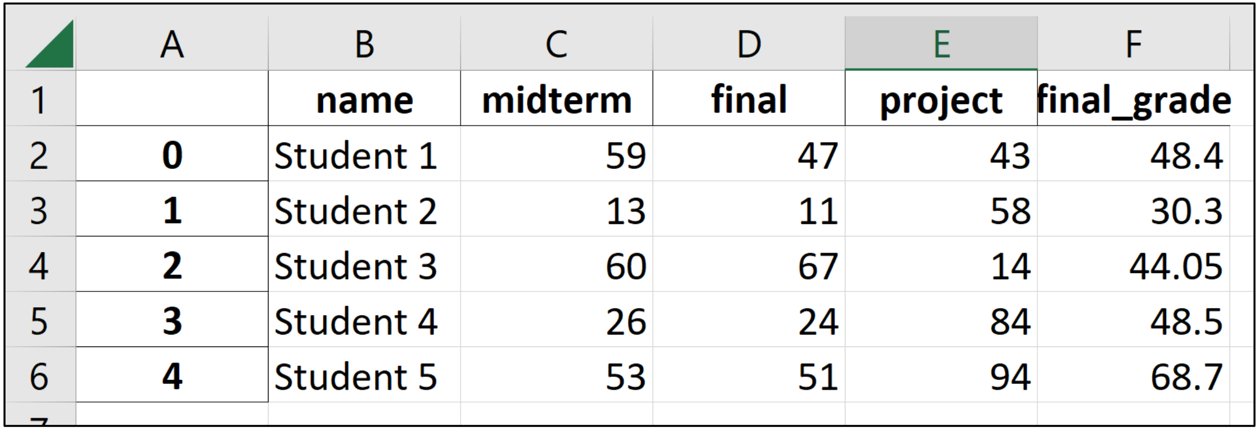
Response:

Here is a Python program that generates a test dataset with random scores for five students, stores the data in a dictionary, and then writes the data to an Excel file:

```python
import random
import pandas as pd

# Generate test data for 5 students
students = []
for i in range(5):
    student = {
        "name": f"Student {i+1}",
        "midterm": random.randint(0, 100),
        "final": random.randint(0, 100),
        "project": random.randint(0, 100),
    }
    students.append(student)

# Weight for each component
midterm_weight = 0.25
final_weight = 0.35
project_weight = 0.4

# Create final score
for student in students:
    midterm_score = student["midterm"] * midterm_weight
    final_score = student["final"] * final_weight
    project_score = student["project"] * project_weight
    final_grade = midterm_score + final_score + project_score
    student["final_grade"] = final_grade

# Create DataFrame
df = pd.DataFrame(students)

# Write DataFrame to Excel
df.to_excel("students.xlsx")

print("The data has been written to the 'students.xlsx' file")
```

This program first uses the `random` library to generate random scores for 5 students and store them in a list of dictionaries called "students". Then, it defines the weighting for each component, midterm_weight, final_weight and project_weight. It then iterates through the list of students and calculates the final grade for each student by multiplying their midterm score by the midterm weight, their final score by the final weight, and their project score by the project weight, then adding those three values together. It creates a new key in each student dictionary called 'final_grade' which stores the final score of each student. Then, it creates a `pandas` DataFrame from the `students` list. Finally, it uses the `to_excel()` method of the DataFrame to write the data to an Excel file called "students.xlsx". Please note that the program uses the `pandas` library to write data to an Excel file and it needs to be installed before using the code above. If you don't have it install it by running `pip install pandas` in your command prompt.

Note: The following is the `students.xlsx` file generated by the Python program



	A	B	C	D	E	F
1		**name**	**midterm**	**final**	**project**	**final_grade**
2	**0**	Student 1	59	47	43	48.4
3	**1**	Student 2	13	11	58	30.3
4	**2**	Student 3	60	67	14	44.05
5	**3**	Student 4	26	24	84	48.5
6	**4**	Student 5	53	51	94	68.7

MI: The names for the 5 students' names are: Mary, John, Bill, July, and Paul. Also the score are normally distributed with mean of 80 and and 10 for standard deviation. The score should be between 0 and 100. Try again

Response:

Here is a Python program that generates test data based on normal distribution with a mean of 80 and standard deviation of 10 for the midterm, final, and project scores of five students. It also includes the names for the 5 students and makes sure that the score is between 0 and 100. Then, it stores the data in a dictionary and writes the data to an Excel file.

```
import random
import numpy as np
import pandas as pd

# Names of 5 students
students_names = ["Mary", "John", "Bill", "July", "Paul"]

# Generate normally distributed test data for 5 students
np.random.seed(0)
midterm_scores = np.random.normal(80, 10, 5)
final_scores = np.random.normal(80, 10, 5)
project_scores = np.random.normal(80, 10, 5)

students = []
for i in range(5):
    student = {
        "name": students_names[i],
        "midterm": max(0, min(int(midterm_scores[i]), 100)),
        "final": max(0, min(int(final_scores[i]), 100)),
        "project": max(0, min(int(project_scores[i]), 100)),
    }
    students.append(student)

# Weight for each component
midterm_weight = 0.25
final_weight = 0.35
project_weight = 0.4

# Create final score
for student in students:
    midterm_score = student["midterm"] * midterm_weight
    final_score = student["final"] * final_weight
    project_score = student["project"] * project_weight
    final_grade = midterm_score + final_score + project_weight
    student["final_grade"] = final_grade

# Create DataFrame
df = pd.DataFrame(students)

# Write DataFrame to Excel
df.to_excel("students.xlsx")
```

```
print("The data has been written to the 'students.xlsx' file")
```

This program first uses the `numpy` library to generate random scores based on a normal distribution with a mean of 80 and standard deviation of 10 using the `numpy.random.normal()` function. Then, it iterates through the list of students' names to populate the students' names and make sure that the scores are between 0 and 100. Then, it defines the weighting for each component, midterm_weight, final_weight and project_weight. It then iterates through the list of students and calculates the final grade for each student by multiplying their midterm score by the midterm weight, their final score by the final weight, and their project score by the project weight, then adding those three values together. It creates a new key in each student dictionary called 'final_grade' which stores the final score of each student. Then, it creates a `pandas` DataFrame from the `students` list. Finally, it uses the `to_excel()` method of the DataFrame to write the data to an Excel file called "students.xlsx" Please note that the program uses the `pandas` library to write data to an Excel file and it needs to be installed before using the code above. If you don't

[Explain code]

MI: What does max(0, min(int(midterm_scores[i]), 100)) mean?

Response:

`max(0, min(int(midterm_scores[i]), 100))` is a function that ensures that the final value is between 0 and 100.

Here's what happens step by step:

- `int(midterm_scores[i])` converts the floating point value generated by numpy.random.normal() to an integer.
- `min(int(midterm_scores[i]), 100)` returns the minimum value between the integer and 100. So if the score is greater than 100 the function will return 100.
- `max(0, min(int(midterm_scores[i]), 100))` returns the maximum value between 0 and the minimum value calculated before. So if the score is lower than 0 the function will return 0.

This is done for all the score values, midterm, final and project score. This makes sure that the score is between 0 and 100, which is the requirement specified in the question.

[Visualize the data: Plot]

MI: Can you draw a chart 20 data points based on np.random.normal(80, 10, 5). Choose the best chart type and tell me why you choose it.

Response:

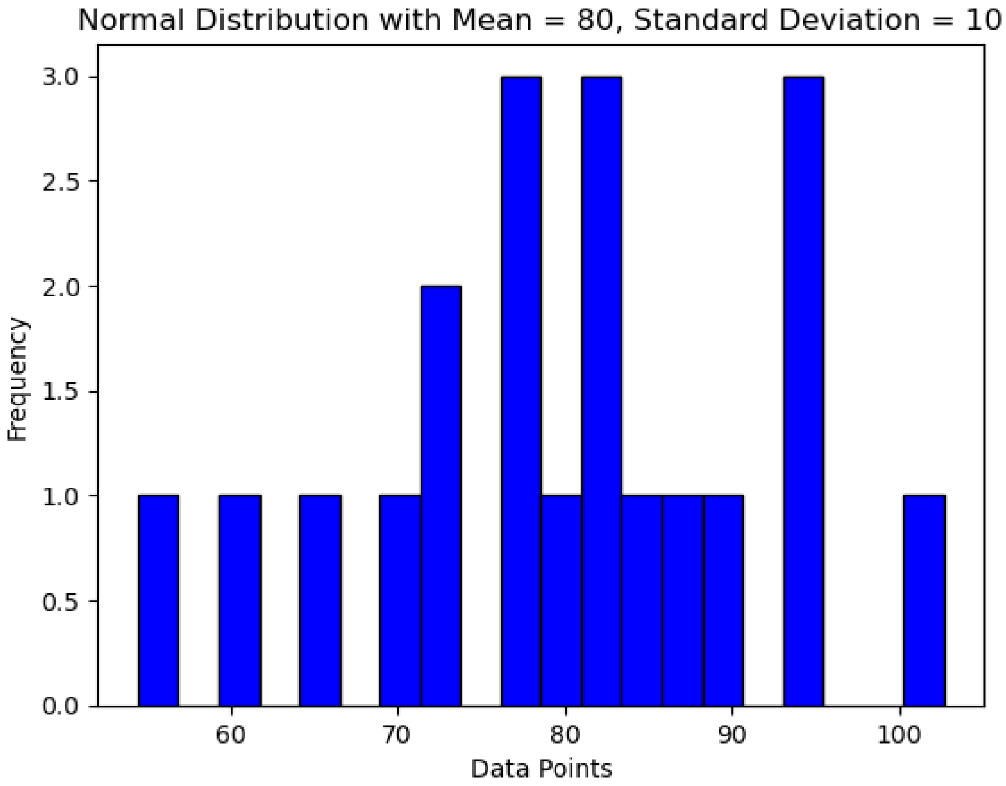
Here is a Python program that generates 20 data points based on the normal distribution with a mean of 80 and standard deviation of 10 using `numpy.random.normal()`, and plots them in a histogram:

```
import matplotlib.pyplot as plt
import numpy as np

# Generate data
data = np.random.normal(80, 10, 20)

# Plot the data in a histogram
plt.hist(data, bins=20, color = 'b', edgecolor = 'k')
plt.xlabel('Data Points')
plt.ylabel('Frequency')
plt.title('Normal Distribution with Mean = 80, Standard Deviation =
10')
plt.show()
```

The above code generates a histogram chart, a **histogram is used in this case as it is an effective way to visualize the distribution of a continuous variable**, the x-axis represents the range of the data points and the y-axis represents the frequency of the data points within that range. Here it shows the frequency of the data points generated by the normal distribution with a mean of 80 and a standard deviation of 10. It helps to visualize the distribution of the generated data points, its shape, and if it follows the normal distribution.

It is also possible to use different types of charts such as boxplot, scatterplot and lineplot with this data. But Histograms are best option to visualize the distribution of continuous variable.

Note: This is the chart drawn by the generated Python Program. The chart drawn by the program will be different because the data are generated randomly each time.



**** Note:** Ask ChatGPT "Choose the best chart type and tell me why you choose it." Involved high level thinking. Choose chart types based on various charaacteristics of you data is an important skill in data visualization.

ChatGPT rewrite my note above as:

"Which chart type would you recommend for my data set and **why**? Can you provide **reasoning** for your choice based on the **characteristics of my data**?"

[Creata an assignment]

MI: Create a new Python assignment and put everything we have learned so far into it.

Response:

Here is a new Python program that puts everything we have learned so far into it:

```
import random
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

# Names of 5 students
students_names = ["Mary", "John", "Bill", "July", "Paul"]

# Generate normally distributed test data for 5 students
np.random.seed(0)
midterm_scores = np.random.normal(80, 10, 5)
final_scores = np.random.normal(80, 10, 5)
project_scores = np.random.normal(80, 10, 5)

students = []
for i in range(5):
    student = {
        "name": students_names[i],
        "midterm": max(0, min(int(midterm_scores[i]),
100)),
        "final": max(0, min(int(final_scores[i]), 100)),
        "project": max(0, min(int(project_scores[i]),
100)),
    }
    students.append(student)

# Weights
midterm_weight = 0.25
final_weight = 0.35
project_weight = 0.4
```

```
# Calculate the final score for each student
for student in students:
    midterm_score = student["midterm"] * midterm_weight
    final_score = student["final"] * final_weight
    project_score = student["project"] * project_weight
    final_grade = midterm_score + final_score + project_weight
    student["final_grade"] = final_grade

# Create DataFrame
df = pd.DataFrame(students)

# Write DataFrame to Excel
df.to_excel("students.xlsx")
print("The data has been written to the 'students.xlsx' file")

# Plot histogram of final grade
plt.hist(df['final_grade'], bins=20, color = 'b', edgecolor = 'k')
plt.xlabel('Final Grade')
plt.ylabel('Frequency')
plt.title('Final Grade Distribution')
plt.show()
```

The program first generates scores for 5 students based on normal distribution with a mean of 80 and standard deviation of 10, then ensure that the scores are between 0 and 100. Then, it populates the students names, defines the weighting for each component, midterm_weight, final_weight and project_weight, and calculates the final grade for each student. It then creates a Pandas DataFrame from the students data and write the data to an Excel file called 'students.xlsx' Finally, it plots a histogram of the final grade distribution which can be used to visualize the distribution of the final grade.

This program covers all the aspects that we have learned so far including creating a dataset, calculating final scores, writing data to excel file and plotting the data in histogram. Please note that the program uses the `pandas` library to write data to an Excel file, `numpy` for generating normal distribution data and `matplotlib` for plotting the histogram, make sure you have those libraries installed in order to use the above code.

[Create an assignment for Python but no code]
MI: Can you just create the assignment in plain English as detail as possible but *do not generate Python code*.

Response:

Sure, here is a detailed plain English version of the assignment:

1. Create a dataset containing students midterm, final, and project scores. The scores can be between 0 and 100. There are 5 students in the class: Mary, John, Bill, July, and Paul. The weighting for midterm, final, and project scores is 25%, 35%, and 40% respectively.
2. Write a program that calculates the final score for each student based on their midterm, final, and project scores, as well as the weighting for each component.
3. Generate test data set by normally distributed data with a mean of 80 and standard deviation of 10 for the midterm, final, and project scores. Also, ensure that the score is between 0 and 100.
4. Store the data in a dictionary and write the data to an Excel file.
5. Create a histogram chart showing the final grade distribution.

Note that the final grade calculation is:

final_grade = (midterm_score * midterm_weight) + (final_score * final_weight) + (project_score * project_weight)

In order to complete this assignment, you'll need to use a variety of skills, including working with data structures like lists and dictionaries, basic math and statistics concepts, file I/O, and data visualization using a library like matplotlib. It's also expected that you should be familiar with basic libraries like numpy, pandas and excel.

Please make sure to read the instruction carefully and also to test your code as you go along to ensure that everything works as expected.

Please let me know if there is anything more I could assist.

[Explain data structure written in Python]
MI: Explain the data structure for me:
students = [{"name": "Mary", "midterm": 0, "final": 100, "project": 0},
{"name": "John", "midterm": 50, "final": 60, "project": 70},
{"name": "Bill", "midterm": 80, "final": 85, "project": 80},
{"name": "July", "midterm": 90, "final": 80, "project": 90},
{"name": "Paul", "midterm": 75, "final": 70, "project": 75},]

Response:

The data structure in this example is a list of dictionaries, where each element in the list represents a student, and each dictionary contains the properties of a student, including the name, midterm score, final score, and project score.

The list is named `students` and contains 5 elements, each of which is a dictionary. Each dictionary has keys 'name', 'midterm', 'final' and 'project' where the values of the key are associated with the respective student. For example, the first element in the list represents the student named "Mary" with a midterm score of 0, a final score of 100, and a project score of 0. The second element represents the student named "John" with midterm score of 50, final score of 60, and project score of 70.

This data structure is easy to understand and efficient for storing student data, and it can be easily indexed, iterated, and modified, which makes it convenient for data processing and analysis.

You can access individual students data using the index of the list, For example, to get the data of student Bill in the list you can use students[2], where 2 is the index of Bill in the list.

Also, you can access the individual keys of the dictionary using the keys name, for example, to get the midterm score of student Bill you could use students[2]['midterm']

[Complete the Python program based on an assignment description]

MI: Write a Pyhton program for the following task:

Here is a detailed description of the assignment:

1. Create a dataset containing students midterm, final, and project scores. The scores can be between 0 and 100. There are 5 students in the class: Mary, John, Bill, July, and Paul. The weighting for midterm, final, and project scores is 25%, 35%, and 40% respectively.
2. Write a program that calculates the final score for each student based on their midterm, final, and project scores, as well as the weighting for each component.
3. Generate test data set by normally distributed data with a mean of 80 and standard deviation of 10 for the midterm, final, and project scores. Also, ensure that the score is between 0 and 100.
4. Store the data in a dictionary and write the data to an Excel file.
5. Create a histogram chart showing the final grade distribution.

Note that the final grade calculation is:

final_grade = (midterm_score * midterm_weight) + (final_score * final_weight) + (project_score * project_weight)

In order to complete this assignment, you'll need to use a variety of skills, including working with data structures like lists and dictionaries, basic math and statistics concepts, file I/O, and data visualization using a library like matplotlib. It's also expected that you should be familiar with basic libraries like numpy, pandas and excel.

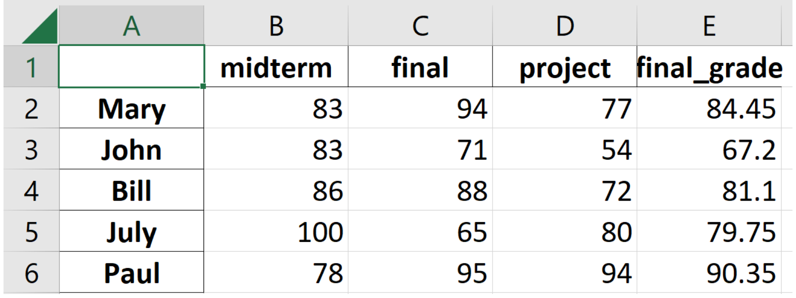
Please make sure to read the instruction carefully and also to test your code as you go along to ensure that everything works as expected.

Response:

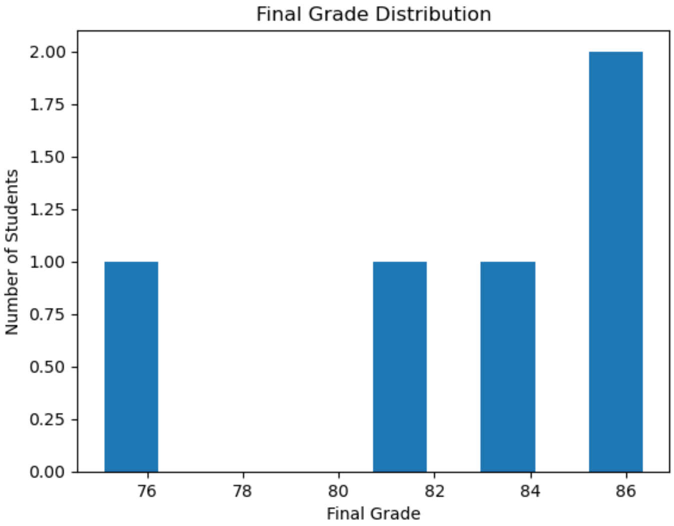
```
import random
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

# Define the weighting for each component
midterm_weight = 0.25
final_weight = 0.35
project_weight = 0.40

# Create a dictionary containing the student names and their scores
students = {
    'Mary': {'midterm': 0, 'final': 0, 'project': 0, 'final_grade': 0},
    'John': {'midterm': 0, 'final': 0, 'project': 0, 'final_grade': 0},
    'Bill': {'midterm': 0, 'final': 0, 'project': 0, 'final_grade': 0},
    'July': {'midterm': 0, 'final': 0, 'project': 0, 'final_grade': 0},
    'Paul': {'midterm': 0, 'final': 0, 'project': 0, 'final_grade': 0}
}

# Generate test data set by normally distributed data with a mean of
80 and standard deviation of 10
# for the midterm, final, and project scores
for student in students:
    students[student]['midterm'] = max(0, min(100,
int(np.random.normal(80, 10))))
    students[student]['final'] = max(0, min(100,
int(np.random.normal(80, 10))))
    students[student]['project'] = max(0, min(100,
int(np.random.normal(80, 10))))

# Calculate the final grade for each student based on their scores
and the weighting
for student in students:
    midterm_score = students[student]['midterm']
    final_score = students[student]['final']
    project_score = students[student]['project']
    final_grade = (midterm_score * midterm_weight) + (final_score *
final_weight) + (project_score * project_weight)
    students[student]['final_grade'] = final_grade

# Write the data to an Excel file
df = pd.DataFrame(students).T
df.to_excel('student_scores.xlsx')
```

```
# Create a histogram chart showing the final grade distribution
final_grades = [students[student]['final_grade'] for student in
students]
plt.hist(final_grades, bins=10)
plt.xlabel('Final Grade')
plt.ylabel('Number of Students')
plt.title('Final Grade Distribution')
plt.show()
```

Note: The Excel file 'student_scores.xlsx' and the histogram generated by the program. There are only 5 data points such that the histogram does not show a pattern of a normally distributed dataset.



	A	B	C	D	E
1		**midterm**	**final**	**project**	**final_grade**
2	**Mary**	83	94	77	84.45
3	**John**	83	71	54	67.2
4	**Bill**	86	88	72	81.1
5	**July**	100	65	80	79.75
6	**Paul**	78	95	94	90.35



MI: **Explain the statement in detail:**

df = pd.DataFrame(students).T

Is it for transpose?

Response:

Yes, that line of code is used to transpose the data in the "students" dictionary.

- The `pd.DataFrame(students)` function creates a DataFrame object from the "students" dictionary.
- The `.T` attribute is then used to transpose the data in the DataFrame, meaning the rows and columns are switched.

So the resulting DataFrame, `df`, will have the student names as the row indices, and the columns will be 'midterm', 'final', 'project', and 'final_grade' . This is done because the `to_excel` function in the next line writes the data in the DataFrame to an Excel file, and it would look better in the excel if student names were in the first column and their scores in the next columns.