Exemplar_Use Python syntax

January 7, 2024

Exemplar: Use Python syntax

Note: The following notebook shows the completed activity. This exemplar notebook includes all code that you were asked to complete in the prior activity. The purpose of this notebook is to act as a resource for you to compare your work. No action is required from you with this notebook. After reviewing this exemplar notebook, please move on to the next item.

0.1 Introduction

In this lab, you will practice Python syntax by creating variables, checking the data types of variables, creating precise variable names, and completing data conversions.

For this activity, imagine you're a data professional analyzing the users that log in to purchase tickets at a movie theater. You'll save the theater's unique identification code to a variable, study the number of tickets sold, and the data types you'll be working with.

Throughout this lab, you will practice assigning variables, viewing the values saved to the variables, and checking their data types.

0.2 Tips for completing this lab

As you navigate this lab, keep the following tips in mind:

- `### YOUR CODE HERE ###` indicates where you should write code. Be sure to replace this with your own code before running the code cell.
- Feel free to open the hints for additional guidance as you work on each task.
- To enter your answer to a question, double-click the markdown cell to edit. Be sure to replace the "[Double-click to enter your responses here.]" with your own answer.
- You can save your work manually by clicking File and then Save in the menu bar at the top of the notebook.
- You can download your work locally by clicking File and then Download and then specifying your preferred file format in the menu bar at the top of the notebook.

0.3 Task 1: Assign the theater ID to a variable

In your work as an analyst, imagine you are considering sales at a specific theater and you want to save the theater's unique identification, which is `"b79cn10k"`, to a variable. By doing this, you'll be able to access the ID easily if you need it later. In the following code cell:

1. Assign the theater's unique identification to a variable named `theater_id`.
2. Display the contents of the `theater_id` variable.

Be sure to replace each `### YOUR CODE HERE ###` with your own code before you run the following cell.

[1]:
```
# 1.
### YOUR CODE HERE ###
theater_id = "b79cn10k"

# 2.
### YOUR CODE HERE ###
print(theater_id)
```

```
b79cn10k
```

Hint 1

You can refer to what you've learned about variables and Python's `print()` function.

Hint 2

To assign a value to a variable, use an `=` operator, with the variable name to the left and the value to the right.

To display the contents of the variable, use the `print()` function.

Hint 3

Assign the value `"b79cn10k"` to the variable `theater_id`.

To display the contents of `theater_id`, call the `print()` function, and pass in `theater_id` as the argument.

0.4 Task 2: Create a ticket type variable

As you continue your work, you're provided a list of the different types of tickets sold by the theater: `"Senior"`, `"Adult"`, and `"Youth"`. In this task, focus on the `"Adult"` ticket type.

1. Create a variable called `ticket_type` and assign it a value of `"Adult"`.
2. Display the contents of the `ticket_type` variable.

[2]:
```
# 1.
### YOUR CODE HERE ###
ticket_type = "Adult"

# 2.
### YOUR CODE HERE ###
print(ticket_type)
```

```
Adult
```

Hint 1

You can refer to what you've learned about naming variables and using the Python `print()` function.

Hint 2

To assign a value to a variable in Python, place the name of the variable to the left of the `=` operator, and place the value to the right of the `=` operator.

Hint 3

To name the variable, make sure to place `ticket_type` to the left of the `=` operator. To assign it a value, place `"Adult"` to the right of the `=` operator.

To display the variable, make sure to call `print()` and pass in `ticket_type`.

0.5 Task 3: Save the number of adult tickets to a variable and check the data type

Now that you've recorded the types of tickets being sold, it's time to record the number of adult tickets sold so far.

1. Create a variable called `adult_tickets_sold` that represents the current number of "adult" tickets sold and assign it a value of `59.0`.
2. Check the data type of `adult_tickets_sold`.

[3]:
```
# 1.
### YOUR CODE HERE ###
adult_tickets_sold = 59.0

# 2.
### YOUR CODE HERE ###
type(adult_tickets_sold)
```

[3]:
```
float
```

Hint 1

Refer to what you learned about assigning variables.

Hint 2

To assign a value to a variable, place the name of the variable to the left of the `=` operator and the value to the right of the `=` operator.

To find the data type of a variable, pass the variable to the `type()` function.

Hint 3

Place an `=` operator between the variable name (`adult_tickets_sold`) and the value (`59`).

When calling `type()`, make sure to pass `adult_tickets_sold` as its argument.

0.6 Task 4: Convert data types and print the result

The data you work with as a data professional often needs to be converted to different data types. In this case, the number of tickets sold should be recorded as integers and not floats.

1. Convert the data saved in the `adult_tickets_sold` variable from a **float** to an **integer**. Reassign the result back to the `adult_tickets_sold` variable.
2. Check the data type of `adult_tickets_sold`.

[4]:
```
# 1.
### YOUR CODE HERE ###
adult_tickets_sold = int(adult_tickets_sold)

# 2.
### YOUR CODE HERE ###
type(adult_tickets_sold)
```

[4]:
```
int
```

Hint 1

Refer to what you learned about reassigning variables and checking data types using the `type()` function.

Hint 2

The `adult_tickets_sold` variable is assigned in the previous task, so make sure to complete Task 3 prior to Task 4.

To reassign a variable using the same name but a different data type, write the name of the variable to the left of the `=` operator, then write the data type function (such as `int()`) to the right of the `=` operator. The name of the function is written inside the type function parentheses.

To check the data type of the variable, use the `type()` function with the name of the variable in the parentheses.

Hint 3

- To convert `adult_tickets_sold` to an integer, pass it as an argument to the `int()` function. Reassign the result back to the `adult_tickets_sold` variable using the `=` operator.
- To display the data type, pass `adult_tickets_sold` to the `type()` function.

0.7 Conclusion

What are your key takeaways from this lab?

- There are many useful operators in Python that help you work with variables.
  - The `=` assignment operator allows you to assign or reassign a specific value to a variable.
- The `print()` function in Python allows you to display information.
  - It can take in a value directly or a variable that stores a value.
- The `type()` function in Python helps you to determine the data type of an object.

  - If you pass in a variable to `type()`, it will output the data type of the value stored in the variable.
- Python is able to convert some data types from one to another.
  - Using the `int()` function on a variable that stores data as a **float** will convert the data from a floating point number to an integer.

Congratulations! You've completed this lab. However, you may not notice a green check mark next to this item on Coursera's platform. Please continue your progress regardless of the check mark. Just click on the "save" icon at the top of this notebook to ensure your work has been logged.