# Activity_Assign Python variables

April 8, 2024

# 1 Activity: Assign Python variables

## 1.1 Introduction

Variables help security analysts to keep track of a variety of security-related information. For example, analysts may need to create Python variables for the users who are allowed to log in, the number of login attempts that they're permitted, and the current number of attempts that a user has made.

## 1.2 Scenario

You are a security analyst who is responsible for writing code that will automate analysis of login attempts made to a specific device. As the first step, you'll need to create variables to keep track of information relevant to the login process. This information includes the device ID, list of approved usernames, maximum login attempts allowed per user, current login attempts made by a user, and login status.

Throughout this lab, I'll assign these variables and check the data types of the variables.

## 1.3 Task

In my work as an analyst, I imagine there is a device only users specified on an allow list can access, and its device ID is `"72e08x0"`.

In the following code cell, assign this value to a variable named `device_id`. Then, display the contents of the variable and observe the output.

[1]:
```
# Assign the `device_id` variable to the device ID that only specified users␣
↪can access

device_id = "72e08x0"

# Display `device_id`

print(device_id)
```

```
72e08x0
```

### 1.4 Task

Now that the variable `device_id` is defined, you can return its data type.

In this task, I'll use a Python function to find the data type of the variable `device_id`. Store the data type in another variable called `device_id_type`. Then, display `device_id_type` to examine the output.

[2]:
```python
# Assign the `device_id` variable to the device ID that only specified users␣
↪can access

device_id = "72e08x0"

# Assign `device_id_type` to the data type of `device_id`

device_id_type = type(device_id)

# Display `device_id_type`

print(device_id_type)
```

```
<class 'str'>
```

**Question 1 Based on the output above, what do you observe about the data type of `device_id`?**

[The data type is string data.]

### 1.5 Task

As I continue my work, I'm provided a list of usernames of users who are allowed to access the device. The usernames with this access are `"madebowa"`, `"jnguyen"`, `"tbecker"`, `"nhersh"`, and `"redwards"`.

In this task, I'll create a variable called `username_list`. Assign a list with the approved usernames to this variable. Then, display the value of the `username_list` variable.

[3]:
```python
# Assign `username_list` to the list of usernames who are allowed to access the␣
↪device

username_list = ["madebowa", "jnguyen", "tbecker", "nhersh", "redwards"]

# Display `username_list`

print(username_list)
```

```
['madebowa', 'jnguyen', 'tbecker', 'nhersh', 'redwards']
```

To create a list in Python, use square brackets. Inside the square brackets, write the elements of the list, with a comma between elements and quotation marks for each element.

## 1.6 Task

In this task, I'll find the data type of the `username_list`. Store the type in a variable called `username_list_type`. Then, display `username_list_type` to examine the output.

[4]:
```
# Assign `username_list` to the list of usernames who are allowed to access the␣
↪device

username_list = ["madebowa", "jnguyen", "tbecker", "nhersh", "redwards"]

# Assign `username_list_type` to the data type of `username_list`

username_list_type = type(username_list)

# Display `username_list_type`

print(username_list_type)
```

```
<class 'list'>
```

The `type()` function allows you to get the data type of a given value. To get the data type of `username_list`, call the `type()` function and pass in `username_list`.

**Question 2 Based on the output above, what do you observe about the data type of `username_list`?**

[The data type is list data.]

## 1.7 Task

Now, I imagine that I've been informed that the previous list is not up-to-date and that there is another employee that now has access to the device. I'm given the updated list of usernames with access, including the new employee, as follows: `"madebowa"`, `"jnguyen"`, `"tbecker"`, `"nhersh"`, `"redwards"`, and `"lpope"`.

In this task, 'll reassign the variable `username_list` to the new list. Run the code to display the list before and after it's been updated to observe the difference.

[5]:
```
# Assign `username_list` to the list of usernames who are allowed to access the␣
↪device

username_list = ["madebowa", "jnguyen", "tbecker", "nhersh", "redwards"]

# Display `username_list`
```

```
print(username_list)

# Assign `username_list` to the updated list of usernames who are allowed to␣
↪access the device

username_list = ["madebowa", "jnguyen", "tbecker", "nhersh", "redwards",␣
↪"lpope"]

# Display `username_list`

print(username_list)
```

```
['madebowa', 'jnguyen', 'tbecker', 'nhersh', 'redwards']
['madebowa', 'jnguyen', 'tbecker', 'nhersh', 'redwards', 'lpope']
```

**Question 3 Based on the output above, what do you observe about the contents of `username_list`?**

[The variable username_list is reassigned. The second output is the updated list.]

### 1.8 Task

In this task, I define a variable called `max_logins` that represents the maximum number of login attempts allowed per user. Store the value `3` in this variable. Then, store its data type in another variable called `max_logins_type`. Display `max_logins_type` to examine the output.

[6]:
```
# Assign `max_logins` to the value 3

max_logins = 3

# Assign `max_logins_type` to the data type of `max_logins`

max_logins_type = type(max_logins)

# Display `max_logins_type`

print(max_logins_type)
```

```
<class 'int'>
```

When assigning numerical data(integer/float) to variables, The value is not placed between quotation marks.

**Question 4 Based on the output above, what do you observe about the data type of `max_logins`?**

[The data type is integer data.]

### 1.9 Task

In this task, I define a variable called `login_attempts` that represents the current number of login attempts made by a user. Store the value 2 in this variable. Then, store its data type in a variable called `login_attempts_type`. Display `login_attempts_type` to observe the output.

[7]:
```
# Assign `login_attempts` to the value 2

login_attempts = 2

# Assign `login_attempts_type` to the data type of `login_attempts`

login_attempts_type = type(login_attempts)

# Display `login_attempts_type`

print(login_attempts_type)
```

```
<class 'int'>
```

**Question 5 Based on the output above, what do you observe about the data type of `login_attempts`?**

[The data type is integer data]

### 1.10 Task

In this task, I'll determine the Boolean value that represents whether the current number of login attempts a user has made is less than or equal to the maximum number of login attempts allowed.

[8]:
```
# Assign `max_logins` to the value 3

max_logins = 3

# Assign `login_attempts` to the value 2

login_attempts = 2

# Determine whether the current number of login attempts a user has made is␣
 ↪less than or equal to the maximum number of login attempts allowed,
# and display the resulting Boolean value

print(max_logins <= login_attempts)
```

```
False
```

In Python, you can use comparison operators such as '==', '<', '>=' etc. These are used to return boolean values; either True or False.

Hint 2

To determine whether the current number of login attempts a user has made is less than or equal to the maximum number of login attempts allowed, use the `<=` operator. Place `login_attempts` to the left of the `<=` operator, and place `max_logins` to the right of the `<=` operator.

To make sure the resulting Boolean value is displayed, write this code inside of the parantheses where `print()` is called.

**Question 6 What is the output? What does this mean?**

[The output is False. The max_logins is greater than login_attempts.]

### 1.11 Task

This code continues to check for the Boolean value of whether `max_logins` is less than or equal to `login_attempts`. In this task, I'll reassign other values to `login_attempts`.

[10]:
```python
# Assign `max_logins` to the value 3

max_logins = 3

# Assign `login_attempts` to a specific value

login_attempts = 1

# Determine whether the current number of login attempts a user has made is␣
 ↪less than or equal to the maximum number of login attempts allowed,
# and display the resulting Boolean value

print(login_attempts <= max_logins)
```

```
True
```

**Question 7 Based on the different values you assigned to `login_attempts`, what did you observe about the output?**

[The output is True. The number of login attempts is less than max_logins.]

### 1.12 Task

One can also assign a Boolean value of `True` or `False` to a variable.

In this task, I'll create a variable called `login_status`, which is a Boolean that represents whether a user is logged in. Assign `False` to this variable and store its data type in a variable called `login_status_type` and display it.

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

[12]:
```python
# Assign `login_status` to the Boolean value False

login_status = False

# Assign `login_status_type` to the data type of `login_status`

login_status_type = type(login_status)

# Display `login_status_type`

print(login_status_type)
```

```
<class 'bool'>
```

Note that Boolean values should not have quotation marks around them in code.

**Question 8 Based on the output above, what do you observe about the data type of `login_status`?**

[The data type is Boolean data.]

## 1.13 Conclusion

### What are your key takeaways from this lab?

[Assigning and reassigning values to variables. One can assign string, integer, list or boolean values to a variable. The type of data in a variable can also be output using 'type()'. I learnt that one can also use operators to determine the boolean values of different data.]