LAB_Activity_CreateAConditionalStatement

April 8, 2024

1 Activity: Create a conditional statement

1.1 Introduction

Conditional statements are a powerful structure that help in achieving automation when you need to make sure conditions are met before certain actions are executed. Security analysts can for example use conditional statements in Python to check if users are approved to access a device.

1.2 Scenario

I'm working as a security analyst. First, I am responsible for checking whether a user's operating system requires an update. Then, I need to investigate login attempts to a specific device. I must determine if login attempts were made by users approved to access this device and if the login attempts occurred during organization hours.

1.3 Task

I am asked to help automate the process of checking whether a user's operating system requires an update. Imagine that a user's device can be running one of the following operating systems: OS 1, OS 2, or OS 3. While OS 2 is up-to-date, OS 1 and OS 3 are not. My task is to check whether the user's system is up-to-date, and if it is, display a message accordingly. To do this, I'll write a conditional statement using the keyword `if`.

[1]:
```python
# Assign a variable named `system` to a specific operating system, represented␣
↪as a string
# This variable indicates which operating system is running
# Feel free to run this cell multiple times; each time try assigning `system`␣
↪to different values ("OS 1", "OS 2", "OS 3") and observe the result

system = "OS 2"

# If OS 2 is running, then display a "no update needed" message

if system == "OS 2":
    print("No update needed")
```

```
No update needed
```

The `==` comparison operator is used to mean 'is equal to'. The colon is always added at the end of a conditional statement header.

1.4 Task

I now assign the `system` variable to different values (`"OS 1"`, `"OS 2"`, and `"OS 3"`), run the cell, and observe what happens.

[2]:
```
# Assign `system` to a specific operating system
# This variable represents which operating system is running
# Feel free to run this cell multiple times; each time try assigning `system`␣
↪to different values ("OS 1", "OS 2", "OS 3") and observe the result

system = "OS 1"

# If OS 2 is running, then display a "no update needed" message

if system == "OS 2":
    print("No update needed")
```

Question 1 What happens when OS 2 is running? What happens when OS 1 is running?

[When OS 2 is running, "No update needed" is output. When OS 1 is running, there is no output.]

1.5 Task

Nothing is displayed when the `system` is not equal to `"OS 2"` because the condition didn't evaluate to `True`. It would be beneficial if an alternative message is provided to them when updates are needed.

I will now add the appropriate keyword after the first conditional so that it will display a message that conveys that an update is needed when the `system` is not running OS 2. Then, I set the value of the `system` variable to indicate that OS 2 is running and run the cell. After observing what happens, set the value of `system` to indicate either that OS 1 is running or that OS 3 is running and run the cell.

[4]:
```
# Assign `system` to a specific operating system
# This variable represents which operating system is running

system = "OS 1"

# If OS 2 is running, then display a "no update needed" message
# Otherwise, display a "update needed" message
```

```python

if system == "OS 2":
    print("No update needed")
else:
    print("Update needed")

```

```
Update needed
```

Hint 1

Use the `else` keyword.

Question 2 In this setup what happens when OS 2 is running? And what happens when OS 2 is not running?

[When OS 2 is running, “No update needed” is the output. When OS 2 is not running, “Update needed” is output.]

1.6 Task

This setup is still not ideal because if the variable `system` contains a random string or integer, the conditional above would still display `update needed`.

To improve the conditional, I need to add the `elif` keyword. In the following cell, I added two `elif` statements after the `if` statement. The first `elif` statement will display `update needed` if `system` is `"OS 1"`. The second `elif` statement will display the same message, if `system` is `"OS 3"`.

[6]:
```python
# Assign `system` to a specific operating system
# This variable represents which operating system is running

system = "OS 4"

# If OS 2 is running, then display a "no update needed" message
# Otherwise if OS 1 is running, display a "update needed" message
# Otherwise if OS 3 is running, display a "update needed" message

if system == "OS 2":
    print("No update needed")
elif system == "OS 1":
    print("Update needed")
elif system == "OS 3":
    print("Update needed")

```

Question 3 Under this setup what happens when OS 2 is running? What happens when OS 1 is running? What happens when OS 3 is running? What happens when

neither of those three operating systems are running?

[When OS 2 is running, “No update needed” is output. When OS 1 and OS 3 are running, “Update needed” is the output. When none of those systems are running, there is no output.]

1.7 Task

Writing code that is readable and concise is a best practice in programming. Conditionals can be written more concisely. I will use a logical operator to combine the two `elif` statements from the previous setup into one `elif` statement.

[13]:
```python
# Assign `system` to a specific operating system
# This variable represents which operating system is running

system = "OS 1"

# If OS 2 is running, then display a "no update needed" message
# Otherwise if either OS 1 or OS 3 is running, display a "update needed" message

if system == "OS 2":
    print("No update needed")
elif system == "OS 3" or "OS 1":
    print("Update needed")
```

```
Update needed
```

Question 4 What do you observe about this conditional?

[If the system is running OS 3 or OS 1, “Update needed” is the output.]

1.8 Task

Next I’ve been asked to investigate login attempts to a specific device. Only approved users should log on to this device. I start with two authorized users, stored in the variables `approved_user1` and `approved_user2`. I’ll need to write a conditional statement that compares those variables to a third variable, `username`. This will be the username of a specific user trying to log in.

[21]:
```python
# Assign `approved_user1` and `approved_user2` to usernames of approved users

approved_user1 = "elarson"
approved_user2 = "bmoreno"

# Assign `username` to the username of a specific user trying to log in

username = "dabaly"
```

```
# If the user trying to log in is among the approved users, then display a␣
↪message that they are approved to access this device
# Otherwise, display a message that they do not have access to this device

if username == approved_user1 or username == approved_user2:
    print("This user has access to this device.")
else:
    print("This user does not have access to this device.")
```

```
This user does not have access to this device.
```

1.9 Task

The number of approved users has now expanded to five. Rather than storing each of the approved users' usernames individually, it would be more concise to store them in an allow list called `approved_list`.

The `in` operator in Python can be used to determine whether a given value is an element of a sequence. Using the `in` operator in a condition can help check whether a specific username is part of a list of approved usernames. For example, in the code below, `username in approved_list` evaluates to `True` if the value of the `username` variable is included in `approved_list`.

[23]:
```
# Assign `approved_list` to a list of approved usernames

approved_list = ["elarson", "bmoreno", "tshah", "sgilmore", "eraab"]

# Assign `username` to the username of a specific user trying to log in

username = "dabaly"

# If the user trying to log in is among the approved users, then display a␣
↪message that they are approved to access this device
# Otherwise, display a message that they do not have access to this device

if username in approved_list:
    print("This user has access to this device.")
else:
    print("This user does not have access to this device.")
```

```
This user does not have access to this device.
```

Question 5 What happens when an approved user tries to log in? What happens when an unapproved user tries to log in?

[When an approved user tries to log in, “This user has access to this device” is displayed. When an unapproved user tries to log in, “This user does not have access to this device” is displayed.]

1.10 Task

Now I’ll write another conditional statement. This one will use a `organization_hours` variable to check if the user logged in during specific organization hours. When that condition is met, the code should display the string `"Login attempt made during organization hours."`. When that condition isn’t met, the code should display the string `"Login attempt made outside of organization hours."`.

The `organization_hours` variable will have a Boolean data type. If `organization_hours` has a Boolean value of `True`, that means the user is logged in during the specified organization hours. If `organization_hours` has a Boolean value of `False`, that means the user is not logged in during those hours.

[25]:
```python
# Assign `organization_hours` to a Boolean value that represents whether the␣
↪user is trying to log in during organization hours

organization_hours = False

# If the entered `organization_hours` has a value of True, then display "Login␣
↪attempt made during organization hours."
# Otherwise, display "Login attempt made outside of organization hours."

if organization_hours == True:
    print("Login attempt made during organization hours.")
else:
    print("Login attempt made outside of organization hours.")
```

```
Login attempt made outside of organization hours.
```

Question 6 What happens when the user logs in during organization hours? What happens when they log in outside of organization hours?

[When user logs in during organisation hours, “Login attempt made during organization hours” is displayed. When a user logs in outside of organisation hours, “Login attempt made outside of organization hours” is displayed.]

1.11 Task

The following cell assembles the code from the previous tasks. It includes the conditional statement that checks if a user is on the allow list and the conditional statement that checks if the user logged in during organization hours.

[26]:
```
# Assign `approved_list` to a list of approved usernames

approved_list = ["elarson", "bmoreno", "tshah", "sgilmore", "eraab"]

# Assign `username` to the username of a specific user trying to log in

username = "bmoreno"

# If the user trying to log in is among the approved users, then display a␣
↪message that they are approved to access this device
# Otherwise, display a message that they do not have access to this device

if username in approved_list:
    print("This user has access to this device.")
else:
    print("This user does not have access to this device.")

# Assign `organization_hours` to a Boolean value that represents whether the␣
↪user is trying to log in during organization hours

organization_hours = True

# If the entered `organization_hours` has a value of True, then display "Login␣
↪attempt made during organization hours."
# Otherwise, display "Login attempt made outside of organization hours."

if organization_hours == True:
    print("Login attempt made during organization hours.")
else:
    print("Login attempt made outside of organization hours.")
```

```
This user has access to this device.
Login attempt made during organization hours.
```

[The code runs perfectly.]

1.12 Task

You can also provide a single message about the login attempt. To do this, I will join both conditions into a single conditional statement using a logical operator. This will make the code more concise.

[28]:
```
# Assign `approved_list` to a list of approved usernames

approved_list = ["elarson", "bmoreno", "tshah", "sgilmore", "eraab"]

# Assign `username` to the username of a specific user trying to log in
```

```
username = "bmoreno"

# Assign `organization_hours` to a Boolean value that represents whether the␣
↪user is trying to log in during organization hours

organization_hours = True

# If the user is among the approved users and they are logging in during␣
↪organization hours, then convey that the user is logged in
# Otherwise, convey that either the username is not approved or the login␣
↪attempt was made outside of organization hours

if username in approved_list and organization_hours == True:
    print("Login attempt made by an approved user during organization hours.")
else:
    print("Username not approved or login attempt made outside of organization␣
↪hours.")
```

```
Login attempt made by an approved user during organization hours.
```

1.13 Conclusion

What are your key takeaways from this lab?

[I learnt to write conditional statements using the 'if, else and elif' functions. I also learnt to combine conditional statements using logical operators. This helps to make the code more concise and improve readability. Conditionals are useful in automating tasks.]