Create a Conditional Statement 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. For example, security analysts can use conditional statements in Python to check if users are approved to access a device.

In this lab, you will practice writing conditional statements in Python.Scenario

You're working as a security analyst. First, you are responsible for checking whether a user's operating system requires an update. Then, you need to investigate login attempts to a specific device. You must determine if login attempts were made by users approved to access this device and if the login attempts occurred during organization hours.-----Task 1: Basic `if` Statement

Automate the process of checking whether a user's operating system requires an update. Imagine 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. Your task is to check whether the user's system is up-to-date, and if it is, display a message accordingly using the keyword `if`.

```
# Assign a variable named `system` to a specific operating system, represented as a string
# This variable indicates which operating system is running
system = "OS 2"

# If OS 2 is running, then display a "no update needed" message
if system == "OS 2":
 print("no update needed")
```

Output:

```
no update needed
```

-----Task 2: Testing the `if` Statement

Try assigning the `system` variable to different values (`"OS 1"`, `"OS 2"`, and `"OS 3"`) and observe what happens with the conditional statement from Task 1.

```
# Assign `system` to a specific operating system
system = "OS 1"

# If OS 2 is running, then display a "no update needed" message
if system == "OS 2":
 print("no update needed")
```

Output: (None, if system is "OS 1" or "OS 3")Question 1:

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

Answer: When OS 2 is running, `no update needed` is displayed. When OS 1 is running, nothing is displayed. The `print` statement is only executed when the condition `system == "OS 2"` evaluates to `True`.-----Task 3: Introducing `else`

It would be beneficial if an alternative message is provided when updates are needed. 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.

```
# Assign `system` to a specific operating system
system = "OS 3"

# If OS 2 is running, then display a "no update needed" message
# Otherwise, display a "update needed" message
if system == "OS 2":
 print("no update needed")
else:
 print("update needed")
```

Output:

```
update needed
```

Question 2:

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

Answer: In this setup, when OS 2 is running, `no update needed` is displayed. And when OS 2 is not running, `update needed` is displayed.-----Task 4: Improving with `elif`

The current setup is not ideal; if `system` contains a random string, it would still display `update needed`. To improve this, add two `elif` statements to specifically check for "OS 1" and "OS 3" and display `update needed`.

```
# Assign `system` to a specific operating system
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")
```
Output: (None, if system is "OS 4")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?

Answer: Under this setup, when OS 2 is running, `no update needed` is displayed. When either OS 1 or OS 3 is running, `update needed` is displayed. When neither of those three operating systems are running, nothing is displayed.-----Task 5: Conciseness with Logical Operators (`or`)

Writing readable and concise code is a best practice. Use a logical operator to combine the two `elif` statements from the previous setup into one.

```
# Assign `system` to a specific operating system
system = "OS 4"

# 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 1" or system == "OS 3":
 print("update needed")
```
Output: (None, if system is "OS 4")Question 4:

What do you observe about this conditional?

Answer: This conditional behaves the same way as the previous conditional. The only difference is that the syntax in this conditional is more concise. The use of the `or` operator allows you to combine the two conditions into one `elif` statement, which is more concise than having the two separate `elif` statements that were written previously.-----Task 6: Checking for Approved Users

Now, investigate login attempts. Write a conditional statement that checks if a `username` trying to log in is one of two authorized users (`approved_user1` or `approved_user2`).

```
# 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 = "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 == 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.")
```

Output:

```
This user has access to this device.
```

-----Task 7: Using the `in` Operator with a List

The number of approved users has expanded to five. Store them in an allow list called `approved_list` and use the `in` operator to check if a specific username is part of the list.

```
# 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 = "jhill"

# 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.")
```

Output:

```
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?

Answer: 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.-----Task 8: Checking Organization Hours with Boolean

Write a conditional statement that uses a Boolean variable `organization_hours` to check if the user logged in during specified organization hours.

```
# 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.")
```

Output:

```
Login attempt made during organization hours.
```

Question 6:

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

Answer: When the user logs in during organization hours, the condition in the `if` statement evaluates to `True`, and a message about the login attempt during organization hours is displayed. When the user logs in outside of organization hours, the condition in the `if` statement evaluates to `False`. This means the `else` statement is executed, and a message about the login attempt outside of organization hours is displayed.-----Task 9: Assembling Multiple Conditional Statements

Combine the code from the previous tasks to check for both approved users and organization hours.

```
# 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.")
```

Output:

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

Question 7:

What happens when the user trying to log in is not among the approved users? What happens when the user trying to log in is among the approved users? What happens when the user tries to log in outside of organization hours?

Answer: When the user trying to log in is not among the approved users, a message is displayed about the user not having access to the device. The code then goes on to check whether the user attempted to log in during organization hours. When the user trying to log in is among the approved users, a message is displayed about the user having access to the device and then the code checks whether they attempted to log in during organization hours. When the user trying to log in is doing so outside of organization hours, a message is displayed about the login attempt outside of

organization hours.-----Task 10: Single Conditional Statement with Logical Operator (`and`)

Join both conditions (approved user AND during organization hours) into a single, more concise conditional statement using a logical operator.

```
# 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.")
```

Output:

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

Question 8:

In this setup, what happens when the user trying to log in is an approved user and doing so during organization hours? What happens when the user either is not approved or attempts to log in outside of organization hours?

Answer: In this setup, when the user is approved and attempts to log in during organization hours, the message `"Login attempt made by an approved user during organization hours."` is displayed. If the user is either not approved or attempts to log in outside of organization hours, the message `"Username not approved or login attempt made outside of organization hours."` is displayed. This code checks the same conditions as the code in the previous task. However, it joins them into one