Developing An Algorithm Using Python

Case Study:

You're working as a security analyst and you're responsible for developing an algorithm that connects users to their assigned devices. You'll write code that indicates if a user is approved on the system and has brought their assigned device to the security team.

Task 1

There's a new employee joining the organization, and they need to be provided with a username and device ID. In the following code cell, you are given a username and device ID of this new user, stored in the variables `new_user` and `new_device`, respectively. Use the `.append()` method to add these variables to the `approved_users` and `approved_devices` respectively. Afterwards, display the `approved_users` and `approved_devices` variables to confirm the added information.

```
# Assign `approved_users` to a list of approved usernames

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

# Assign `approved_devices` to a list of device IDs that correspond to the usernames in `approved_users`

approved_devices = ["8rp2k75", "hl0s5o1", "2ye3lzg", "4n482ts", "a307vir"]

# Assign `new_user` to the username of a new approved user

new_user = "gesparza"

# Assign `new_device` to the device ID of the new approved user

new_device = "3rcv4w6"

# Add that user's username and device ID to `approved_users` and `approved_devices` respectively

approved_users.append(new_user)
approved_devices.append(new_device)

# Display the contents of `approved_users`

print(approved_users)

# Diplay the contents of `approved_devices`

print(approved_devices)
```

```
['elarson', 'bmoreno', 'tshah', 'sgilmore', 'eraab', 'gesparza']
['8rp2k75', 'hl0s5o1', '2ye3lzg', '4n482ts', 'a307vir', '3rcv4w6']
```

Task 2

An employee has left the team and should no longer have access to the system. In the following code cell, you are given the username and device ID of the user to be removed, stored in the variables `removed_user` and `removed_device` respectively. Use the `.remove()` method to remove each of these elements from the corresponding list. Afterwards, display both the `approved_users` and the `approved_devices` variables to view the removed users.

```
# Assign `approved_users` to a list of approved usernames

approved_users = ["elarson", "bmoreno", "tshah", "sgilmore", "eraab", "gesparza"]

# Assign `approved_devices` to a list of device IDs that correspond to the usernames in `approved_users`

approved_devices = ["8rp2k75", "hl0s5o1", "2ye3lzg", "4n482ts", "a307vir", "3rcv4w6"]

# Assign `removed_user` to the username of the employee who has left the team

removed_user = "tshah"

# Assign `removed_device` to the device ID of the employee who has left the team

removed_device = "2ye3lzg"

# Remove that employee's username and device ID from `approved_users` and `approved_devices` respectively

approved_users.remove(removed_user)
approved_devices.remove(removed_device)

# Display `approved_users`

print(approved_users)

# Diplay `approved_devices`

print(approved_devices)
```

```
['elarson', 'bmoreno', 'sgilmore', 'eraab', 'gesparza']
```

Task 3

As part of verifying a user's identity in the system, you'll need to check if the user is one of the approved users. Write a conditional statement that verifies if a given username is an element of the list of approved usernames. If it is, display `"The user ______ is approved to access the system."`. Otherwise, display `"The user ______ is not approved to access the system."`

```
# Assign `approved_users` to a list of approved usernames

approved_users = ["elarson", "bmoreno", "sgilmore", "eraab", "gesparza"]

# Assign `approved_devices` to a list of device IDs that correspond to the usernames in `approved_users

approved_devices = ["8rp2k75", "hl0s5o1", "4n482ts", "a307vir", "3rcv4w6"]

# Assign `username` to a username

username = "sgilmore"

# Conditional statement
# If `username` belongs to `approved_users`, then display "The user _____ is approved to access the sy
# Otherwise display "The user _____ is not approved to access the system."
if username in approved_users:
    print("The username", username, "is approved to access the system.")
else:
    print("The username", username, "is not approved to access the system.")
```

```
The username sgilmore is approved to access the system.
```

Task 4

The next part of the algorithm uses the `.index()` method to find the index of `username` in the `approved_list` and store that index in a variable named `ind`.

When used on a list, the `.index()` method will return the position of the given value in the list.

Add a statement to display `ind` in the following code cell to explore the value it contains.

```
# Assign `approved_users` to a list of approved usernames
approved_users = ["elarson", "bmoreno", "sgilmore", "eraab", "gesparza"]
# Assign `approved_devices` to a list of device IDs that correspond to the usernames in `approved_users`
approved_devices = ["8rp2k75", "hl0s5o1", "4n482ts", "a307vir", "3rcv4w6"]
# Assign `username` to a username
username = "sgilmore"
# Assign `ind` to the index of `username` in `approved_users`
ind = approved_users.index(username)
# Display the value of `ind`
print(ind)
```

```
2
```

Task 5

This task will allow you to build your understanding of list operations for the algorithm that you'll eventually build. It will demonstrate how you can find an index in one list and then use this index to display connected information in another list. First, use the `.index()` method again to find the index of `username` in the `approved_users` and store that in a variable named `ind`. Then, connect `ind` to the `approved_devices` and display the device ID located at the index `ind`. Afterwards, run the cell to observe the result.

```
# Assign `approved_users` to a list of approved usernames
approved_users = ["elarson", "bmoreno", "sgilmore", "eraab", "gesparza"]
# Assign `approved_devices` to a list of device IDs that correspond to the usernames in `approved_users`
approved_devices = ["8rp2k75", "hl0s5o1", "4n482ts", "a307vir", "3rcv4w6"]
# Assign `username` to a username
username = "sgilmore"
# Assign `ind` to the index of `username` in `approved_users`
ind = approved_users.index(username)
# Display the device ID at the index that matches the value of `ind` in `approved_devices`
print(approved_devices[ind])
```

```
4n482ts
```

Task 6

Your next step in creating the algorithm is to determine if a username and device ID correspond. To do this, write a conditional that checks if the `username` is an element of the `approved_devices` and if the `device_id` stored at the same index as `username` matches the `device_id` entered. You'll use the logical operator `and` to connect the two conditions. When both conditions evaluate to `True`, display a message that the username is approved and another message that the user has their assigned device.

```
# Assign `approved_users` to a list of approved usernames

approved_users = ["elarson", "bmoreno", "sgilmore", "eraab", "gesparza"]

# Assign `approved_devices` to a list of device IDs that correspond to the usernames in `appr

approved_devices = ["8rp2k75", "hl0s5o1", "4n482ts", "a307vir", "3rcv4w6"]

# Assign `username` to a username

username = "sgilmore"

# Assign `device_id` to a device ID

device_id = "4n482ts"

# Assign `ind` to the index of `username` in `approved_users`

ind = approved_users.index(username)

# Conditional statement
# If `username` belongs to `approved_users`, and if the device ID at `ind` in `approved_devic
# then display a message that the username is approved,
# followed by a message that the user has the correct device

if username in approved_users and device_id == approved_devices[ind]:
    print("The username", username, "is approved to access the system.")
    print(device_id, "is the assigned device for", username)
```

```
The username sgilmore is approved to access the system.
4n482ts is the assigned device for sgilmore
```

Task 7

It would also be helpful for users to receive messages when their username is not approved or their device ID is incorrect.

Add to the code by writing an `elif` statement. This `elif` statement should run when the `username` is part of the `approved_users` but the `device_id` doesn't match the corresponding device ID in the `approved_devices`. The statement should also display two messages conveying that information.

```
# Assign `approved_users` to a list of approved usernames

approved_users = ["elarson", "bmoreno", "sgilmore", "eraab", "gesparza"]

# Assign `approved_devices` to a list of device IDs that correspond to the usernames in `approved_users`

approved_devices = ["8rp2k75", "hl0s5o1", "4n482ts", "a307vir", "3rcv4w6"]

# Assign `username` to a username

username = "sgilmore"

# Assign `device_id` to a device ID

device_id = "4n482ts"

# Assign `ind` to the index of `username` in `approved_users`

ind = approved_users.index(username)

# If statement
# If `username` belongs to `approved_users`, and if the element at `ind` in `approved_devices` matches `device_id`,
# then display a message that the username is approved,
# followed by a message that the user has the correct device

if username in approved_users and device_id == approved_devices[ind]:
    print("The user", username, "is approved to access the system.")
    print(device_id, "is the assigned device for", username)

# Elif statement
# Handles the case when `username` belongs to `approved_users` but element at `ind` in `approved_devices` does not m
```

```
# and displays two messages accordingly

elif username in approved_users and device_id != approved_devices[ind]:
    print("The user", username, "is approved to access the system, but", device_id, "is not their assigned device.")
```

```
The user sgilmore is approved to access the system.
4n482ts is the assigned device for sgilmore
```

Task 8

In this task, you'll complete your algorithm by developing a function that uses some of the code you've written in earlier tasks. This will automate the login process.

There are multiple ways to use conditionals to automate the login process. In the following code, a nested conditional is used to achieve the goals of the

algorithm. There is a conditional statement inside of another conditional statement. The outer conditional handles the case when the `username` is approved and the case when `username` is not approved. The inner conditional, which is placed inside the first `if` statement, handles the case when the `username` is approved and the `device_id` is correct, as well as the case when the `username` is approved and the `device_id` is incorrect.

To complete this task, you must define a function named `login` that takes in two parameters, `username` and `device_id`. Afterwards, call the function and pass in different username and device ID combinations to experiment and observe the function's behavior.

```
:
# Assign `approved_users` to a list of approved usernames

approved_users = ["elarson", "bmoreno", "sgilmore", "eraab", "gesparza"]

# Assign `approved_devices` to a list of device IDs that correspond to the usernames in `approved_users`

approved_devices = ["8rp2k75", "hl0s5o1", "4n482ts", "a307vir", "3rcv4w6"]

# Define a function named `login` that takes in two parameters, `username` and `device_id`

def login(username, device_id):

    # If `username` belongs to `approved_users`,

    if username in approved_users:

        # then display "The user _____ is approved to access the system.",

        print("The user", username, "is approved to access the system.")

        # assign `ind` to the index of `username` in `approved_users`,

        ind = approved_users.index(username)

        # and execute the following conditional
        # If `device_id` matches the element at the index `ind` in `approved_devices`,

        if device_id == approved_devices[ind]:

          # then display "_____ is the assigned device for _____"

          print(device_id, "is the assigned device for", username)
```

```
        # Otherwise,

        else:

          # display "_____ is not their assigned device"

          print(device_id, "is not their assigned device.")

    # Otherwise (part of the outer conditional and handles the case when `username` does not belong to `approved_use

    else:

        # Display "The user _____ is not approved to access the system."

        print("The username", username, "is not approved to access the system.")

# Call the function you just defined to experiment with different username and device_id combinations

login("bmoreno", "hl0s5o1")
login("elarson", "r2s5r9g")
login("abernard", "4n482ts")
```

```
The user bmoreno is approved to access the system.
hl0s5o1 is the assigned device for bmoreno
The user elarson is approved to access the system.
r2s5r9g is not their assigned device.
The username abernard is not approved to access the system.
```