**Activity - Debug Python Code**

**Introduction**

One of the biggest challenges faced by analysts is ensuring that automated processes run smoothly. Debugging is an important practice that security analysts incorporate in their work to identify errors in code and resolve them so that the code achieves the desired outcome.

Through a series of tasks in this lab, you'll develop and apply your debugging skills in Python.

**Tips for completing this lab**

**Scenario**

In your work as a security analyst, you need to apply debugging strategies to ensure your code works properly.

Throughout this lab, you'll work with code that is similar to what you've written before, but now it has some errors that need to be fixed. You'll need to read code cells, run them, identify the errors, and adjust the code to resolve the errors.

**Task 1**

The following code cell contains a syntax error. In this task, you'll run the code, identify why the error is occuring, and modify the code to resolve it. (To ensure that it has been resolved, run the code again to check if it now functions properly.)

*# For loop that iterates over a range of numbers*

*# and displays a message each iteration*

​

**for** i **in** range(10):

print("Connection cannot be established")

Connection cannot be established

Connection cannot be established

Connection cannot be established

Connection cannot be established

Connection cannot be established

Connection cannot be established

Connection cannot be established

Connection cannot be established

Connection cannot be established

Connection cannot be established

**Hint 1**

**Hint 2**

**Question 1**

**What happens when you run the code before modifying it? How can you fix this?**

[Double-click to enter your responses here.]

**Task 2**

In the following code cell, you're provided a list of usernames. There is an issue with the syntax. In this task, you'll run the cell, observe what happens, and modify the code to fix the issue.

*# Assign `usernames\_list` to a list of usernames*

​

usernames\_list **=** ["djames", "jpark", "tbailey", "zdutchma", "esmith", "srobinso", "dcoleman", "fbautist"]

​

*# Display `usernames\_list`*

​

print(usernames\_list)

['djames', 'jpark', 'tbailey', 'zdutchma', 'esmith', 'srobinso', 'dcoleman', 'fbautist']

**Hint 1**

**Hint 2**

**Question 2**

**What happens when you run the code before modifying it? How can you fix it?**

[Double-click to enter your responses here.]

**Task 3**

In the following code cell, there is a syntax error. Your task is to run the cell, identify what is causing the error, and fix it.

*# Display a message in upper case*

​

print("update needed".upper())

UPDATE NEEDED

**Hint 1**

**Hint 2**

**Question 3**

**What happens when you run the code before modifying it? What is causing the syntax error? How can you fix it?**

[Double-click to enter your responses here.]

**Task 4**

In the following code cell, you're provided a usernames\_list, a username, and code that determines whether the username is approved. There are two syntax errors and one exception. Your task is to find them and fix the code. A helpful debugging strategy is to focus on one error at a time and run the code after fixing each one.

*# Assign `usernames\_list` to a list of usernames that represent approved users*

​

usernames\_list **=** ["djames", "jpark", "tbailey", "zdutchma", "esmith", "srobinso", "dcoleman", "fbautist"]

​

*# Assign `username` to a specific username*

​

username **=** "esmith"

​

*# For loop that iterates over the elements of `usernames\_list` and determines whether each element corresponds to an approved user*

​

**for** name **in** usernames\_list:

​

*# Check if `name` matches `username`*

*# If it does match, then display a message accordingly*

​

**if** name **==** username:

print("The user is an approved user")

The user is an approved user

**Hint 1**

**Hint 2**

**Hint 3**

**Question 4**

**What happens when you run the code before modifying it? What is causing the errors? How can you fix it?**

[Double-click to enter your responses here.]

**Task 5**

In this task, you'll examine the following code and identify the type of error that occurs. Then, you'll adjust the code to fix the error.

*# Assign `usernames\_list` to a list of usernames*

​

usernames\_list **=** ["elarson", "bmoreno", "tshah", "sgilmore", "eraab"]

​

*# Assign `username` to a specific username*

​

username **=** "eraab"

​

*# Determine whether `username` is the final username in `usernames\_list`*

*# If it is, then display a message accordingly*

​

**if** username **==** usernames\_list[4]:

print("This username is the final one in the list.")

This username is the final one in the list.

**Hint 1**

**Hint 2**

**Hint 3**

**Question 5**

**What happens when you run the code before modifying it? What type of error is this? How can you fix it?**

[Double-click to enter your responses here.]

**Task 6**

In this task, you'll examine the following code. The code imports a text file into Python, reads its contents, and stores the contents as a list in a variable named ip\_addresses. It then removes elements from ip\_addresses if they are in remove\_list. There are two errors in the code: first a syntax error and then an exception related to a string method. Your goal is to find these errors and fix them.

*# Assign `import\_file` to the name of the text file*

​

import\_file **=** "allow\_list.txt"

​

*# Assign `remove\_list` to a list of IP addressess that are no longer allowed to access the network*

​

remove\_list **=** ["192.168.97.225", "192.168.158.170", "192.168.201.40", "192.168.58.57"]

​

*# With statement that reads in the text file and stores its contents as a list in `ip\_addresses`*

​

**with** open(import\_file, "r") **as** file:

ip\_addresses **=** file.read()

​

*# Convert `ip\_addresses` from a string to a list*

​

ip\_addresses **=** ip\_addresses.split()

​

*# For loop that iterates over the elements in `remove\_list`,*

*# checks if each element is in `ip\_addresses`,*

*# and removes each element that corresponds to an IP address that is no longer allowed*

​

**for** element **in** remove\_list:

**if** element **in** ip\_addresses:

ip\_addresses.remove(element)

​

*# Display `ip\_addresses` after the removal process*

​

print(ip\_addresses)

['ip\_address', '192.168.25.60', '192.168.205.12', '192.168.6.9', '192.168.52.90', '192.168.90.124', '192.168.186.176', '192.168.133.188', '192.168.203.198', '192.168.218.219', '192.168.52.37', '192.168.156.224', '192.168.60.153', '192.168.69.116']

**Hint 1**

**Hint 2**

**Question 6**

**What happens when you run the code before modifying it? What is causing the errors? How can you fix them?**

[Double-click to enter your responses here.]

**Task 7**

In this final task, there are three operating systems: OS 1, OS 2, and OS 3. Each operating system needs a security patch by a specific date. The patch date for OS 1 is "March 1st", the patch date for OS 2 is "April 1st", and the patch date for OS 3 is "May 1st".

The following code stores one of these operating systems in a variable named system. Then, it uses conditionals to output the patch date for this operating system.

However, this code has logic errors. Your goal is to assign the system variable to different values, run the code to examine the output, identify the error, and fix it.

*# Assign `system` to a specific operating system as a string*

​

system **=** "OS 2"

​

*# Assign `patch\_schedule` to a list of patch dates in order of operating system*

​

patch\_schedule **=** ["March 1st", "April 1st", "May 1st"]

​

*# Conditional statement that checks which operating system is stored in `system` and displays a message showing the corresponding patch date*

​

**if** system **==** "OS 1":

print("Patch date:", patch\_schedule[2])

​

**elif** system **==** "OS 2":

print("Patch date:", patch\_schedule[0])

​

**elif** system **==** "OS 3":

print("Patch date:", patch\_schedule[2])

Patch date: March 1st

**Hint 1**

**Hint 2**

**Hint 3**

**Question 7**

**What happens when you run the code before modifying it? What is causing the logic errors? How can you fix them?**

[Double-click to enter your responses here.]

**Conclusion**

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

[Double-click to enter your responses here.]