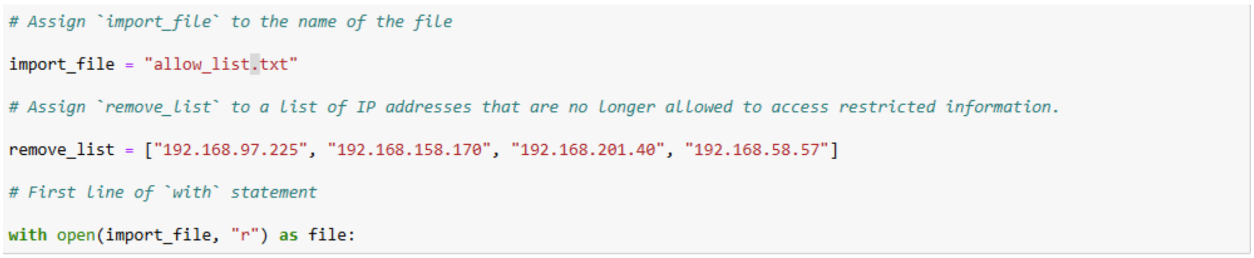
Algorithm for file updates in Python

Project description

In this project, the goal was to update an allow list of IP addresses by removing unwanted addresses. The project involves reading an existing text file, processing the data, and then updating the file with a revised list of IP addresses. Below are the steps taken to complete the task using Python.

Open the file that contains the allow list

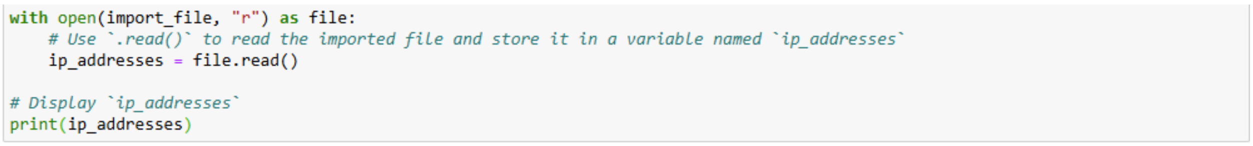
In this task, the file containing the list of allowed IP addresses is opened using the open() function with the "r" parameter to read the contents.



```
# Assign `import_file` to the name of the file

import_file = "allow_list.txt"

# Assign `remove_list` to a list of IP addresses that are no longer allowed to access restricted information.

remove_list = ["192.168.97.225", "192.168.158.170", "192.168.201.40", "192.168.58.57"]

# First line of `with` statement

with open(import_file, "r") as file:
```

Read the file contents

The file contents are read using the .read() method and stored in the ip_addresses variable. The data is displayed to verify its current format.



```
with open(import_file, "r") as file:
    # Use `.read()` to read the imported file and store it in a variable named `ip_addresses`
    ip_addresses = file.read()

# Display `ip_addresses`
print(ip_addresses)
```

Convert the string into a list

To work with the data in a more manageable form, the string containing all IP addresses is split into a list using the .split() method.

```
# Use `.split()` to convert `ip_addresses` from a string to a list
ip_addresses = ip_addresses.split()  # Split by whitespace to create a list of IPs

# Display `ip_addresses` to verify the update took place
print(ip_addresses)
```

Iterate through the remove list

The program then loops through the remove_list and checks each IP address in the ip_addresses list to determine if it should be removed.

```
# Build iterative statement
# Name loop variable `element`
# Loop through `ip_addresses`

for element in ip_addresses:
    # Display `element` in every iteration
    print(element)
```

Remove IP addresses that are on the remove list

In each iteration, a conditional check is used to determine if the current IP address should be removed. If it is found in the remove_list, the .remove() method is applied to remove that IP address.

```
for element in ip_addresses:

  # Build conditional statement
  # If current element is in `remove_list`,

    if element in remove_list:

        # then current element should be removed from `ip_addresses`

        ip_addresses.remove(element)

# Display `ip_addresses`

print(ip_addresses)
```

Update the file with the revised list of IP addresses

After removing the unwanted IPs, the list is converted back into a string using .join() so it can be written back to the file.

```
ip_addresses = "\n".join(ip_addresses)

# Build `with` statement to rewrite the original file
with open(import_file, "w") as file:
    # Rewrite the file, replacing its contents with the updated `ip_addresses`
    file.write(ip_addresses)
```

Summary

This Python project automates the process of updating an IP address allow list by removing specified addresses. It reads an input file, processes the list, and then writes the updated list back into the original file. The solution involves reading, iterating, modifying, and writing data efficiently using Python's built-in methods such as .read(), .split(), .remove(), and .join(). The project demonstrates how to manage and update data in text files using basic Python functionalities.

Italo B.