Update File Through Python Algorithm

Project Description

The Python script addresses the need for updating a file, specifically an allow list file containing IP addresses. The scenario involves removing certain IP addresses specified in the **'remove_list'** from the original file (**'allow_list.txt'**). The script utilizes file handling and string manipulation techniques to achieve this. It begins by reading the initial contents of the file, converting them into a list of IP addresses, and then iteratively removing addresses that match those in the **'remove_list'**. The modified list is then converted back into a string and written back to the file. The final contents of the updated file are displayed as output, showcasing the effective removal of the specified IP addresses.

Python Code

```
# 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"]

# Display `import_file`

print(import_file)

# Display `remove_list`

print(remove_list)
```

```
allow_list.txt
['192.168.97.225', '192.168.158.170', '192.168.201.40', '192.168.58.57']
```

```
# Build `with` statement to read in the initial contents of the file

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)
```

```
192.168.6.9 192.168.90.124 192.168.133.188 192.168.218.219 192.168.156.224 192.168.69.116
```

```
# Use `.split()` to convert `ip_addresses` from a string to a list

ip_addresses = ip_addresses.split()

# Display `ip_addresses`

print(ip_addresses)
```

```
['192.168.6.9', '192.168.90.124', '192.168.133.188', '192.168.218.219', '192.168.156.224', '192.168.69.116']
```

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

for element in ip_addresses:

    # Display `element` in every iteration

    print(element)
```

```
192.168.6.9
192.168.90.124
192.168.133.188
192.168.218.219
192.168.156.224
192.168.69.116
```

```
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)
```

```
['192.168.6.9', '192.168.90.124', '192.168.133.188', '192.168.218.219', '192.168.156.224', '192.168.69.116']
```

```
# Convert `ip_addresses` back to a string so that it can be written into the text file

ip_addresses = " ".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 `ip_addresses`

  file.write(ip_addresses)

# Build `with` statement to read in the updated file

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

    # Read in the updated file and store the contents in `text`

    text = file.read()

# Display the contents of `text`

print(text)
```

```
192.168.6.9 192.168.90.124 192.168.133.188 192.168.218.219 192.168.156.224 192.168.69.116
```

```python
# Define a function named `update_file` that takes in two parameters: `import_file` and `remove_list`
# and combines the steps you've written in this lab leading up to this

def update_file(import_file, remove_list):

  # Build `with` statement to read in the initial contents of the file

  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()

  # Use `.split()` to convert `ip_addresses` from a string to a list

  ip_addresses = ip_addresses.split()

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

  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)

  # Convert `ip_addresses` back to a string so that it can be written into the text file

  ip_addresses = " ".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 `ip_addresses`

    file.write(ip_addresses)

# Call `update_file()` and pass in "allow_list.txt" and a list of IP addresses to be removed

update_file("allow_list.txt", ["192.168.25.60", "192.168.140.81", "192.168.203.198"])

# Build `with` statement to read in the updated file

with open("allow_list.txt", "r") as file:

  # Read in the updated file and store the contents in `text`

  text = file.read()

# Display the contents of `text`

print(text)
```

```
192.168.6.9 192.168.90.124 192.168.133.188 192.168.218.219 192.168.156.224 192.168.69.116
```

Summary

The algorithm first reads the contents of the specified file (**'allow_list.txt'**) into a string and converts it into a list of IP addresses. It then iterates through the list, removing any addresses that match those in the **'remove_list'**. Afterward, the updated list is converted back into a string and overwritten onto the original file. The key components include file reading and writing with the **with** statement, string manipulation using **.split()** and **.join()**, and a **for** loop with conditional statements for removing specific IP addresses. The modular design is further encapsulated in the **'update_file'** function, making it reusable for other scenarios. The provided code includes an example usage of the function, demonstrating its effectiveness in removing a new set of IP addresses from the file.