Updating a File Using a Python Algorithm

Project Description

In my organization, access to restricted content is managed using an allow list that contains authorized IP addresses. This file, named `"allow_list.txt"`, identifies these addresses. Another file, `"remove_list.txt"`, contains IP addresses that should no longer have access. I created an algorithm to automate updating the allow list by removing the IPs specified in the remove list.

Opening the Allow List File

The first step in the algorithm is to open the `"allow_list.txt"` file. First, I assigned the file name to a variable:

```
import_file = "allow_list.txt"
```

Then, I used a `with` statement to open the file in read mode:

```
with open(import_file, "r") as file:
  content = file.read()
```

The `with` statement ensures that resources are properly managed, automatically closing the file after reading it. The `open()` function takes two arguments: the first specifies the file name, and the second (`"r"`) indicates that the file should be opened in **read mode**. The file's content is then stored as a string in the `content` variable.

Reading and Converting the File into a List

Since the IP addresses are stored as raw text, I used the `.split()` method to convert them into a list:

```
ip_addresses = content.split()
```

The `.split()` function separates the string into a list of elements, using whitespace and newline characters as default delimiters. This makes it easier to manipulate individual IP addresses.

Iterating Through the Remove List

The algorithm must iterate over the IP addresses in `"remove_list.txt"` and remove them from `ip_addresses` if they are present. I used a `for` loop for this:

```
with open("remove_list.txt", "r") as file:
  remove_list = file.read().split()

for ip in remove_list:
  if ip in ip_addresses:
    ip_addresses.remove(ip)
```

The `"remove_list.txt"` file is opened in read mode, and its content is converted into a list using `.split()`. Then, a `for` loop checks if each IP address in `remove_list` exists in `ip_addresses`. If it does, it is removed using `.remove()`.

Updating the File with the Revised IP List

Once the unauthorized IP addresses are removed, the updated list needs to be written back to `"allow_list.txt"`. Before doing this, I converted the list back into a string using `.join()`:

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

This ensures that each IP address appears on a new line in the file.

Finally, I overwrote the existing `"allow_list.txt"` file using write mode (`"w"`):

```
with open(import_file, "w") as file:
  file.write(new_content)
```

The `"w"` mode replaces the file's existing content with the updated allow list.

Algorithm Summary

1. **Open** `"allow_list.txt"` and **read** its content.
2. **Convert** the content into a list for easier removal of IP addresses.
3. **Open** `"remove_list.txt"`, convert it into a list, and **remove** any matching IPs from `ip_addresses`.
4. **Rewrite** the updated list into `"allow_list.txt"`.

This algorithm ensures that only valid IP addresses remain in the allow list after each execution.