Algorithm for file updates in Python

Project description

As part of my job at the healthcare company, I regularly update a file that identifies employees with access to restricted content. This file is based on who works with personal patient records, and access is restricted based on IP addresses. I maintain an allow list of permitted IP addresses for the restricted subnetwork. Additionally, there's a remove list that specifies which employees should be removed from the allow list. My task involves creating a Python algorithm to check whether any IP addresses from the remove list appear in the allow list. If they do, I remove those IPs from the file containing the allow list.

Open the file that contains the allow list

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

- import_file = "allow_list.txt": This line assigns the string "allow_list.txt" to the variable import_file. It indicates that the file named "allow_list.txt" will be read.
- with open(import_file, "r") as file:: This line opens the file specified by the import_file variable in read mode ("r"). The with statement ensures that the file is properly closed after reading. The file object is assigned to the variable file.
- The indented block following this line represents the code that will execute while the file is open. In this case, it would typically involve reading the contents of the file line by line or performing some other operation.

Read the file contents

```
text = file.read()
```

- file.read(): This method reads the entire contents of the file that was opened using the with open(...) statement. It reads the file as a single string and assigns it to the variable text.

Convert the string into a list

```
text = text.split()
```

- text.split(): This method splits the string stored in the text variable into a list of substrings. By default, it splits the string at whitespace characters (spaces, tabs, and newlines). The resulting list contains each word from the original string.

Iterate through the remove list

```
for element in text:
        if element in remove_list:
```

- for element in text:: This line initiates a loop that iterates over each element (word) in the text list. The variable element takes on the value of each word during each iteration.
- if element in remove_list:: Within the loop, this conditional statement checks whether the current element (word) exists in the remove_list. If it does, the indented block of code (not shown in the snippet) will execute. Otherwise, it will skip to the next iteration.

Remove IP addresses that are on the remove list

```
text.remove(element)
```

- text.remove(element): This line attempts to remove the value of the element variable from the text list. If the value exists in the list, it will be removed; otherwise, it will raise an error.

Update the file with the revised list of IP addresses

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

- with open(import_file, "w") as file:: This line opens the file specified by the import_file variable in write mode ("w"). When a file is opened in write mode, any existing content is overwritten, and the file is ready to accept new data. The with statement ensures that the file is properly closed after writing. The file object is assigned to the variable file.
- file.write(text): Within the indented block, this line writes the contents of the text variable (which contains modified data) to the file. The existing content is replaced with the new content.

Summary

1. I provided a code snippet that reads from a file named “allow_list.txt” and assigned its content to the variable `text`.
2. I discussed each line of the code snippet, including opening the file, reading its content, splitting the text into words, and removing specific elements from the list.
3. Finally, I wrote the modified `text` back to the same file.