Algorithm for file updates in Python

Project description

You are a security professional working at a health care company. As part of your job, you're required to regularly update a file that identifies the employees who can access restricted content. The contents of the file are based on who is working with personal patient records. Your task is to create an algorithm that uses Python code to check whether the allow list contains any IP addresses identified on the remove list. If so, you should remove those IP addresses from the file containing the allow list.

Open the file that contains the allow list

We first need to reference the file we want to open, which is the "allow_list.txt":

```
import_file = "allow_list.txt" # Assign allow_list.txt to a variable
```

Then pass it through our open() function, the first parameter being the file we want to open, and the second parameter being the read-only format. we then assign the import_file to the file variable using the as argument:

```
with open(import_file, "r") as file: # Opening the allow list as read-only
```

Read the file contents

Then, with the file open, we will assign the file contents to a variable using the read() function:

```
ip_addresses = file.read() # Assigning the text file contents to a variable
```

Convert the string into a list

Now we need to convert the string to a list, to do this we will use the split() function:

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

Iterate through the remove list

We'll need to create a for loop that checks for matching IP addresses in ip_addresses and remove_list. In order to do this we will create a for loop using the iterator "element" that loops through remove_list and checks to see if the "element" iterator matches with text in ip_addresses:

```
for element in remove_list: # Loops through the remove list
```

Remove IP addresses that are on the remove list

If any matches, then use the remove() function to remove the matching text in remove_list from ip_addresses:

```
if element in ip_addresses: # If the remove_list matches with text in
ip_addresses
      ip_addresses.remove(element) # Remove the matched text from ip_addresses
```

Update the file with the revised list of IP addresses

With the removed IP addresses, we need to update the allow_list, to do this we will first need to turn the list back into a string using the join() function.

```
ip_addresses = "\n".join(ip_addresses) # Turn the ip_addresses variable back into
a string, with a new line between each text
```

Then use a with open() function to write the updated list.

```
with open(import_file, "w") as file: # Open allow_list with write permissions
    file.write(ip_addresses) # Write the contents of ip_addresses to the
allow_list file
```

Summary

The algorithm begins with us assigning the allow_list to a variable, so we can use it throughout the code. We then convert the contents of the allow_list file to string format, in order for us to then turn it into list format. We then use a for loop to iterate through the list of denied IP addresses and check those IP addresses against the list of approved IP addresses. If there are any matches, we delete the denied IP addresses from the allow list. Once that's complete, we need to convert the contents back into a string using the join() function, then use the with *open() function to write the updated allow list. This gives us the updated text file which has the denied IP addresses removed.*