Algorithm for file updates in Python

Project description

I have been tasked to create an algorithm using python code to check a list of allowed IP addresses against another list of IP addresses that are required to be removed.

Open the file that contains the allow list

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

import_file = "allow_list.txt"
```

First the file containing the list of allowed IP addresses (allow_list.txt) is stored in a variable called import_file.

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

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

Next I used a ‘with’ statement to open and prepare the file to be read. The first argument is the file (import_file), the next argument (“r”) instructs the open command to open the file in read mode.

Read the file contents

```
  # Use `.read()` to read the imported file and store it in a variable named `ip_addresses`

  ip_addresses = file.read()
```

Here I have stored the output of the file.read() (reads the contents of the file) command into a variable called ip_addresses, this can be displayed using the print command.

Convert the string into a list

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

ip_addresses = ip_addresses.split()
```

In order to remove elements from the file, I need to convert the contents into a list. I did this with the .split() command. This will separate the data using any white space as a marker for separation. This was then stored in the ip_addresses variable.

Iterate through the remove list

```
# 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:
```

Next I had to design a ‘for’ loop which would iterate through the ip_addresses list and compare it to the remove_list.

Remove IP addresses that are on the remove list

```
   # then current element should be removed from `ip_addresses`

   ip_addresses.remove(element)
```

The .remove(element) part of code will remove the element (ip address) from the ip_addresses list if it is found in the remove_list.

Update the file with the revised list of IP addresses

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

  file.write(ip_addresses)
```

With the updates now made to the file, all that remains is to convert the list back into string data and re-write it to the file. The “\n”.join() instructs the program to join the list using a \n (new line) as a separator. The next with statement opens the file (import_file) in write mode (“w”).The final line of the with statement (file.write(ip_addresses)) overwrites the new list of IP addresses into the original file.

Summary

I created an algorithm that removes IP addresses identified in a remove_list variable from the "allow_list.txt" file of approved IP addresses. This algorithm involved opening the file, converting it to a string to be read, and then converting this string to a list stored in the variable ip_addresses. I then iterated through the IP addresses in ip_addresses. With each iteration, I evaluated if the element was part of the remove_list. If it was, I applied the .remove() method to it to remove the element from ip_addresses.. After this, I used the .join() method to convert the ip_addresses back into a string so that I could write over the contents of the "allow_list.txt" file with the revised list of IP addresses.