Algorithm for file updates in Python

Project description

My employer stores sensitive information which is controlled access only. Access is determined using an Allow List which stores the IP addresses of employees who are authorized to sign into the restricted subnetwork. There is also a Remove List which designates IP addresses whose access needs to be rescinded.

The objective of this piece is to show the steps necessary to create an algorithm which will automatically read the two lists, compare them, and remove from the Allow List any IP addresses which are on the Remove List.

Open the file that contains the allow list

After creating a the variable 'import_file' to hold the Allow List and adding the Remove List data, I open the Allow List text using 'with open', calling the import_file variable in read access, storing the contents as the 'file' local variable (Fig. 1, new commands highlighted)

Fig. 1

```
import_file = "allow_list.txt"

remove_list = ["192.168.97.225", "192.168.158.170", "192.168.201.40", "192.168.58.57"]

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

Read the file contents

As part of the 'with open' statement I create a variable 'ip_addresses' to hold the contents of the original Allow List, using .read() on the local variable 'file'. This will output the contents as a string. As part of the 'with open' function, the ip_addresses variable will be a global variable usable outside the 'with open' statement. Also 'with open' will automatically close the imported file.

As a check I print the ip_addresses variable to ensure the ‘with open’ statement if working properly (Fig. 2, new commands highlighted)

Fig. 2

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

print(ip_addresses)
```

```
ip_address
192.168.25.60
192.168.205.12
192.168.97.225
192.168.6.9
192.168.52.90
192.168.158.170
192.168.90.124
192.168.186.176
192.168.133.188
192.168.203.198
192.168.201.40
192.168.218.219
192.168.52.37
192.168.156.224
192.168.60.153
192.168.58.57
192.168.69.116
```

Convert the string into a list

As the content of ip_addresses is currently a string I convert it to a list (Fig. 3, new commands highlighted) so it can be used iteratively in future steps. Again, printing the outcome to ensure the desired results.

Fig. 3

```
ip_addresses = ip_addresses.split()

print(ip_addresses)
```

```
['ip_address', '192.168.25.60', '192.168.205.12', '192.168.6.9', '192.168.52.90', '192.168.90.124', '192.168.186.176', '192.16
8.133.188', '192.168.203.198', '192.168.218.219', '192.168.52.37', '192.168.156.224', '192.168.60.153', '192.168.69.116']
```

Iterate through the remove list and remove relevant IP addresses

Next, I need to compare the list of allowed IP addresses with the list of IP addresses to remove and when a match is found remove the relevant IP address from the allowed list.

For this I create a 'for' loop which looks at each IP address (called 'element' in the loop) in ip_addresses, then looks for that IP address in the remove_list. When an IP address is found in the remove_list, that IP address is removed from the ip_addresses list (Fig. 4)

Fig. 4

```
for element in ip_addresses:
    if element in remove_list:
        # remove current element from allowed IP
        ip_addresses.remove(element)
```

Update the file with the revised list of IP addresses

Now we need to update the original Allow List with the new data using a 'with open' statement.

As 'with open' only works with string data we first need to convert ip_addresses from a list to a string. We do this by using .join, designating a blank space (" ") as the separator for the elements of the string and using the same ip_addresses name to overwrite the original.

We then call 'with open' using the import_file and write ("w") parameters to place the contents of import_file into the local variable 'file'. Next we use 'file' with .write to designate which data to overwrite import_file with, in this case the previously modified ip_addresses (Fig. 5)

Fig. 5

```
ip_addresses = " ".join(ip_addresses)


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

CREATE THE FUNCTION

Now that we have these operational pieces we can easily use them to create a new function called ‘update_file’ which allows us to pass two parameters (one for the allowed IP addresses and one for the IP addresses which need to be removed) and automatically remove any IP addresses which need access to be rescinded from the original Allow List (Fig. 6).

Fig. 6

```
def update_file(import_file, remove_list)

    # open the imported file and read it into a string
    with open(import_file, "r") as file:
        ip_addresses = file.read()

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

    # Loop through ip_addresses, remove disallowed IP's
    for element in ip_addresses:
        if element in remove_list:
            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)

    #`with open` statement to rewrite the original file
    with open(import_file, "w") as file:
        file.write(ip_addresses)
```