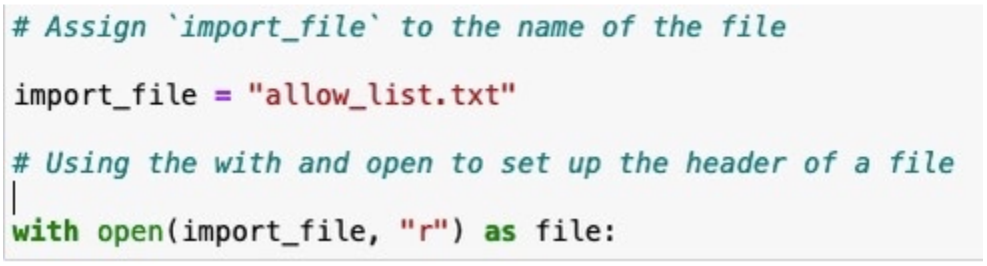
Algorithm for file updates in Python

Project description

At my organization, access to restricted content is controlled with an allow list of IP addresses. The "allow_list.txt" identifies these IP addresses. A separate remove list identifies IP addresses that should no longer have access to this content. I created an algorithm to automate updating the "allow_list.txt" and remove these IP addresses that should no longer have access.

Open the file that contains the allow list

[I first assigned a string called import_file to the "allow_list.txt". Then, I used the with in conjunction with the open function. I placed the import_file as the first parameter, and “r” as the second, because we were only reading the file. Then, I used the as keyword to assign file as the variable.

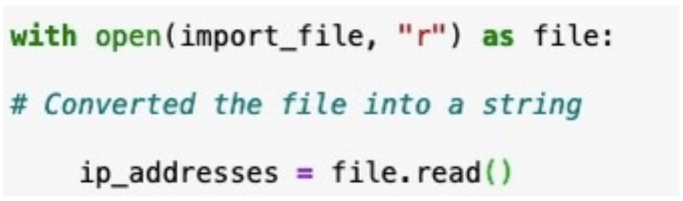


```
# Assign `import_file` to the name of the file

import_file = "allow_list.txt"

# Using the with and open to set up the header of a file

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

]

Read the file contents

[I used the read method to convert the file into a string, and to allow the file to be displayed

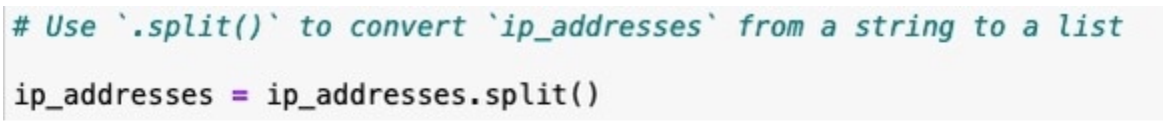


```
with open(import_file, "r") as file:

# Converted the file into a string

    ip_addresses = file.read()
```

]

Convert the string into a list

[I used the split() method to convert the string into a list

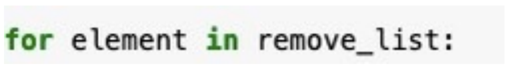


```
# Use `.split()` to convert `ip_addresses` from a string to a list

ip_addresses = ip_addresses.split()
```

]

Iterate through the remove list

[I created a list of all the IP addresses that should be removed, and then iterated through the list using a for loop. I assigned a variable called element as the loop variable.

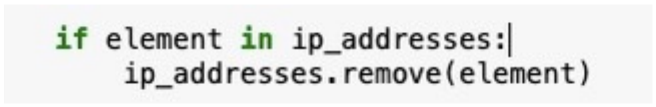


```
for element in remove_list:
```

]

Remove IP addresses that are on the remove list

[In the body of the aforementioned for loop, I created a conditional statement to see if the element is inside of the ip_addresses list, if it is, I used the remove() method to remove the element from the ip_addresses list.

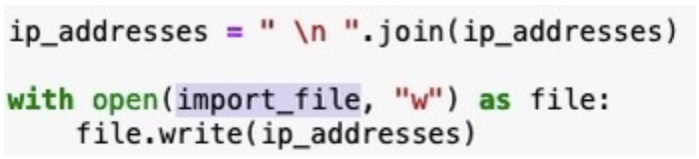


```
    if element in ip_addresses:
        ip_addresses.remove(element)
```

]

Update the file with the revised list of IP addresses

[I had to first change the list back into a string using the join() method. I proceeded the .join() method with “\n”, to separate the elements on a separate line. Then, I used the with keyword, in conjunction with the open method. I used the import_file variable as the first parameter, and “w” as the second parameter, since I was updating the file. In the body I used the write() method to update the elements in the file.



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

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

]

Summary

[I created an algorithm that removes IP addresses identified in a remove_list variable from the "allow_list.txt" of approved IP addresses. This algorithm involved opening the "allow_list.txt", 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 remove_list. With each iteration, I evaluated if the element was part of the ip_addresses list. If it was, I applied the .remove() method to remove the element from ip_addresses list. 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" with the revised list of IP addresses.]