Algorithm for file updates in Python

NOTE: ‘import_file’ was later renamed to ‘allow_file’ in the code

Project description

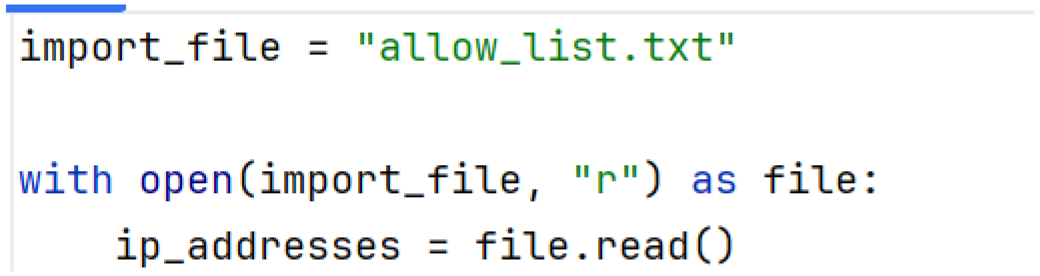
Our task is to create a Python algorithm to check if an IP address from the remove_list (list of IP addresses that should not access restricted content on a healthcare service’s network) is in the allow_list (list of IP addresses that have permission to access restricted content). The algorithm involves understanding file handling and parsing in Python, as well as incorporation of conditional and iterative logic. I was able to create functions that made my code more efficient and reusable.

Open the file that contains the allow list

To open the file that contains the allow list, we simply create a new variable, say “import_file” and assign it the “allow_list.txt” file like this: **`import_file = “allow_list.txt”`**. Next, we use a with open() statement to open our file like this: `with open(import_file, "r") as file:`

Read the file contents

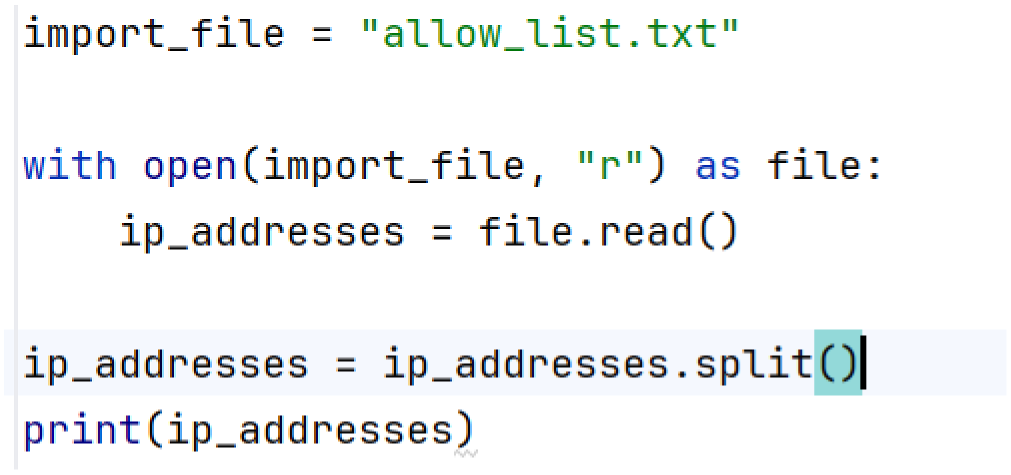
We read the contents of the file by using the .read() method on file, and assign it to our ‘ip_addresses’ variable.



```
import_file = "allow_list.txt"

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

Convert the string into a list

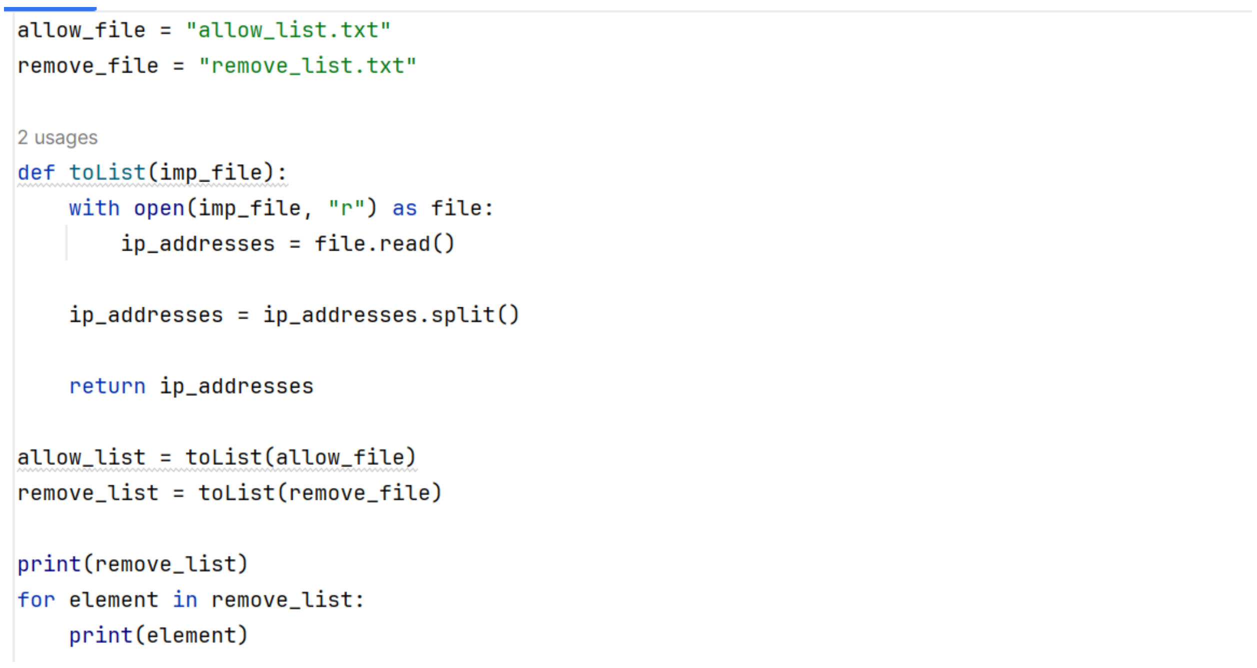
We convert the string into a list by using the split() method on the ip_addresses string variable and reassign the result to ip_addresses.



```
import_file = "allow_list.txt"

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

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

Iterate through the remove list

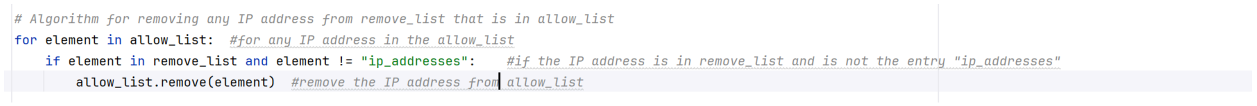
Initially, we convert the remove_list.txt to a list using the toList() function for better formatting. We iterate through the remove_list by using a for loop and the indexed element variable “element”. The output returns all the elements in list form.



```
allow_file = "allow_list.txt"
remove_file = "remove_list.txt"

2 usages
def toList(imp_file):
    with open(imp_file, "r") as file:
        ip_addresses = file.read()

    ip_addresses = ip_addresses.split()

    return ip_addresses

allow_list = toList(allow_file)
remove_list = toList(remove_file)

print(remove_list)
for element in remove_list:
    print(element)
```

Remove IP addresses that are on the remove list

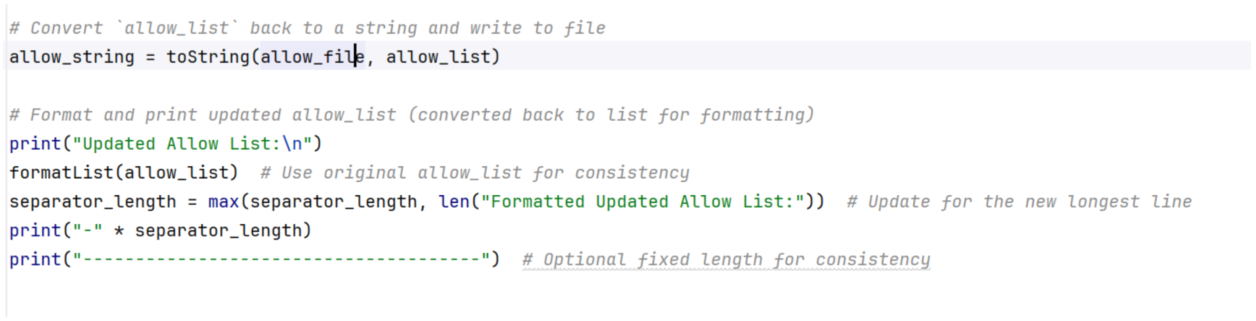
I was able to remove the IP addresses that are on the remove list by using iterative statements and conditional statements. I created a for loop in the allow_list, and inside that I included a conditional statement with “if” to remove the matched IP addresses:



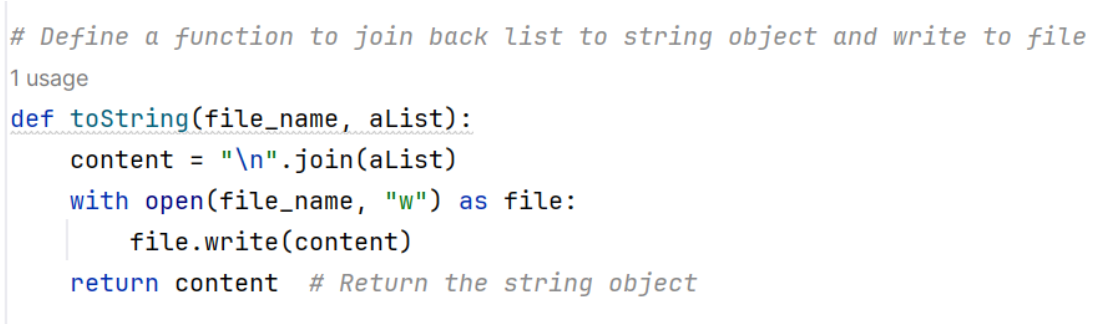
```
# Algorithm for removing any IP address from remove_list that is in allow_list
for element in allow_list:  #for any IP address in the allow_list
    if element in remove_list and element != "ip_addresses":    #if the IP address is in remove_list and is not the entry "ip_addresses"
        allow_list.remove(element)  #remove the IP address from allow_list
```

Update the file with the revised list of IP addresses

We updated the file with the revised list of IP addresses by converting the updated allow_list back to a string using the .join() method. Then, we use the with() statement and write() method to write back the updated allow list to allow_file (or import_file in the context of the directions), and display the updated allow list.



```
# Convert `allow_list` back to a string and write to file
allow_string = toString(allow_file, allow_list)

# Format and print updated allow_list (converted back to list for formatting)
print("Updated Allow List:\n")
formatList(allow_list)  # Use original allow_list for consistency
separator_length = max(separator_length, len("Formatted Updated Allow List:"))  # Update for the new longest line
print("-" * separator_length)
print("--------------------------------------")  # Optional fixed length for consistency
```



```
# Define a function to join back list to string object and write to file
1 usage
def toString(file_name, aList):
    content = "\n".join(aList)
    with open(file_name, "w") as file:
        file.write(content)
    return content  # Return the string object
```

Summary

The algorithm aims to iterate through the allow_list list object and check to see if any of the IP addresses match with the IP addresses in the remove_list object. We do this because our task is to eliminate any unwanted/suspicious IP addresses from accessing restricted content in the network of the healthcare services company. The algorithm is achieved with the help of file parsing and handling, as well as functions, iterative logic, and conditional logic. The removal algorithm logic is achieved by creating a for loop that iterates through all elements in the allow_list. During the iteration, the system checks to see if any IP address from the allow_list matches an IP address from remove_list and is not equal to the entry “ip_addresses” (as this is just a label to make clear the list consists of IP addresses), then remove the IP address from allow_list if a match is found.

Github link to final project and code repo: Git Repo