Algorithm for file updates in Python

Project description

This project is to demonstrate the ability to use Python in regards to the following actions:

1. Open the file that contains the allow list
2. Read the file contents
3. Convert the string into a list
4. Iterate through the remove list
5. Remove the IP address from the allow list that are on the remove list
6. Update the file with the revised list of IP addresses.

Open the file that contains the allow list

Code Block:

```
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:
      ip_addresses = file.read()
print(ip_addresses)
```

1. The `“allow_list.txt”` is located in the `“import_file”` variable.
2. The `“with”` statement is used with the `“open()”` function to open the file. The first argument is the `“import_file”` variable. The second argument is `“r”`, which represents “read”. This is telling Python how to interact with the file after opening it.

Read the file contents

```
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:
      ip_addresses = file.read()
print(ip_addresses)
```

3. The third line assigns the result of the `“with”` statement into the variable `“ip_addresses”` It’s intended to tell Python this code is to be done within the `“with”` statement.

4. The fourth line tells Pyton to print the results. There is no indentation to tell Python to execute the code outside of the **"with"** method.

Convert the string into a list

Code Block:

```
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:
      ip_addresses = file.read()
ip_addresses = ip_addresses(split)
```

5. Since the list in the **"ip_addresses"** is a string, it has to be converted into a list for Python to iterate through it. The **".split()"** method is used to split the elements in the **"allow_list"**, and convert it to a list.

Iterate through the remove list

Code Block:

```
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:
      ip_addresses = file.read()
print(ip_addresses)
for element in ip_addresses:
```

6. The **"for"** loop is used to iterate through the list using the **"element"** variable to determine which address meets the condition of the statement.

Remove IP addresses that are on the remove list

Code block:

```
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:
      ip_addresses = file.read()
print(ip_addresses)
```

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

7. The `"if"` statement is used to build the conditional statement by giving a parameter to the `"for"` loop. The second part of the if statement, with indentation, contains the `".remove()"` method. The `".remove()"` method is used to tell Python to remove any element that meets the condition based on the argument. In this case, the `"element"` variable is the argument.

Update the file with the revised list of IP addresses

Code Block:

```
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:
      ip_addresses = file.read()
for element in ip_addresses:
      if element in remove_list:
            ip_addresses.remove(element)
ip_addresses = "\n".join(ip_addresses)
with open(import_file, "w") as file:
      file.write(ip_addresses)
```

8. The `".join()"` method is used to convert the `"ip_addresses"` variable back into a list. The `"\n"` argument at the beginning of the `".join()"` method will tell Python to separate the items in the list by printing them of different lines
9. In the next line of code, the `"with"` statement is to reopen the `"import_file"` to be able to "write" into it (symbolized by the `"w"`)
10. Then the results of the `"with"` statement is stored in a new variable, `"text"`
11. The last line of code is to write into the `"ip_addresses"` file

Summary

What was done here was a routine update for a list of ip_addresses that require all ip addresses no longer in use to be removed from the list. In this example, the following techniques were used:

`"with"` statement: This statement
`".join()"` method: converts a file into a list, in some cases it can also be used to split a
`"for"` loop: an iterative statement that repeats the code for a specific sequence.
`"if"` statement: statement that executes a command when the conditions are met

"`.remove()`" method: removes elements specified in parenthesis
"`.read()`" method: converts files into strings
"`.write()`" method: allows for the file to be written in/replaces contents of file after it has been opened
"`open()`" function: used to open the file; used alongside the "with" statement
"`.split()`" method: converts the contents of the file into a string that splits a list
"`print()`" function: prints to contents of the file