Update a file through a Python algorithm

Project description

My organization is implementing an access list control to allow the access on for certain IPs. My role as a security analyst is to create an automatization so whenever the list changes, the restrictions shall apply accordingly.

Open the file that contains the allow list

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

import_file = "allow_list.txt"

# Assign `remove_list` to a list of IP addresses that are no longer allowed to access restricted information.

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

# Display `import_file`

print(import_file)

# Display `remove_list`

print(remove_list)
```

```
allow_list.txt
['192.168.97.225', '192.168.158.170', '192.168.201.40', '192.168.58.57']
```

1. There is a list of allowed IPs stored in a text file. A variable called `import_file` is assigned to the name of the file

2. The variable `remove_list` is assigned to a list of Ips no longer required to access restricted information

3. The print function is used twice: once to display the `import_file` variable and the second time, to display the `remove_list` variable

Read the file contents

```
# First line of `with` statement

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

```
  File "<ipython-input-4-b925af1022fc>", line 11
    with open(import_file, "r") as file:
                                        ^
SyntaxError: unexpected EOF while parsing
```

1. `with open()` is a function used to open the file. It takes two parameters: `import_file` (the variable that stores the file to be opened) and `"r"`, which instructs the function to read the file.
2. `as file` tells how to create the file object created by `open()` function.

Convert the string into a list

```
  ip_addresses = file.read()

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

ip_addresses = ip_addresses.split()

# Display `ip_addresses`

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', '192.16
8.90.124', '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']
```

1. `.read()` method reads the contents of the imported file and stores them as a string in the variable `ip_addresses`.
2. The `.split()` method then converts the string stored in `ip_addresses` into a list, assigned to a variable named `ip_addresses`. Each element in the list represents a separate IP address.
3. The `print()` function displays the contents of the `ip_addresses` list, showing each allowed IP address on a new line.

Iterate through the remove list

```
for element in ip_addresses:

    # Display `element` in every iteration

    print(element)
```

```
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
```

1. Every element in `ip_addresses` variable is iterated using `for…in` expression

Remove IP addresses that are on the remove list

```
    if element in ip_addresses:

        # then current element should be removed from `ip_addresses`

        ip_addresses.remove(element)

# Display `ip_addresses`

print(ip_addresses)
```

```
['192.168.25.60', '192.168.97.225', '192.168.52.90', '192.168.90.124', '192.168.133.188', '192.168.201.40', '192.168.52.37',
92.168.60.153', '192.168.69.116']
```

1. Conditional statement if checks if the `ip_addresses` variable contains an element from the `remove_list`

Update the file with the revise 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)
```

1. After the elements are removed, the `ip_addresses` variable is converted back to a string so that it can be written into the text file
2. This is achieved by the `.join()` method. In Python it is used to combine elements of a list (or any iterable) into a single string, using a specified separator. In this case, `"\n"` is the separator.
3. `with open()` takes the parameters import_file and `“w”`. Parameter `“w”` is used to signal the writing.
4. `.write()` method performs the overwriting of the file object
5. the import_file variable’s contents are replaced with ip_addresses’s contents

Summary

In this lab, “allow_list.txt” was a list of approved IP addresses that could access certain restricted content within an organization. The goal was to create an algorithm that can automatically remove IP addresses from the list if they are no longer authorized to access the restricted content.