Algorithm for file updates in Python

Project description

This project is intended to help me learn how to create functions from algorithms in python. The main theme of this project is using an algorithm to update a file containing a list of allowed IP addresses.

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"]
# Build `with` statement to read in the initial contents of the file
with open(import_file, "r") as file:
  # Use `.read()` to read the imported file and store it in a variable named `ip_addresses`
 ip_addresses = file.read()
```

First, I had to import the file using the 'With' statement, which then allowed me to read, write or append the file I opened with it.

Read the file contents

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

  # Use `.read()` to read the imported file and store it in a variable named `ip_addresses`

 ip_addresses = file.read()

# 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
192.168.90.124
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
```

To read the file, I just had to make a new variable and assign it to the .read() statement so that when it came to printing it, it had the information inside of the file in it.

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.
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']
```

To convert the list into a string and make it more usable in python, I used the .split() which turns a string into a list.

Iterate through the remove list

```
ip_addresses = ip_addresses.split()

# Build iterative statement
# Name loop variable `element`
# Loop through `ip_addresses`

for element in ip_addresses:
    if element in remove_list:
        ip_addresses.remove(element)
    else:
        continue

    # Display `element` in every iteration

    print(element)
```

```
192.168.97.225
192.168.158.170
192.168.201.40
192.168.58.57
```

The next step is to filter the results based on who is on the remove list so that I can remove them in the next step. This is where the 'for' algorithm comes in.

Remove IP addresses that are on the remove list

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

# Display `ip_addresses`

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.
168.133.188', '192.168.203.198', '192.168.218.219', '192.168.52.37', '192.168.156.224', '192.168.60.153', '192.168.69.116']
```

This code includes the remove element too, so the printed list no longer contains the remove list IP addresses.

Update the file with the revised list of IP addresses

```
  # Convert `ip_addresses` back to a string so that it can be written into the text file
  ip_addresses = " ".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)
# Call `update_file()` and pass in "allow_list.txt" and a list of IP addresses to be removed
update_file### YOUR CODE HERE ###
# Build `with` statement to read in the updated file
with open("allow_list.txt", "r") as file:
  # Read in the updated file and store the contents in `text`
    text = file.read()
# Display the contents of `text`
    print(text)
update_file("allow_list.txt", ["192.168.25.60", "192.168.140.81", "192.168.203.198"])
```

```
ip_address 192.168.25.60 192.168.205.12 192.168.6.9 192.168.52.90 192.168.90.124 192.168.186.176 192.168.133.188 192.168.20
3.198 192.168.218.219 192.168.52.37 192.168.156.224 192.168.60.153 192.168.69.116
```

This shows the process of updating the original file. I also turned this algorithm into a single function to streamline it for the future.

Summary

I learned a lot about how python can be applied, especially when used for cybersecurity. I feel like the things I learned in this module will help me a lot in progressing my knowledge of security and how tools like python are applied in the industry.