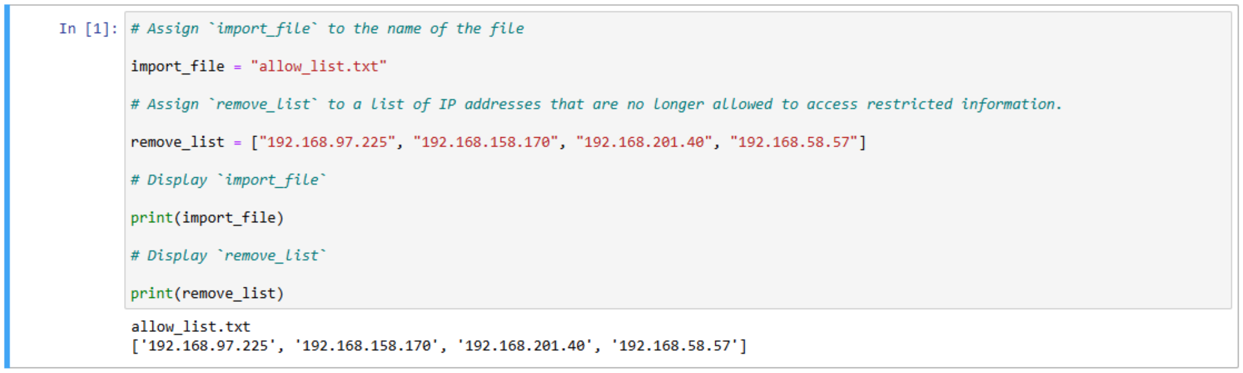
Algorithm for file updates in Python

Project description

This document demonstrates my ability to use python to develop algorithms that involve opening and parsing elements to a file. In this scenario, I am a security professional at a small health care company. As part of my tasks, I am required to regularly update a file of employees with access permission to some restricted content. The contents of this file are based on who is working on the personal patient records. Restrictions are based on the IP addresses of the employees. There is a list of allowed IP addresses with permissions to sign into the restricted subnetwork and a list of IP addresses that should be removed from the allow list.
In this project I used the Jupyter kernel with Python 3.

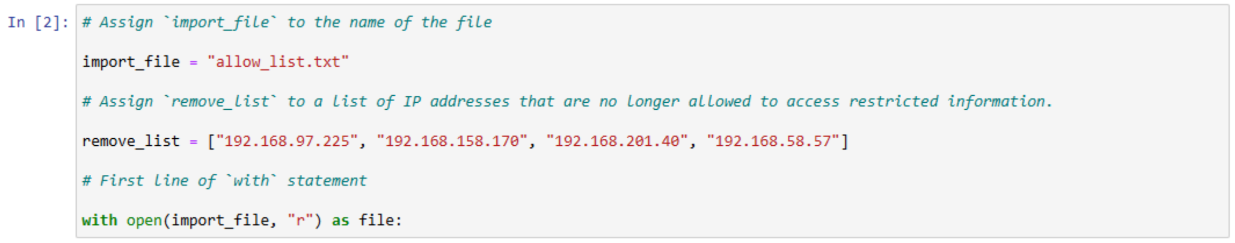


```
In [1]: # Assign `import_file` to the name of the file

        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']
```

Open the file that contains the allow list

To open the file that contains the allow list I used the following functions and commands:

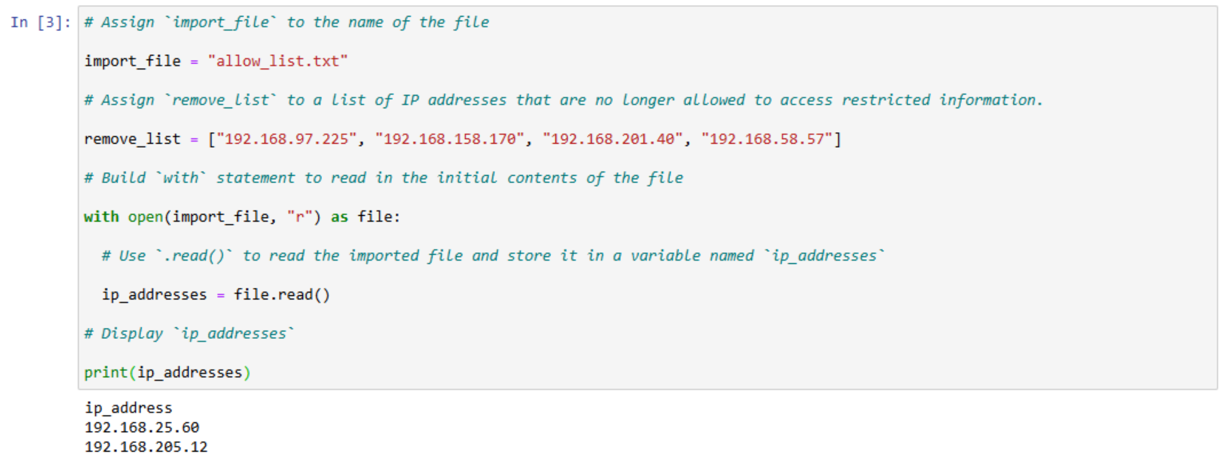
- with - handles errors and manages external resources.
- open() - a function that opens a file in python. I passed 2 arguments as required, the first being the variable name for the file I am interested in.
- “r” - tells python we want to read the file. That’s also the second argument in the open() function.
- as file - assigns a variable that references a file.



```
In [2]: # 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"]

        # First line of `with` statement

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

Read the file contents

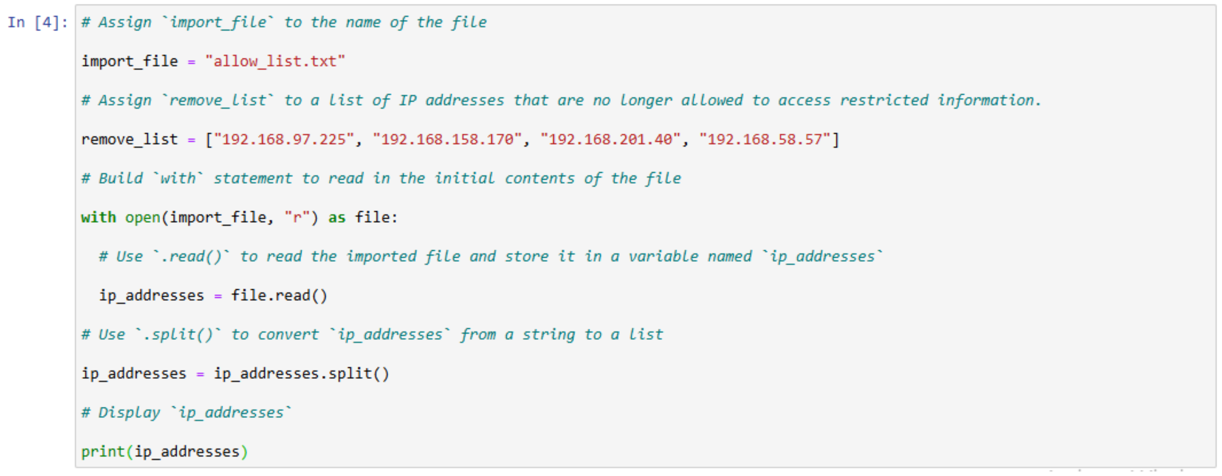
I defined a variable ip_addresses that returns the file as string data. .read() converts the file into strings and print(ip_addresses) displays the contents of the as a string.



```
In [3]: # 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()

        # Display `ip_addresses`

        print(ip_addresses)

        ip_address
        192.168.25.60
        192.168.205.12
```

Convert the string into a list

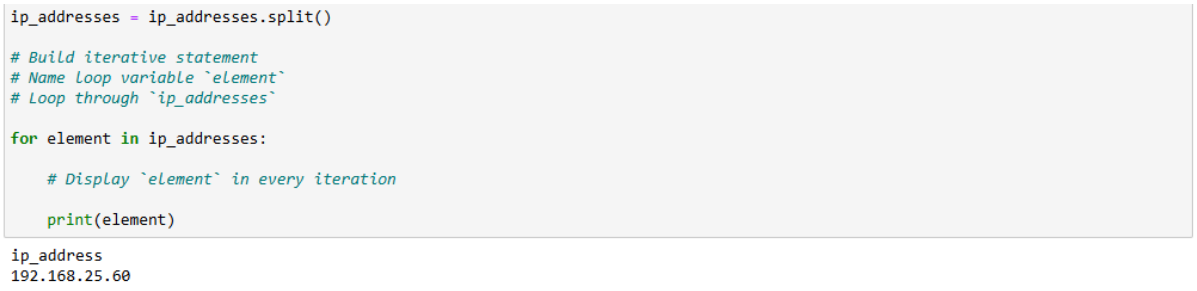
To convert the string into a list, I used the .split() function. I reassigned the variable ip_addresses. The .split() function converts a string into a list



```
In [4]: # 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()

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

        ip_addresses = ip_addresses.split()

        # Display `ip_addresses`

        print(ip_addresses)
```

Iterate through the remove list

I then used the for loop to iterate through the list. I assigned the name loop variable to element for readability.



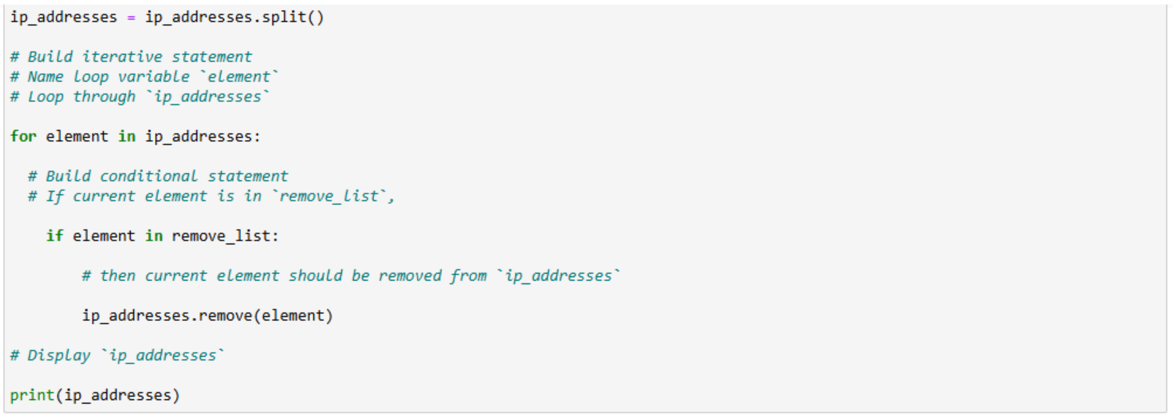
```
ip_addresses = ip_addresses.split()

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

for element in ip_addresses:

    # Display `element` in every iteration

    print(element)
```

```
ip_address
192.168.25.60
```

Remove IP addresses that are on the remove list

In order to remove the IP addresses that are on the list, I:

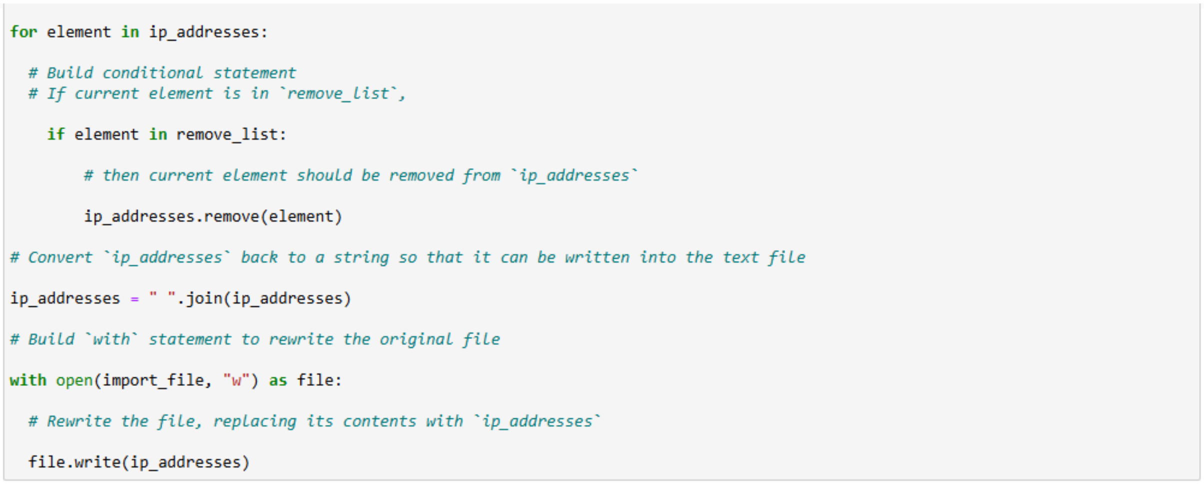
1. iterated through the list of IP addresses that tried to access the restricted file.
2. included a conditional statement using the if statement, The condition checks if each element being iterated in the ip_addresses has is in the remove_list. If the condition is true, the .remove() function removes the element(IP address).



```
ip_addresses = ip_addresses.split()

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

for element in ip_addresses:

  # Build conditional statement
  # If current element is in `remove_list`,

    if element in remove_list:

        # then current element should be removed from `ip_addresses`

        ip_addresses.remove(element)

# Display `ip_addresses`

print(ip_addresses)
```

Update the file with the revised list of IP addresses

In order to update the file, I used the .write() function. I had to first open the import_file again, but this time with a "w" argument. This argument asks python to give us permission to write to the file. I then used the .write() function with the updated list (ip_addresses) as the argument. This then rewrites the file and replaces the contents.



```
for element in ip_addresses:

  # Build conditional statement
  # If current element is in `remove_list`,

    if element in remove_list:

        # then current element should be removed from `ip_addresses`

        ip_addresses.remove(element)

# 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)
```

Summary

I have mastered the foundational concepts that allow me to automate tasks with python and this was an example of how I created an algorithm that replaces the contents of a file.