# Algorithm for file updates in Python

## Project description:

My organization wants me to create an algorithm in python to check whether or not the allowed IP addresses in their systems are identified on the remove list. If there is, the IP should be removed from the file containing the allow list. The “allow\_list.txt” file identifies these IP addresses. A separate remove list identifies IP addresses that should no longer have access to this content. I created an algorithm to automate the updates of the “allow\_list.txt” file and remove the IP’s that should no longer have access.

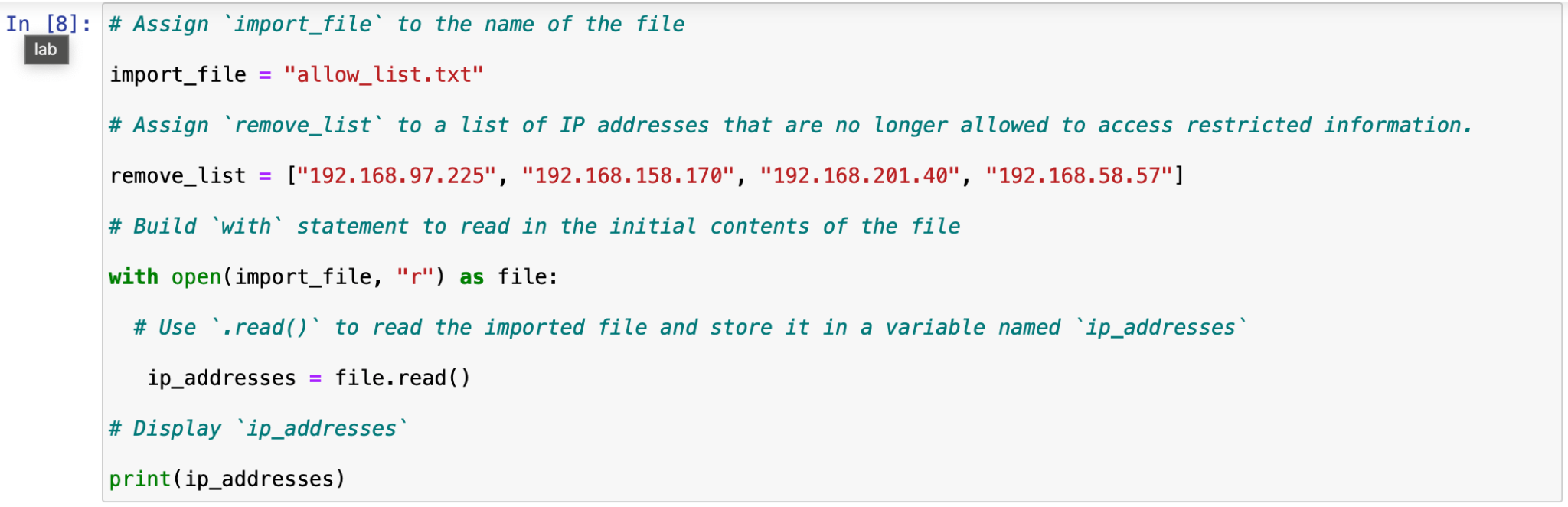
## Opening the file:

## 

To start the algorithm, I had to open the “allow\_list.txt” file. To do that, I assigned the file name as a string variable named “import\_file”. Then, I used a with statement, which is “with open (import\_file, “r”) as file:” and it allows me to open the file containing the allowed IP’s. The purpose of opening the file is to allow me to access the IP’s stored in the allow list file. The with keyword will help resource management by closing the file after exiting the with statement.

## Read the file contents

In order to read the file contents, I used the .read() method to convert it into the string.

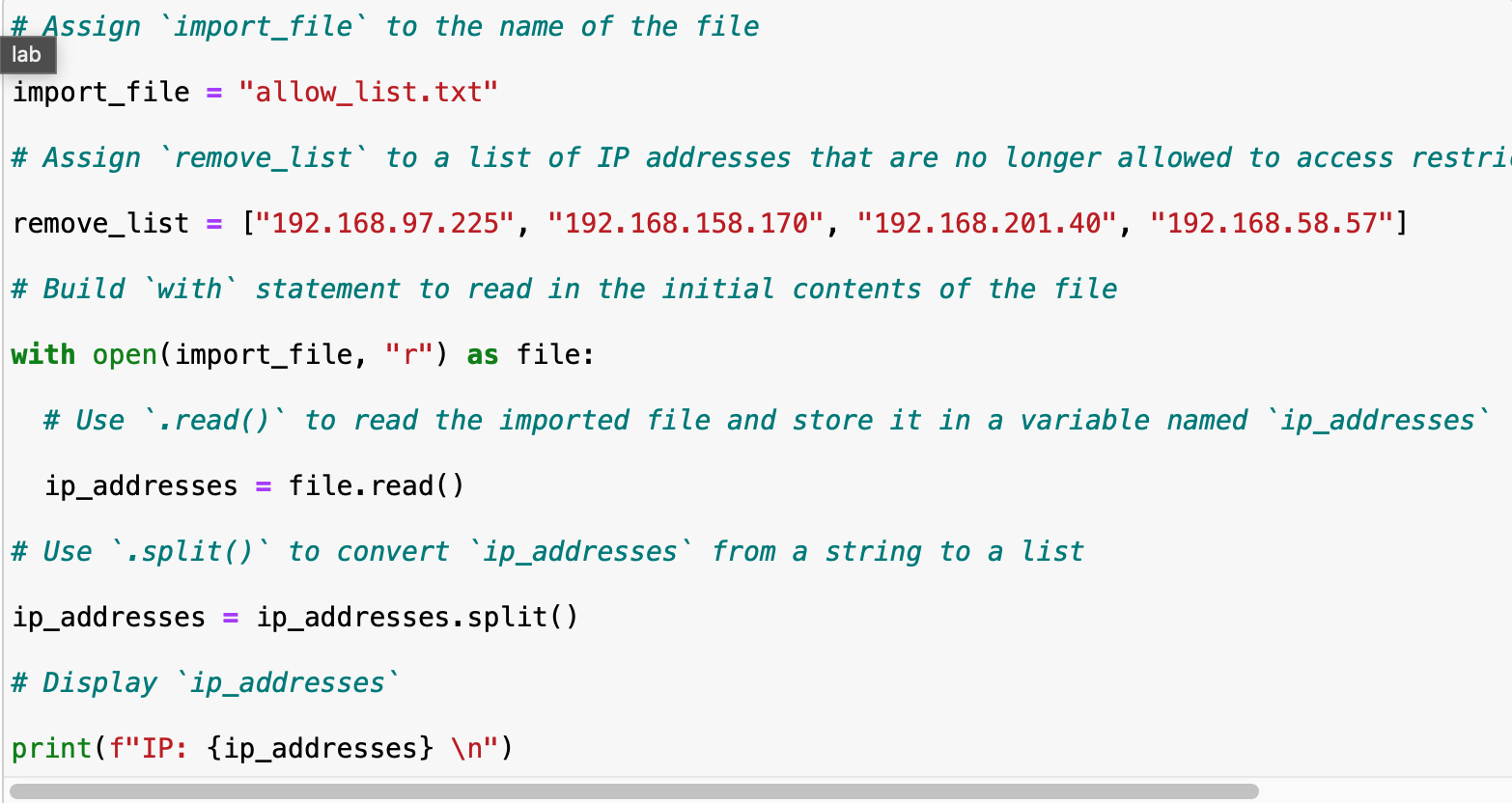


When using .open() function, I included “r” for “read”, I can call the .read() function in the body of the with statement. The .read() method converts the file into a string for reading purposes. I applied the .read() method to the file variable identified in the with statement. Then, I assigned the output to the ip\_addresses variable.

This code reads the content of the “allow\_list.txt” file into a string that allows me to later use the string to organize and extract data in the program.

Convert the string into a list

In order to remove the individual IP addresses from the allow\_list.txt file, I need to first convert it to a list format. I then used the .split() method to convert the ip\_addresses string into a list.



The .split() function is called by appending it to a string valuable and it works by converting the contents of a string into a list with individual values. The purpose is to make it easier to remove individual IP addresses from the allow\_list.txt. By default, the .split() function splits the text by whitespace into list elements.

Iterate through the remove list

A Key part of the algorithm involves iterating through each IP that are in the remove\_list. To do this, I incorporated a for loop.

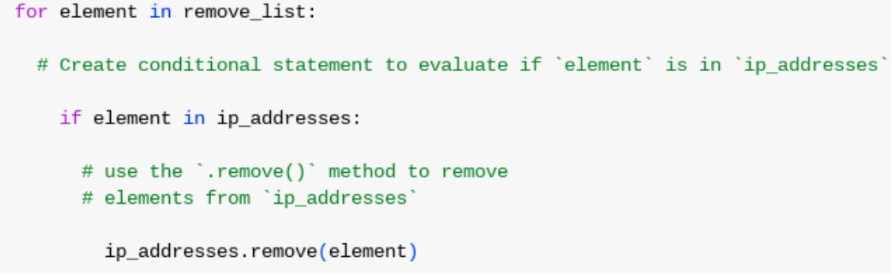
A close-up of a computer screen

Description automatically generated

The for loop in Python repeats code for a specified sequence. The purpose of a for loop in the algorithm is to apply specific code statements to all elements in a sequence. The for keyword starts the for loop, it is then followed by a loop variable element and the keyword in. The keyword in indicates to iterate through the sequence ip\_addresses and assign each value to the loop variable element.

Remove IP addresses that are on the remove list

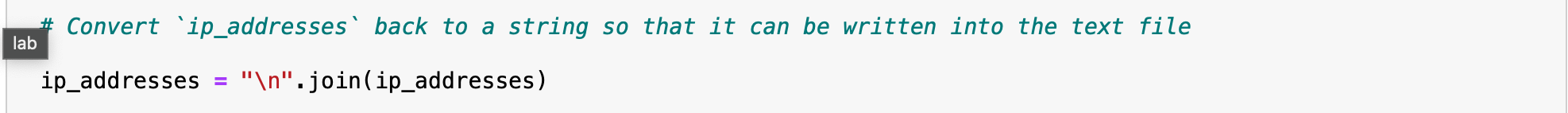
My algorithm requires removing any IP addressed from the allow list, ip\_addresses, that is also contained in remove\_list. Because there were not any duplicates in ip\_addresses, I was able to use the following:



First, within the for loop, I created a conditional that evaluated whether or not the variable element was found in the ip\_addresses list. Applying .remove() to elements that were not found in ip\_addresses would result in an error. Then, within the conditional statement I applied .remove() to ip\_addresses. I passed in the loop variable element as the argument so that each IP address that was in the remove\_list would be removed from ip\_addresses.

## Update the file with the revised list of IP addresses

The final step in the algorithm, I needed to update the allow list file with the revised list of IP addresses. To do so, I first needed to convert the list back into a string. I used the .join() method:



The .join() method combines all items in an iterable into a string. The .join() method is applied to a string containing characters that will separate the elements in the iterable once joined into a string. In this algorithm, I used the .join() method to create a string from the list ip\_addresses so that I could pass it in as an argument to the .write() method when writing to the file “allow\_list.txt”. I used the string (“\n”) as the separator to tell Python to place each element on a new line.

I then used another with statement and the .write() method to update the file:

A white screen with blue text

Description automatically generated

I used a second argument of “w” with the open() function in my with statement. This argument indicates that I want to open a file to write over its contents. The .write() function writes string data to a specified file and replaces any existing file content.

In this algorithm, I wanted to write the updated allow list as a string to the file “allow\_list.txt”. After this, the restricted content will no longer be accessible to any IP addresses that were removed from the allow list. In order to re-write the file, I appended the .write() function to the file object file that I identified in the with statement. I passed the ip\_addresses variable as the argument to specify the contents of the file.

Summary

I created an algorithm that removes IP addresses identified in a remove\_list variable from the “allow\_list.txt” file of approved IP’s. This algorithm involved opening the file, converting it into a string, and then converting the string into a list stored in the variable ip\_addresses. I then iterated through the IP addresses in remove\_list. With each iteration, I evaluated if the element was part of the ip\_addressess list. If it was, I applied the .remove() method to it to remove the element from ip\_addresses. After this, I used the .join() method to convert the ip\_addresses back into a string so that I could write over the “allow\_list.txt” file with the revised list of IP addresses.