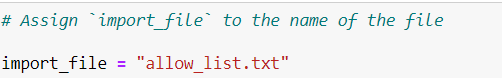
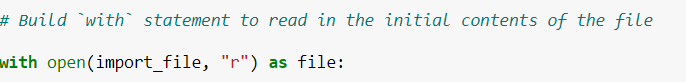
Algorithm for file updates in Python

Project description

Working for a healthcare company as a security analyst, where for my task I must regularly update a file to detect the employees who are permitted access to the restricted content. The contents behind the file are only managed by the ones who conduct with confidential patient records. Based on the IP address, employees are restricted access. There is an allow list established for the accepted IP addresses that can enter the restricted subnetwork. However, there is also a remove list which determines which employees must be eradicated from the allow list.

Open the file that contains the allow list





In my algorithm, the with statement is used with the .open() function in read mode to open

the allow list file for the purpose of reading it. The purpose of opening the file is to allow me to

access the IP addresses stored in the allow list file. The with keyword will help manage the

resources by closing the file after exiting the with statement. In the code with

open(import\_file, "r") as file:, the open() function has two parameters. The first

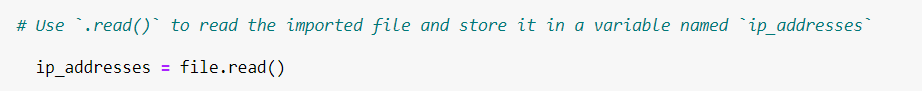
identifies the file to import, and then the second indicates what I want to do with the file. In this

case, "r" indicates that I want to read it. The code also uses the as keyword to assign a

variable named file; file stores the output of the .open() function while I work within the

with statement.

Read the file contents



When using an .open() function that includes the argument "r" for “read,” I can call the

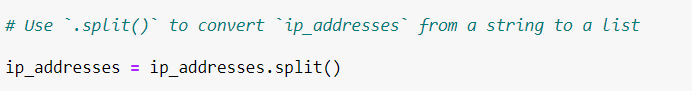
.read() function in the body of the with statement. The .read() method converts the file

into a string and allows me to read it. I applied the .read() method to the file variable

identified in the with statement. Then, I assigned the string output of this method to the

variable ip\_addresses.

Convert the string into a list



The .split() function is called by appending it to a string variable. It works by converting the

contents of a string to a list. The purpose of splitting ip\_addresses into a list is to make it

easier to remove IP addresses from the allow list. By default, the .split() function splits the

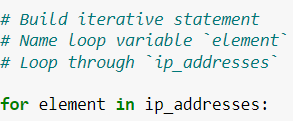
text by whitespace into list elements. In this algorithm, the .split() function takes the data

stored in the variable ip\_addresses, which is a string of IP addresses that are each

separated by a whitespace, and it converts this string into a list of IP addresses. To store this

list, I reassigned it back to the variable ip\_addresses.

Iterate through the remove list



The for loop in Python repeats code for a specified sequence. The overall purpose of the for

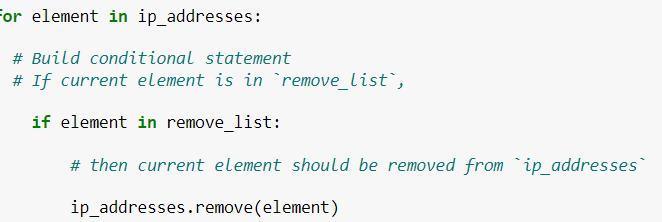
loop in a Python algorithm like this is to apply specific code statements to all elements in a

sequence. The for keyword starts the for loop. It is followed by the 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



First, within my for loop, I created a conditional that evaluated whether or not the loop

variable element was found in the ip\_addresses list. I did this because applying

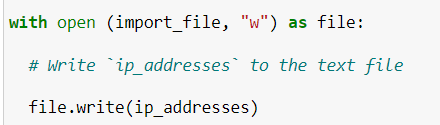
.remove() to elements that were not found in ip\_addresses would result in an error.

Then, within that conditional, 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



This time, 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. When using this

argument "w", I can call the .write() function in the body of the with statement. The

.write() function writes string data to a specified file and replaces any existing file content.

Summary

I created an algorithm that removes IP addresses identified in a remove\_list variable from

the "allow\_list.txt" file of approved IP addresses. This algorithm involved opening the

file, converting it to a string to be read, and then converting this string to 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\_addresses 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 contents of the "allow\_list.txt" file with the revised list of IP addresses.