# Algorithm for file updates in Python

## Project description

## In our organization, access to restricted content is managed using an allow list of IP addresses, identified in the "allow\_list.txt" file. We also maintain a separate remove list to identify IP addresses that should no longer have access to this content. I've developed an algorithm to automate the updating of the "allow\_list.txt" file, ensuring that these obsolete IP addresses are promptly removed from access permissions.

## 

## Open the file that contains the allow list

I opened the allow\_list.txt then assigned this file name as a string to the import\_file variable

#Assign ‘import\_file’ to the name of file

Import\_file = “allow\_list.txt”

Then used a with statement to open the file

#Build ‘with’ statement to read initial contents of file

With open(import\_file, “r”) as file:

In my algorithm, I use the with statement with the .open() function in read mode to open the allow list file for the purpose of reading it. This allows me to access the IP addresses stored in the allow list file. The with keyword helps manage resources by automatically 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 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

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

With open(import\_file, “r”) as file:

# Use .read() to read the imported file and store it in a variable names ip\_addresses

Ip\_addresses = file.read()

When using an .open() function with the argument "r" for “read,” I can call the .read() function within the body of the with statement. The .read() method converts the file into a string, allowing me to read its contents. 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. In summary, this code reads the contents of the "allow\_list.txt" file into a string format, which I can later use to organize and extract data in my Python program.

## Convert the string into a list

to remove individual IP addresses from the allow list, I needed it to be in list format.

I next used the .split() method to convert the ip\_addresses string into a list:

# Use .split() to convert ip\_addresses from a string to a list

Ip\_addresses = ip\_addresses.split()

The .split() function is called by appending it to a string variable. I 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. 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

Add a for loop;

# Build iterative statement

# Name a loop variable element

# Loop through remove list

For element in remove\_list:

The for loop in Python repeats code for each item in a sequence. Its purpose is to apply specific code statements to every element in the sequence. The for keyword initiates the loop, followed by the loop variable (element) and the keyword in, indicating iteration through the sequence (ip\_addresses) and assigning each value to the loop variable.

## Remove IP addresses that are on the remove list

This algorithm removes ip addresses from the allow list, ip\_addresses, that are also in the remove\_list. We did not have many duplicates in ip\_addresses to begin with, we can use the following code ;

For element in remove\_list:

# Create conditional statement to evaluate if ‘element’ is in “ip\_addresses”

If element in ip\_addresses:

# use the .remove() method to remove

# elements from ip\_addresses

ip\_addresses.remove(element)

Firstly, in my for loop, I checked if the loop variable (element) was present in the ip\_addresses list to avoid errors when using .remove(). Then, if the element was found in ip\_addresses, I used .remove() to delete it from the list. This ensured that each IP address from the remove\_list was successfully removed from ip\_addresses.

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

Need to update current allowed list with revised list of ip\_addresses. To do so convert the list back to a string, to do this use .join()

#Convert list back to a string using .join()

Ip\_addresses = “\n”.join(ip\_addresses)

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 instruct

Python to place each element on a new line.

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

# 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)

In this instance, I included a second argument, "w", in the open() function within my with statement. This argument indicates my intention to overwrite the file's existing contents. With this "w" argument, I can use the .write() function within the with statement. The .write() function is used to write string data to a specified file, replacing any existing content. In this scenario, I aimed to update the allow list by writing it as a string to the file "allow\_list.txt". To achieve this, I added the .write() function to the file object (file) identified in the with statement. I passed the ip\_addresses variable as the argument to indicate that the contents of the file specified in the with statement should be replaced with the data stored in this variable.

## Summary

I developed an algorithm to remove IP addresses listed in a remove\_list variable from the "allow\_list.txt" file containing approved IP addresses. The algorithm involves opening the file, converting its content into a string for reading, and then converting this string into a list stored in the ip\_addresses variable. Next, I iterated through the IP addresses in the remove\_list. During each iteration, I checked if the element was part of the ip\_addresses list. If it was, I used the .remove() method to eliminate the element from ip\_addresses. Finally, I employed the .join() method to convert ip\_addresses back into a string, allowing me to overwrite the contents of the "allow\_list.txt" file with the updated list of IP addresses.