# **Updating a File with a Python Algorithm** **Project Description**

In my company, access to restricted content is controlled by an allowlist of IP addresses. The file "allow\_list.txt" contains these IP addresses. A separate blacklist identifies the IP addresses that should no longer have access to this content. I created an algorithm to automate the process of updating the "allow\_list.txt" file and removing the IP addresses that should no longer have access.

## **Opening the File Containing the Allowlist**

For the first part of the algorithm, I opened the "allow\_list.txt" file. First, I assigned this filename as a string to the variable import\_file:



Then, I used a with statement to open the file:



In my algorithm, the with statement is used with the .open() function in read mode to open the allowlist file for the purpose of reading it. The goal of opening the file is to allow me to access the IP addresses stored in the allowlist file. The with keyword helps manage resources by automatically closing the file once the with block is exited. In the code with open(import\_file, "r") as file:, the open() function has two parameters. The first one identifies the file to be imported, and the second specifies 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 block.

## **Read the File Content**

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



When I use the .open() function with the "r" argument 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 it. I applied the .read() method to the file variable identified in the with statement. Then, I assigned the output string from this method to the variable ip\_addresses.

In summary, this code reads the content of the "allow\_list.txt" file into a string format, which allows me to later use the string to organize and extract data within my Python program.

## **Convert the String to a List**

To remove individual IP addresses from the allowlist, I needed them to be in the form of a list. Therefore, I used the .split() method to convert the ip\_addresses string into a list:



I called the .split() function by applying it to a string variable. It works by converting the contents of a string into a list. The purpose of splitting ip\_addresses into a list is to facilitate the removal of IP addresses from the allowlist. By default, the .split() function splits the text by spaces into list elements. In this algorithm, the .split() function takes the data stored in the ip\_addresses variable, which is a string of space-separated IP addresses, and converts that string into a list of IP addresses. To store this list, I reassigned it to the ip\_addresses variable.

## **Iterating through the Removal List**

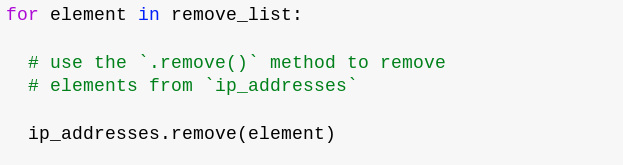
A key element of my algorithm is iterating through the IP addresses that are elements of the remove\_list. To do this, I incorporated a for loop:



The for loop in Python repeats the code for a specified sequence. The general purpose of the for loop in an algorithm like this is to apply specific code instructions to all elements of a sequence. The for keyword starts the loop. It is followed by the loop variable element and the in keyword. The in keyword indicates to iterate over the ip\_addresses sequence and assign each value to the loop variable element.

## **Removing IP Addresses in the Removal List**

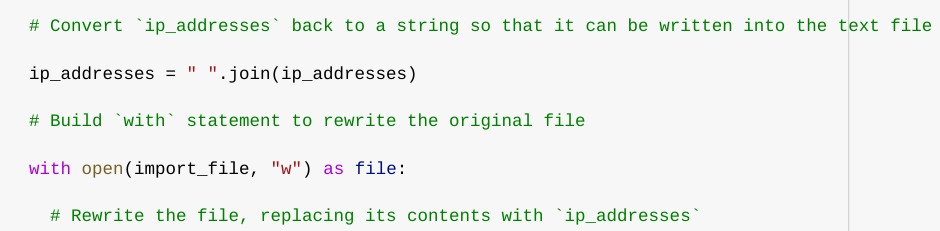
My algorithm needs to remove any IP address from the allowlist, ip\_addresses, that is also contained in the remove\_list. Since all elements of remove\_list are also present in the ip\_addresses list and ip\_addresses contains no duplicates, I was able to incorporate the .remove() method into the body of my for loop as follows:



Since the IP addresses in remove\_list need to be removed from the ip\_addresses list, I applied the .remove() method to ip\_addresses. I passed the loop variable element as an argument so that each IP address found in remove\_list would be removed from ip\_addresses.

## **Updating the File with the Revised List of IP Addresses**

As the final step of my algorithm, I needed to update the allowlist file with the revised list of IP addresses. To do this, I first had to convert the list into a string. I used the .join() method for this:



The .join() method combines all elements of an iterable into a string. The .join() method is applied to a string containing the characters that will separate the elements of the iterable once they are joined into a string. In this algorithm, I used the .join() method to create a string from the ip\_addresses list so that I could pass it as an argument to the .write() method when writing to the "allow\_list.txt" file. I used a single space (" ") as the separator.

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



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 the file to write its content. When using this "w" argument, I can call the .write() function within the body of the with statement. The .write() function writes string data to a specified file and replaces any existing file content.

In this case, I wanted to write the updated allowlist as a string into the "allow\_list.txt" file. This way, the restricted content will no longer be accessible to the IP addresses that were removed from the allowlist. To rewrite the file, I added the .write() function to the file object I identified in the with statement. I passed the ip\_addresses variable as an argument to specify that the content of the file specified in the with statement should be replaced with the data from this variable.

## **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 into a readable string, and then converting that string into a list stored in the ip\_addresses variable. I then iterated over the IP addresses in remove\_list and removed each of those IP addresses from the ip\_addresses list using the .remove() method. After that, I used the .join() method to convert ip\_addresses into a string so that I could write the revised list of IP addresses into the "allow\_list.txt" file.