# Algorithm for file updates in Python

## Project description

In this organization, access to restricted content is controlled with an allow list of IP addresses. The “allow\_list.txt” file identifies these IP addresses. Additionally, a separate remove list identifies IP addresses that should no longer have access to this content. I created an algorithm that automates the updating process for the “allow\_list.txt” file and remove the IP addresses that should have their access rights rescinded.

## Open the file that contains the allow list

The first part of the algorithm involves assigning a name to the “allow\_list.txt” file and subsequently opening it using a ‘with’ statement.





This second part of the code is interpreted as using the “.open()” function on the file that was defined in the first picture to read as a file in this program. The “r” indicates the action of read-only for the filename in the “.open()” function.

## Read the file contents

The with statement shown above requires an additional line to actually read the contents of the file and that is where the “.read()” function comes in.



Linking this line named “ip\_addresses = file.read()” to the “r” permission in the “with” statement allows for the file to be converted into a string for reading purposes.

This code essentially reads the contents of the original “allow\_list.txt” file into a string format that allows a user to organize and extract data using this particular Python program.

## Convert the string into a list

Now, additional executables need to be implemented to edit this file that we are now allowed to read. For the purposes of removal, the list of IP addresses need to be in a list format, so the “.split()” method is utilized to convert the “ip\_addresses” string into a list.



The purpose of splitting “ip\_addresses” into a list is to make it easier to remove IP addresses from the allow list. Split does this by separating each element in the list with a whitespace and in this case allows for the list of IP addresses to be easily read. This entire process is being overwritten into the variable named “ip\_addresses” for usability.

## Iterate through the remove list

A major part of this algorithm involves the process of iteration through the list of IP addresses that are stored as elements belonging to the “remove\_list” variable. A “for” loop is utilized to achieve this process.



This code allows for specific code statements to be applied to each element in the list. This will help us find specific elements that need to be removed.

## Remove IP addresses that are on the remove list

This algorithm requires removing any IP addresses from the allow list that also exist in the remove list.



Within this “for” loop exists a conditional statement for evaluating if there exists an element in the remove list that also exists in the allowed “ip\_addresses” list. When an IP address is found in both lists, then the conditional statement on the third line of code will be run. “ip\_addresses.remove(element)” selects the current element in the iteration and removes it from the allowable list. This process will happen for every element in the “ip\_addresses” list.

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

As a final touchup to this algorithm, the allow list file should be updated with the revision of the IP addresses that were changed in the process. To achieve this task, the “.join()” function will be utilized.



The above code utilized the “.join()” method to be applied to the string of “ip\_addresses” so that it is able to be passed in as an argument to the “.write()” method to update the original file of “allow\_list.txt”, which is required for updating the file. The added use of “\n” allows for a return between each element for easier readability.

The following code is using the “with” statement and “.write()” method to update the file.



This code is similar, but different from the “.read()” method by changing the “r” to a “w” and the “file.read(ip\_addresses)” to “file.write(ip\_addresses)” and will overwrite the file and will show the updated list when called again.

This update to the list successfully changes permissions to the allow list and prevents access from UP addresses that were removed from being allowed access to the content.

## Summary

In this example, I created an algorithm that removed IP addresses identified in a “remove\_list” variable from the “allow\_list.txt” file of approved IP addresses. The algorithm opened the existing file, converted it to a string to be read, then converted the string into a stored list, “ip\_addresses”. After this stored list was created, an iterative process was able to comb through each element and determine which IP addresses required removal. This was achieved using the “.remove()” method. Finally, the “.join()” method was used to convert the “ip\_addresses” back into a string format for writing purposes to update the original “allow\_list.txt” file with only the appropriate IP addresses.