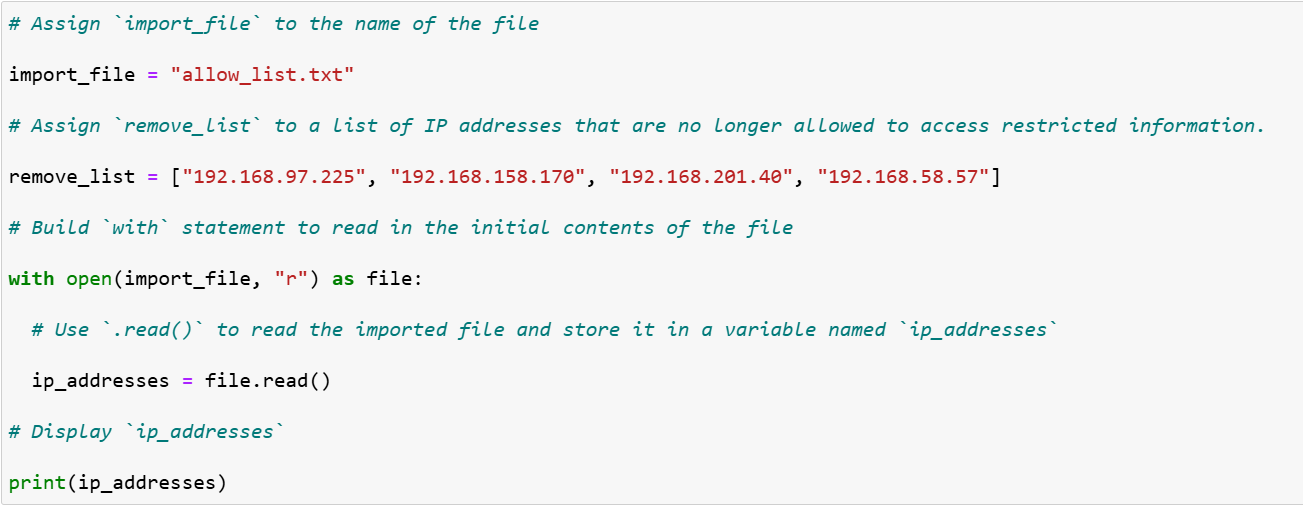
# Algorithm for file updates in Python

## Project description

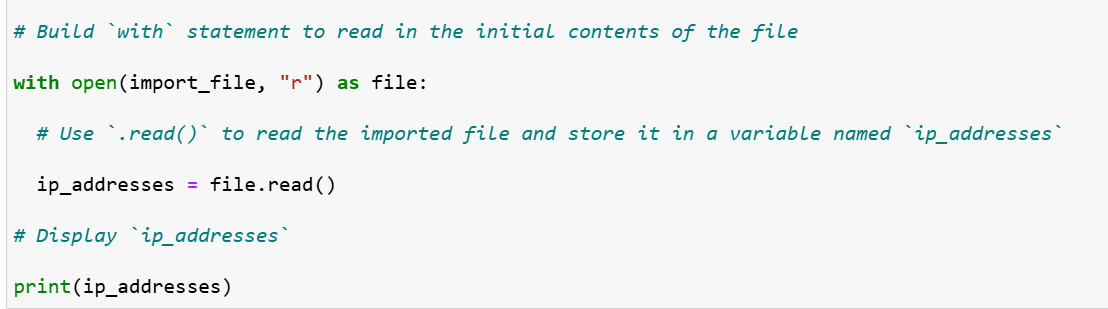
At this organization, access to restricted content is controlled by a list of allowed IP addresses. The "allow\_list.txt" file identifies these IP addresses. There is also a separate remove list of IP addresses that should no longer have access to this content. Using Python, I made an algorithm that helps automate updating the "allow\_list.txt" file and remove these IP addresses that should no longer have access.

## Open the file that contains the allow list



## For the first part of the code, I opened the "allow\_list.txt" file, and assigned this file name as a string to the import\_file variable. Then, by using the with statement to open the file with the .open() function in read mode to open the allow list file for the algorithm to read from. By opening the file, it is able to access the IP addresses stored in the allow list file. with will help manage the resources by closing the file once exited. 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 as to assign a variable named file, which stores the output of the .open() function while working within the with statement.

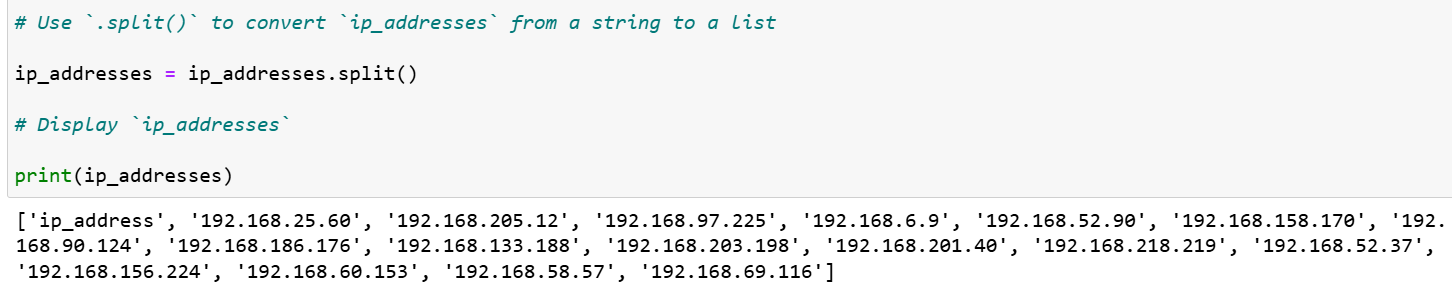
## Read the file contents



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

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 to make it easier to read. Then, I assigned the string output of this method to the variable ip\_addresses. This allows the code to read the contents of the "allow\_list.txt" file into a string format that allows for later use of the string to organize and extract data in the Python program.

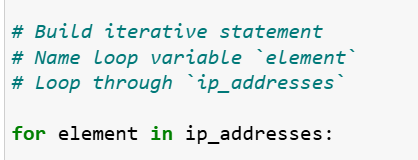
## Convert the string into a list



To be able to remove individual IP addresses from the allow list, I first needed it to be in list format, so I used the .split() method to convert the ip\_addresses string into a list.

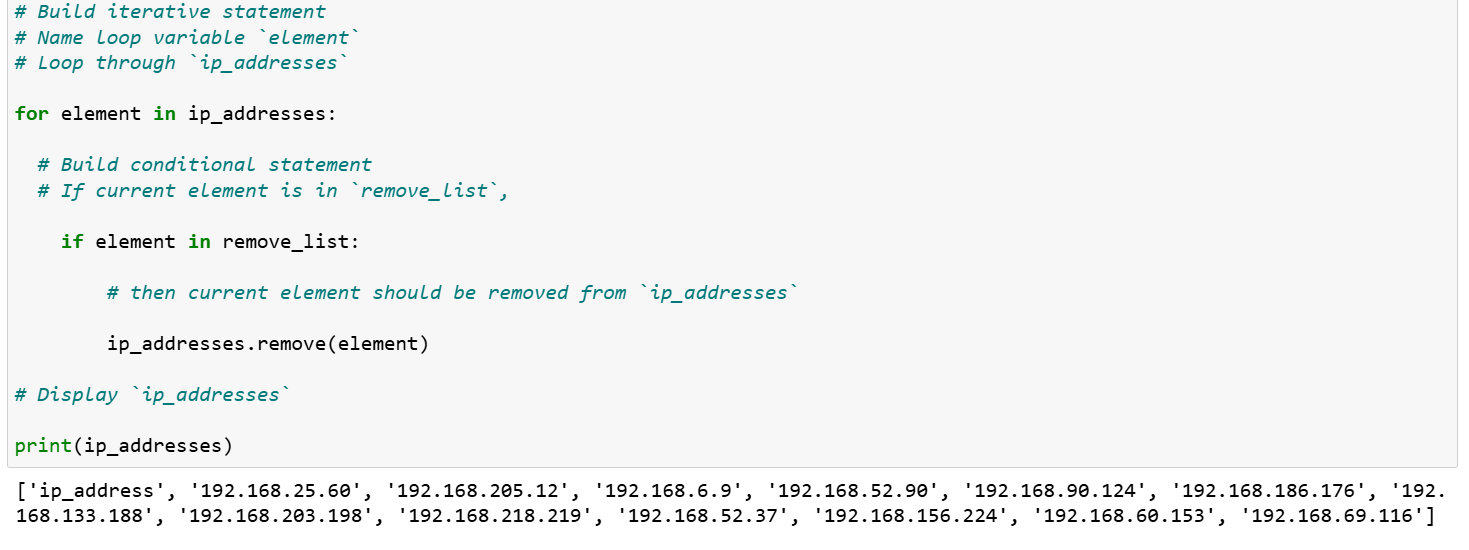
The .split() function is called by appending it to a string variable by converting the contents of a string to a list. The purpose of splitting ip\_addresses into a list is to make it easier to remove IP addresses from the allow list. Here, the .split() function takes the data stored in the variable ip\_addresses, which is a string of IP addresses that’s separated by a whitespace, and then gets converted into a list of IP addresses. To store this list, I reassigned it back to the variable ip\_addresses to stay recorded and up to date.

## Iterate through the remove list



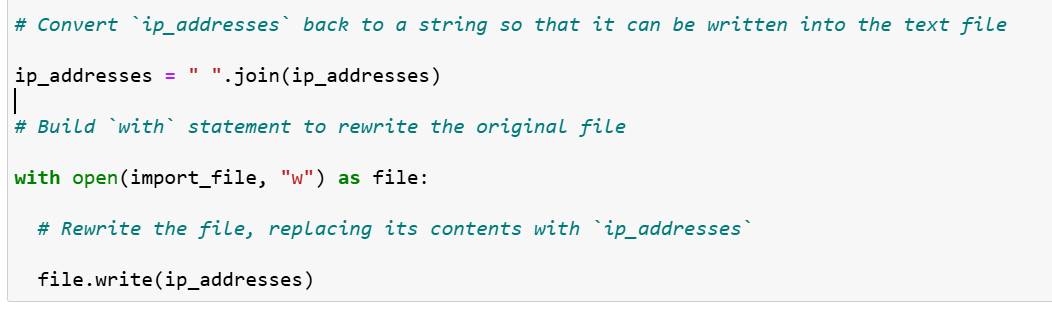
Another main part of the algorithm is going through the IP addresses that are elements in the remove\_list. To do this, I used a for loop to apply specific code statements to all elements in a sequence. The for keyword is followed by the loop variable element and the keyword in, which indicates to go through the sequence ip\_addresses and assign each value to the loop variable element.

## Remove IP addresses that are on the remove list

Because there were no duplicates in the allow list, ip\_addresses that are also contained in remove\_list, I was able to use the for loop to create 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 not found in ip\_addresses would result in an error. Within that condition, I applied .remove() to ip\_addresses To pass 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



As the last part of the algorithm, I needed to update the allow list file with the revised list of IP addresses. First I needed to convert the list back into a string, so I used the .join() method to combine 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 (" ") as a whitespace 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. By using a second argument of "w" with the open() function in my with statement to be able to open a file and write over its contents. The .write() function would then write string data to a specified file and replace the existing file content. Because I needed to write the updated allow list as a string to the file "allow\_list.txt", the restricted content will no longer be accessible to any IP addresses that were removed from the allow list. I then passed in the ip\_addresses variable as the argument to specify that the contents of the file specified in the with statement should be replaced with the data in this variable.

## Summary

By making an algorithm that removes IP addresses identified in a remove\_list variable from the "allow\_list.txt" file of approved IP addresses, it was able to This open the file, convert it to a string to be read, and then convert the string to a list stored in the variable ip\_addresses. Then it checked the IP addresses in the 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 it from ip\_addresses Variable. 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.