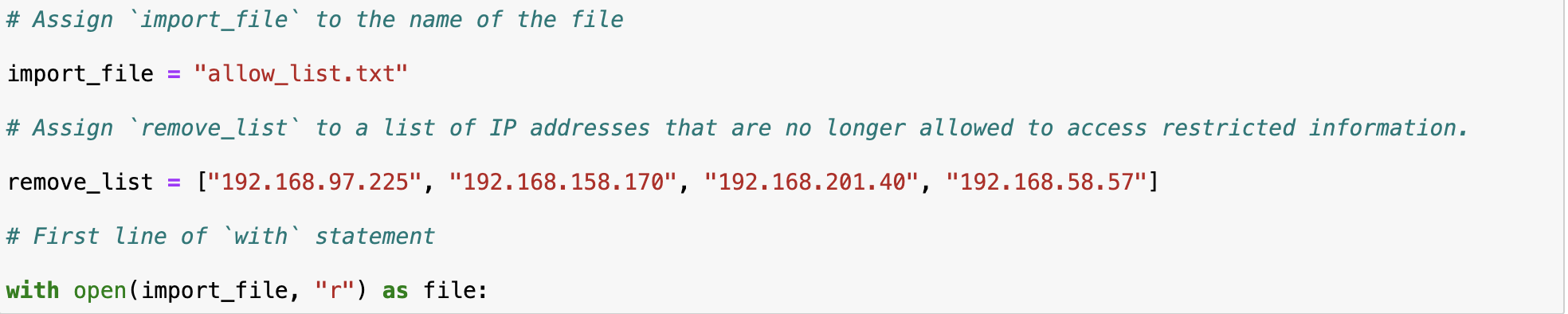
# Algorithm for file updates in Python

## Project description

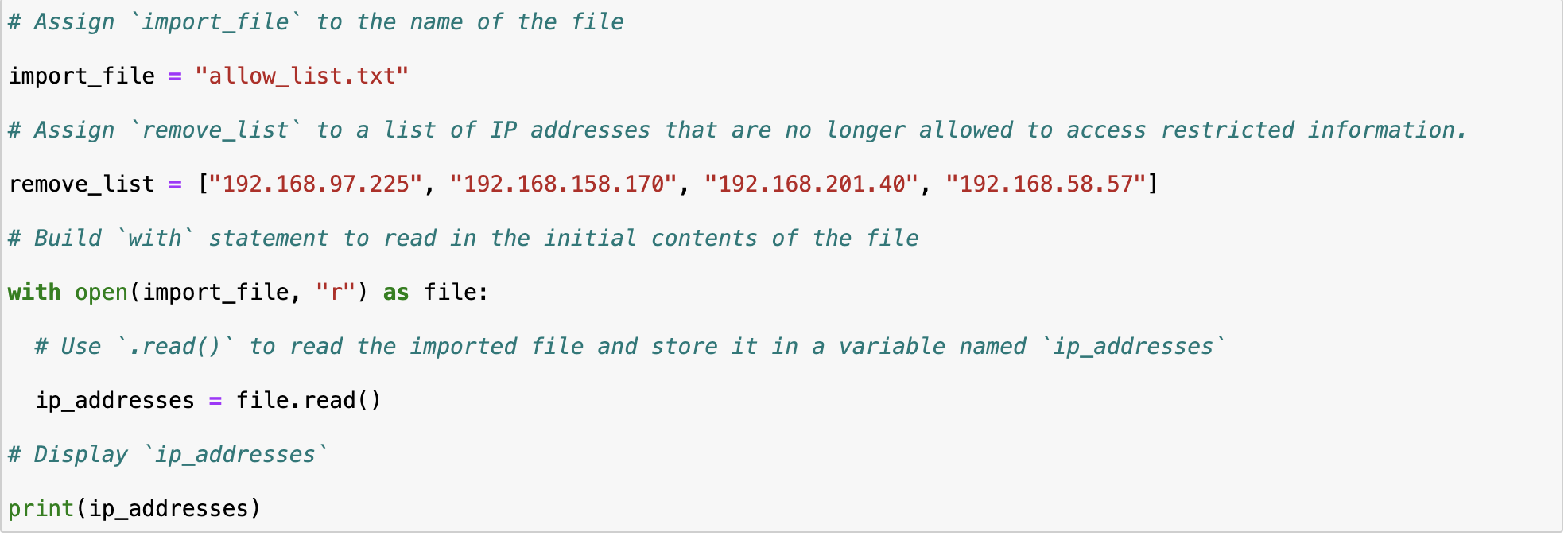
In this scenario, there is content that is restricted to an allowed list of IP addresses. These IP addresses are stored in a file called “allow\_list.txt”. There is a separate remove list that contains IP addresses that should not be allowed to access the content. I used a Python algorithm to update the “allow\_list.txt” file and remove the IP address that should no longer have access.

## Open the file that contains the allow list



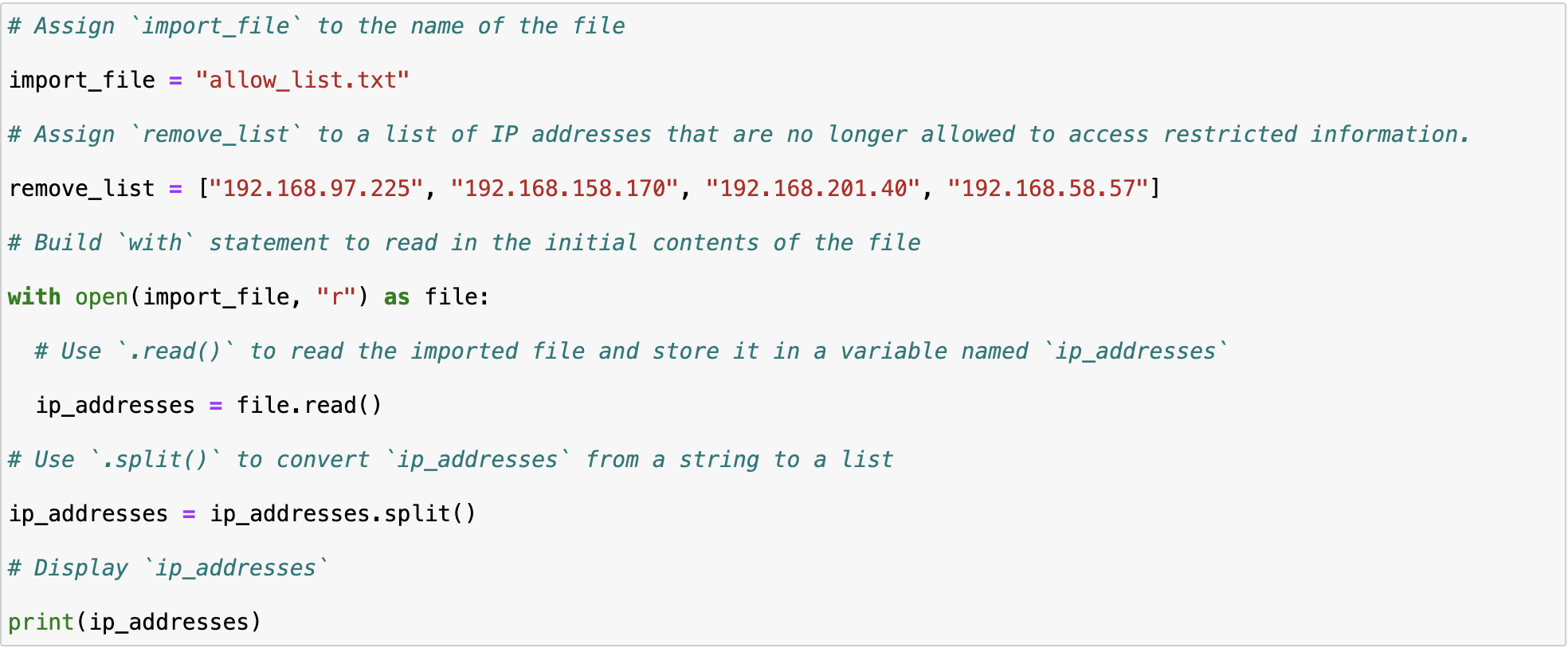
To open a file using Python, I used the following code. **With** allows me to open the file and close it after use, which is best Python practice. **Open** indicated that I am opening the file. The following parameter, **import\_file,** is the file that I want to open. **“R”** shows that I am reading the file. And lastly, **as file:** is to open the file.

## Read the file contents



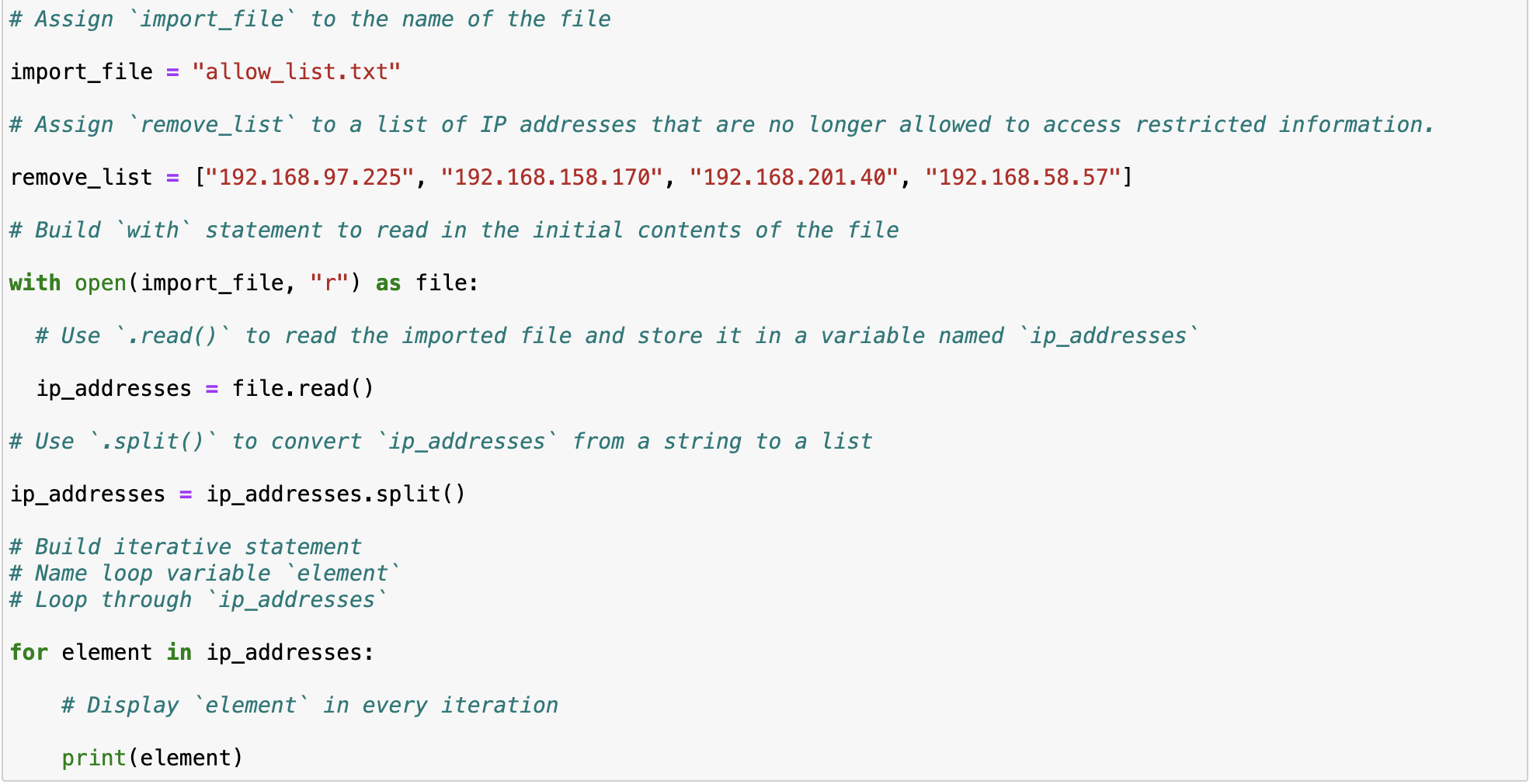
The next step is to read the contents of the file. To do this, I assign **ip\_addresses = file.read** to show that the imported file will be stored in a variable called ip\_addresses. Then, I **print(ip\_addresses)** to read the contents of the file.

## Convert the string into a list



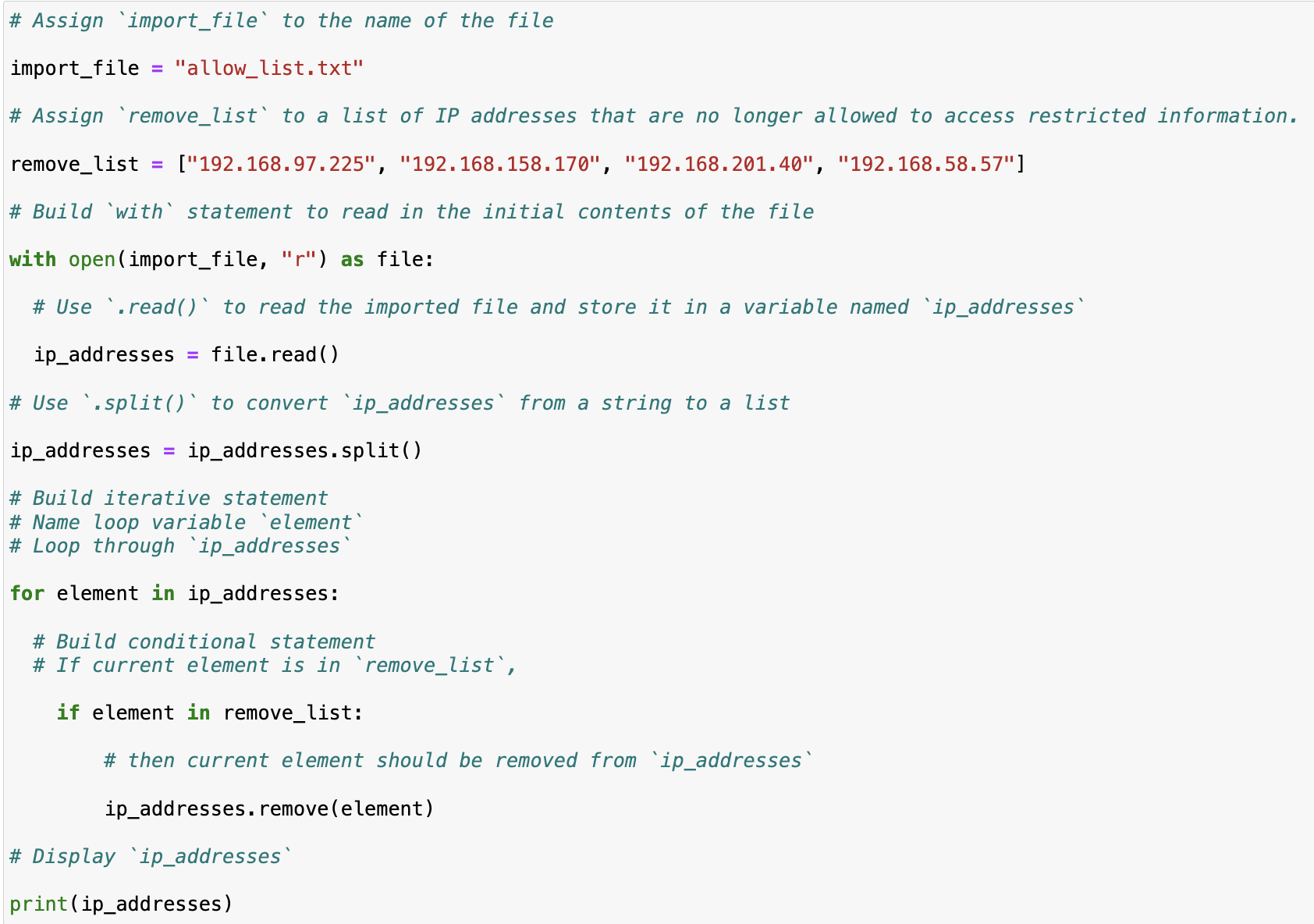
To convert the ip\_addresses to a more readable format, i used the **ip\_addresses = ip\_addresses.split().** The split() function allows me to convert data from a string to a list.

## Iterate through the remove list



To check each element in the list, I use the iterative statement **for element in ip\_addresses.** This creates a loop variable named ‘element’ and will loop through every item in ip\_addresses.

## Remove IP addresses that are on the remove list



Next, I wrote a conditional statement, **if element in remove\_list** to check if the element needs to be removed. **ip\_addresses.remove(element)** indicates that from the ip\_address list, the element needs to be removed if it is also in the remove\_list. The remove() method is possible to use because there are no duplicates in the ip\_addresses list.

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



Next, I use the .join() to convert the list back to a string: **ip\_addresses = “ “. join(ip\_addresses).** To rewrite the original file, I use the same syntax as before : **with open(import\_file, “w”) as file.** In this instance, “w” indicates that the file will be rewritten.

## Summary

I created an algorithm to remove IP address that were in the remove\_list from the “allow\_list.txt” file of allowed IP addresses. To do this, I opened a file, converted it to a string to be read, and converted the string to a list, which was stored in the variable ip\_addresses. Then I iterated through each IP address in remove\_list. During the iteration, I checked if the element was a part of the ip\_address list. If it was, I used the .remove() method to remove the element from ‘ip\_addresses’. After, I used the ‘.join()’ method to convert ‘ip\_addresses’ back into a string so I could write the contents of ‘allowed\_list.txt’ file with the new list of IP addresses.