# Algorithm for file updates in Python

## Project description

As a security professional working at a health care company, I am required to regularly update a file that identifies the employees who are allowed to access restricted content. The contents of the file are based on who is working with personal patient records. Employees are restricted access based on their IP address. There is an allow list for IP addresses permitted to sign into the restricted subnetwork. There’s also a remove list that identifies which employees I must remove from this allow list. My task consists of creating an algorithm that uses Python code to check whether the allow list contains any IP addresses identified on the remove list. If it is so, then I should be able to remove those IP addresses from the file containing the allow list.

## Open the file that contains the allow list



In order to open the file that contains the allow list as a Python algorithm, I need to first assign the name of the file called “allow\_list.txt” to the variable name called import\_file. Once there, I use the with keyword followed by the open() function its two parameters, and the as keyword followed by a variable name. I also need to place a colon ( : ) at the end of the line.

The keyword with handles errors and manages external resources when used with other functions. The open() function opens a file in Python which has the import\_file variable name with the same name of the file assigned to it as the first parameter and the “r” (reads a file) as the second parameter. I used the as keyword to assign a variable that references another object followed by the file variable name to reference the output of the open() function within the indented code block that follows it.

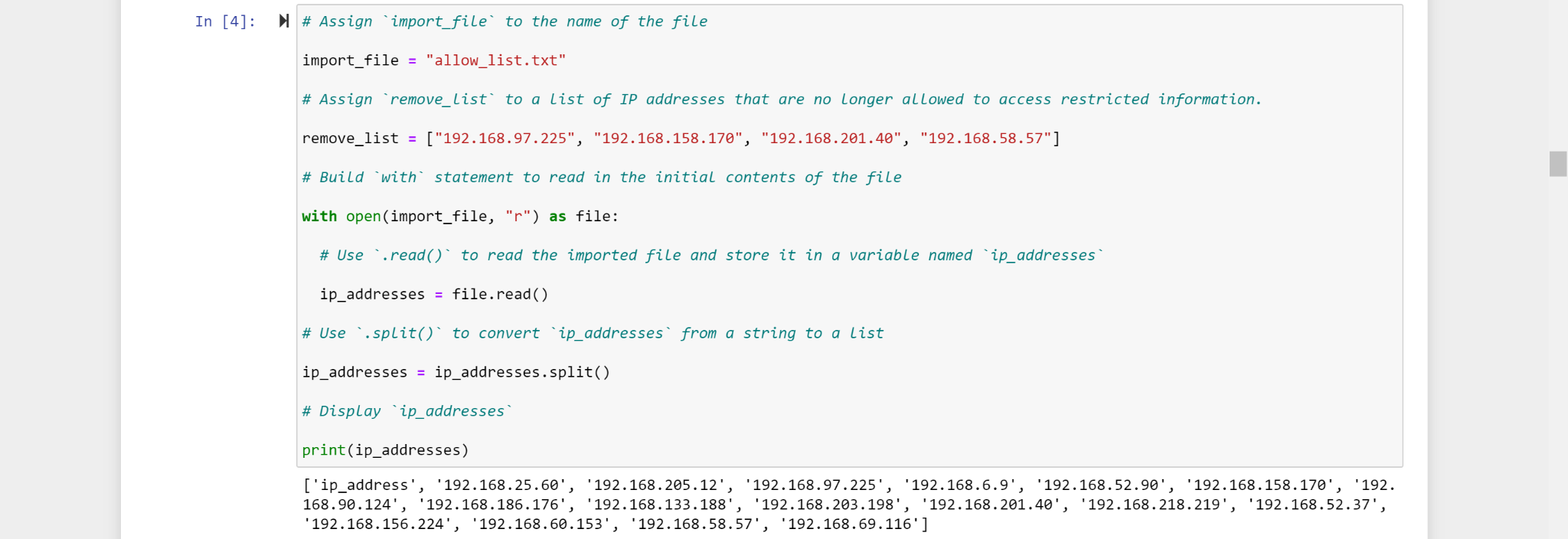
## Read the file contents





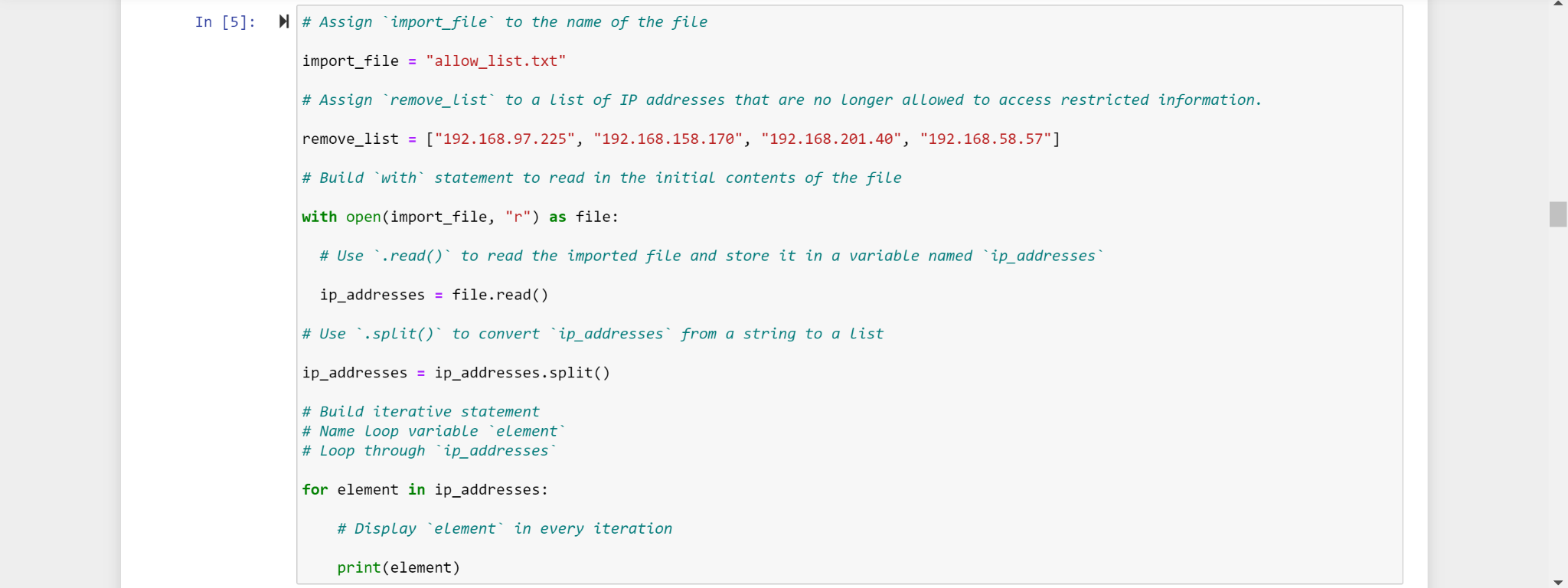
Within the with clause statement, I used the .read() method and placed it with the file variable name after the file variable name itself and placed a period between the file variable name and the .read() method as well just for the file variable name to register the .read() method properly. The .read() method is used to read the imported file and store it in a variable named ip\_addresses.

## Convert the string into a list



Once I accumulated all the ip\_addresses from the import\_file, I used the .split() method to convert all the list of IP addresses in the import\_file into their own elements which are all originally string variables but then I converted them to list. The .split() method just separates each word, number, or item mainly by space or comma or both.

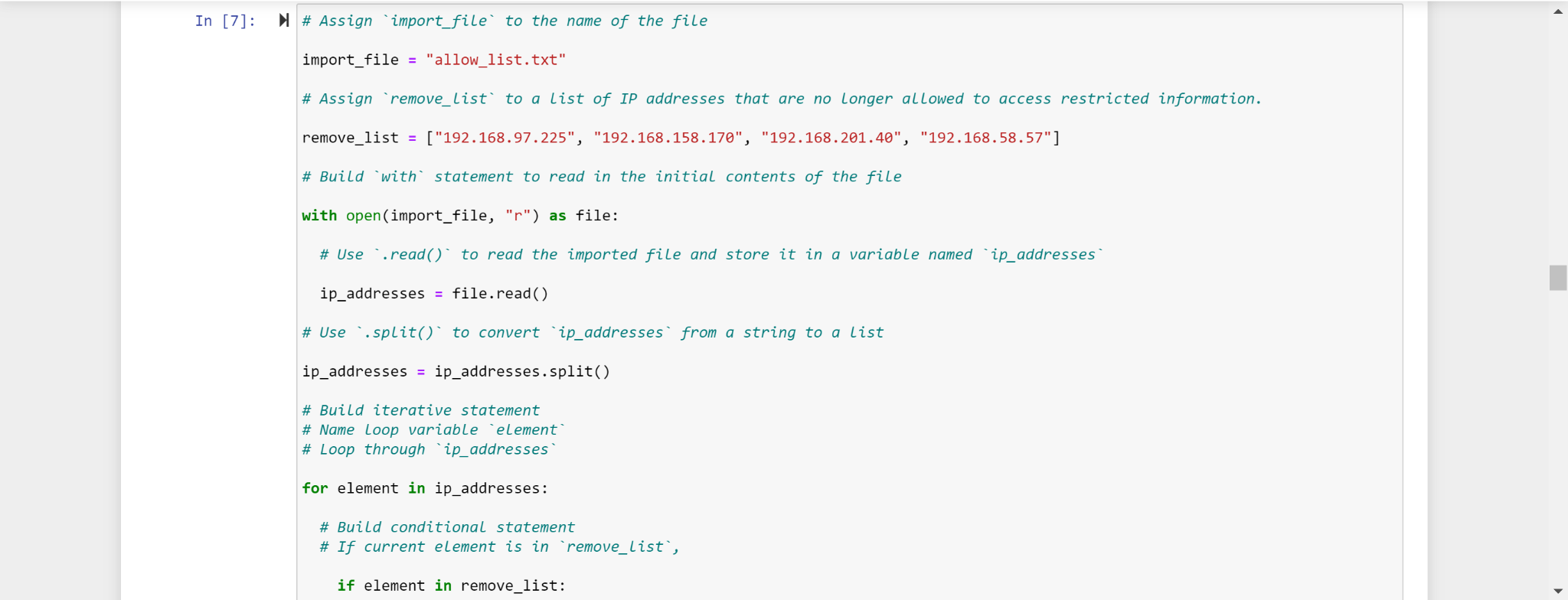
## Iterate through the remove list

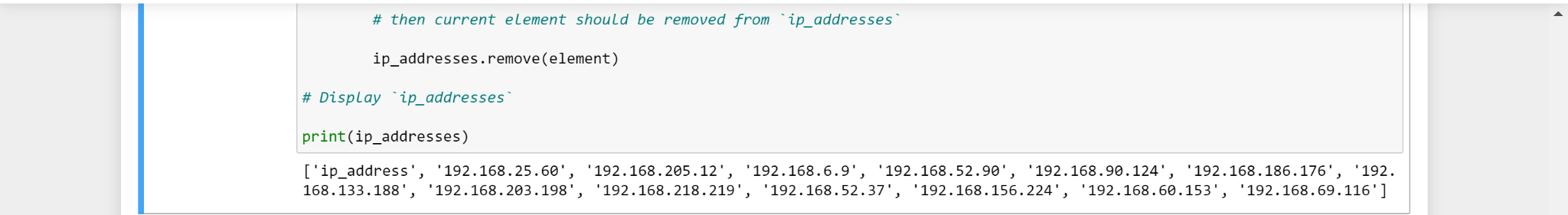




I used the for loop to iterate through the ip\_addresses for each element in it in order to print out all the elements in the list of IP addresses.

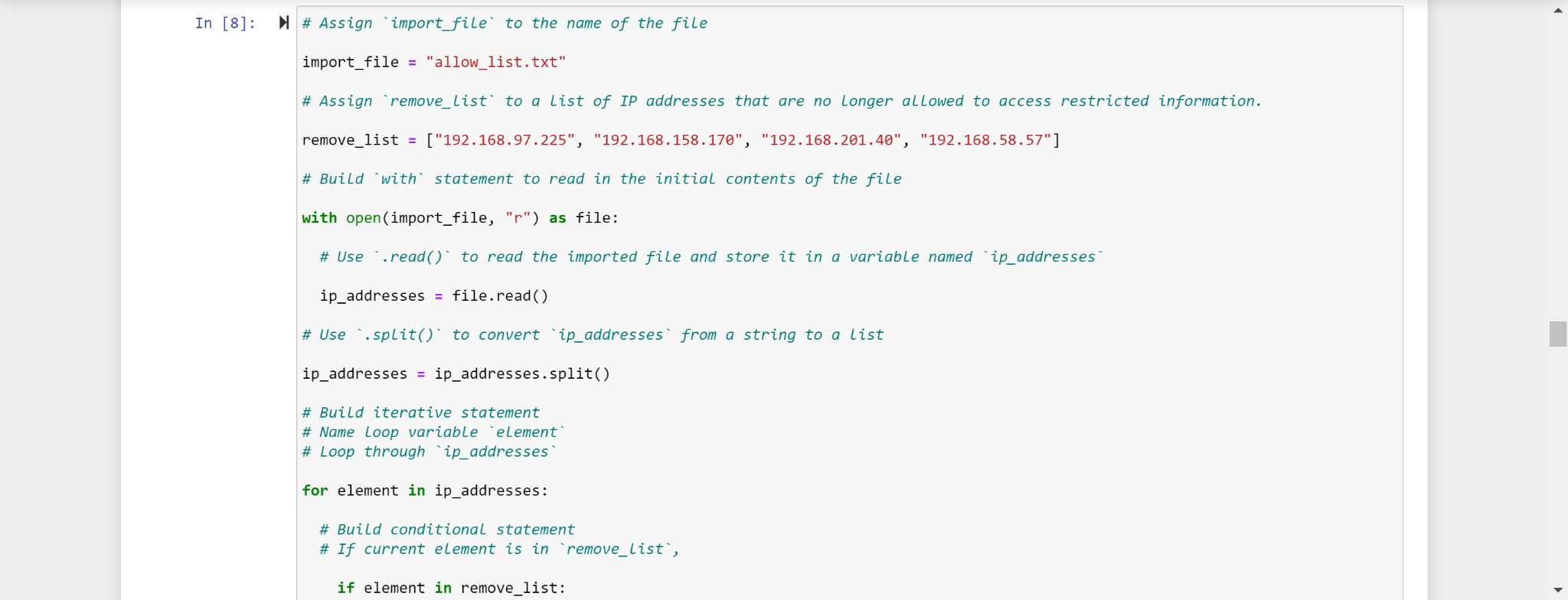
## Remove IP addresses that are on the remove list

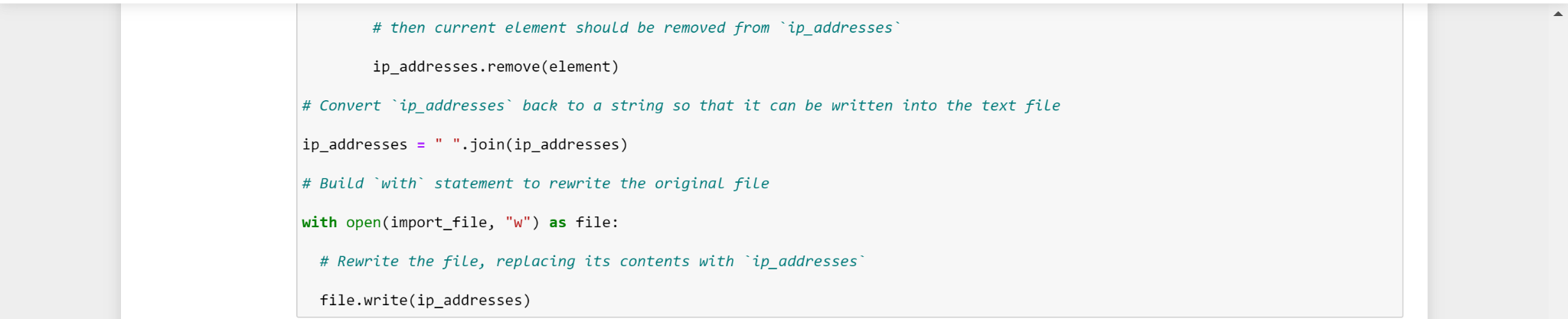




In order to remove IP addresses that are on the remove list, I have to check if an element is in the remove\_list with an if statement. If it is so, then I used the .remove() method to remove the element by placing the element variable name in the .remove() method along with the ip\_addresses being connected to the same .remove() method with a period to indicate that the current element is removed from ip\_addresses.

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





I used the .join() method to join all the lists of ip\_addresses together in a string which, in other words, does the exact opposite of what the .split() method does. Once there, I used the with open clause to write the ip\_addresses back to the import\_file with the “w” argument as the second parameter in the open() function as well as with the import\_file as the first parameter in the same open() function which leads to the file variable name being connected to the .write() method with the ip\_addresses instead of that function.

## Summary

I was successfully able to remove all the IP addresses that are on the remove list which belongs to certain employees. It is extremely important to remove all the IP addresses so that we don’t come across some malicious attackers tampering with personal patient records who would use them for their own personal gains. Now that I removed all the IP addresses, all the important data is kept safe like they should and will only grant access to those who will use the information with responsibility and maturity.