# Algorithm for file updates in Python

## Project description

Through this project I will be developing an algorithm that enables automated editing of a file containing a series of ip addresses. The algorithm will parse the given file of allowed ip addresses and compare a series of given ip addresses to be removed, and remove any matching ip addresses from the file.

## Open the file that contains the allow list

The first task facing me is to open the “allow\_list.txt” file. In order to accomplish this, I assigned the file name as a string to the variable import\_file:



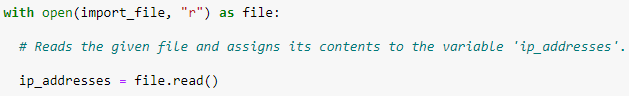
Next, I used a with statement to open the file:



In the algorithm I created, the with statement makes use of the .open() function in read mode to open the allow list file in order to read it. The purpose of opening the file is to allow me to access the IP addresses stored in the file. The with keyword helps with computer resource management by closing the file after exiting the statement, rather than keeping it constantly open, and opening new instances each time the algorithm executes. In the code with open(import\_file, “r”) as file:, the open() function has two given parameters, the file to be opened, and the indicator of what to do with the specified file. In this case since I want to read the file I used “r”. The code also uses the as keyword to assign the output of the function to the variable file while within the with statement.

## Read the file contents

In order to read the contents of the file, I used the .read() method to convert it to a string.



When using a .open() function that includes the read method, I can call .read() in the body of the with statement. The .read() method converts the contents into a string to allow for it to be read. I applied the .read() method to the file variable identified within the with statement header. Then I assigned the output of the .read() method to the variable ip\_addresses. All of this effectively allows for the contents of the given file to be converted into string format for use later in the algorithm.

## Convert the string into a list

In order to remove the individual ip addresses from the allow list, I needed to convert the contents of the ip\_addresses variable to a list, so Python will be able to recognize the contents as individual entries rather than one continuous data point. This is accomplished via use of the .split() method:



The .split() method is called by appending it to the end of the desired string variable. It works by converting the contents of the string to a list. By default, the .split() method searches for whitespaces in the string contents and divides it into new entries in the list. In this algorithm the .split() method takes the contents of ip\_addresses, converts it into a list, and then reassigns the new list to the ip\_addresses variable, effectively overwriting the previously stored string.

## Iterate through the list of IP addresses

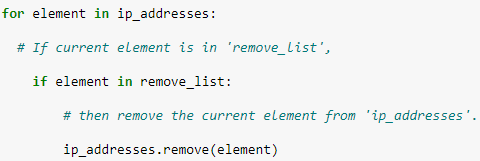
A critical part of my algorithm involves iterating through the elements in the newly created list. In order to accomplish this task, I implemented a for loop:



The for loop in Python repeats code for a specified sequence. The overall purpose of a for loop in Python is to apply specific code statements to all elements in a sequence. The for keyword initializes the loop. It is immediately followed by a local variable, in this case named element, and then the keyword in. The in keyword specifies what sequence to iterate through, in this case the contents of the list ip\_addresses.

## Remove IP addresses that are on the remove list

My algorithm requires removal of any IP addresses on the remove list from the allow list, ip\_addresses. Because ip\_addresses contains only unique entries, meaning no duplicates, I was able to accomplish this with the following code:



First, within my for loop, I created a conditional statement to evaluate whether the loop variable element was found in the remove\_list. The if keyword initializes the conditional, and is immediately followed by the hypothesis, in this case the loop variable element. Next is the keyword in which specifies what the statement should compare the hypothesis to, in this case the remove\_list variable.

Then, within the conditional I applied the .remove() method to ip\_addresses. I passed the loop variable element into the .remove() method so that the element will be removed from ip\_addresses if it is within the remove\_list variable.

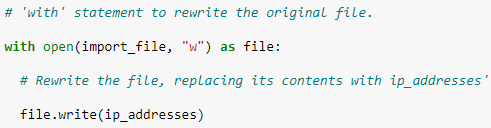
## Update the file with the revised list of IP addresses

As a final step in my algorithm, I needed to update the allow list file with the revised list of IP addresses. To do so, I needed to convert the list back into a string. I accomplish this with the .join() method:



The .join() method combines all items in an iterable, a list in this case, into a string. The .join() method is applied to a string containing characters that will separate the elements in the list once joined into a string. In this algorithm, I used .join() to create a string from ip\_addresses so I could pass it as an argument into the .write() method when writing to the “allow\_list.txt” file. I used the string (“\n”) as the separator to instruct Python to place each element on a new line. This step was not strictly necessary, but improves readability of the file once written.

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

In this case I wanted to write the updated allow list as a string to the file “allow\_list.txt”. This way, the restricted content will no longer be accessible to any IP addresses that were removed from the allow list. To rewrite the file, I appended the .write() function to the file object file that I identified within the with statement. I 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 data in this variable.

## Summary

In 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 to a string to be read, and then converting this string to a list stored in the variable ip\_addresses. I then iterated through the IP addresses in ip\_addresses, and compared each iteration to the contents of remove\_list. If it was, I applied the .remove() method to it to remove it from ip\_addresses. 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.