This program requires the file messages to be in a resource pack located in your project folder.

Code explanation :

“var is = Main.class.getResourceAsStream("/messages");”

* This line is used to get the file messages from my resource folder contained within my project folder.

“BufferedReader reader = new BufferedReader(new InputStreamReader(is, "UTF-8"));”

* I create a bufferedreader with the value is to be able to read the messages file contained in the ‘is’ variable.

“for(int i=0; (line\_str = reader.readLine()) != null; i++) { “

* I use a for loop allowing me to read the file line by line.

“if (line\_str.contains("sshd") & line\_str.contains("Accept")) {

* I filter the lines keeping only those containing the keywords “sshd” and “Accept”

“var indexStart = line\_str.indexOf(before);

var indexEnd = line\_str.indexOf(after);

var a = line\_str.substring(indexStart + before.length(), indexEnd);”

* I had trouble with making the program work when I was using the fact that the IP address is within the 11th column using a split function. While the code was getting me the 11th column or at least seemed to, I would still get from and ports from time to time instead of the IP address.
* Hence, I decided to find the index located between the keyword from and port and store that in a string variable called ‘a’.

“HashMap<String, Integer> repetitions = new HashMap<String, Integer>();”

* We introduce a hashmap that will be used to map the repetitions of the IP addresses.

‘if (repetitions.containsKey(a)) {

repetitions.put(a, repetitions.get(a) + 1);

} else {

repetitions.put(a, 1);

}’

* With this we add one to the existing value of any keys already present in the hashmap. A key = An ip address that was already added to the map.

‘for (Map.Entry<String, Integer> e : repetitions.entrySet()) {

String val\_key = e.getValue() + " " + e.getKey();

sorted\_array.add(val\_key);

}’

* We put the strings into an array. These strings contain line per line the values of the map by following the model : Value\_map Key\_map

‘sorted\_array.sort((o1, o2) -> {

var n1 = Integer.parseInt(o1.split(" ")[0]);

var n2 = Integer.parseInt(o2.split(" ")[0]);

return Integer.compare(n1, n2);

});’

* We sort the array in an ascending order.

‘System.out.println("The following are the statistics of the number of times each ip address was connected successfully.");

for(int j=0; sorted\_array.size()>j; j++){

System.out.println(sorted\_array.get(j));’

* We print the results line by line using the for loop.