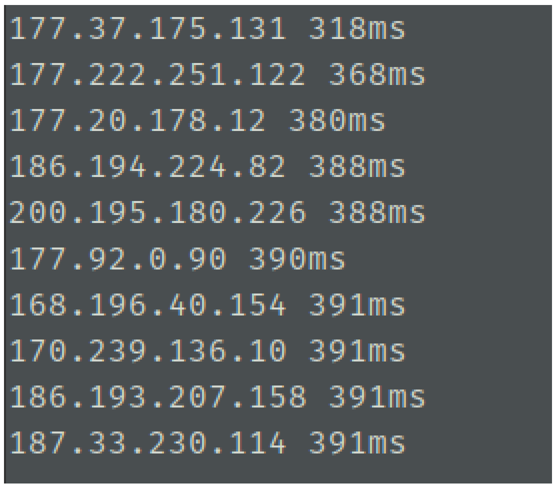
# Assignment 1

## Commands Used

### head -n 10

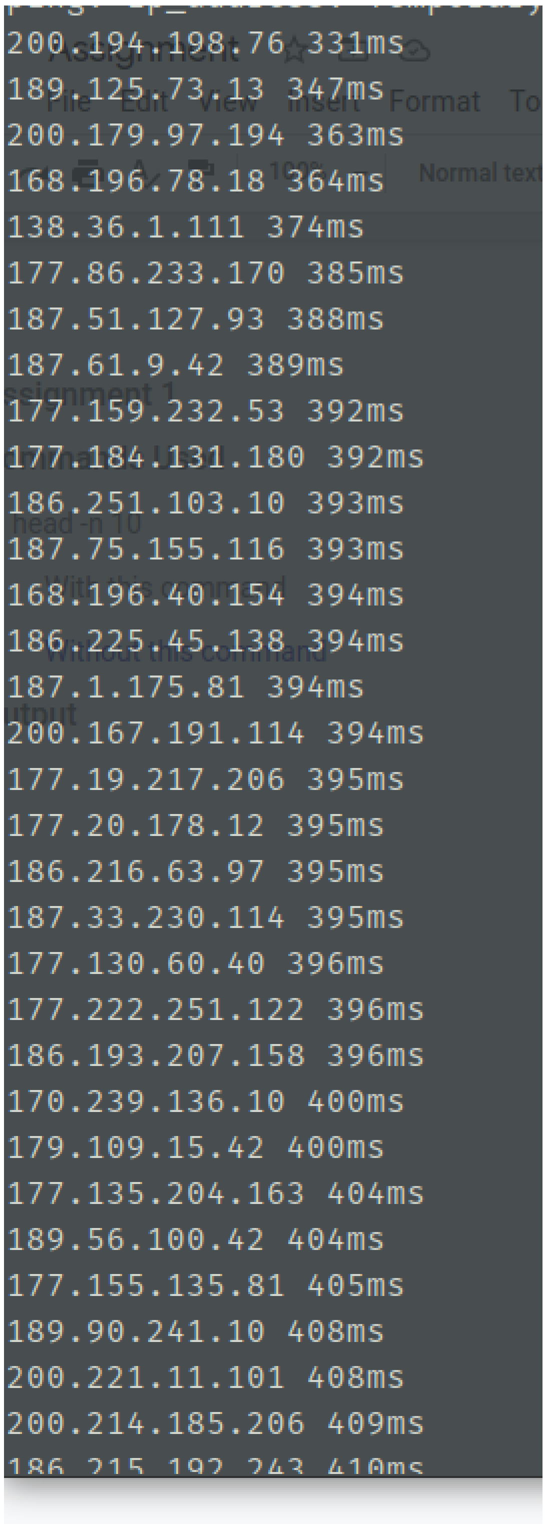
This is added so that only the first 10 lines of the input is printed. -n signifies the number of lines to be printed. If 10 is replaced with 20, then first 20 lines will be printed.

#### With this command



```
177.37.175.131 318ms
177.222.251.122 368ms
177.20.178.12 380ms
186.194.224.82 388ms
200.195.180.226 388ms
177.92.0.90 390ms
168.196.40.154 391ms
170.239.136.10 391ms
186.193.207.158 391ms
187.33.230.114 391ms
```

Without this command

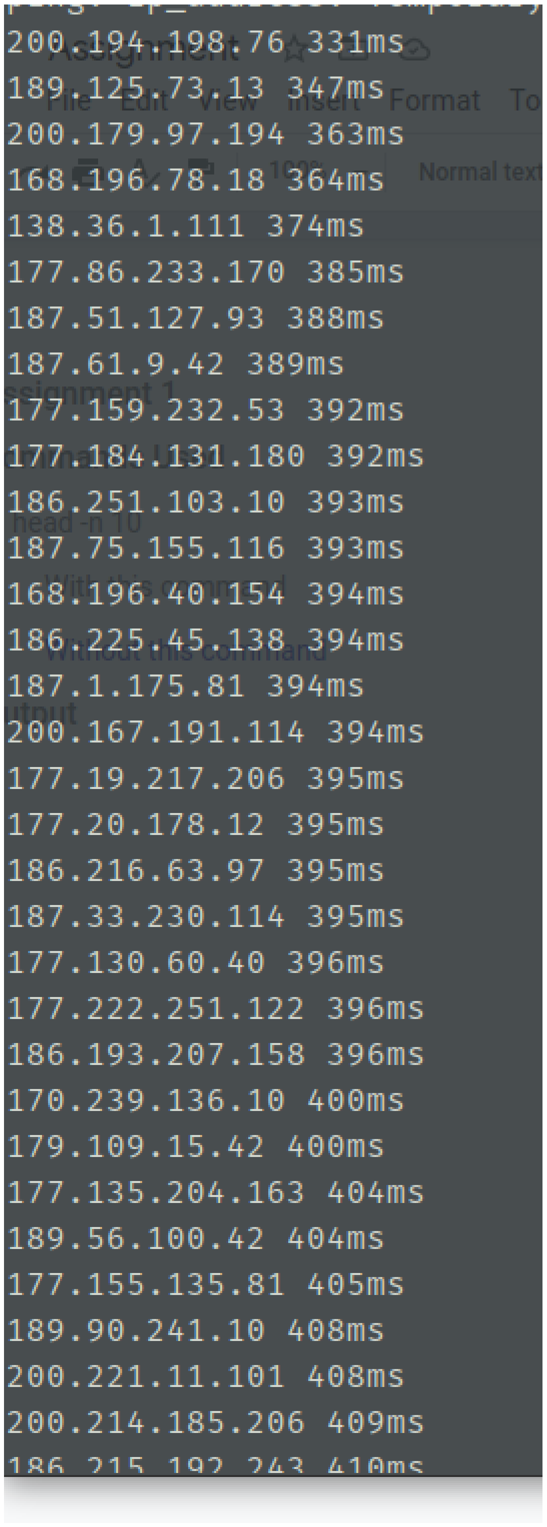


```
200.194.198.76 331ms
189.125.73.13 347ms
200.179.97.194 363ms
168.196.78.18 364ms
138.36.1.111 374ms
177.86.233.170 385ms
187.51.127.93 388ms
187.61.9.42 389ms
177.159.232.53 392ms
177.184.131.180 392ms
186.251.103.10 393ms
187.75.155.116 393ms
168.196.40.154 394ms
186.225.45.138 394ms
187.1.175.81 394ms
200.167.191.114 394ms
177.19.217.206 395ms
177.20.178.12 395ms
186.216.63.97 395ms
187.33.230.114 395ms
177.130.60.40 396ms
177.222.251.122 396ms
186.193.207.158 396ms
170.239.136.10 400ms
179.109.15.42 400ms
177.135.204.163 404ms
189.56.100.42 404ms
177.155.135.81 405ms
189.90.241.10 408ms
200.221.11.101 408ms
200.214.185.206 409ms
186.215.192.243 410ms
```

## awk '{print $2 " " $1 "ms"}'

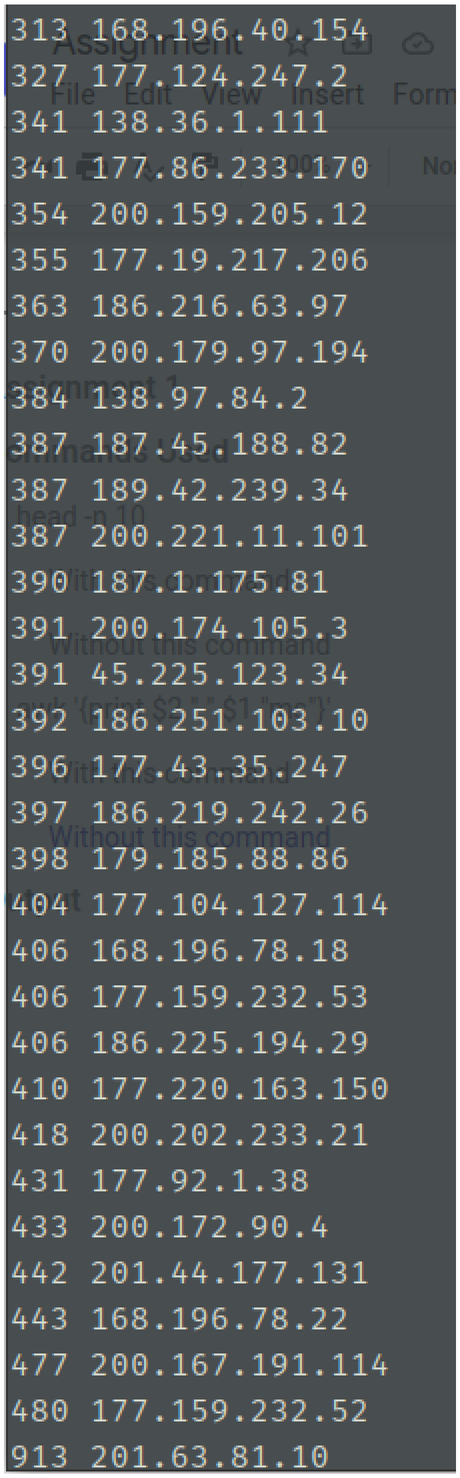
awk is a way of printing the values in the specified pattern. The text inside the curly brackets asks the shell to print the output in the following format: First print the 2nd command line argument and then add a space followed by the 1st command line argument and at the end add “ms”.

With this command



```
200.194.198.76 331ms
189.125.73.13 347ms
200.179.97.194 363ms
168.196.78.18 364ms
138.36.1.111 374ms
177.86.233.170 385ms
187.51.127.93 388ms
187.61.9.42 389ms
177.159.232.53 392ms
177.184.131.180 392ms
186.251.103.10 393ms
187.75.155.116 393ms
168.196.40.154 394ms
186.225.45.138 394ms
187.1.175.81 394ms
200.167.191.114 394ms
177.19.217.206 395ms
177.20.178.12 395ms
186.216.63.97 395ms
187.33.230.114 395ms
177.130.60.40 396ms
177.222.251.122 396ms
186.193.207.158 396ms
170.239.136.10 400ms
179.109.15.42 400ms
177.135.204.163 404ms
189.56.100.42 404ms
177.155.135.81 405ms
189.90.241.10 408ms
200.221.11.101 408ms
200.214.185.206 409ms
186.215.192.243 410ms
```

Without this command



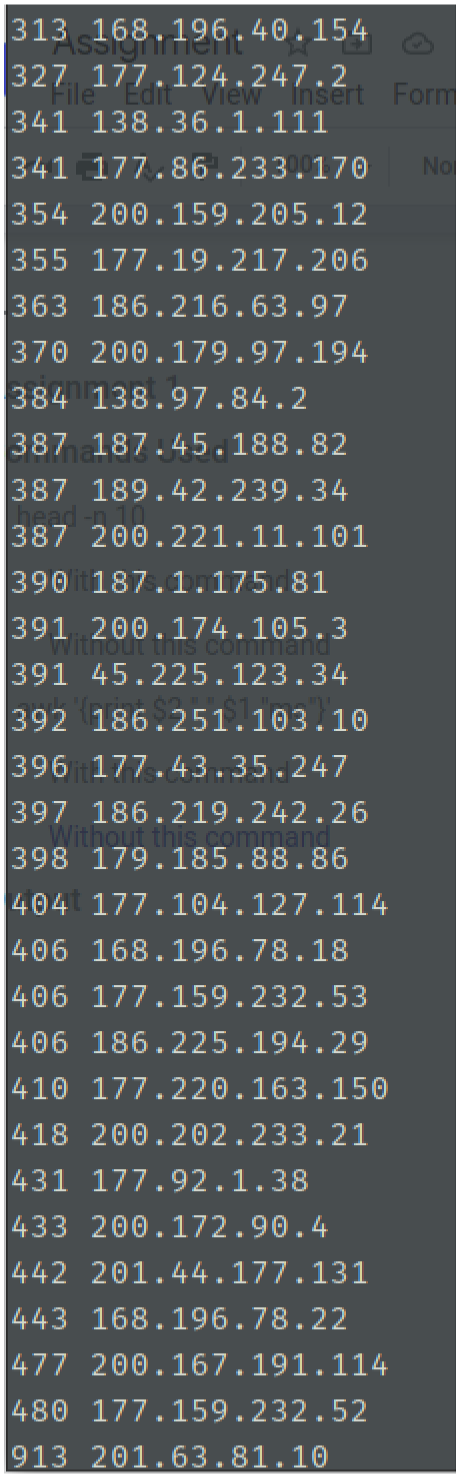
```
313 168.196.40.154
327 177.124.247.2
341 138.36.1.111
341 177.86.233.170
354 200.159.205.12
355 177.19.217.206
363 186.216.63.97
370 200.179.97.194
384 138.97.84.2
387 187.45.188.82
387 189.42.239.34
387 200.221.11.101
390 187.1.175.81
391 200.174.105.3
391 45.225.123.34
392 186.251.103.10
396 177.43.35.247
397 186.219.242.26
398 179.185.88.86
404 177.104.127.114
406 168.196.78.18
406 177.159.232.53
406 186.225.194.29
410 177.220.163.150
418 200.202.233.21
431 177.92.1.38
433 200.172.90.4
442 201.44.177.131
443 168.196.78.22
477 200.167.191.114
480 177.159.232.52
913 201.63.81.10
```

Note: The time is the first argument and IP address is the 2nd.

## sort -n

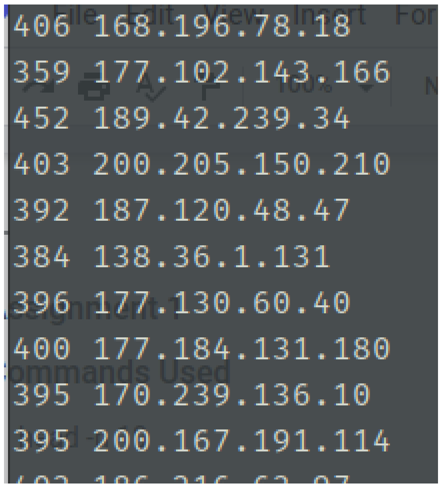
This command is used to print the input in a sorted manner. -n is used to specify that the shell is supposed to compare the string numerical values i.e the time in this case.

With this command



```
313 168.196.40.154
327 177.124.247.2
341 138.36.1.111
341 177.86.233.170
354 200.159.205.12
355 177.19.217.206
363 186.216.63.97
370 200.179.97.194
384 138.97.84.2
387 187.45.188.82
387 189.42.239.34
387 200.221.11.101
390 187.1.175.81
391 200.174.105.3
391 45.225.123.34
392 186.251.103.10
396 177.43.35.247
397 186.219.242.26
398 179.185.88.86
404 177.104.127.114
406 168.196.78.18
406 177.159.232.53
406 186.225.194.29
410 177.220.163.150
418 200.202.233.21
431 177.92.1.38
433 200.172.90.4
442 201.44.177.131
443 168.196.78.22
477 200.167.191.114
480 177.159.232.52
913 201.63.81.10
```

### Without this command

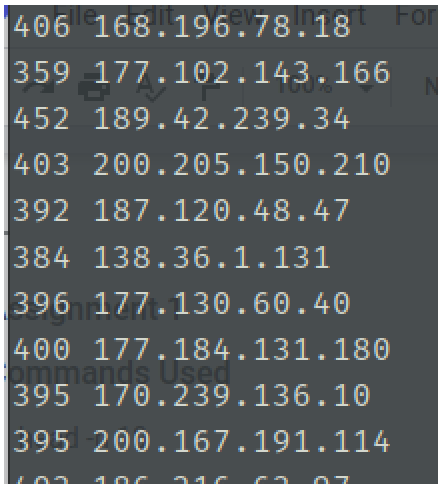


```
406 168.196.78.18
359 177.102.143.166
452 189.42.239.34
403 200.205.150.210
392 187.120.48.47
384 138.36.1.131
396 177.130.60.40
400 177.184.131.180
395 170.239.136.10
395 200.167.191.114
```

## awk '{print substr($7, 6, length($7)) " " substr($4, 1, length($4) -1)}'

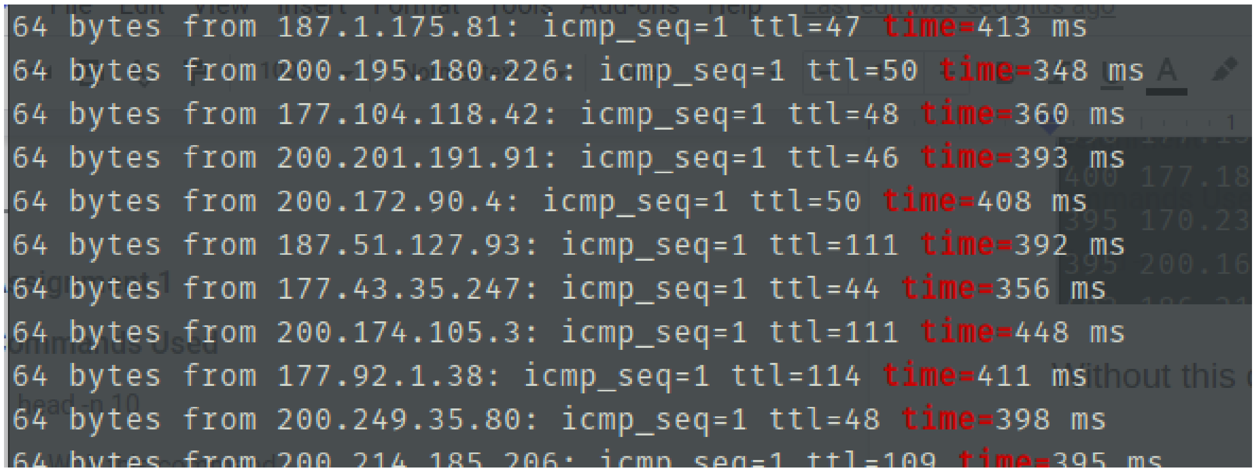
awk command is again used to structure the input so that only the useful information is printed. The print command is used to print the 7th and 4th substrings. The 7th command line argument is in the following format: time=123. We only need ‘123’ in this case. Hence we only print the substring from the 6th character.

### With this command



```
406 168.196.78.18
359 177.102.143.166
452 189.42.239.34
403 200.205.150.210
392 187.120.48.47
384 138.36.1.131
396 177.130.60.40
400 177.184.131.180
395 170.239.136.10
395 200.167.191.114
```

## Without this command



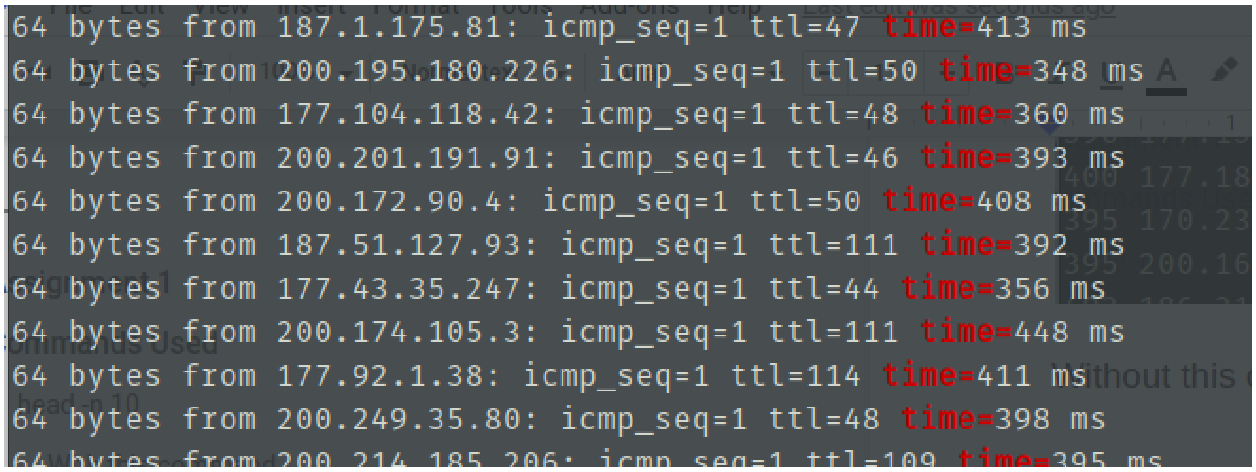
```
64 bytes from 187.1.175.81: icmp_seq=1 ttl=47 time=413 ms
64 bytes from 200.195.180.226: icmp_seq=1 ttl=50 time=348 ms
64 bytes from 177.104.118.42: icmp_seq=1 ttl=48 time=360 ms
64 bytes from 200.201.191.91: icmp_seq=1 ttl=46 time=393 ms
64 bytes from 200.172.90.4: icmp_seq=1 ttl=50 time=408 ms
64 bytes from 187.51.127.93: icmp_seq=1 ttl=111 time=392 ms
64 bytes from 177.43.35.247: icmp_seq=1 ttl=44 time=356 ms
64 bytes from 200.174.105.3: icmp_seq=1 ttl=111 time=448 ms
64 bytes from 177.92.1.38: icmp_seq=1 ttl=114 time=411 ms
64 bytes from 200.249.35.80: icmp_seq=1 ttl=48 time=398 ms
64 bytes from 200.214.185.206: icmp_seq=1 ttl=109 time=395 ms
```

1 2 3 4 5 6 7 8

## grep ‘time=’

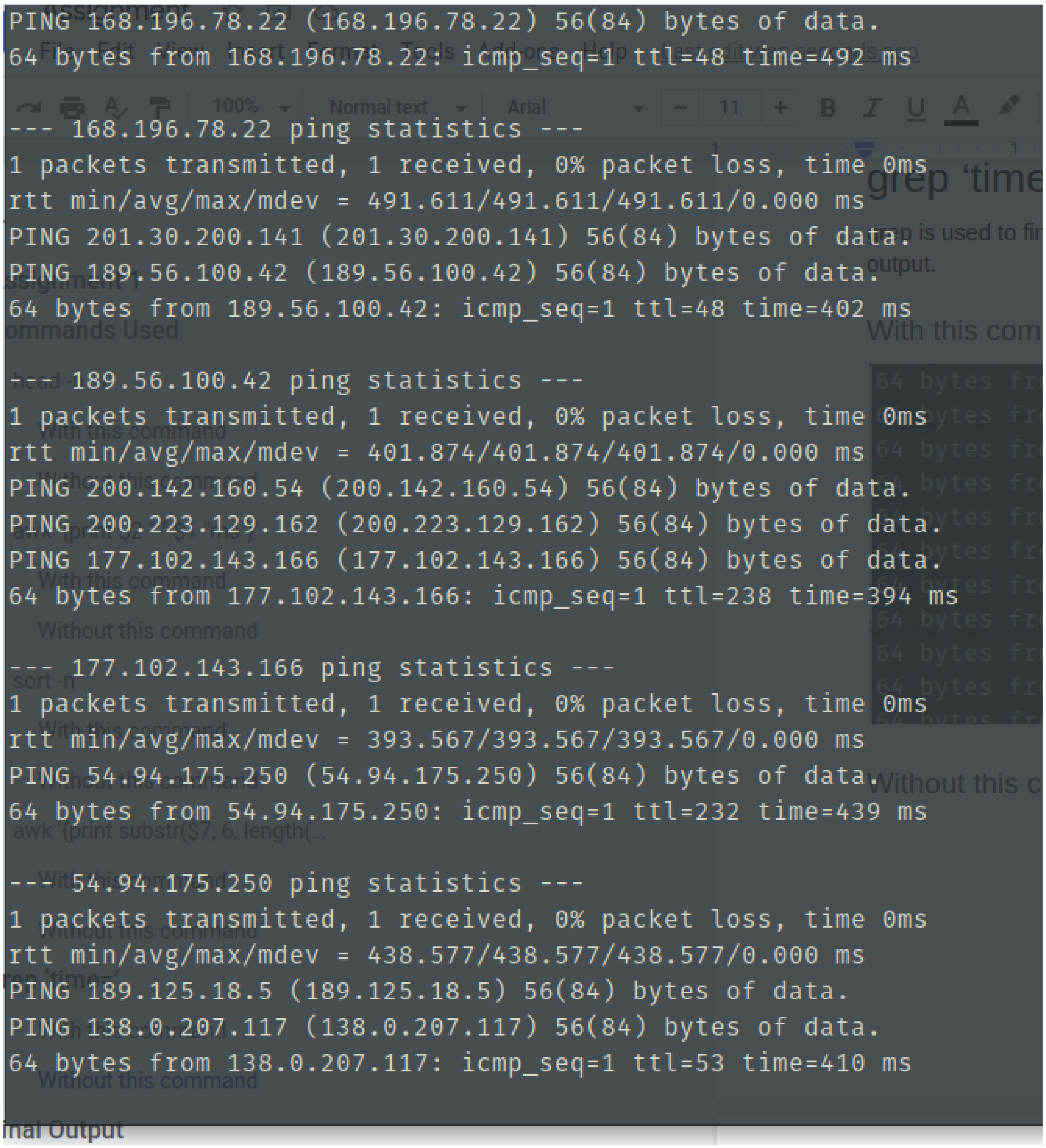
grep is used to find the lines containing the matching string. This is one way of filtering the output.

## With this command



```
64 bytes from 187.1.175.81: icmp_seq=1 ttl=47 time=413 ms
64 bytes from 200.195.180.226: icmp_seq=1 ttl=50 time=348 ms
64 bytes from 177.104.118.42: icmp_seq=1 ttl=48 time=360 ms
64 bytes from 200.201.191.91: icmp_seq=1 ttl=46 time=393 ms
64 bytes from 200.172.90.4: icmp_seq=1 ttl=50 time=408 ms
64 bytes from 187.51.127.93: icmp_seq=1 ttl=111 time=392 ms
64 bytes from 177.43.35.247: icmp_seq=1 ttl=44 time=356 ms
64 bytes from 200.174.105.3: icmp_seq=1 ttl=111 time=448 ms
64 bytes from 177.92.1.38: icmp_seq=1 ttl=114 time=411 ms
64 bytes from 200.249.35.80: icmp_seq=1 ttl=48 time=398 ms
64 bytes from 200.214.185.206: icmp_seq=1 ttl=109 time=395 ms
```

Without this command



```
PING 168.196.78.22 (168.196.78.22) 56(84) bytes of data.
64 bytes from 168.196.78.22: icmp_seq=1 ttl=48 time=492 ms

--- 168.196.78.22 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 491.611/491.611/491.611/0.000 ms
PING 201.30.200.141 (201.30.200.141) 56(84) bytes of data.
PING 189.56.100.42 (189.56.100.42) 56(84) bytes of data.
64 bytes from 189.56.100.42: icmp_seq=1 ttl=48 time=402 ms

--- 189.56.100.42 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 401.874/401.874/401.874/0.000 ms
PING 200.142.160.54 (200.142.160.54) 56(84) bytes of data.
PING 200.223.129.162 (200.223.129.162) 56(84) bytes of data.
PING 177.102.143.166 (177.102.143.166) 56(84) bytes of data.
64 bytes from 177.102.143.166: icmp_seq=1 ttl=238 time=394 ms

--- 177.102.143.166 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 393.567/393.567/393.567/0.000 ms
PING 54.94.175.250 (54.94.175.250) 56(84) bytes of data.
64 bytes from 54.94.175.250: icmp_seq=1 ttl=232 time=439 ms

--- 54.94.175.250 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 438.577/438.577/438.577/0.000 ms
PING 189.125.18.5 (189.125.18.5) 56(84) bytes of data.
PING 138.0.207.117 (138.0.207.117) 56(84) bytes of data.
64 bytes from 138.0.207.117: icmp_seq=1 ttl=53 time=410 ms
```

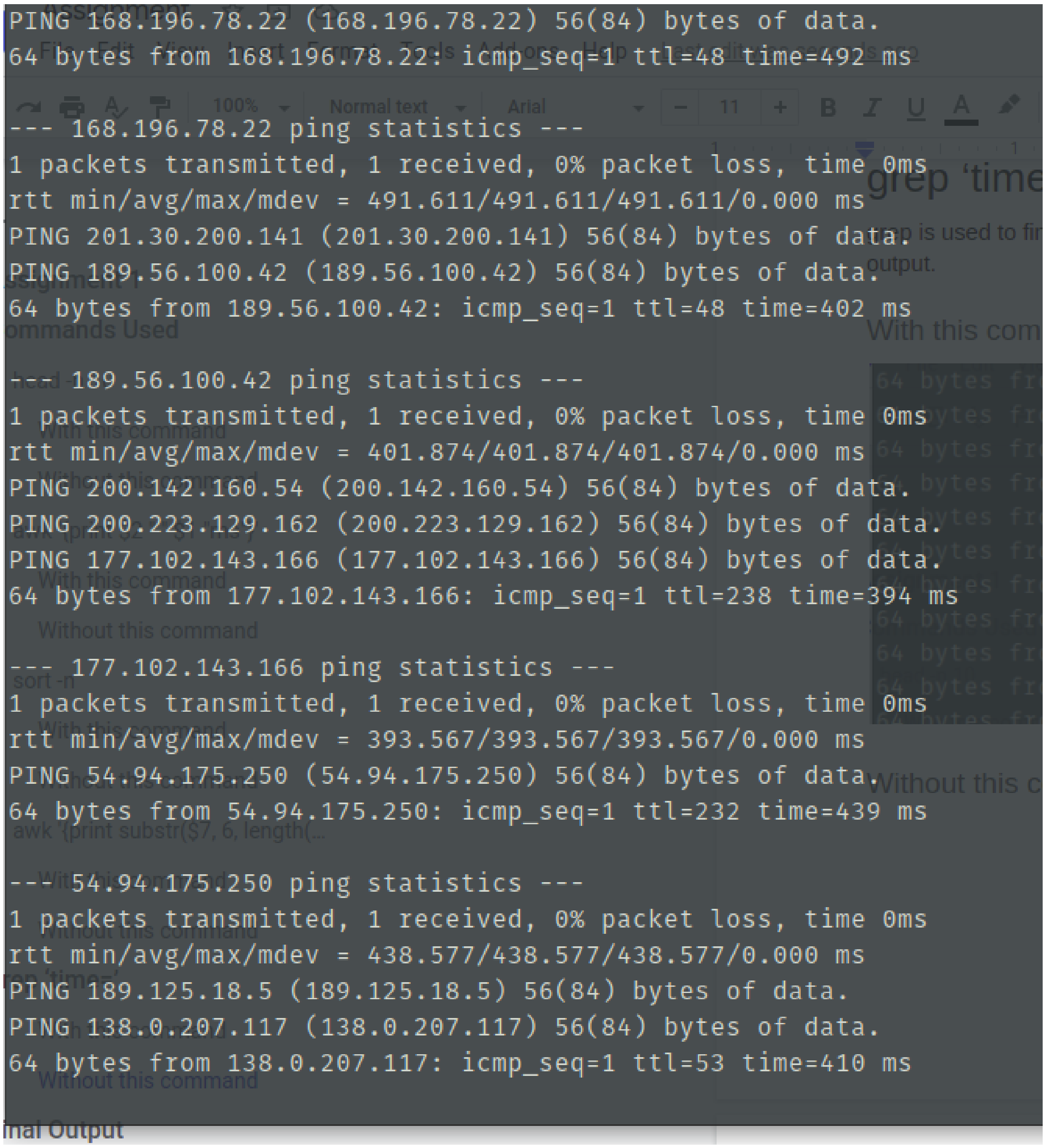
Note: The last line in each section is what is eventually filtered out.

## xargs -i timeout 1 ping -c1 -w 1 {}

xargs is used to build command line argument using the standard input received from the previous command. -i is used to replace the string with the STDIN received.
This command is then set to timeout if it runs for more than 1 second.

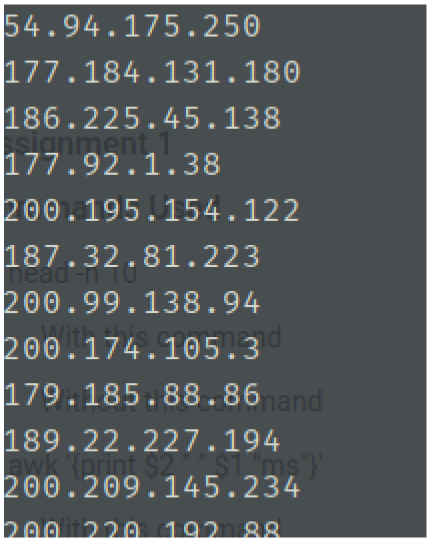
ping is used to send and receive packets from websites. -c is used to specify how many packets needs to be exchanged. -w is the deadline. i.e. if the specified number of packets are not exchanged within the limit, the command will exit after the deadline is hit.
{} - this is where the IP addresses are entered.

## With this command



```
PING 168.196.78.22 (168.196.78.22) 56(84) bytes of data.
64 bytes from 168.196.78.22: icmp_seq=1 ttl=48 time=492 ms

--- 168.196.78.22 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 491.611/491.611/491.611/0.000 ms
PING 201.30.200.141 (201.30.200.141) 56(84) bytes of data.
PING 189.56.100.42 (189.56.100.42) 56(84) bytes of data.
64 bytes from 189.56.100.42: icmp_seq=1 ttl=48 time=402 ms

--- 189.56.100.42 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 401.874/401.874/401.874/0.000 ms
PING 200.142.160.54 (200.142.160.54) 56(84) bytes of data.
PING 200.223.129.162 (200.223.129.162) 56(84) bytes of data.
PING 177.102.143.166 (177.102.143.166) 56(84) bytes of data.
64 bytes from 177.102.143.166: icmp_seq=1 ttl=238 time=394 ms

--- 177.102.143.166 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 393.567/393.567/393.567/0.000 ms
PING 54.94.175.250 (54.94.175.250) 56(84) bytes of data.
64 bytes from 54.94.175.250: icmp_seq=1 ttl=232 time=439 ms

--- 54.94.175.250 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 438.577/438.577/438.577/0.000 ms
PING 189.125.18.5 (189.125.18.5) 56(84) bytes of data.
PING 138.0.207.117 (138.0.207.117) 56(84) bytes of data.
64 bytes from 138.0.207.117: icmp_seq=1 ttl=53 time=410 ms
```

### Without this command



```
54.94.175.250
177.184.131.180
186.225.45.138
177.92.1.38
200.195.154.122
187.32.81.223
200.99.138.94
200.174.105.3
179.185.88.86
189.22.227.194
200.209.145.234
200.220.192.88
```

## tail -n 50

tail is used to print the last n elements of the input. -n is used to specify the number of lines to be printed. 50 in this case.

### With this command

Last 50 lines are printed.

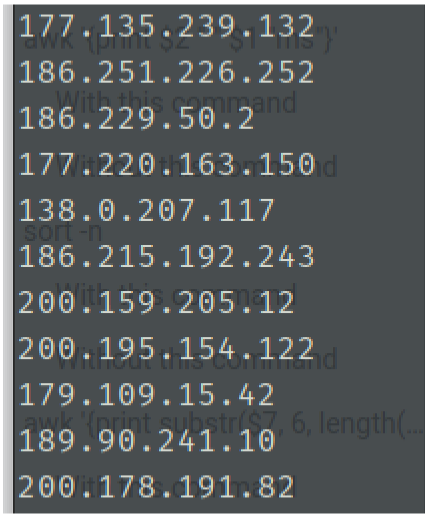
### Without this command

More than 50 lines are printed. (Not adding the screenshot as it is too long)

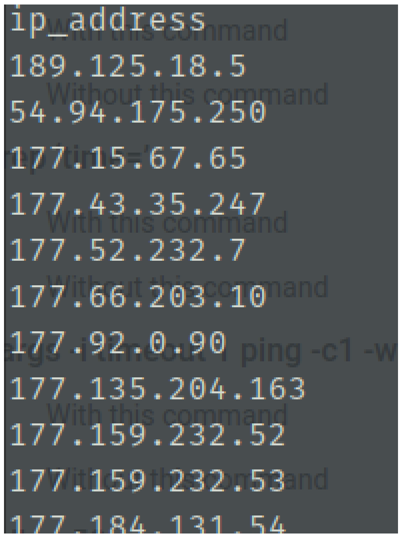
## shuf

This is used to shuffle the lines.

With this command


```
177.135.239.132
186.251.226.252
186.229.50.2
177.220.163.150
138.0.207.117
186.215.192.243
200.159.205.12
200.195.154.122
179.109.15.42
189.90.241.10
200.178.191.82
```

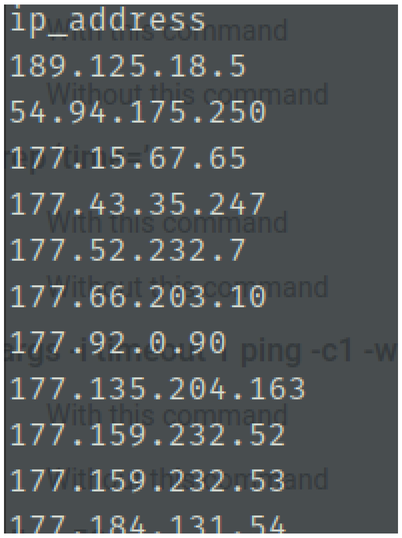
Without this command


```
ip_address
189.125.18.5
54.94.175.250
177.15.67.65
177.43.35.247
177.52.232.7
177.66.203.10
177.92.0.90
177.135.204.163
177.159.232.52
177.159.232.53
177.184.131.54
```

## cut -d, -f1

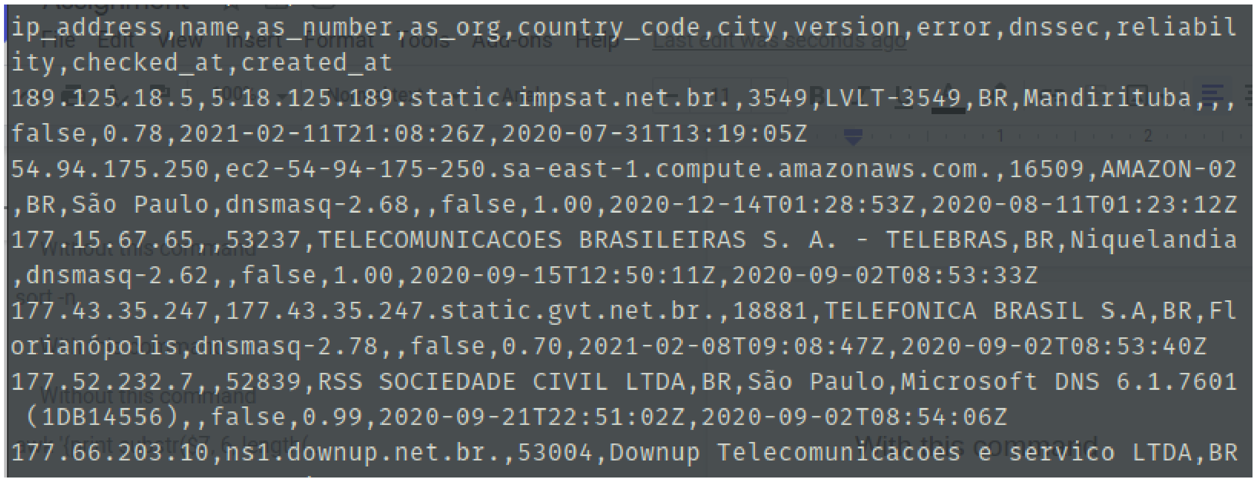
cut is used to remove the unwanted information from the input. In this case we are only reading the first column using -f1. -d is used to specify that a delimiter(‘,’ in this case ) must be used in place of tab spaces.

With this command



```
ip_address
189.125.18.5
54.94.175.250
177.15.67.65
177.43.35.247
177.52.232.7
177.66.203.10
177.92.0.90
177.135.204.163
177.159.232.52
177.159.232.53
177.184.131.54
```

Without this command



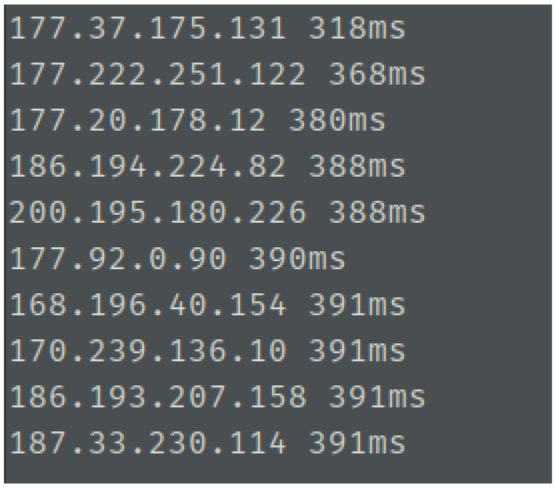
```
ip_address,name,as_number,as_org,country_code,city,version,error,dnssec,reliabil
ity,checked_at,created_at
189.125.18.5,5.18.125.189.static.impsat.net.br.,3549,LVLT-3549,BR,Mandirituba,,,
false,0.78,2021-02-11T21:08:26Z,2020-07-31T13:19:05Z
54.94.175.250,ec2-54-94-175-250.sa-east-1.compute.amazonaws.com.,16509,AMAZON-02
,BR,São Paulo,dnsmasq-2.68,,false,1.00,2020-12-14T01:28:53Z,2020-08-11T01:23:12Z
177.15.67.65,,53237,TELECOMUNICACOES BRASILEIRAS S. A. - TELEBRAS,BR,Niquelandia
,dnsmasq-2.62,,false,1.00,2020-09-15T12:50:11Z,2020-09-02T08:53:33Z
177.43.35.247,177.43.35.247.static.gvt.net.br.,18881,TELEFONICA BRASIL S.A,BR,Fl
orianópolis,dnsmasq-2.78,,false,0.70,2021-02-08T09:08:47Z,2020-09-02T08:53:40Z
177.52.232.7,,52839,RSS SOCIEDADE CIVIL LTDA,BR,São Paulo,Microsoft DNS 6.1.7601
 (1DB14556),,false,0.99,2020-09-21T22:51:02Z,2020-09-02T08:54:06Z
177.66.203.10,ns1.downup.net.br.,53004,Downup Telecomunicacoes e servico LTDA,BR
```

Note: The above file is a csv file.

## curl -s [http://public-dns.info/nameserver/br.csv](http://public-dns.info/nameserver/br.csv)

curl is used to download/transfer data. -s is used to hide the progress bar and error messages.

## Final Output



```
177.37.175.131 318ms
177.222.251.122 368ms
177.20.178.12 380ms
186.194.224.82 388ms
200.195.180.226 388ms
177.92.0.90 390ms
168.196.40.154 391ms
170.239.136.10 391ms
186.193.207.158 391ms
187.33.230.114 391ms
```

The command as a whole is used to, first, download the csv file, then only send the first column (IP addresses) forward. This dada is then shuffled and the last 50 rows are then sent to the xargs command. Each IP address and pinged and the time required to exchange packets is noted. The rows containing the time are filtered out and sorted in ascending order. Then the top 10 rows are displayed.