

## Ping Program Design Document

---

*Note: The code and (in-line) documentation for this project can also be found at the following GitHub link: <https://github.com/ariblack17/ping-program>.*

### Description

**The Program:** The Ping program for this project is intended to be a manual implementation of the ping function and a few of its features. The user, from the command line, calls the *ping* method for some host server. As a result, their system sends an ICMP packet (echo request) to the server, containing the current time. The server sends a response ICMP (echo response) back to the client, and the client is then able to calculate the RTT from the time difference between requests and responses, if the packet received was an expected ICMP. Packets will be “timed out” if they have not been received before their TTL expires (specified by the *-t* command line option). This function repeats until the number of sent packets (specified by the *-c* command line option) is reached, and some relevant statistics (average RTT, min RTT, max RTT, ...) about the session is output to the client’s console.

The command to run the *ping* method is as follows:  
`sudo python3 ICMP.py ping <host_ip> <options>`

### Trade Offs

**Parsing ICMP Error Codes:** In order to parse the error codes received by ICMP messages from the server, I had to manually map each possible error type and code to its respective description, for both IPv4 and IPv6 addresses. While this solution is functional, it is also rather inefficient.

**Catching Errors:** While the code design meets the specifications required by the assignment, there could be more done to improve the project’s robustness and make it more similar to the actual built-in Ping function.

**User Interface:** Given the starter code, I felt that it was most natural to have the user interact with the client-side program directly via the command line. In previous versions of the program, I had developed a client-side driver program that the user would run instead of using the command line interface; this, however, seemed to add some unnecessary complexity to the code.

## Extensions

***Increasing Similarity to Ping:*** The most obvious extension to this project would be to implement more of the functionalities present in the built-in ping function into the manually-implemented version. Some of these may include the *-i* (wait some number of seconds before sending each packet), *-o* (stop just after sending one packet), or the *-q* (quiet, so display nothing except the post-ping summary) options.

***Streamlining Code Structure:*** Currently, there is a large amount of logic located under an *if \_\_name\_\_ == "\_\_main\_\_":* block in the main code file. Again, this part of the assignment is functional but perhaps not optimal. Rewriting and refactoring the code here to better fit the structure of the rest of the code (especially the starter code and helper functions) would improve readability.

## Test Cases

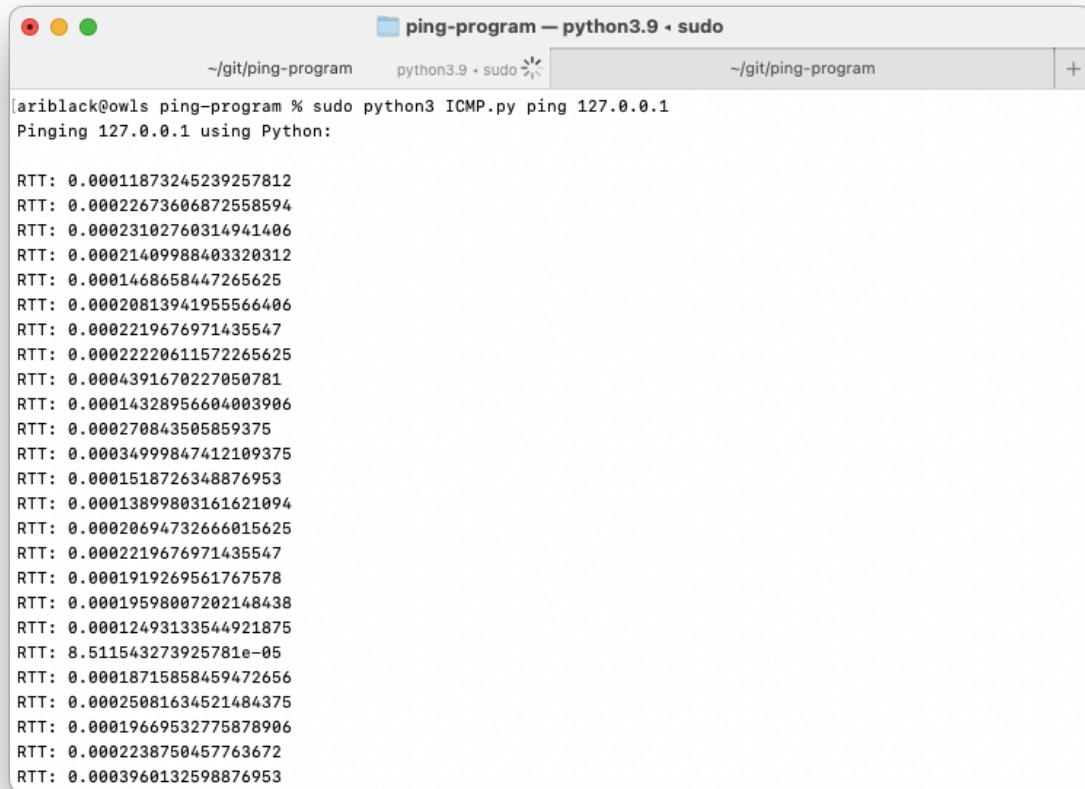
***Structure:*** The following section has a simple structure – first (in bold/italics) is a line containing the relative client-side command used to run the relevant function in *ICMP.py*. (That is, each command is truncated for better readability. In order to actually run, each command needs to begin with *sudo python3 ICMP.py*, which is omitted below. The full commands are visible in their associated screenshots.) Note that commands are separated by |.

Below the command line is a brief description of the given test cases' functionalities, and below this descriptor is the screenshot showing the program's outputs for the given commands.

## Commands:

***ping 127.0.0.1***

*Test ping to localhost with no arguments. Shows that ping works.*



A terminal window titled "ping-program — python3.9 • sudo" is shown. The window has a tab labeled "~/git/ping-program" and a search bar. The terminal content shows the command "sudo python3 ICMP.py ping 127.0.0.1" being executed. The output indicates that the ping is successful, displaying 20 Round Trip Times (RTT) in seconds. The first 19 RTTs are very low, around 0.0001 to 0.0004 seconds, while the 20th RTT is significantly higher at 8.511543273925781e-05 seconds.

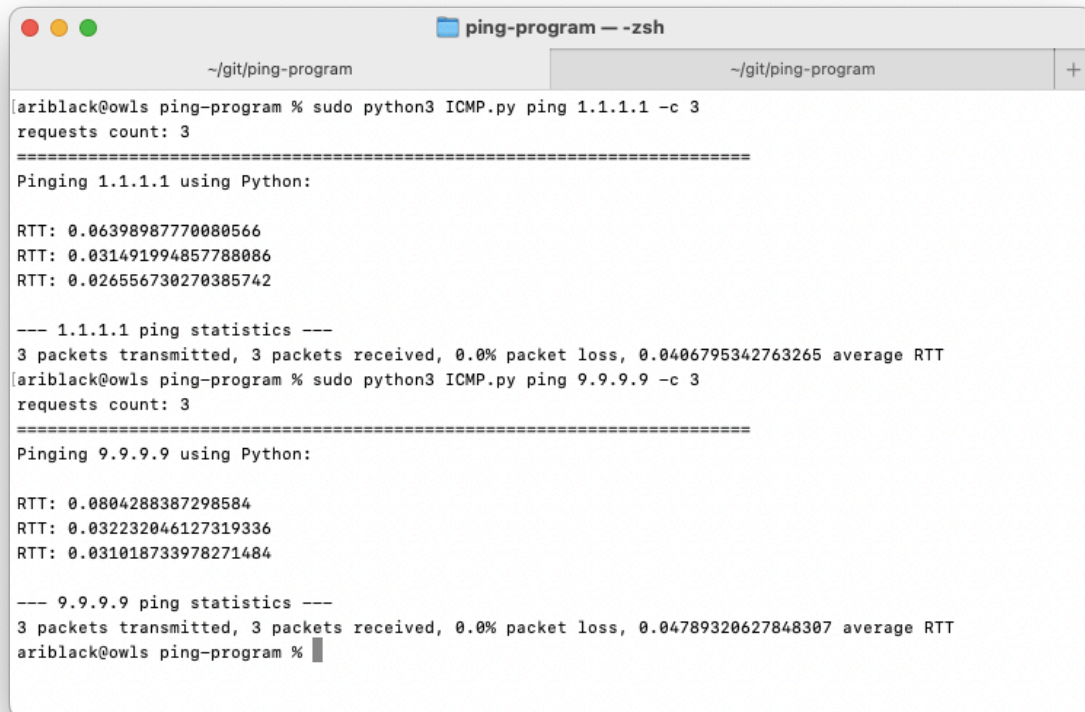
```
[ariblack@owls ping-program % sudo python3 ICMP.py ping 127.0.0.1]
Pinging 127.0.0.1 using Python:

RTT: 0.00011873245239257812
RTT: 0.00022673606872558594
RTT: 0.00023102760314941406
RTT: 0.00021409988403320312
RTT: 0.0001468658447265625
RTT: 0.00020813941955566406
RTT: 0.0002219676971435547
RTT: 0.00022220611572265625
RTT: 0.0004391670227050781
RTT: 0.00014328956604003906
RTT: 0.000270843505859375
RTT: 0.00034999847412109375
RTT: 0.0001518726348876953
RTT: 0.00013899803161621094
RTT: 0.00020694732666015625
RTT: 0.0002219676971435547
RTT: 0.0001919269561767578
RTT: 0.00019598007202148438
RTT: 0.00012493133544921875
RTT: 8.511543273925781e-05
RTT: 0.00018715858459472656
RTT: 0.00025081634521484375
RTT: 0.00019669532775878906
RTT: 0.0002238750457763672
RTT: 0.0003960132598876953
```

### ***ping 1.1.1.1 -c 3 | ping 9.9.9.9 -c 3***

*Test ping to servers on different continents, specifying an argument for -c. 1.1.1.1 belongs to Cloudflare (located in Australia) and 9.9.9.9 belongs to Quad9 (located in Switzerland).*

*Shows that ping -c works for servers on different continents, and shows that session statistics are calculated and output.*

A terminal window titled "ping-program -- zsh" with a tab labeled "~/.git/ping-program". The terminal shows the execution of a Python script to ping 1.1.1.1 and 9.9.9.9. The output for 1.1.1.1 shows three RTT values and statistics: 3 packets transmitted, 3 packets received, 0.0% packet loss, and an average RTT of 0.0406795342763265. The output for 9.9.9.9 shows three RTT values and statistics: 3 packets transmitted, 3 packets received, 0.0% packet loss, and an average RTT of 0.04789320627848307.

```
ariblack@owls ping-program % sudo python3 ICMP.py ping 1.1.1.1 -c 3
requests count: 3
=====
Pinging 1.1.1.1 using Python:

RTT: 0.06398987770080566
RTT: 0.031491994857788086
RTT: 0.026556730270385742

--- 1.1.1.1 ping statistics ---
3 packets transmitted, 3 packets received, 0.0% packet loss, 0.0406795342763265 average RTT
ariblack@owls ping-program % sudo python3 ICMP.py ping 9.9.9.9 -c 3
requests count: 3
=====
Pinging 9.9.9.9 using Python:

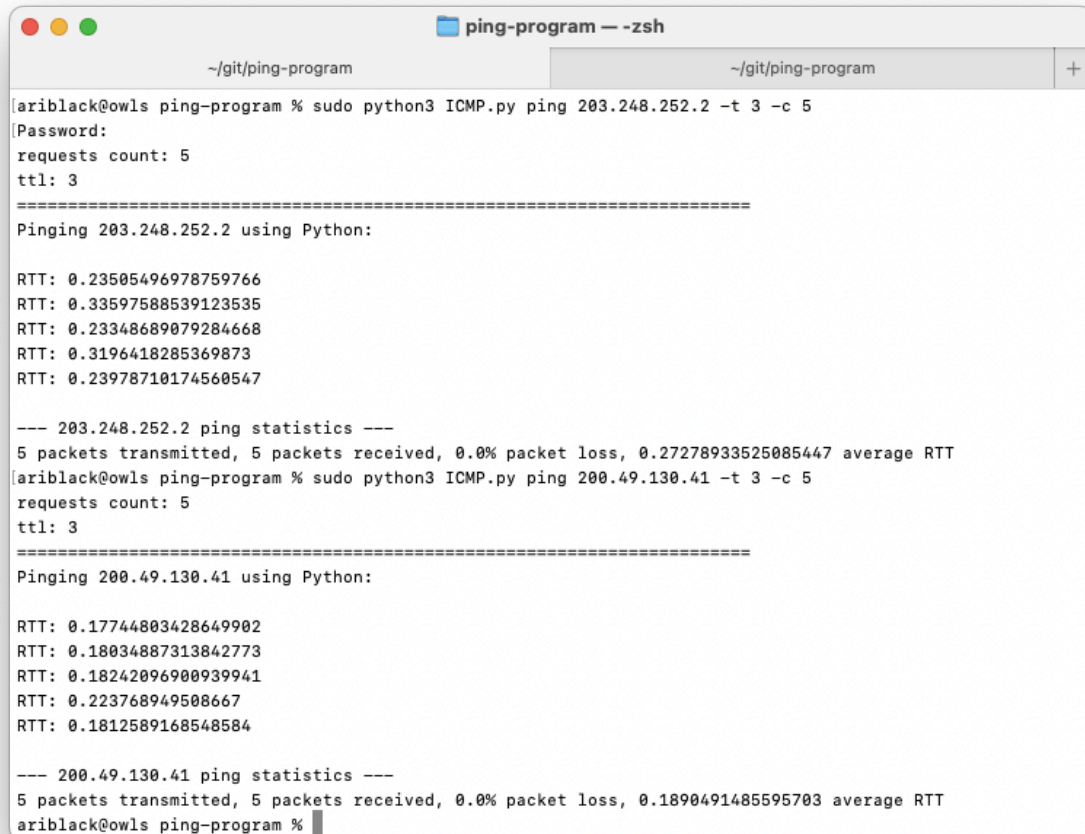
RTT: 0.0804288387298584
RTT: 0.032232046127319336
RTT: 0.031018733978271484

--- 9.9.9.9 ping statistics ---
3 packets transmitted, 3 packets received, 0.0% packet loss, 0.04789320627848307 average RTT
ariblack@owls ping-program %
```

***ping 203.248.252.2 -t 3 -c 5 | ping 200.49.130.41 -t 3 -c 5***

*Test ping to servers on different continents, specifying arguments for -t and (a different) -c. 203.248.252.2 belongs to Korea Telecom (located in Korea) and 200.49.130.41 belongs to Telefonica (located in Argentina).*

*Shows that ping -t -c works for servers on different continents, and that -c evaluates correctly for various values (5 here, versus 3 above).*



```
ping-program --zsh
~/git/ping-program  ~/git/ping-program +
[ariblack@owls ping-program % sudo python3 ICMP.py ping 203.248.252.2 -t 3 -c 5
Password:
requests count: 5
ttl: 3
=====
Pinging 203.248.252.2 using Python:

RTT: 0.23505496978759766
RTT: 0.33597588539123535
RTT: 0.23348689079284668
RTT: 0.3196418285369873
RTT: 0.23978710174560547

--- 203.248.252.2 ping statistics ---
5 packets transmitted, 5 packets received, 0.0% packet loss, 0.27278933525085447 average RTT
[ariblack@owls ping-program % sudo python3 ICMP.py ping 200.49.130.41 -t 3 -c 5
requests count: 5
ttl: 3
=====
Pinging 200.49.130.41 using Python:

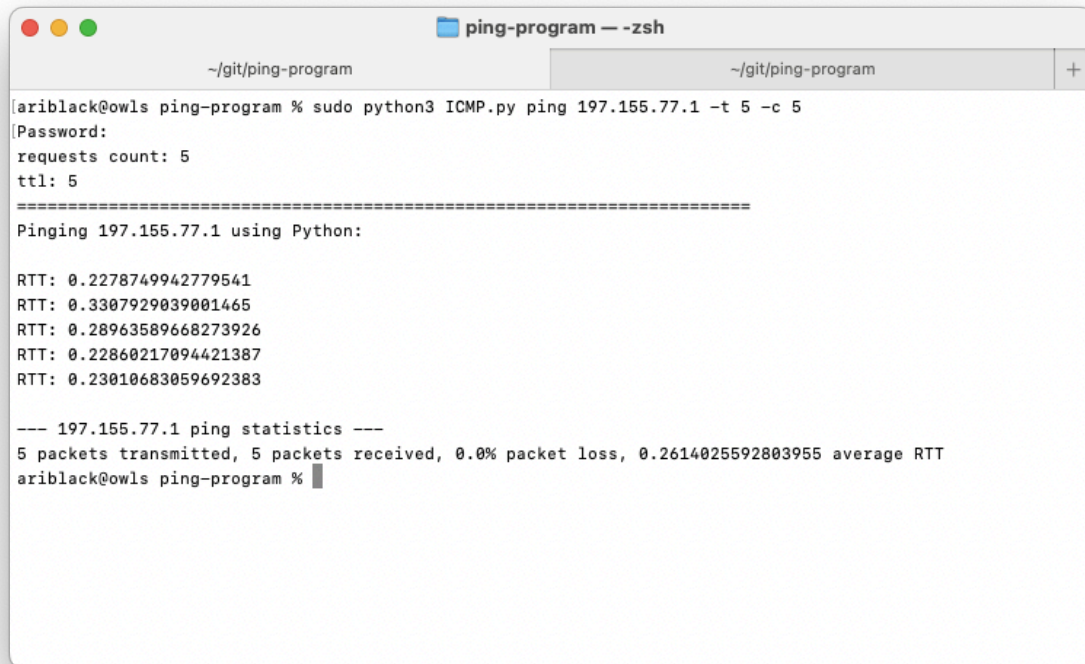
RTT: 0.17744803428649902
RTT: 0.18034887313842773
RTT: 0.18242096900939941
RTT: 0.223768949508667
RTT: 0.1812589168548584

--- 200.49.130.41 ping statistics ---
5 packets transmitted, 5 packets received, 0.0% packet loss, 0.1890491485595703 average RTT
ariblack@owls ping-program %
```

***ping 197.155.77.1 -t 5 -c 5***

*Test ping to servers on all remaining continents (besides Antarctica), specifying arguments for (a different) -t and -c. The IP belongs to Liquid Telecom (located in Kenya).*

*Shows that ping -t -c works for servers on each continent, and that -t works for various values (5 here, versus 3 above).*

A terminal window titled "ping-program -- zsh" with a tab labeled "~/.git/ping-program". The prompt is "ariblack@owls ping-program %". The command "sudo python3 ICMP.py ping 197.155.77.1 -t 5 -c 5" is entered. It prompts for a password, then shows "requests count: 5" and "ttl: 5". A separator line of equals signs follows. The text "Pinging 197.155.77.1 using Python:" is displayed. Five RTT values are listed: 0.2278749942779541, 0.3307929039001465, 0.28963589668273926, 0.22860217094421387, and 0.23010683059692383. Then, a separator line of dashes is shown, followed by "197.155.77.1 ping statistics ---". The statistics are: "5 packets transmitted, 5 packets received, 0.0% packet loss, 0.2614025592803955 average RTT". The prompt "ariblack@owls ping-program %" is shown at the end with a cursor.

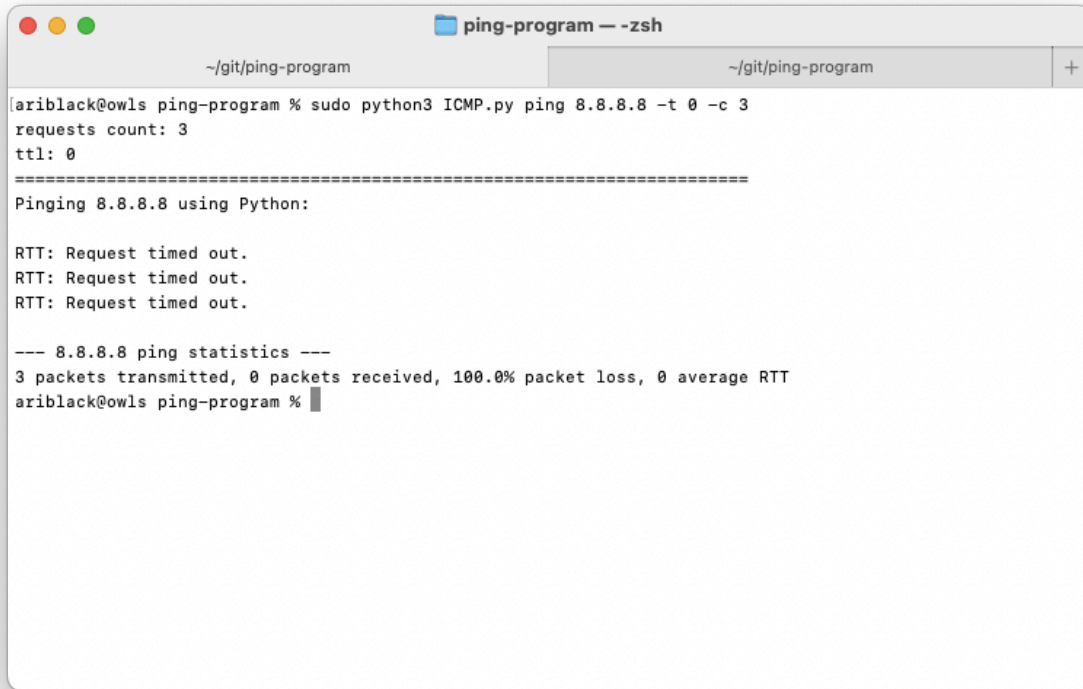
```
ariblack@owls ping-program % sudo python3 ICMP.py ping 197.155.77.1 -t 5 -c 5
[Password:
requests count: 5
ttl: 5
=====
Pinging 197.155.77.1 using Python:

RTT: 0.2278749942779541
RTT: 0.3307929039001465
RTT: 0.28963589668273926
RTT: 0.22860217094421387
RTT: 0.23010683059692383

--- 197.155.77.1 ping statistics ---
5 packets transmitted, 5 packets received, 0.0% packet loss, 0.2614025592803955 average RTT
ariblack@owls ping-program %
```

***ping 8.8.8.8 -t 0 -c 3***

*Test ping to server setting 0 as the value for -t. Shows that -t correctly changes the timeout value for the ping function.*

A terminal window titled "ping-program — zsh" with a tab labeled "ping-program". The window shows the execution of a ping command. The user is "ariblack@owls" in the directory "~/git/ping-program". The command is "sudo python3 ICMP.py ping 8.8.8.8 -t 0 -c 3". The output shows "requests count: 3", "ttl: 0", and a separator line. It then says "Pinging 8.8.8.8 using Python:". Three "RTT: Request timed out." messages follow. Then, it shows "--- 8.8.8.8 ping statistics ---" and "3 packets transmitted, 0 packets received, 100.0% packet loss, 0 average RTT". The prompt returns to "ariblack@owls ping-program %".

```
ariblack@owls ping-program % sudo python3 ICMP.py ping 8.8.8.8 -t 0 -c 3
requests count: 3
ttl: 0
=====
Pinging 8.8.8.8 using Python:

RTT: Request timed out.
RTT: Request timed out.
RTT: Request timed out.

--- 8.8.8.8 ping statistics ---
3 packets transmitted, 0 packets received, 100.0% packet loss, 0 average RTT
ariblack@owls ping-program %
```

***Known Issues:*** The code works as expected for all test cases, though this is not to say that there are no bugs or other minor issues present at the time of submission.

## Resources

No specific sources were referenced for the development of the Ping program.