

The one with the chat logs

Due 20 Oct by 23:59 **Points** 1 **Submitting** an external tool
Available until 20 Oct at 23:59

Successfully solving this problem will award the student **1 point** towards Examination 1

Requirements

- The script must pass all the automatic tests
- The script must pass a manual inspection
- **The student must write the script on his or her own**, any suspicion about cheating, cooperation, and/or plagiarism will be reported to the Discipline Board (see the [Study Guide](https://hkr.instructure.com/courses/5190/files/1027789/download?download_frd=1) ↓ (https://hkr.instructure.com/courses/5190/files/1027789/download?download_frd=1) for more information). The students script will be checked both manually and using anti-plagiarism software

Instructions

1. Analyse the problem below using [Polya's Problem solving Technique](https://math.berkeley.edu/~gmelvin/polya.pdf) ↗ (<https://math.berkeley.edu/~gmelvin/polya.pdf>)
2. Create a script named **Lab2_1.py** that solves the problem according to the given instructions
3. Pay special attention when formatting the output as it must match the specified format **exactly** (character by character) to pass the tests
4. Upload the script using the link below

Task description

With this new *work from home situation* your small team of developers has quickly found out

how much questions you used to ask each other in person. Now that you all work from home most of the time you have resorted to using online chat instead. But this has resulted in that you all have to go back and read endless chat logs when you forget what you agreed on earlier.

To remedy this problem, you have decided to write a small script that **reads a log file and then allow you to search for all messages sent by one user**. To make the script a bit more versatile you have decided that the path to **the log file shall be sent as a command line argument to the script**. As everyone in the team is a bit sloppy when typing, the script needs to have **error handling that clearly informs the user in case the specified path to the log file does not exist**. The format of the error message can be seen in this example:

```
Error: The file '/Users/nnn/messages.txt' could not be found.
```

The script shall have **a main function, a function for reading from the log file, and a function for formatting and displaying a message on the screen**. Here are some rules for how each of the functions shall work:

read_file(filename)

Shall read the file specified by the parameter *filename* and create a list of tuples that is then returned. Each tuple shall consist of a name and a message. In the file, each message takes up two rows, where the first row contains the senders name, and the second row contains the text of the message. Each log file can of course contain many messages sent by many different persons. If the *filename* parameter is not a valid path, the function shall raise an `FileNotFoundError`.

display_entry(name, message)

Shall use the two parameters to display a formatted output on the screen. The output shall follow this format **exactly**:

```
[name] --> message
```





main()

This function shall read the command line argument, ask what name to search the log file for, and use the other functions as needed to display all messages send by the specified name. The function shall also be responsible for the error handling.

To avoid problems with the testing framework, and to build a good habit, call the main function using the following construct:

```
if __name__ == '__main__':  
    main()
```

Comments and additional resources

- Students are highly recommended to first solve all the tasks from the corresponding lectures and exercises before attempting to solve this problem. Having a solid plan before starting to code will also help
- The student can upload as many times as he or she likes to before the deadline
- An example log file can be found [here](https://gist.github.com/andreas-hkr/32ae90cc1206713dc2fb7aaebadada27)  [_\(https://gist.github.com/andreas-hkr/32ae90cc1206713dc2fb7aaebadada27\)](https://gist.github.com/andreas-hkr/32ae90cc1206713dc2fb7aaebadada27)
- Watch [this](https://youtu.be/IOeIDvyRUQs)  [_\(https://youtu.be/IOeIDvyRUQs\)](https://youtu.be/IOeIDvyRUQs) video if you want to learn more about why we call main the way we do
- This [video](https://youtu.be/o1Rxrtiqml8)  [_\(https://youtu.be/o1Rxrtiqml8\)](https://youtu.be/o1Rxrtiqml8) covering how to use command line arguments might be useful
- This [video](https://youtu.be/QgSZ8os-UZQ)  [_\(https://youtu.be/QgSZ8os-UZQ\)](https://youtu.be/QgSZ8os-UZQ) recorded while running the finished script showcases how the script shall look and work when finished

This tool needs to be loaded in a new browser window

Load The one with the chat logs in a new window