corax

## Coding challenge

If you're reading this, that means that Corax is really interested in your engineering abilities. Congratulations!

This is intended to be a small coding exercise meant to gauge your programming style. Now's your chance to show off your design skills (simple but extensible), your testing habits (hey hey unittest), and your keen attention to detail.

Included is an input file that you will write a Python program to process. After you've engineered a solution to your satisfaction, email us with the source code for your solution and the result of running the input file through your program.

Your solution should be documented and tested to production standard. It should follow good Python conventions, and should be easy to get running.

Package up your response in a tar.gz file, send it off to us, and we'll get back to you shortly!

## The problem

You're tasked with taking entries of personal information in multiple formats and normalizing each entry into a standard JSON format. Write your formatted, valid JSON out to a file with two-space indentation and keys sorted alphabetically.

### Input

Your program will be fed an input file of n lines. Each line contains "entry" information, which consists of a first name, last name, phone number, color, and zip code.

The order and format of these lines vary in three separate ways.  The three different formats are as follows:

Lastname, Firstname, (703)-742-0996, Blue, 10013
Firstname Lastname, Red, 11237, 703 955 0373
Firstname, Lastname, 10013, 646 111 0101, Green

Some lines may be invalid and should not interfere with the processing of subsequent valid lines.  A line should be considered invalid if its phone number does not contain the proper number of digits.

### Output

The program should write a valid, formatted JSON object out to result.json. The JSON representation should be indented with two spaces and the keys should be sorted in ascending order.

Successfully processed lines should result in a normalized addition to the list associated with the "entries" key. For lines that were unable to be processed, a line number i (where $0 \leq i < n$) for each faulty line should be appended to the list associated with the "errors" key.

The "entries" list should be sorted in ascending alphabetical order by (last name, first name).

## Sample

For the input

Booker T., Washington, 87360, 373 781 7380, yellow
Chandler, Kerri, (623)-668-9293, pink, 123123121
James Murphy, yellow, 83880, 018 154 6474
asdfawefawea

We should receive the output


```
{
  "entries": [
    {
      "color": "yellow",
      "firstname": "James",
      "lastname": "Murphy",
      "phonenumber": "018-154-6474",
      "zipcode": "83880"
    },
    {
      "color": "yellow",
      "firstname": "Booker T.",
      "lastname": "Washington",
      "phonenumber": "373-781-7380",
      "zipcode": "87360"
    }
  ],
  "errors": [
    1,
    3
  ]
}
```


Questions?

Feel free to email.

Good luck!
Your friends at Corax