Comma-separated values (CSV) files

"Comma Separated Values" is a common file format when exporting data from spreadsheets and databases. Each line of the file is a row, and each column is separated by a comma.

data.csv file contents:

```
Name,Location,URL,Students
Westminster College,Salt Lake City-UT,westminstercollege.edu,2135
Muhlenberg College,Allentown-PA,muhlenberg.edu,2330
University of Maine,Orono-ME,umaine.edu,8677
James Madison University,Harrisonburg-VA,jmu.edu,19019
Michigan State University,East Lansing-MI,msu.edu,38853
```

1. Write a function called `input_by_name` that takes in the name of a csv file and returns a dictionary organized by name like so:

```
college_dict = {
    'James Madison University': {'Location': 'Harrisonburg-VA',
                                 'Students': '19019',
                                 'URL': 'jmu.edu'},
    'Michigan State University': {'Location': 'East Lansing-MI',
                                  'Students': '38853',
                                  'URL': 'msu.edu'},
    'Muhlenberg College': {'Location': 'Allentown-PA',
                           'Students': '2330',
                           'URL': 'muhlenberg.edu'},
    'University of Maine': {'Location': 'Orono-ME',
                           'Students': '8677',
                           'URL': 'umaine.edu'},
    'Westminster College': {'Location': 'Salt Lake City-UT',
                            'Students': '2135',
                            'URL': 'westminstercollege.edu'}
}
```

```python
def input_by_name(filename: str) -> dict:
    file_dict = {}
    with open(filename, 'r') as fileobj:
        header_items = fileobj.readline().strip().split(',')
        print(header_items)
        for line in fileobj:
            entries = line.strip().split(',')
            dict_value = {}
            for sindex in range(1, len(entries)):
                dict_value[header_items[sindex]] = entries[sindex]
            file_dict[entries[0]] = dict_value

    return file_dict
```

2. Write a function called `get_all_students` that takes in the dictionary organized by name and returns the total number of students:

```python
def get_all_students(college_dict: dict) -> int:
    count = 0
    for name in college_dict:
        if 'Students' in college_dict[name]:
            count += int(college_dict[name]['Students'])

    return count
```

3. Write a function called `transpose_by_URL` that takes in the dictionary organized by name and transposes it by URL so it looks like so:

```python
college_dict = {
    'jmu.edu': {'Location': 'Harrisonburg-VA',
                'Name': 'James Madison University',
                'Students': '19019'},
    'msu.edu': {'Location': 'East Lansing-MI',
                'Name': 'Michigan State University',
                'Students': '38853'},
    'muhlenberg.edu': {'Location': 'Allentown-PA',
                       'Name': 'Muhlenberg College',
                       'Students': '2330'},
    'umaine.edu': {'Location': 'Orono-ME',
                   'Name': 'University of Maine',
                   'Students': '8677'},
    'westminstercollege.edu': {'Location': 'Salt Lake City-UT',
                               'Name': 'Westminster College',
                               'Students': '2135'}
}
```

Note that `transpose_by_URL` **is meant to me a mutator function (that is it changes the dictionary in place without creating another dictionary).**

```python
1  def transpose_by_URL(college_dict: dict) -> None:
2      keys = list(college_dict.keys())
3      for name in keys:
4          url = college_dict[name]['URL']
5          college_dict[name]['Name'] = name
6          del college_dict[name]['URL']
7          college_dict[url] = college_dict[name]
8          del college_dict[name]
```

4. With this new dictionary does the code for `get_all_students` need to change? Explain why.

No, still need to go through every key and search for the Students key inside the dictionary associated with it.

5. Write a function called `get_min_students` that takes in the dictionary organized by URL and returns the name of the college with the least number of students:

```python
1  def get_min_students(college_dict: dict) -> str:
2      min = get_all_students(college_dict) #some large value
3      min_name = 'No name'
4      for name in college_dict:
5          if int(college_dict[name]['Students']) < min:
6              min = int(college_dict[name]['Students'])
7              min_name = college_dict[name]['Name']
8
9      return min_name
```

6. Write a function called `input_by_column` that takes in the filename and returns a dictionary organized by the column's name:

```
column_dict = {
    'Location': ['East Lansing-MI',
                 'Salt Lake City-UT',
                 'Allentown-PA',
                 'Orono-ME',
                 'Harrisonburg-VA'],
    'Name': ['Michigan State University',
             'Westminster College',
             'Muhlenberg College',
             'University of Maine',
             'James Madison University'],
    'Students': ['38853', '2135', '2330', '8677', '19019'],
    'URL': ['msu.edu',
            'westminstercollege.edu',
            'muhlenberg.edu',
            'umaine.edu',
            'jmu.edu']
}
```

```python
def input_by_column(filename: str) -> dict:
    file_dict = {}
    with open(filename, 'r') as fileobj:
        header_items = fileobj.readline().strip().split(',')
        for item in header_items:
            file_dict[item] = []
        for line in fileobj:
            entries = line.strip().split(',')
            for sindex in range(len(entries)):
                file_dict[header_items[sindex]] += [entries[sindex]]

    return file_dict
```

7. Re-write `get_min_students_column` that takes in the dictionary organized by column name and returns the name of the college with the least number of students:

```python
def get_min_students_column(column_dict: dict) -> str:
    min_index = 0
    for index in range(len(column_dict['Students'])):
        if int(column_dict['Students'][index])
                    < int(column_dict['Students'][min_index]):
            min_index = index

    return column_dict['Name'][min_index]
```