Work with your neighbor. (This will be graded for participation only.)

1. Create a 2-d dictionary called `contacts` that holds contact information on the people given below. There are two groups of people. The first are the people in the Family group:

   John – 520-470-3621

   Maria – 820-690-5241

   The second are the people in the Friend group:

   Javier – 512-820-5861

   Katie – 202-890-4200

   The first level keys of the dictionary `contacts` should be the strings `"Family"` and `"Friends"`. The second level dictionaries contain key/value pairs of the person's name and the corresponding number, for example: `{"John": "520-470-3621", …}`. Create the dictionary below:

   Now that your `contacts` dictionary is created, answer these two questions:

   a) What is the code to access Maria's number?

   b) What is the code to add another friend to your contacts? This friend is Brandon, with phone number 313-682-6800.

2. Given the dictionary below:

```
>>>catalog
{ 'MIS': {'mis 101': 4, 'mis 102': 3, 'mis 202': 2},
  'CSC': {'csc 110': 4, 'csc 120': 4, 'csc 352': 3},
  'ECE': {'ece 111': 3, 'ece 222': 3, 'ece 333': 4}}
```

   Add the 3-unit course `'csc 144'` to `catalog`. (This should be added to the `'CSC'` dictionary.)

3. Given the dictionary from the problem above, we need to add a course from a *new* department to the `catalog`. We want to add a key/value pair for the English department. The key is `'ENGL'`. Now add the 3-unit course `'engl 101'` to `catalog` for the English department.

4. Write a Python function `num_keys(d)` that takes as argument a 2-level dictionary `d` and returns the total number of *keys* in `d`, counting keys at both levels of `d`. Duplicates should be considered as distinct and counted separately. For example, in the dictionary

```
mydict = { 12 : { 'a' : 11, 'b': 22},
           23 : { 'm' : 33, 'b': 44, '5': 55 } }
```

   there are two keys at the first level (12 and 23) and five keys at the second level, for a total of seven keys. Thus, `num_keys(mydict)` should return 7.

(**Note:** We may not get to these next problems.)

5. **Tuples.** Write a function `min_max(L)` that takes a list `L` of integers and returns a tuple of the smallest and largest even numbers in `L`. You may use the built-in min and max functions.

6. Use the `catalog` dictionary shown in the previous problems above and the `items()` method.

   a) Print the keys and values of the `catalog`, separated by a ":"

   b) Print the keys and value of `catalog`, separated by a ":", in sorted order of the keys.