Compsci 06, Spring 2011 Test Redux: Dictionary

You're given code that creates a dictionary of students for lab attendance. The keys are student names, the value for each student is a list of labs the student has attended, with the labs labeled by "*lab1*", "*lab2*", "*lab3*", and so on. For example:

```
labs = {
   "Fred"   : ["lab1", "lab2", "lab4", "lab6"],
   "Ethel"  : ["lab1", "lab2", "lab3", "lab4", "lab5", "lab6"],
   "Alex"   : ["lab3"],
   "Mary"   : ["lab4", "lab3", "lab1", "lab6"]
}
```

Write the function `labify` that has as a parameter a dictionary in the format above and which returns a dictionary in which each possible lab that appears in some value list in `attend` is the key. The value associated with this key is a list of strings: the people who attended that lab. Each list of strings should be sorted alphabetically.

For example, for the data above the call `labify(labs)` should return the dictionary diagrammed below:

```
{
    "lab1" : ["Ethel", "Fred", "Mary"],
    "lab2" : ["Ethel", "Fred"],
    "lab3" : ["Alex", "Ethel", "Mary"],
    "lab4" : ["Ethel", "Fred", "Mary"],
    "lab5" : ["Ethel"],
    "lab6" : ["Ethel", "Fred", "Mary"]
}
```

Write your code below:

```
def labify(attend):
    """
    attend is a dictionary in format name : [strings of labs attended],
    return a dictionary in which key is a string "labX" for all named
    labs that appear in values of attend, value for "labX" is
    alphabetized list of people attending that lab
    """
```

Compsci 06, Spring 2011, Files/Dictionaries, April 24

Name____________________ net-id _________

Name____________________ net-id _________

Name____________________ net-id _________

The data from sites that rate faculty members might look like what's shown below: Each line shows a school, the name of a faculty member, the department, a rating on a 0-5 scale, and the IP address from which the rating was made (last names are shown for brevity, the full-name would be recorded).

```
Duke:Astrachan:Computer Science:4:152.3.250.1
Duke:Forbes:Computer Science:4:152.3.250.1
Duke:Astrachan:Computer Science:5:153.39.0.22
Stanford:Parlante:Computer Science:3:152.3.250.1
Stanford:Sahami:Computer Science:4:152.3.150.1
Stanford:Lattin:Business:5:153.39.0.22
Duke:Fullenkamp:Economics:4:152.3.250.1
Duke:Leachman:Econonmics:3:152.3.250.1
Duke:Bonk:Chemistry:4:152.3.250.1
```

Write a function that takes a filename for a file in the format shown and the name of a school (both strings) as parameters and which returns a list of `(name,department,rating)` three-tuples that are rank-ordered by average-rating, highest-to-lowest, for all professors in the school. If there are ties, break them by the alphabetic/lexicographical ordering of professor name.

```
def rankings(filename,school):
   """
   filename is name of file in format shown (string)
   school is a string

   return list of 3-tuples ordered by faculty rating, high-to-low
   """
```