PC 204 Exercises, Fall 2000

The theme for these exercises is maintaining a database of articles (authors, title, journal and publication date). This is a very small subset of what a digital library does, but is useful in illustrating the programming techniques.

Week 1

The code below defines a list of articles, with each article being represented by a dictionary.

```
articles = [
    {
        'authors': [ 'Burkhard Rost', 'Chris Sander' ],
        'title': 'Prediction of protein secondary structure '
                'at better than 70% accuracy',
        'journal': 'Journal of Molecular Biology',
        'date': 1993
    },
    {
        'authors': [ 'R J Zauhar', 'R S Morgan' ],
        'title': 'Computing the Eletric Potential of Biomolecules: '
                'Application of a New Method of '
                'Molecular Surface Triangulation',
        'journal': 'Journal of Computational Chemistry',
        'date': 1990
    },
    {
        'authors': [ 'Mike Carson', 'Charles E Bugg' ],
        'title': 'Algorithm for ribbon models of proteins',
        'journal': 'Journal of Molcular Graphics',
        'date': 1986
    },
]
```

1-1. Print all of the keys and values in the second article of the list (remember that list indexing starts at zero).

1-2. Print the title of the third article.

1-3. Add another articles from the following reference:

Elaine C Meng and Richard A Lewis, ‘‘Determination of Molecular Topology and Atomic Hybridization States From Heavy Atom Coordinates,’’ *Journal of Computational Chemistry*, 12(7):891-898, 1991.

Week 2

For these exercises, use the list of articles from Week 1, including the article added for exercise 1-3.

2-1. Print the authors and titles from articles published prior to 1992.

2-2. Create a non-redundant list (*i.e.*., one without duplicate entries) of journals from all of the articles. (Hint: keys in dictionaries are unique.)

Week 3

3-1. Using the list of articles from Week 1, write a function that takes two arguments, the name and value of a field, and prints the authors and title from all articles that have the matching entry. For example, if the function is called `find`, then

```
find('date', 1993)
```

should print

```
authors: ['Burkhard Rost', 'Chris Sander']
title: Prediction of protein secondary structure at better than 70% accuracy
```

Call the function to print the authors and titles of articles from the Journal of Computational Chemistry.

3-2. Rewrite the function so that it normally prints the authors and title, but can also print all fields from a user-supplied list. For example,

```
find('date', 1993, ['authors', 'journal'])
```

should print

```
authors: ['Burkhard Rost', 'Chris Sander']
journal: Journal of Molecular Biology
```

Call the function to print the articles from the Journal of Computational Chemistry, but print only the title and date from the articles. (Hint: use default arguments.)

Week 4

Rewrite the function from 3-2 so that incorrect key names are reported instead of causing exceptions.

Week 5

Instead of using article fields directly as Python strings and integers, design sets of functions that manipulate and compare the fields. (Hint: what should be the objects in an object-oriented design? What are the necessary functions associated with each type of object?)

Week 6

Implement the module design from the lecture. Make each set of functions a separate module. Write a main program that demonstrates the use of the functions. Handle errors gracefully.

Week 7

7-1. Design and implement classes to replace the functions from Week 6. Articles should be stored as instances of a class rather than as dictionaries. Keys and values in the dictionaries should become attributes of the instances. Some of the values should also become instances of classes because they have methods associated with them.

7-2. Recreate the list from Week 3 using instances instead of dictionaries. Print the list of articles.

Week 8

Add representation and comparison methods to the author class. The `__repr__` method should return the author name in the form *first middle last* name. The `__cmp__` should compare names by last, first and then middle name.

8-1. Demonstrate that two instances of authors (`a1` and `a2`) may be compared with:

```
print a1
print a2
print a1 < a2
```

should print

```
Mike Carson
Charles E Bugg
0
```

8-2. Print the list of articles sorted by first author.

Week 9 and 10

Final Project.