Pickling Class Instances

Way back in history, in a time before refrigeration was available in most homes, it was common to preserve food by submerging it in vinegar, with sugar and/or spices (often dill) added to make it palatable. This process is generally known as *pickling.* That is **not** what this document is about.

Pickling, in Python, is a simple way to preserve class instances that you've created. (So, not entirely unrelated, we are preserving something in both cases.) This allows a user to gradually build up a database of objects/instances that the user wants to work with, and have a really easy way to load/save them all.

Example: Swindle eReader

In Lab 12, you designed a Swindle eReader that allows you to set up a user with some books that they can choose from, some that they have bought, and how long they want each page to be when they read their books. The main Lab does not include any way to preserve that information from one run of the program to the next. Enhancement #1 suggests that you read/write the data to a text file. Let's try it using pickling instead (Note: I am not dealing with the problem of checking whether or not the file exists and all that, which you should think about):

```
ereader.py:

import pickle

class Swindle():
    # All your existing code

def saveData(instance):
    with open('swindle.dat', 'wb') as outfile:
        pickle.dump(instance, outfile)
def loadData(instance):
    with open('swindle.dat', 'rb') as infile:
        instance = pickle.load(infile)
    return instance
```

Quick quiz: Why is there a letter `b` in the read/write part of the `open()` command? That tells Python that the file is expected to be a *binary* file instead of a text file. So, if you open this file in Atom, it will at least partly look like gibberish.

How to use this: You can feed in an instance of the `Swindle()` class, and `saveData()` will write all that data to a file. Then, later, you use `loadData()` to recall all of what you saved before. This is really helpful if you actually want to read a book and save your place for next time.

You can also read/write instances of your Book class this way.

What kinds of things can I `pickle`?

Cucumbers (most common), green beans, beets, watermelon rinds (surprisingly good!), oh wait… that's pickle, not `pickle`.

You can `pickle` any Python object. What's an object in Python? *Everything* is an object in Python. You can `pickle` a class instance. You can `pickle` a list of class instances. You can `pickle` a dictionary whose values are lists of class instances. You could `pickle` the entire contents of a module if you wanted to, but I'm not sure why you'd want to because `import`ing it is much more natural. Any time you want to preserve some information from one run of a Python program to the next, you can `pickle` that information and write to a file, and then load next time the program runs.

The thing to be careful about is to make sure you know what object you're fetching when you read the pickled data back in: Did you store an instance of a class? Which class? Did you store a list of those class instances? Was it a dictionary? Then you have to keep that in mind when you unpack the `pickle`d data.

For the Swindle program, probably the ideal situation would be to set up a dictionary that has two keys: `"books"` and `"ereaders"`, each of which has a value that is a list of instances of those classes.