600.112: Intro Programming for Scientists and Engineers

Assignment 9: Course Database *

Peter H. Fröhlich
phf@cs.jhu.edu

Joanne Selinski
joanne@cs.jhu.edu

Due Date: 11:30pm on Friday 12/4 - only 1 hour late grace period!

Introduction

The last assignment for *600.112: Introduction to Programming for Scientists and Engineers* explores how we can work with *data* in many different formats. We'll also work with inheritance a little bit. Most of the code for this project is already written for you - you'll need to download the posted zip file from the main assignment page. Also, before you begin, make sure that the following modules are installed in your environment: python-urllib, python-xlrd and python-xlwt.

There are only two things to do, and both will be due the same day: First you'll write a class `SemesterListing` to manipulate a database (PYTHON dictionary) consisting of `CourseListing` objects, with their course number IDs as the keys. Second you'll complete a program that manages a course catalog and semester listings with a menu-driven program, managing a full course database. You must write the options to read and save course data from/to text files (local or URLS). For extra credit, you can write the parts to save and load semester course listings to/from excel files.

You must submit a complete zip file with all files needed to run your solutions as detailed below (including the README and courses.db), on Blackboard before the deadline. **Note: there is only a one hour late grace period for this assignment, not a full 24 hour day!**

1 Background

Databases are virtual collections of data. They are everywhere in modern daily life. For example, in order to enroll in this class, Intro to Programming for Scientists and Engineers, you had to register on ISIS. This meant that a database associated with ISIS that connected your student ID to a list of your other classes had to add this class in particular.

Another example would be how a student housing database contains and groups the buildings, room numbers, and students so that you can choose from available options and the university can also appropriately bill and contact the correct students.

Furthermore, this assignment will be of help for those of you who plan on taking subsequent programming classes - basic databases are an extremely common theme in these. Being able to quickly look-up and filter data according to a key is a fundamental operation.

*Disclaimer: This is *not* a course in physics or biology or epidemiology or even mathematics. Our exposition of the science behind the projects cuts corners whenever we can do so without lying outright. We are *not* trying to teach you anything but computer science!

We will also explore a slightly different style of program this week - one which features a menu from which our program user can repeatedly and interactively manipulate the database we'll be creating. We call this style of application a *menu-driven* program.

2 Part A: SemesterListing [25 points]

We'll begin by programming a class that will keep track of course listings for a particular semester, using a Python dictionary. Before you begin, you should read through the README outline file and all the provided code in the starter zip file. Understanding it is essential to completing the assignment!

For the first part of this project we'll edit the starter file `SemesterListing.py`. The purpose of this class is to keep track of all courses being offered in a particular semester. We'll write a function to initialize a SemesterListing given the semester information (season and year), creating an empty dictionary. We'll then code important functions to create a string representation of a SemesterListing (for printing), add a course which may be either a core Course or a complete CourseListing, and delete a course if it's there.

Instead of providing the code to do all this in this handout, for this last assignment we'll walk through the steps in class together, although we may not finish all the parts. If you miss class, you'll need to figure it out on your own, in lab, or with TA help. The provided skeleton code has docstrings to clarify the operation of each method.

3 Part B: Data Files [15 points + 10 extra]

For the second part of this project, you must complete the `CoursesMain.py` driver program so that it can save and load catalog information using plain text files. The format of the files must be the same as in the provided sample input file, "courses.txt". See the docstrings for these methods in the provided driver program for more details.

For extra credit, implement the menu operations to save and load a semester to/from an Excel file (5 points each). You can either implement these operations within the CoursesMain.py file, or as methods in the SemesterListing.py class definition, which you then call from the CoursesMain.py main function.

Here are some details regarding the extra credit component. The name of the excel file must be the semester code, with .xls as the extension, as in 'FA15.xls' for example if saving or loading the Fall 2015 semester. Each Excel file (workbook) must have one sheet, named 'CourseList'. Then each course in the semester listing should correspond to a row in the spreadsheet, in order according to the full course number. You must use a column for each individual piece of information, using this exact column order: course division, course department, course number, course title, course credits, course instructor, course times. Make sure that number values are saved to the excel sheet as numbers, not as strings.