CS 224 Introduction to Python | Spring 2022

Closed Lab 6

University of Wisconsin – La Crosse | *March 11*

The Wonders of `zip`

Description: For this assignment, you will generate data (randomly) and sort it. That sounds a lot like the first sentence of a recent lab. That's because it is a lot like the first sentence of a recent lab. The assignment is almost the same. The key difference is that you will use `zip` this week. You will also sort the data a couple of different ways. In the interest of full-disclosure, "wonders" may be overselling it a bit but "The Occasional Usefulness of `zip`" doesn't really pop.

Details:

- Write a function that generates a list of experiment numbers. The list has length `num`, a parameter to the function. The numbers are consecutive, beginning with 0. The function should return the list. Note that this is simpler than the first function from last week's lab.
- Write a function that generates a list of data values. There are two types of data: integer-valued and real-valued. The function should take a parameter that indicates which type of data to generate (with a default value indicating integers). There should also be parameters for the number of values to generate, the minimum value, and the maximum value. These have default values of 50, 100, and 1000, respectively. The function should return the list generated.
- Using the functions you've written, write a function called `main` that generates a list of experiment numbers and three lists of data: one real-valued list and two integer-valued lists. All of the lists should contain the same number of elements. The experiment numbers and data lists are related by position. That is, the experiment number and data values at index i in each list represent the results for an experiment. Test your program.
- Write a function that takes the four lists as parameters as well as an optional parameter called `rev_order` with a default value of `False`. The function will use zip to help achieve the following: the data will be sorted using data from the first integer-valued list as the primary sort key and data from the real-valued list as the secondary sort key.

  After sorting the data should be "unzipped" into four lists. The data in the lists will be ordered as described above and will still be "grouped" with the same

experiment number and data values as before the sort. For example, if the real-value in position i before the sort is in position j after the sort, then all of the other values that were in position i before the sort must also be in position j after the sort. `rev_order` indicates whether the data should be in non-decreasing order (`False`) or non-increasing order (`True`). The function should return the four sorted lists.

- The final result of your program is the four lists, sorted as indicated above. Print the data before and after sorting to verify that your program works correctly.