Handin 2 B2

The handins in the course will consist of two parts: 1) a set of short exercises to practice a particular part of the curriculum (e.g. loops), and 2) a project part. In the project part we will consider a specific Data Science case, each week working on the same (or similar) data, and gradually building up a complete analysis of the data.

Note that the tools required to solve the handins will be generally covered in the lectures in the week where the assignment is given. Sometimes, you might have to  wait until the Friday lecture before you have all the tools to complete the assignment.

Part 1

This week, we'll look into tuples, dictionaries and functions.

1. Create a file called handin2.py. Inside this file, create a variable called beatles\_container1, and assign to this variable a list of tuples, where each tuple consists of the the name and the instrument played by one of the Beatles:

|  |  |
| --- | --- |
| Paul McCartney | bass guitar |
| John Lennon | rhythm guitar |
| George Harrison | lead guitar |
| Ringo Starr | drums |

1. The list should thus have 4 members, and each entry should be a tuple with two strings: name and instrument.
2. We want to be able to look up the instruments by the name of the band member. Inside the same handin2.py file, create a variable called beatles\_container2, and choose an appropriate data type, such that we can write:

print(beatles\_container2['Ringo Starr'])

to get the output drums. When writing this code, you should use the values from your beatles\_container1 - i.e. you should not type in the values again.

1. Now, let's put the functionality above into a function. Again, inside handin2.py, create a function called beatle\_lookup and copy&paste your code from above into the body of the function. The function should have an argument called name, and then return the instrument of the band member with that name. If the name passed to the function is not one of the 4 Beatles, the function should return a string with an error message that provides the user with information on how the function can be called - this message should mention the four names for which the function works. For instance:

ERROR. Name 'Mick Jagger' not found. Available names: ['Paul McCartney', 'John Lennon', 'George Harrison', 'Ringo Starr']

You should do this without typing in the four Beatles names again. Note: we'll see later in the course how to deal properly with error message in Python.

Part 2: Project

Last week, we opened the temperature anomaly data file, read it in, and then printed lines within a particular year range. This week, we'll put the data into a container variable, and wrap the code in a function, so we can make the selection for particular years more flexible.

1. Create a file called handin2\_project.py. Write similar code as last week to iterate over the lines in the file (i.e. feel free to copy&paste), but now, instead of printing out the lines to screen, add them to a list called data\_list. Read the data lines for all years into this list, only ignoring comment lines and empty lines.
2. Now, we'll put this functionality inside a function. Create a function called read\_data that takes a single argument: a string argument called filename. The function should read in the data from the specified filename and return a list of strings (in the same format as we had in data\_list in the previous exercise). Feel free to copy&paste from the previous question.
3. Duplicate (Copy&paste) your read\_data function into a new function called read\_data2, but change it in the following way: The function should now take two arguments: 1) a string argument called filename as before and 2) a tuple of two integers called year\_range, which specifies the (start,end) years for which we which to read in data from the file. As is common for Python ranges, the start value of the range should be included, while the end value should be excluded. For example, if I write year\_range=(2000, 2010) I should get the values from 2000 to 2009. If year\_range is not specified by the caller of the function, the function should return data for all available years.

When you are done, click on the "Load Handin2 in a new window" button below, which will take you to the CodeGrade server. Here, please submit the handin2.py, handin2\_project.py files. CodeGrade will then automatically check the code for you, and upgrade your grade for the assignment within Absalon. You can submit as many times as you want.