Fundamentals of Computer Programming

Building a Programming Portfolio

Week 7

You should be able to complete the following programs by the end of the week. By now you should understand why you should be saving your work to GitHub or similar. Possible solutions will be uploaded to the main module GitHub repository every week. If you follow that repo you should be able to receive notifications.

1. Write and test a function that takes a string as a parameter and returns a sorted list of all the unique letters used in the string. So, if the string is `cheese`, the list returned should be `['c', 'e', 'h', 's']`.
2. Write and test three functions that each take two words (strings) as parameters and return sorted lists (as defined above) representing respectively:

   Letters that appear in at least one of the two words.
   Letters that appear in both words.
   Letters that appear in either word, but *not* in both.

   Hint: These could all be done programmatically, but consider carefully what topic we have been discussing this week! Each function can be exactly one line.
3. Write a program that manages a list of countries and their capital cities. It should prompt the user to enter the name of a country. If the program already "knows" the name of the capital city, it should display it. Otherwise it should ask the user to enter it. This should carry on until the user terminates the program (how this happens is up to you).

   Note: A good solution to this task will be able to cope with the country being entered variously as, for example, "Wales", "wales", "WALES" and so on.
4. One approach to analysing some encrypted data where a substitution is suspected is frequency analysis. A count of the different symbols in the message can be used to identify the language used, and sometimes some of the letters. In English, the most common letter is "e", and so the symbol representing "e" should appear most in the encrypted text.

   Write a program that processes a string representing a message and reports the six most common letters, along with the number of times they appear. Case should not matter, so "E" and "e" are considered the same.

   Hint: There are many ways to do this. It is obviously a dictionary, but we will want zero counts, so some initialisation is needed. Also, sorting dictionaries is tricky, so best to ignore that initially, and then check the usual resources for the runes.