Dictionaries in Python

A data structure is a way of organizing and storing data in computer memory so that it can be used in a program to solve a program. To this point, we have studied two data structures:

1) A variable which stores a single piece of data
2) A list which stores many pieces of data. This data is stored in an ordered manner - the first piece of data is stored in the first element which is followed by the second piece of data, etc.

There is a third data structure in Python called a **dictionary** which is defined as a mutable unordered set of key:value pairs. The English translation of this is:

mutable - can be changed once it is created (like a list but not a variable)
unordered - the data is not stored in any secific order in memory (unlike lists which are stored in numeric order)
set - one or more pieces of data of the same or different types can be stored
key:data pairs - individual pieces of data are accessed using a **key** instead of their relative position

Key:Data Pairs - An Example

A phone book is a good example of how key:data pairs work when organizing data. Phone numbers could be listed numerically (ie. 544-0000, 544-0001, etc) which would not be particularly useful for users looking for a friend's phone number.

Instead, a phone book is organized alphabetically by name along with the corresponding phone number. A person looking for a phone number looks up their friend's name in the phone book to find the phone number. The friend's name is used as a **key** to find the phone number (the data).

Uses of Dictionaries

Dictionaries have a number of uses including grouping data (keeping it together) so it is easier to work with. For example, instead of using four variables to store the four pieces of data about a person, a single dictionary with four keys (fields) is used. (See *DictionaryBasic.py* and *DictionarySingle.py* examples).

An even more useful application is using dictionaries along with lists to store a great deal of related data. (See DictionaryList.py example)

A Basic Example of Using a Dictionary and Related Functions in a Program

```
''' DictionaryBasic.py - a basic program using a dictionary and related functions '''

dataAboutPete = {"Name":"Pete Brown","Grade":"9","Average":89.2} # Initializes a 3 item dictionary called dataAboutPete
                                                                 # Dictionaries follow the same naming rules as variables
print ("A list of the all the key:value pairs in the dictionary 'dataAboutPete' is", dataAboutPete.items() )
# The .items() function outputs all key:value pairs in dictionary
```

```
print ("A list of the all the keys in the dictionary" 'dataAboutPete' is",dataAboutPete.keys() )
# The .keys() function outputs all the keys in the dictionary

print ("A list of the all the values in the dictionary 'dataAboutPete' is",dataAboutPete.values() )
# The .values() function outputs all the values in dictionary

print ("The average in dataAboutPete is called ", dataAboutPete ["Average"] )
# 89.2 is outputted because it is the value matched with the key "average" Note the square brackets around the key.

whatKey = input ("What data would you like to see?") # Key inputted by the user

if whatKey in dataAboutPete:  # Checks to see if the key in whatKey is a key in the dictionary dataAboutPete

    print (dataAboutPete [whatKey] )          # Outputs the value if the key was found.
                                              # Note the square brackets around the key.
else:
   print ("No such key")     # Outputs message if the value in whatKey is not a key in the dictionary.
```

An Example of Using a Dictionary to Input and Output Data

```
''' DictionarySingle.py - a program that uses a dictionary to store four pieces of data about a user. Note that instead of using
four variables to store the four pieces of data, a single dictionary with four keys (fields) is used.'''

student = {} # Creates an empty dictionary called student

student ["Name"] = input ("Enter student's name: ")
# Inputs a piece of data into the dictionary. "Name" is the key which will be  used to access this data.
# Note the square brackets which signify "Name" is a key for the data the user inputs'''

student ["Homeform"] = input ("Enter student's homeform: ")
# Inputs another piece of data accessed using the key called  "Homeform"

student ["Age"] = int (input ("Enter student's age: "))
# Inputs another piece of data accessed using the key called "Age". Note it is converted to an integer value

student ["Average"] = float (input("Enter student's average: "))
# Inputs average accessed using the key called "Average". Note it is converted to a float value

# Data must be outputted one key at a time (ie. print (student) will give you information that is not useful to the user)
print ("The student's name is ", student ["Name"])
print ("The student's homeform is ", student["Homeform"])
print ("The student's age is ", student["Age"])
print ("The student's average is ", student["Average"])
```