**Project 5 Guidelines**

**Introduction**

For Project 5, you will be creating a dictionary in Python. Python already has a built-in dictionary but we'll be making our own version.

**Dictionary**

A dictionary is sort of like an array but you can index it with anything, not just an integer. For example:

myDict[3822] = 'fun'  
myDict['Temple'] = 12345  
myDict[3.84] = 'an important number'

The general format is

myDict[key] = value

**Implementation**

Your dictionary will consist of two arrays (one for keys and one for values). When the user asks to insert a new value into the array (such as myDict['Temple'] = 12345), you create a hash value from the key (convert 'Temple' into an integer that can index the arrays). Once you have the hash value, you should insert the key accordingly into the key array, and the value into the value array.

When the user wants to read a value from the dictionary, you use the same process. If the user wants to see what value is at myDict['Temple'], you should first convert 'Temple' into its hash value, and then return valueArray[hashValue].

**What Code to Write**

At a bare minimum, your code should do the following - this will get you a maximum of 35/50 points:

Write a dictionary class that allows the user to **insert** and **retrieve** values. For this part, you can assume there are no collisions with hash values. Your class should also allow you to **print** the valid contents of the dictionary. Allowable keys are strings, integers, and floats.

**For an additional 5 points:**

Implement the \_\_setitem\_\_ and \_\_getitem\_\_ functions in your class so that instead of having to do:

myDict.insert(3822 , 'fun')

you can instead say:

myDict[3822] = 'fun'

You can easily Google these functions to see what they do and how to use them.

**For an additional 5 points:**

Implement a collision avoidance strategy, in case two keys map to the same hash values.

**For an additional 5 points:**

Modify the print function so that the dictionary contents are printed alphabetically by key. Edit: for this part you can assume there are no collisions - that should simplify things a lot.