CS160 Computer Science I
Lab 12

Objective
Work with files
Working with formatted output
Work with dictionaries

Assignment
Part 1
Write a program to ask for a series of menu items and prices. Ask for the menu items until the user enters a blank value. For every valid menu item ask for a price. The prices may be floating point values. Fill a dictionary with this data. Once the user has finished entering data ask for a file name. You may use `input` or `FileUtils` to ask for the file name. Write out each menu item/price pair in the dictionary to the data file, one pair per line. Separate the menu item and price with a tab (**"**`\t`**"**).

Part 2
Write a program that asks the user for a data file and then displays some information about the data.

First ask the user for the name of the text file. You may use `input` or `FileUtils` to ask for the file name. Fill a dictionary with the values from the data file using `readMenuItems()`.

Next, after the dictionary has been filled, you will need to write a variety of functions that will work with the menu items and their prices. The function header for each function must be implemented exactly as specified.

Finally, write a main() function that tests each of your functions.

Do not use a global variable to store the dictionary. This will be an automatic 50% deduction before any other grading.

Display:
- The total number of menu items
- A list of menu items
- All menu items with a price within a certain range
- Take an order
- Add menu item
- Update menu item
- Access a menu items price
- The average of all prices
- Print menu

Required Functions

`def readMenuItems (fileName)` – Creates a dictionary in the function and fills the dictionary with the menu items/prices from the `fileName` file. Then it returns the dictionary. No error checking is required.

`def totalMenuItems (theDictionary)` – Returns the number of menu items in the dictionary.

`def getMenuItems (theDictionary)` – Returns a list of the menu items. The list should be sorted by the menu items. The list should be created in the function.

`def getMenuItemsWithinRange (theDictionary, lowerLimit, upperLimit)` – Returns the menu items whose price falls inclusively within the range specified by `lowerLimit` and `upperLimit`. The list must be created in the function.

`def addMenuItem (theDictionary, item, price)` – If `item` does not currently exist in the dictionary add it, with its `price`, to the dictionary and return `True`. If `item` does currently exist in the dictionary do nothing to the dictionary, and return `False`.

`def updateMenuItem (theDictionary, item, price)` – If `item` does exist in the dictionary update its price and return `True`. If `item` does not currently exist in the dictionary do nothing to the dictionary, and return `False`.

`def getMenuItemPrice (theDictionary, item)` – If `item` does exist in the dictionary return its price. If `item` does not currently exist in the dictionary return `None`.

`def avergePrice (theDictionary)` – Returns the average of all menu items in the dictionary. Use `round()` to ensure that the average price has exactly 2 places after the decimal point.

`def takeOrder (theDictionary)` – This function will ask for menu items until the user enters a blank value. For each menu item, ask for the quantity for that item. Calculate the cost of the overall order. You do not need to store each item in the order, just calculate the total cost. This function will require input and output. This function should not return the total cost of the order.

`def printMenu (theDictionary)` – This function WILL write to the display a table of each menu item and its price. Include column headers in the output. Make sure the price has two places after the decimal point. Also make sure the columns are neatly aligned, with the menu item column being left justified and the price column being right justified. This function should not return a value.

Questions?
Ask, sooner is better than later.