CS160 Computer Science I
Lab 11

Objective

Work with files
Working with formatted output
Work with dictionaries

Assignment

Part 1

Write a program to ask for a series of student names and locker numbers. You can assume that each student will have their own locker. Ask for a student name until the user enters a blank value. For every valid student name ask for their locker number. 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 name/locker number pair in the dictionary to the data file, one pair per line. Separate the student name and locker number with a tab (`“\t”`).

For this assignment, limit the student names to the single name, such as “Tom”, “Scott”, or “Sue”. The locker numbers will be integer values, with a maximum of 3 digits.

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.

Next, after the dictionary has been filled, write functions that will return various information about the students and their lockers. Do not use a global variable to store the dictionary. Remember, unless the function specifically requests that you write information to the display, do not write anything to the display.

Make sure your main program thoroughly tests each required function. Ensure your function headers are written EXACTLY as specified below.

Required Functions

`def totalStudents (theDictionary)` – Returns the number of students with a locker.

`def matchingByName (theDictionary, firstLetter)` – Returns a list of the student names where the first letter of their name starts with `firstLetter`. The list to be returned should be created in the function.

For example, Using the sample data mentioned earlier, if the function was used like this:
`names = matchingStudents (lockingInfo, “S”)`
`names` would be a list containing two values, “Scott” and “Sue”

`def matchingByLocker (theDictionary, lowerLimit, upperLimit)` – Returns a list of student names whose locker number falls within the inclusive range of `lowerLimit` to `upperLimit`. The list to be returned should be created in the function.

`def firstLocker (theDictionary)` – Returns the “smallest” locker number.

`def lastLocker (theDictionary)` – Returns the “largest” locker number.

`def findLocker (theDictionary, studentName)` – Returns the locker number of the specified student (`studentName`). Returns `None` if `studentName` is not the name of a current student.

`def findStudent (theDictionary, lockerNum)` – Returns the student name associated with the specified locker (`lockerNum`). Returns `None` if `lockerNum` is not the locker number of a current student.

`def students (theDictionary)` – Returns a list of all student names, sorted by name. The list to be returned should be created in the function..

`def printInfoSortedByStudent (title, theDictionary)` – This function WILL write to the display a table of each student and their locker number. Print the title on its own line before printing the table. In the table include column headers in the output. Ensure the columns are neatly aligned, with the student name being left justified and the locker number being right justified. This function should not return a value. The info should be sorted by student name.

`def printInfoSortedByLocker (title, theDictionary)` – This function is identical to the `printInfoSortedByStudent`, with the difference that the info should be sorted by locker number.

Questions?

Ask, sooner is better than later.