Honors Computer Science Python
Mr. Clausen
Program 5A, 5B, 5C, 5D

PROGRAM 5A I'm Sort of Searching for Monty Python (35 points)

This program is going to practice the sorts and searches that we cover in every programming language. Python has some built in methods to help achieve this goal. This program will practice the Python list methods as well as the standard algorithms.

1) Use a `DocString` at the beginning of the program for your comments. Type comments at the beginning of the program to display your name and other information just like those used for program 1A. **Be sure to change the program name, program number, and program description.**

2) Leave a blank line after the DocString.

3) import random

4) Leave a blank line after the "import" statement.

5) Initialize all of the variables that are to be used in this program. Initialize each variable on a separate line. (Initialize integers to 0 (zero), decimal numbers to 0.0, strings to "" (double quotes with nothing in between them, and Boolean variables to False). **Initialize lists to [ ] and make sure that the word "list" is part of the identifier name for the list (ie. wordList).** Make sure that you use descriptive identifiers for all of your variables to model "self-documenting code", and that all variables are initialized at this place in the program.
Please use the names originalList and listCopy. The word "list" is a reserved word in Python. Also use the variable logicalSize to represent how many elements are in the list.

6) Leave a blank line after the variable initialization statements.

7) Type the following comment:
#----------Display My Information----------
Follow this comment with print statements to display your name and period output just like those used for program 1A. **Be sure to change the program name, and program number.**

8) Leave a blank line after the print statements listed above.

9) **This program does not follow the input, calculations, output model.**

10) Start a "while loop" that continues until the user enters an upper case Q or a lower case q to quit the program.

11) Display the menu (as pictured below) inside the "while loop" and ask the user to enter their choice. The menu needs to repeat every time the while loop iterates.

12) Implement the "if elif" statements for each of the menu choices listed below. Your "if elif" statements need to work for the upper case and lower case letters in the menu choices.

13) For this program we will implement menu choices 1, 2, 0 (zero), P, S, I, and D. **Please implement these parts of the program in this order.** Upper case and lower case values should work for the menu choices. For all the other menu choices use a 'print' statement that says that this menu choice hasn't been implemented yet.

14) In menu choice #1, ask the user how many elements they want in the list. Generate random integers from 1 to this value to fill the list with that many elements.

15) In menu choice #2, properly copy the list.

16) For menu choice **0 (zero)** use a loop to iterate through the list and display each number using format specifiers to allow for spaces between each of the numbers. Print ten numbers across each row, then print a new line character, and repeat this process.

17) For menu choices P, S, I, and D use Python's List Methods to implement these choices.
For choice P use the Python sort method.
For choice I, ask the user for the index number where they would like to insert the number, what number they would like to insert, and then use the insert method to do the rest. Don't forget to increment the variable logicalSize which keeps track of how many elements are in the list
For choice D, ask the user for the index number where they would like to delete the number, and use the pop method to delete the number at this index location. Don't forget to decrement the variable logicalSize which keeps track of how many elements are in the list.

18) For menu choice "S" ask the user for the number they wish to find. Display the index number of **every occurrence** of this number as well as how many times the number was found in the list. To find the number of occurrences in the list, please use Python's count method. I suggest checking to see if the number is in the list first, then use the linear (sequential) search to display the index numbers of every occurrence of this number in the list.

19) Finish your program with these last 2 lines of code.
```
print ("")
input("Press enter to quit the program")
```

20) Save your program as LastNameFirstNameP5A.py.

21) When you are finished with your program, have tested it thoroughly to make sure that your program is correct, and are sure that you don't need to make any changes, then save your program in the "T" network mapping, in the Program 5A folder.

```
 I'm Sort Of Searching For Monty Python

1.  Generate Random Numbers For The Original List
2.  Copy The Original List
3.  Sequential (Linear) Search
4.  Binary Search (The List Must be Sorted)
5.  Binary Search Recursion (The List Must be Sorted)
6.  Selection Sort
7.  Bubble Sort
8.  Insertion Sort
9.  Quick Sort
P.  Python's Sort Method
S.  Python's Search and Count Method
I.  Insert an element in the List at any position
D.  Delete an element from the List at any position
0.  Display the numbers using a 'loop' (press zero)
Q.  Quit
```

PROGRAM 5B Functional Sphere (35 points)

This program is going to practice writing functions as we rewrite Program 2A using functions. Write a program to calculate the diameter, circumference, surfaceArea, and volume of a sphere. Ask the user to enter a value for the radius. Make your program user friendly by prompting them for this value. The formulas are listed at the end of these instructions.

1) Use a `DocString` at the beginning of the program for your comments. Type comments at the beginning of the program to display your name and other information just like those used for program 1A. **Be sure to change the program name, program number, and program description.**

2) Leave a blank line after the DocString.

3) import math.

4) Leave a blank line after the import statement.

5) Type a one line comment of all equal signs.
   #==

6) Define the **main** function. The main function should declare and initialize all of the variables and call all of the other functions. The last line of code in the main function should be: input("Press enter to quit the program")

7) Initialize all of the variables that are to be used in this program **in the main function** and call all of the other functions from the main function. Initialize each variable on a separate line. (Initialize integers to 0 (zero), decimal (float) numbers to 0.0, strings to “” (double quotes with nothing in between them, and Boolean variables to False). Make sure that you use descriptive

identifiers for all of your variables to model "self-documenting code", and that all variables are initialized at this place in the program.

8) After the main function, have a one line comment of subtraction signs. Include this comment after each function as well.
   #----------

9) Define the function displayMyInfo. Inside this function use print statements to display your name and period output just like those used for program 1A. **Be sure to change the program name, and program number.** Include a comment line of subtraction signs after the function. Call this function from the main function and run your program to see if it works without errors. **See step #17.**

10) For all function definitions from here to the end of the program, remember to include a DocString at the beginning of each function with a brief description of what the function does, a "type contract", and some sample function calls with sample expected outputs.

11) Define the function getData that asks the user for the radius (float) and returns this value. Pass the radius to this function when calling it from the main function. Call this function from the main function and run your program to see if it works without errors. **See step #17.**

12) Define the function calcDiameter that accepts the radius (float) as an argument and returns the diameter. Pass the radius to this function when calling it from the main function. Call this function from the main function and run your program to see if it works without errors. **See step #17.**

13) Define the function calcCircumference that accepts the diameter (float) as an argument and returns the circumference. Pass the diameter to this function when calling it from the main function. Call this function from the main function and run your program to see if it works without errors. **See step #17.**

14) Define the function calcSurfaceArea that accepts the radius (float) as an argument and returns the surfaceArea. Pass the radius to this function when calling it from the main function. Call this function from the main function and run your program to see if it works without errors. **See step #17.**

15) Define the function calcVolume that accepts the radius (float) as an argument and returns the volume. Pass the radius to this function when calling it from the main function. Call this function from the main function and run your program to see if it works without errors. **See step #17.**

16) Define the function displayOutput that accepts the radius, diameter, circumference, surfaceArea, and volume (floats) as arguments and displays the values of these variables. Echo out the value of the radius as well as the results of your calculations for the diameter, circumference, surfaceArea, and volume of a sphere in this function. Call this function from the main function and run your program to see if it works without errors. **See step #17.**

17) On a line by itself after all of the functions type: **main()** #You will need this line of code to call the main() function, or your program will not run.

18) Type two comment lines of equal signs to indicate that this is the end of the program.
```
#=======================================
#=======================================
```

19) Save your program as LastNameFirstNameP5B.py.

20) When you are finished with your program, have tested it thoroughly to make sure that your program is correct, and are sure that you don't need to make any changes, then save your program in the "T" network mapping, in the Program 5B folder.

diameter = 2 * radius
circumference = diameter * math.pi
surface area = 4 * math.pi * radius **2
volume = 4/3 * math.pi * radius ** 3

PROGRAM 5C Compounding Your Interest (40 points)

This program is going to practice writing functions as we solve a compound interest problem. Write a program to calculate the final amount of interest earned or owed. Ask the user to enter the principal, interest rate, number of times the interest is compounded per year, and the number of years that the principal is invested or borrowed. Each of these inputs need their own separate function since we can return only one value from each function. Make your program user friendly by prompting them for these values.

1) Use a `DocString` at the beginning of the program for your comments. Type comments at the beginning of the program to display your name and other information just like those used for program 1A. **Be sure to change the program name, program number, and program description.**

2) Leave a blank line after the DocString.

3) Type a one line comment of all equal signs.
```
#============================================================
```

4) Define the **main** function. The main function should declare and initialize all of the variables and call all of the other functions. The last line of code in the main function should be: input("Press enter to quit the program")

5) Initialize all of the variables that are to be used in this program **in the main function** and call all of the other functions from the main function. Initialize each variable on a separate line. (Initialize integers to 0 (zero), decimal (float) numbers to 0.0, strings to "" (double quotes with nothing in between them, and Boolean variables to False). Make sure that you use descriptive identifiers for all of your variables to model "self-documenting code", and that all variables are initialized at this place in the program. I know that this formula is a "famous formula", but for this program don't use any one letter variables. Please use the variable names: interest, principal,

amount, time, rate, and numCompounded.

6) After the main function, have a one line comment of subtraction signs. Include this comment after each function as well.
   #----------

7) Define the function displayMyInfo. Inside this function use print statements to display your name and period output just like those used for program 5B. **Be sure to change the program name, and program number.** Include a comment line of subtraction signs after the function. Call this function from the main function and run your program to see if it works without errors. **See step #16.**

8) **For all function definitions from here to the end of the program, remember to include a DocString at the beginning of each function with a brief description of what the function does, a "type contract", and some sample function calls with sample expected outputs.**

9) Define the function getPrincipal that asks the user for the principal (float) and returns this value. Pass the variable principal to this function when calling it from the main function. Call this function from the main function and run your program to see if it works without errors. **See step #16.**

10) Define the function getTime that asks the user for the length of time for the investment or loan in a variable named time (float) and returns this value. Pass the variable time to this function when calling it from the main function. Call this function from the main function and run your program to see if it works without errors. **See step #16.**

11) Define the function getNumCompounded that asks the user for the number of times the interest is calculated per year in a variable named numCompounded (float) and returns this value. Pass the variable numCompounded to this function when calling it from the main function. Call this function from the main function and run your program to see if it works without errors. **See step #16.**

12) Define the function getInterestRate that asks the user for the interest rate expressed as a decimal (ie. 5% = 0.05) in a variable named rate (float) and returns this value. Pass the variable rate to this function when calling it from the main function. Call this function from the main function and run your program to see if it works without errors. **See step #16.**

13) Define the function calcAmount. Pass the variables principal, time, numCompounded, and rate to this function when calling it from the main function. This function should calculate the amount you have or owe using the formula listed below. Call this function from the main function and run your program to see if it works without errors. **See step #16.**

14) Define the function calcInterestOnly. Pass the variables amount and principal to this function when calling it from the main function. This function should calculate the interest only that you have or owe using the formula listed below. Call this function from the main function and run your program to see if it works without errors. **See step #16.**

15) Define the function displayOutput that accepts the principal, time, numCompounded, amount, rate, and interest (floats) as arguments and displays the values of these variables. Echo out the values for principal, time, number of times compounded, and interest rate as well as the results of your calculations for the final amount and interest earned/owed in this function. Call this function from the main function and run your program to see if it works without errors. **See step #16.**

16) On a line by itself after all of the functions type: **main()** #You will need this line of code to call the main() function, or your program will not run.

17) Type two comment lines of equal signs to indicate that this is the end of the program.
    #==========
    #==========

18) Save your program as LastNameFirstNameP5C.py.

19) When you are finished with your program, have tested it thoroughly to make sure that your program is correct, and are sure that you don't need to make any changes, then save your program in the "T" network mapping, in the Program 5C folder.

amount = principal * (1+ rate/numCompounded)**(numCompounded*time)
interest = amount – principal

PROGRAM 5D PhoneBook (50 points)

This program is going to practice writing functions as we practice Python dictionaries. Write a phone book program to perform all of the operations illustrated in the menu below. Each of these menu choices need their own separate function. Make your program user friendly by prompting them for these values. While much of the code can be found in our textbook, remind me to teach the class how to "invert a dictionary" for the reverse number lookup function. Since dictionaries can only lookup "keys" and not "values", we have to invert the dictionary into another dictionary called invertedPhoneBook in order to invert the keys and values (making what used to be keys as values and what used to be values as keys). To keep this a simple inversion, make sure that each name and phone number in the original dictionary is unique (no repeated names or phone numbers). It can be done otherwise, but the duplicate values would be grouped in lists.

```
 Phone Book
1. Look up a phone number
2. Add a phone number
3. Change a phone number
4. Delete a person and their phone number
5. Display all names and phone numbers in the phone book
6. Display all the names in the phone book in alphabetical order
7. Display the dictionary and inverted dictionary for debugging purposes
8. Display all the names & numbers in the phone book in alphabetical order
9. Reverse phone number lookup
Q. Quit the program
Enter your choice
```

1) Use a `DocString` at the beginning of the program for your comments. Type comments at the beginning of the program to display your name and other information just like those used for program 1A. **Be sure to change the program name, program number, and program description.**

2) Leave a blank line after the DocString.

3) Type a one line comment of all equal signs.
   #==

4) Define the **main** function. The main function should declare and initialize all of the variables and call all of the other functions. The last line of code in the main function should be:
   input("Press enter to quit the program")

5) Initialize all of the variables that are to be used in this program **in the main function** and call all of the other functions from the main function. Initialize each variable on a separate line. (Initialize integers to 0 (zero), decimal (float) numbers to 0.0, strings to "" (double quotes with nothing in between them, and Boolean variables to False). Make sure that you use descriptive identifiers for all of your variables to model "self-documenting code", and that all variables are initialized at this place in the program.

   Here are the only "variables" that need to be declared in the main function:
   Declare the dictionary phoneBook and include data for two names and corresponding phone numbers. Declare invertedPhoneBook and leave this dictionary empty for now, invertedPhoneBook = {}. Use the names phoneBook and invertedPhoneBook for the dictionaries.
   Declare a variable for menuChoice and initialize this to an empty string.
   Leave a blank line.
   Call the function displayMyInfo in the main function.
   Leave a blank line.
   Create a while loop that will continue until the user enters a lowercase "q" or an uppercase "Q" to quit running the program. Inside the while loop, call a function named "menu" that displays all of the menu choices for this program and gets the user's choice.
   Next (inside the while loop) type all of the "if" "elif" choices for the program. For now, just place a print statement in each choice saying what choice it represents. **As you define each function, call the function from here.**

The last line of code in the main function should be:
input("Press enter to quit the program")

6) After the main function, have a one line comment of subtraction signs. Include this comment after each function as well.
#----------

7) Define the function displayMyInfo. Inside this function use print statements to display your name and period output just like those used for program 5B. **Be sure to change the program name, and program number.** Include a comment line of subtraction signs after the function. Call this function from the main function and run your program to see if it works without errors.

8) **For all function definitions from here to the end of the program, remember to include a DocString at the beginning of each function with a brief description of what the function does. (This program does not lend itself well to type contracts and sample function calls.) DO NOT NAME ANY FUNCTIONS WITH NAMES LIKE: menuChoice1, etc. THE NAME OF THE FUNCTION SHOULD DESCRIBE WHAT THE FUNCTION DOES, NOT HOW IT RELATES TO THE MENU.**

9) Define and implement the function named **menu** which prints all of the menu choices, inputs and **returns** the user's choice. **(Don't forget to call it in an assignment statement from the main function.)** Include a comment line of subtraction signs after the function. Call this function from the main function and run your program to see if it works without errors.

10) Define and implement the function named **numberLookUp.** Pass the phoneBook dictionary into this function. If the name the user chose to look up is not in the phoneBook, let them know without the program crashing, otherwise display the phone number. Include a comment line of subtraction signs after the function. Call this function from the main function and run your program with menu choice 1 to see if it works without errors.

11) Define and implement the function named **addAPerson.** Pass the phoneBook dictionary into this function. Ask for the name that the user wants to add to the phoneBook, check to see if this name is already in the phoneBook. If it is not in the phoneBook, ask for the phone number and add the name and phone number to the phoneBook. Include a comment line of subtraction signs after the function. Call this function from the main function and run your program with menu choice 2 to see if it works without errors.

12) Define and implement the function for menu choice #7 to display both dictionaries for debugging purposes. In order for this to work, you will have to invert the phoneBook in this function. This will be useful as you work on the program. Include a comment line of subtraction signs after the function. Call this function from the main function and run your program with menu choice 7 to see if it works without errors.

13) Define and implement the functions for all of the other menu choices using function names that describes what the function does, not which menu choice it is. Write one function at a time and test the function to make sure it works. Pass the phoneBook (and possibly the invertedPhoneBook where necessary) into these functions.

For the function that changes the phone number, check to see if the name is in the dictionary first before trying to change the phone number.
Also, when deleting a "contact" check to see if the name is in the dictionary before deleting it.
Other than menu choice #7, when displaying the contents of the entire dictionary, use a "for loop" to display each choice without just saying **print (phoneBook)**.
Include a comment line of subtraction signs after each function. Call each function from the main function and run your program with the appropriate menu choice to see if it works without errors.

14) In the two functions that display the names only, and the names and numbers in alphabetical order, the dictionaries need to be assigned to a list which can be sorted and displayed using a "for loop".

15) Remember that keys in a dictionary need to be immutable types, so be sure to put quotes around the names and phone numbers (so we can invert it) to treat them as strings.

16) For the function that performs the reverse phone number lookup, make the inverted dictionary an empty dictionary, then invert the dictionary. After asking for which phone number to look up, check to see if the number is in the dictionary before trying to look it up. Include a comment line of subtraction signs after each function. Call each function from the main function and run your program with the appropriate menu choice to see if it works without errors.

17) On a line by itself after all of the functions type: **main()** #You will need this line of code to call the main() function, or your program will not run.

18) Type two comment lines of equal signs to indicate that this is the end of the program.
#==========
#==========

19) Save your program as LastNameFirstNameP5D.py.

20) When you are finished with your program, have tested it thoroughly to make sure that your program is correct, and are sure that you don't need to make any changes, then save your program in the "T" network mapping, in the Program 5C folder.