COMP1005/1405 Fall 2024 Tutorial 10

Do not post the tutorial or the tutorial and/or solutions on any website.

Objectives

- Practice writing/running Python code in the VSCode
- Practice coding **File I/O, Recursion, and Try-Except.**

Expectations

To receive full grades for this tutorial, you must complete Problems 1-3.

Grading Scheme for Tutorials

For each tutorial, you will be graded based on the following scale:

- 2/2 for demonstrating problems 1-3 and your **YourStudentNumber_T10.zip** file
    - Section A Tutorial Sessions: submit your work to the tutorial BrightSpace. The zip file should contain your solutions to all the required problems.
    - Section C, D, and E Tutorial Sessions: By the end of the tutorial session, demonstrate your tutorial work in person to a teaching assistant.
- 1/2 if you are missing problems or your solutions need significant improvement.
- 0/2 if you do not submit to BrightSpace or demonstrate to a teaching assistant
    - Section A Tutorial Sessions: no submission to the tutorial BrightSpace
    - Section C, D, and E Tutorial Sessions: not demonstrating your tutorial work to a teaching assistant by the end of the tutorial session

---

Problem 1 (Recursion)

Write a Python file named **`sumBetween.py`** in VSCode to implement recursive functions.

1. **Define a Recursive Function called sumBetween():**
    a. The function takes two integer parameters: start, end
    b. The function **calculates and returns** the sum of all integers from start to end (inclusive).
    c. The function must apply a recursive design.
2. **Create and Execute the main Function:**
    a. Prompt the user to input two integer values separated by a comma.
    b. The first value is start, and the second is end.
    c. Ensure the start is greater than or equal to the end and both inputs are integers.
    d. If the input is invalid, display an error message and prompt the user until valid inputs are provided.
    e. Call function **sumBetweenNums(start, end)** and print out the result.

Problem 2 (Recursion)

Write a Python file named **`peaches.py`** in VSCode to write a program for Recursive Implementation.

A monkey picks a certain number of peaches on the first day, eats half of them, and one more. On each subsequent day, the monkey eats half of the remaining peaches and one more. On the last day, there is only one peach left.

Enter the number of days (nDays) the monkey eats peaches, and calculate and return the number of peaches the monkey originally had on the first day.

a. **Prompts** the user to input a value for **nDays** that must meet the following conditions:
    1. The number has to be an integer:
        Required: Use **Try-Except** to capture the **ValueError** if the user enters a non-integer value.
    2. The number has to be a positive integer.

b. **Create** a **recursive** function called **calcNumPeach():**

- The function takes one parameter: an integer **nDays**.
- The function should calculate the number of peaches the monkey picked on the first day.

c. Sample Output:

```
n = 1; initial_peaches = 1
n = 2; initial_peaches = 4
n = 5; initial_peaches = 46
```

Problem 3 (File Input/Output)

Write a Python file named **`fileProcessing.py`** in VSCode to apply File Input/Output.

a. **Data file**:

i. **sample_data.txt** **(found in the tutorial 10 BrightSpace module)**

ii. The file includes the animal names and their age.

b. Create a **function** called **readData**()

- The function takes 2 parameters - a string **filename** and an integer **n**.
- Read the data in the file: **sample_data.txt.**
- The function should read the file and store **every n'th** line in a list, starting from line 1 but not including it. For example, if there are 10 lines in the file and n = 3, the function will include lines 3, 6, and 9.
- **Return** the list.

Sample Output:

```
print(readData("sample_data.txt", 3))
# Every 3rd line of the file:
# ['wombat, 2', 'quokka, 5', 'kookaburra, 1']
print(readData("sample_data.txt", 2))
# Every 2nd line of the file:
# ['capybara, 7', 'koala, 3', 'quokka, 5', 'dingo, 2']
```

c. Create a **function** called **ageLessThan**()

- The function takes 2 parameters - a string **filename** and an integer **n**.
- The function should store all the names of all the animals with ages less than **n** in a list and return the list.
- Read the data in the file **sample_data.txt**

Sample Output:

```
print(ageLessThan("sample_data.txt", 4))
#['wombat', 'koala', 'dingo', 'kookaburra']
print(ageLessThan("sample_data.txt", 9))
#['kangaroo', 'capybara', 'wombat', 'koala', 'wallaby', 'quokka', 'dingo', 'kookaburra']
```

d. Define a **`main`**( ) function that calls both the **`readData`**( ) and **`ageLessThan`**( ) functions and then prints the values returned by each.

Final Step

For Section A (Submit the work before the tutorial ends):

1. **Submit** your **zip** file to our Merged Tutorial Brightspace. The due date of your submission is aligned with your tutorial session.
2. **After** you submit the file, download your submission from Brightspace and confirm that it is a zip file containing **`sumBetween.py, peaches.py, and fileProcessing.py.`**
3. **Extract** the .py files and execute the extracted files again to ensure they work properly. Occasionally, a problem can occur during uploading, and files can become corrupted.

For Sections C, D, and E (Show the TAs your work before the tutorial ends):

1. **Problem 1-3:** Run your Python programs in VS Code to demonstrate they are working.
2. Answer the questions the TA may ask.
3. Show the TA your zip file and extract the files.