Non-Textbook Lab 13: Recursive Functions (100 points)

Objective: Practice writing recursive functions

Using the starter code **`recursive_functions.py`**, write the five recursive functions described below. For each function, **before** you write the code, figure out how to solve it conceptually: write down the base case (when recursion stops) and how each recursive function-call moves towards the base case. The functions should not print anything (except you may add temporary print statements to help debug them).

Use the provided program **`test_recursive_functions.py`** to test your program. **Do *not* edit the test program.** Put it into the same directory (folder) with your **`recursive_functions.py`** and run it. It will import your functions as a module, test your functions, and tell you when each function is returning correct results.

1. Factorial (20 points)

In math, if you have a number **n**, the **factorial** function (written n!) computes **n** x **(n-1)** x **(n-2)** x **(n-3)** x **…** x **1**. For example:

- **0! is defined to be 1**
- **1! = 1**
- **2! = 2** x **1 = 2**
- **3! = 3** x **2** x **1 = 6**
- **4! = 4** x **3** x **2** x **1 = 24**
- **5! = 5** x **4** x **3** x **2** x **1 = 120**

Add your code to the provided function signature so it computes the factorial of the integer it is given. You may not use math.factorial() in your function.

2. Recursively Sum (20 points)

Write a recursive function that takes an integer argument **n** and returns the **sum** of the integers from **1** to **n**. E.g., if **n=10**, the function should return **55** (1 + 2 + 3 + . . . 10).

3. Recursively Sum List (20 points)

Write a *recursive* function `sumlist_recursively(l)` that accepts a list of numbers as an argument, and returns the sum of all the numbers in the list. E.g., if the function is passed the list [1, 4, 8, 3, 0, 16], it should return 32.

4. Recursively Multiply (20 points)

Write a recursive function that multiplies its two integer arguments (**x** and **y**) recursively. The function should return the value of **x** times **y**. Hint: multiplication can be performed as repeated addition, e.g.:

7 x 4 = 4 + 4 + 4 + 4 + 4 + 4 + 4 = 28

5. Recursively Reverse a List (20 points)

Write a recursive function that reverses a list. For example, given **[1, 2, 3, 4]**, the function would return this list **[4, 3, 2, 1]**. It does **not** print anything.

Submitting

Upload **`recursive_functions.py`** and a **`readme.txt`** to NT Lab 13 in Canvas.