Carleton University
Department of Systems and Computer Engineering
ECOR 1041 – Computation and Programming

Lab 10 – Loops: for and while

Objective

- To develop some functions that use loops to process lists and handle user inputs.

Overview

Your solutions can use:

- the [] operator to get and set list elements (e.g., `lst[i]` and `lst[i] = a`)
- the `in` and `not in` operators (e.g., `a in lst`)
- the list concatenation operator (e.g., `lst1 + lst2`)
- the list replication operator (e.g., `lst * n` or `n * lst`)
- Python's built-in `len`, `min`, and `max` functions, **unless otherwise noted.**

Your solutions **cannot** use:

- list slicing (e.g., `lst[i : j]` or `lst[i : j] = t`)
- the `del` statement (e.g., `del lst[0]`)
- Python's built-in `reversed` and `sorted` functions.
- any Python methods that provide list operations, e.g., `sum`, `append`, `clear`, `copy`, `count`, `extend`, `index`, `insert`, `pop`, `remove`, `reverse`, `sort`, etc.
- list comprehensions (not taught in the course)

Getting Started

Begin by creating a new file within Wing 101. **Save it as lab10.py**

Automated testing is required for this lab.

It will be helpful to bring the functions you developed for automated testing in previous labs. In this lab, you will need to write a new function to test the functions that return a list.

Exercise 1 (…/20)

Use the function design recipe to develop a function named `Fibonacci_sequence`. The function takes an integer n. The function returns a list containing the Fibonacci sequence until the nth term. The Fibonacci sequence is defined as $F_n = F_{n-1} + F_{n-2}$. For example, if n = 6, then the Fibonacci sequence is 0, 1, 1, 2, 3, 5. The function returns a list [0, 1, 1, 2, 3, 5]

Exercise 2(…/20)

Use the function design recipe to develop a function named `max_min`. The function prompts the user to enter integers until the user enters zero. The function stores the entered integers in a list. . The function returns the largest and smallest elements in the list as a string. This is an example of the interaction of the function with the user and the return value of the function:

```
Please enter an integer (enter zero to quit): <user types <3>>
Please enter an integer (enter zero to quit): <user types <10>>
Please enter an integer (enter zero to quit): <user types <7>>
Please enter an integer (enter zero to quit): <user types <-6>>
Please enter an integer (enter zero to quit): <user types <11>>
Please enter an integer (enter zero to quit): <user types <0>>
```

The function returns

```
'The list = [3, 10, 7, -6, 11], max = 11, min = -6'
```

For this exercise, your function cannot call Python's **min** and **max** functions. It is true that Python offers a very elegant and efficient solution, but in this exercise, we want you to practice writing loops.

Your solution must include at least, one while loop and one for loop.

As you are dealing with inputs from the user, no automated testing is required for this function.

Exercise 3 (…/20)

Use the function design recipe to develop a function named `max_occurrences`. The function takes a list of integers, which may be empty. The function returns the value with the maximum number of occurrences in a given list. For example, when the function's argument is [2, 4, 7, 9, 8, 2, 6, 5, 1, 6, 1, 2, 3, 4, 6, 9, 1, 2], the function returns the value with the maximum number of occurrences which is 2.

Exercise 4 (…/20)

Use the function design recipe to develop a function named `bank_statement`. The function has two input parameters: (1) a floating-point value representing the account balance and (2) a list of floating-point numbers, which will always have at least one number. Positive numbers represent deposits into a bank account, and negative numbers represent withdrawals from the account. The function returns a floating-point value representing the new account balance. After the decimal point, the account balance must be rounded to two digits of precision (read Chapter 3, pages 33-34).

Your function must have exactly one loop.

Note: when the value returned by the function is displayed, a number such as 15.0 or -17.3 will be displayed with one digit after the decimal point instead of two. This is ok.

Exercise 5 (…/20)

Use the function design recipe to develop a function named `prime_numbers`. The function takes two positive integers (lower and upper). It returns a list containing all the prime numbers in the range of lower and upper numbers. For example, if `prime_numbers` is called with arguments 1 and 4, the list will contain [1, 2, 3]. If `prime_numbers` is called with arguments 4 and 1, the list will also contain [1, 2, 3]

Wrap Up

Ensure that your code meets the posted marking rubrics for the labs.

- Make sure that you included your name and student number
- Check proper use constants (UPPER_CASE) and variables (lower_case) (There is a 10/100 deduction for misuse of UPPER & lower case)
- Check the indents of the function bodies. (There is a 10/100 deduction for misuse of indentation)
- Check file organization: (1) imports, (2) CONSTANTS, (3) all function definitions; (4) Main Script (There is a 10/100 deduction for not organizing the file according to the instructions)
- Confirm that your filename matches exactly.
- Confirm that your .py script runs properly; otherwise, the TA will assign a zero.
- Submit the file on Brightspace.

You are required to keep a backup copy of (all) your work for the duration of the term.

Last edited: February 9, 2022