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

Final Exam Review Exercises

Use automated testing to test all the functions.

Exercise 1

The area of a cylinder can be calculated by the following function:

$$f : h, r \rightarrow f(h, r); f(h, r) = 2\pi r(h + r)$$

where h is the height of the cylinder and r is the radius of the base.

Using the FDR, design and implement a function to calculate the area of a cylinder

Follow the 5-step FDR. The only limits that you have to follow are those made to help to mark

- The name of your function must be: `area_cylinder`
- Function takes two integers parameters which are radius and height.
- Function returns the area of the cylinder

Exercise 2

Use the function design recipe (FDR) to develop a function named `triangle_type` to check if a triangle is equilateral, isosceles, or scalene. The function takes three integers arguments which are the lengths of the triangle sides. The function returns the triangle type, calculated as follows:

- An equilateral triangle is a triangle in which all three sides are equal.
- A scalene triangle is a triangle that has three unequal sides.
- An isosceles triangle is a triangle with (at least) two equal sides.

Exercise 3

A supermarket provides discounts depending on the price of items. For example, if the item price is $50, you will get a discount worth 9% of that amount ($4.5). The following table shows the percentage used to calculate the discount awarded for different price items:

Item Price	**Discount Percentage**
Less than $10	No discount
From $10 to $65	9%
More than $65 to $120	13%
More than $120 to $200	17%
More than $200	20%

Use the function design recipe to develop a function named `discount`. The function's input parameter is the item price. It returns the value of the discount in dollars.

Exercise 4

Use the function design recipe to develop a function named `sum_uniques`. This function takes three integer values, a, b and c, and returns their sum. However, if one of the values is the same as another value, that value is not used when the sum is calculated. For example, if the values of a, b and c are 3, 2, and 3, respectively, the sum is 2 (the two 3's are not used). If the values of a, b and c are 3, 3, and 3, the sum is 0 (the three 3's are not used).

Exercise 5

Use the function design recipe to develop a function named `same_first_last`. The function takes a list of integers and an integer x (e.g., 6, 90,etc). (The list may be empty.) The function returns True if the list is not empty and if the first and last elements are equal to the integer x. Otherwise, the function returns False.

Exercise 6

Use the function design recipe to develop a function named `canada_letters`. The function has no arguments. It returns a list of length six containing the letters of CANADA.

Exercise 7

Use the function design recipe to develop a function named `rotate_right`. The function takes a list containing three integers. The function returns a new list containing the same elements, but they are "rotated right". For example, when the argument is [1, 2, 3], the function returns [3,1,2].

Exercise 8

Use the function design recipe to develop a function named `average2`. The function takes a list of integers. The function returns the average of the first two list elements. It should return 0 if the list is empty. It should return the element if the list has one element.

Exercise 9

Use the function design recipe to develop a function named `make_ends`. The function takes a list of integers. (Assume that the list will not be empty.) The function returns a new list containing the first and last elements from the original list. For example, when the argument is [4, 5, 6, 7], the function returns [4, 7].

Exercise 10

Use the function design recipe to develop a function named `count_evens`. The function takes a list of integers, which may be empty. The function returns the number of even integers in the list.

Exercise 11

Use the function design recipe to develop a function named `unique_list`. The function takes a list of strings, which may be empty. The function returns a new list that includes the first occurrence of each value in a list and omits later repeats. The returned list should include the first occurrences of values in a list in their original order. For example, list contains the following animals **['cat', 'dog', 'cat', 'bug', 'dog', 'ant', 'dog', 'bug'],** the unique_list function returns a unique list: **['cat', 'dog', 'bug', 'ant'].**

Exercise 12

Use the function design recipe to develop a function named `sum_range`. The function takes a list of integers, which may be empty, and two indices m, n that specify the start and end range. The function returns the sum of the numbers in the list between the indices of a specified range, returning 0 for an empty list. For example, when the function's argument is [2,1,5,6,8,3,4,9,10,11,8,12], and indices are m=8, and n = 10, the function returns the sum of the specified range which is 29.