בס"ד

Home Exercise Nr 2 - Functional Programming with Python

Important Note for this Home Exercise: In class we learned basic principles of the functional approach to programming. In all the questions in this home exercise, you are required to write several functions – all those functions (excluding those that need to be in direct interaction with the user) need to be written as *pure* functions; that is, functions *without side effects*:

a. The returned value needs to be dependent only on the input values; that is, that the returned value will be the result of some appropriate transformation of the input values.
b. The function will be stateless.
c. State changes, if there will be, they will occur only inside the local environment of the function, without any effects on the external environment, relative to the function. All this, excluding the value returned by the function to its caller.

If there will be cases in which it will be extremely difficult to have stateless functions, do the most minimal state changes possible, by writing helper functions that will implement purely functional solutions for sub-problems of the whole problem.

It is important to note that in the solutions to the questions of this home exercise, it is forbidden to use non-functional language constructs, like for loops and while loops, excluding cases where it will be explicitly said that it is allowed to used such constructs.

1. Numbers called "penta numbers" are defined as follows:

$$\frac{n(3n-1)}{2} \; for\ each\ n = 1,2, \ldots.$$

   a. Write a function called pentaNumRange(n1,n2) that returns a list containing all the penta numbers that correspond to values of n within the interval [n1,n2). In order to calculate all those penta numbers, create a variable called getPentaNum, local to the pentaNumRange function. The value of getPentaNum will be a functional object which will be created by a lambda expression which expresses the formula above.
   b. Write a function which will ask the user to enter two Natural numbers, n1 and n2 (n1 < n2), and will print all the penta numbers corresponding to the values of n, between n1 and n2, 10 per line. Implement the way of printing those lines, by two alternative ways: (1) by using a for loop; (2) by using only functional programming tools which we learned in class.

2. Write a function which receives a Natural number n, and returns the sum of its digits.
   a. Write a first version sumDigits1(n), by applying the sum built-in function upon the list of digits of the number n. That list will be built by a function which will decompose the number n, by repeatedly applying the division operator and the modulo operator, storing the digits exactly in the same order as they were in the input number.

   b. Write a second version sumDigits2(n), by applying the sum built-in function upon the list of digits of the number n. That list will be built by a function which will use (part or all of) the built-in functions str, abs, list, int, and functional tools only.
   c. Write a function which will ask the user to enter a Natural number, calls the two versions of sumDigits, that you wrote in a' and b' above, and prints, in two different lines, the values that both versions return.
3. Write a function that receives a Natural number n, and returns the number with its digits reversed.
   a. Write a version reverseNum1(n) which implements the function by composing built-inn functions like str, abs, int, and by using a slice expressing the reverse of a string.
   b. Write a second version reverseNum2(n) by applying the built-in reversed function upon the list of the digits of the number n. This list will be built by one of the functions you wrote in question 2 above.
   c. Write a function that will ask the user to enter a Natural number, it will call the two versions of reverseNum, separately, will print both results in different lines.
4. Write a boolean function called isPalindrome(n) which receives a Natural number n, and returns True if n is a palindrome, and returns False, otherwise.
   Note: implement this function by using one of the versions of reverseNum, that you wrote in question 3, above.
   Write a function that will ask the user to enter a Natural number, calls the isPalindrome function, and prints "It is a palindrome!!" or "It is not a palindrome", in accordance with the boolean value that isPalindrome returned.
5. Write a function m(n) which receives a Natual number n, and returns the value of calculating the following series:

$$\sum_{i=1}^{i=n} \frac{i}{(i+1)}$$

   a. Write a version that implements the required function by applying the sum built-in function upon a list of values of i/(i+1) for each i, inside the interval [1,n]. This list will be built and returned by some helper function. In order to calculate the values of i/(i+1) for each i, inside the interval [1,n], apply an anonymous function (created by an appropriate lambda expression) which expresses that formula. <u>It is mandatory to use functional tools, from those that we learned in class</u>.
   b. Write a function which will ask the user to enter a Natural number n, calls the above function, and prints the values of i and m(i), for each value of i in the interval [1,n].
6. Write a function pi(n) which receives a Natural number n, and returns an approximation of the value of the number π, according to the following formula:

$$pi(n) = 4\left(\sum_{i=1}^{i=n} \frac{(-1)^{i+1}}{2i-1}\right)$$

   a. Write a first version of this function by applying the built-in function sum upon a list of values of the formula $(-1)^{i+1}/(2i-1)$ for each i in the interval [1,n]. This list will be built by a helper function. In order to calculate the value of the formula $(-1)^{i+1}/(2i-1)$

for each i in the interval [1,n], apply an anonymous function created by a lambda expression which expresses that formula. <u>It is mandatory to use only functional tools, from those that we learned in class.</u>

   b. Write a function which will ask the user to enter a Natural number n, calls the above function, and prints on every line the values of i and pi(i), for every value of in in [1,n].

7. Two prime numbers are called "twin primes" if the difference between them is 2.
   a. Write a function twinp(n) which receives a Natural number n, and returns a dictionary whose keys are primes and their bound values are their twins. In order to calculate the list of twin primes in the interval [1,n], import the module eratosthenes, used by us at class, and use the function called napa, from that module.
      A reminder: one of the methods to create a dictionary is as follows:
      ```
      >>> dict(zip(['a','b','c'],[3,4,5]))
      {'a':3,'b':4,'c':5}
      ```
   b. Write a function which will ask the user to enter a Natural number n, calls the above function, and prints the twin primes that the function found in [1,n], every pair in a different line.

8. Write a function add2dicts(d1,d2,d3) that receives three dictionaries, and returns a new dictionary which will be built as follows:
   Every key which is shared by all the input dictionaries, will be a key in the new one, and the value bound to that key will be a tuple which contains all the corresponding values taken from the input dictionaries (without duplicates!!). Every pair whose key is not shared by all the input dictionaries, will be inserted in the new one, as it is.
   Note: write helper functions based on lambda expressions that express operations on sets, which calculate the set of shared keys, and separately, set of non-shared keys.
   Run example:
   ```
   >>> d1 = dict([(1,'a'),(3,'d'),(5,'e')])
   >>> d2 = dict([(1,'b'), (3,(11,22)), (7,'f'), (4,'q')])
   >>> d3 = dict([(2,'c'), (3,'x'), (4,'t'), (8,'g')])
   >>> add3dicts(d1,d2,d3)
   {1:('a','b'), 2:'c', 3:('d',(11,22),'x'), 4:('q','t'), 5:'e', 7:'f', 8:'g'}
   ```

   Write a function which will ask the user to enter the three input dictionaries, will call the function add3dicts, and will print what that function returns.

9. Write a program which will behave as the main program of all this home exercise. The program will display a menu from which the user will choose the function he wishes to run (from all the functions written in the questions 1 thru 8, above. In order to manage the application of all those functions, in accordance to the menu, a dictionary will be created in the main program, in which the keys will be the options of the menu, and the bound values will be functional objects of the above functions, according to the menu options. The program will call a function according to the option that user chooses, then, the called function will receive its input data from the user, and the program will print the values returned by that function. The option "0" will mean to quit from the menu

loop. The behavior of the menu will be very similar to the menu your used in the Home exercise Nr 1 – you certainly may use that menu as a code basis to this new menu.