NYU, Tandon School of Engineering

CS-UY 1114 Introduction to Programming and Problem Solving — Fall 2015

Homework # 9

Due _Nov. 23rd 11:55pm_

Submission instruction:

- Note the unusual deadline. It is the Saturday before the second midterm.
- Submit one .py file which contains your answers to all the questions.
- Unless specified otherwise, you should have a main program for each question. For example, def main_q1():, def main_q2():, etc.
- Grading will be done on Question 1-4. Question 5-7 are extra practice problems.

Question 1:

Implement function `max_abs_val(lst)`, which returns the maximum absolute value in lst.

For example, given a list lst [-19, -3, 20, -1, 0, -25], the function should return 25.

Question 2:

Write a function `find_all(lst, val)`, which returns a list containing indices of the occurrences of val in lst.

For example, given a list lst [‘a’, ‘b’, 10, ‘bb’, ‘a’] and val ‘a’, the function should return a list [0, 4].

Question 3:

Implement following functions.

a. `reverse1(lst)`

This function creates a new list containing the elements of lst in reverse order and returns it..

b. `reverse2(lst)`

It changes the order of the elements in lst **(in place)** to be in the reverse order.

Hint: think about the how the positions change before and after the reverse.

You may use the following main program to check your implementation of `reverse1` and `reserse2`.

```
def main_q3():
    lst1 = [1, 2, 3, 4, 5, 6]
    rev_lst1 = reverse1(lst1)
    print(”After reverse1, lst1 is ”, lst1, " and the returned 
list is ”, rev_lst1)
    lst2 = [1, 2, 3, 4, 5, 6]
    reverse2(lst2)
    print(”After reverse2, lst2 is ”, lst2)
```

Expect output:

```
After recerse1, lst1 is [1, 2, 3, 4, 5, 6] and the returned list
is [6, 5, 4, 3, 2, 1]
After reverse2, lst2 is [6, 5, 4, 3, 2, 1]
```

Question 4:

Design and implement functions to perform run length encoding of strings.

https://en.wikipedia.org/wiki/Run-length_encoding

a. Run length string encoder. The encoder should take a string, and return the encoded string in a list.

For example, if given string ”aadccccaa”, after encoding it becomes: [[’a’, 2], [’d’, 1], [’c’, 4], [’a’, 2]].

b. Run length string decoder. The decoder should take an encoded string (in a list), and returns the original string.

For example, if given an encoding [[’a’, 2], [’d’, 1], [’c’, 4], [’a’, 2]], the decoded string is: ”aadccccaa”.

Hint: carefully design your functions. If any of your functions seem to be too complex, you may want to write some 'helper' functions that solves some part of your problem.

Optional questions for practice:

Question 5:

Write a function `add_list( lst1, lst2)` that takes two lists of numbers of the same length and returns a list consisting of the sum of the first numbers, the sum of the second numbers, etc.

For example `add_list([1,2,3], [4,5,6])` should return the list [5, 7, 9] since 1+ 4 == 5, 2 + 5 == 7, etc.

Your main program should prompt user for input. It should read a list of numbers, one number per line, followed by "done", then another list of numbers, one per line, followed by done. If the lists have different lengths, your main program should print "Lists are different lengths." If they are the same length, the main program should call `add_list` and print the resulting list, one number per line.

Note: you may define additional 'helper' functions in your program.

Question 6:

Implement the function create_prefix_lists(lst) that will return a sequence of lists, each containing a prefix of lst. All of the lists should be collected as one big list. For example for [2,4,6,8,10] the function should return [[], [2], [2,4], [2,4,6], [2,4,6,8], [2,4,6,8,10]]

The challenge:

Question 7:

Write a function to check if a given lsit is a valid Sudoku board. Sudoku: https://en.wikipedia.org/wiki/Sudoku

You may test your program with the following two lists:

board1 = [

```
  [5,3,4,6,7,8,9,1,2],
  [6,7,2,1,9,5,3,4,8],
  [1,9,8,3,4,2,5,6,7],
  [8,5,9,7,6,1,4,2,3],
  [4,2,6,8,5,3,7,9,1],
  [7,1,3,9,2,4,8,5,6],
  [9,6,1,5,3,7,2,8,4],
  [2,8,7,4,1,9,6,3,5],
  [3,4,5,2,8,6,1,7,9]
]
board2 = [
  [5,3,4,6,7,8,9,1,2],
  [6,7,2,1,9,5,3,4,8],
  [1,9,8,3,4,2,5,9,7],
  [8,5,9,7,6,1,4,2,3],
  [4,3,6,8,5,3,7,9,1],
  [7,1,3,9,2,4,8,5,6],
  [9,6,1,5,3,7,2,8,4],
  [2,8,9,4,1,9,6,3,5],
  [3,4,5,2,8,6,1,7,9]
]
```