Week: 05 Date: 3/4/2021

|  |
| --- |
| 15-110 Recitation Week 5 |

**Reminders**

* Check 3 due Monday 3/8 @ Noon EDT
* Check 2 and HW 2 revisions due Tuesday 3/9 @ Noon EDT
* Quiz 2 is Wednesday 3/10, same procedure as Quiz 1
  + Quiz 2 small groups!
* Recitation feedback form: <https://forms.gle/WKrrbawKktmRu1xp9>

**Overview**

* Lists
* 2D lists
* Aliasing
* Recursion (code writing)

|  |
| --- |
| Problems |

# **LIST CODE WRITING: ALTERNATING SUM**

Write a function alternatingSum(L) that takes in a list of numbers L, and then returns the alternating sum (where the sign switches from positive to negative or negative to positive at each index).

For example, alternatingSum([5,3,8,4]) returns 6 as (5-3+8-4) = 6

See starter file for more tests and function header!

Code

# **LIST ALIASING**

Code trace and compare the following two options for ways to create “empty” 2D lists:

Option 1:

rows = 3

cols = 2

a = [ [0] \* cols ] \* rows

Option 2:

def make2dList(rows, cols, defaultValue=None):

a=[]

for row in range(rows):

a.append([defaultValue]\*cols)

return a

rows = 3

cols = 2

a = make2dList(rows, cols, 0)

**For each option**, after running the code, what is a?

Option 1: a = answer

Option 2: a = answer

After adding the following line of code:

a[0][0] = 42

What is a?

Option 1: a = answer

Option 2: a = answer

Be sure you can explain what difference you are seeing, and which option you should use and why.

# **RECURSION INTRO**

General notes on recursion:

Notes

Recreate the following function using recursion:

|  |  |
| --- | --- |
| def double(lst):  result = []  for elem in lst:  result.append(2 \* elem)  return result  #double([1,2,3]) -> [2,4,6] | def doubleRecursive(lst): |

# **RECURSIVE CODE WRITING**

Write the function rangeSum(lo, hi) which takes in two integers (where lo <= hi) and sums all values in between them inclusive.

Code

Write the function reverseList(L) which takes in a list L and nondestructively returns a reversed copy of L

Code