Lab 6
Lists

Z. Hutchinson
`zachary.s.hutchinson@maine.edu`

October 19, 2022

Goals

The goal of this lab is to practice using Python 3. Specifically, you will practice:

- Lists

Instructions

Work is due by the end of your lab session and must be submitted to Brightspace in the proper place. Unless otherwise instructed, submissions must be python files (e.g. files that end with *.py*). Any other format, even if it is plain text, will **not** be graded. Messy or otherwise unreadable code will lose points. Lab submissions can be all in the same file, but please label with comments to which task code belongs. IMPORTANT: Any code that is commented out will not be graded. **RUN YOUR CODE TO MAKE SURE IT WORKS!!!**

Task 1 - Lists and Functions

A

Write a function, *BuildRandomList* that builds a list, using append(), of random integers. The function has two parameters: *N* the number of random integers to add to the list and *R*, the maximum random integer to draw. In other words, BuildRandomList should return a list of size N in which each integer is a random number drawn from the range 1 to R. The function should return the newly built list.

B

Write a function, *PruneList* that takes a list of integers as a parameter. This function builds a new list from the old list with all duplicates removed. The function returns the list.

C

Using the functions from A and B, generate a list of 20 random numbers from 1 to 10 and then remove all duplicates. Print the list after both actions to test things are working correctly.

D

Write a function called *RemoveAll*. The function takes two parameters: *L*, a list, and *e* a possible element of L. The function should remove all instances of *e* from L. The function should alter the original list (i.e., not build a new one). It should remove ALL instances: there might be more than one.

E

Write a function, *CutAndPaste*. The function has two parameters: *L* a list and *i* an integer. The function cuts the list L in two, slicing it at the index, i. The function then concatenates the two halves back together in reverse order.

Example:

```
alist = [13,6,7,65,10,99,-5]
new_list = CutAndPaste(alist,3)
print(new_list)
```

would print: [65,10,99,-5,13,6,7].

Task 2 - 2D Lists (and functions)

A

Write a function *Print2DList* that takes a two dimensional list as a parameter and prints the list in a block. For example, if passed the following list:

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

it would print:

```
987
654
321
```

It should print a 2d list of any size or shape.

B

Write a function, *Copy2DList* which returns a **deep copy** of a 2D list. It has one parameter, *L*, which it expects to be a 2d list. Make sure that the returned list does not share any references in common with the original. You can check this by altering an element in the new list and checking whether the alteration also changes the original list.

C

Repeat Task 1 part D except *L* is a 2D list and *e* is an element of the 2D list. The function will now remove entire lists from *L* if any of the elements of the sublists match *e*. I suggest you generate a 5x5 random 2D list of integers drawn from the range 1 to 20.

D

Write a 2D list slicing function called *Slice2DList*. The function has three parameters: *L*, a 2D list; *rc* a string of a single character which is always either 'r' or 'c'; and *index* which is an integer. Depending on whether *rc* contains an 'r' or 'c', the function slices a row or column from the list. The index parameter is the row or column to slice. The function should return the slice.

Example:

```
Slice2DList(mylist,'c',2)
```

would return the list: [7,4,1] from the original mylist.

```
Slice2DList(mylist,'r',1)
```

would return: [6,5,4]. Make sure that the returned slice is a new list and not a reference to a part of mylist (HINT: a deep copy).

E - Extra

Write a function, *ReverseList*, which expects a 2D list as a parameter. The function reverses the list. Given mylist above, it would reorder the elements in the list such that afterwards mylist would contain:

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

It should work for any sized list.