Pre-lab 08: Loops, Strings and Lists

Study Chapter 7 and 8 of the textbook.

We encourage you to work together on the Pre Lab. The Pre Lab is not graded but will help you prepare for your lab session. In the Pre Lab, you may find questions to answer. We do not require you to provide us the answers, but we do recommend you to try to answer these questions. If you have any questions on the material in the pre lab, first check the book and recitation slides, if you do not find your answer please email your recitation TA or the course instructors.

Indexing in a nested list

You already know how to access an element of a list using an index. Now we will review how to index nested lists.

```python
>>> students = [['CS', 'MATH', 'MUSIC', 'BIO'], [10, 20, 30, 40], ['Freshman', 'Sophomore', 'Junior', 'Senior']
>>> print (students[0])
['CS', 'MATH', 'MUSIC', 'BIO']
>>> print (students[1][0])
10
>>> print (students[2][0])
'Freshman'
```

In the above code students[0] is ['CS', 'MATH', 'MUSIC', 'BIO']. So, students[0][1] is actually ['CS', 'MATH', 'MUSIC', 'BIO'][1], which clearly is MUSIC. In other words, the first [idx] is indexing into the outer list, returning the inner list, and the second [idx] is actually being applied to the list returned from the first index operation.

Using for-loops with Lists

If we want to display each element in a specific List we can use a “for” loop to traverse each index of the List:

```python
students = [['CS', 'MATH', 'MUSIC', 'BIO'], [10, 20, 30, 40], ['Freshman', 'Sophomore', 'Junior', 'Senior']]

for i in range(len(students)):
    print (students[i][0])

#output:
CS
10
```
Freshman

Or we can use the for-loop to access all the second (index=1) elements in each sub-list:

```python
students = [['CS', 'MATH', 'MUSIC', 'BIO'], [10, 20, 30, 40], ['Freshman', 'Sophomore', 'Junior', 'Senior']]

for i in range(len(students)):
    print (students[i][1])
```

#output:
MATH
20
Sophomore

If we want to print all the lists[i] and all the values from each sub-list we need to use two for loops:

```python
for i in range(len(students)):
    print ("List: "+str(i))
    for n in range(len(students[i])):
        print (students[i][n])
```

#output:
list: 0
CS
MATH
MUSIC
BIO
list: 1
10
20
30
40
list: 2
Freshman
Sophomore
Junior
Senior

We can store the values of different elements in a new list. The following example shows how to create a new list and append to it the new values. The “newlist” works like an accounting variable which keeps track of multiple values every time the outer loop iterates.

```python
students = [['CS', 'MATH', 'MUSIC', 'BIO'], [10, 20, 30, 40], ['Freshman', 'Sophomore', 'Junior', 'Senior']]

for i in range(len(students)):    # this is called outer loop
    newlist=[]
    for n in range(len(students[i])):
        newlist.append(students[i][n])
    print ("List: "+str(i))
    print (newlist)
```
for n in range(len(students[i])):  # this is called inner loop
    newlist.append(students[i][n])  # append the values of each list
print(newlist)

#output:
List: 0
['CS', 'MATH', 'MUSIC', 'BIO']
List: 1
[1, 2, 3, 4]
List: 2
['Freshman', 'Sophomore', 'Junior', 'Senior']

Printing a matrix

We use matrices in programming very frequently. In Python, there are multiple ways of representing a matrix with a data structure. A common way is list of lists:

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

```
for i in range(len(matrix)):
    for j in range(len(matrix[i])):
        print(matrix[i][j])
```

This will print:

```
1
2
3
4
5
6
7
8
9
```

Removing characters without using built-in functions

```
>>> a = 'hello world'
>>> a.replace('o', 'x')
'hellx wxrld'
```

We can have the same functionality using:

```
>>> a = 'hello world'
>>> b = ''
```
```python
>>> for c in a:
...     if c == 'o':
...         b += 'x'
...     else:
...         b += c
...>>> b
'hellx wxrld'
```