Python Workshop 3 Exercises

Examine and test the example code below, then design and write your code to answer the 7 questions that follow.

Lists and indexing in Python

Consider a list of numbers: nums = [90, 80, 70, 60, 50, 40, 30, 20, 10]

90	80	70	60	50	40	30	20	10
0	1	2	3	4	5	6	7	8
-9	-8	-7	-6	-5	-4	-3	-2	-1

Each character in the list has a notional position called the index,
to access one of the elements, the name of the list with the index (in square brackets) must be provided E.g:

```
print(nums[2]) # index from the front => 70
```

In Python lists may also be indexed from the back by using a negative number for the index so

```
print(nums[-2]) # index from the back => 20
```

Of course where you use a number in a program, you can replace with a variable to produce interesting results

Example 1

Write a Python function that returns the sum of two given vectors (represented as lists of the same length).

Solution

```
def sumVectors(x, y): vect = len(x)*[0] # create a list of the same length as x, filled with 0s i = 0 while i < len(x): vect[i] = x[i] + y[i] i += 1 return vect
```

Note that we have to create a vector of the required length before we can store the resulting values in it. Try a call such as sumVectors([1,2,3], [6,5,4]) to test the working of the function.

Example 2

Write a Python function that creates and returns a table with a given numbers of rows and columns, filled with None values.

Solution

```
def buildTable(nRows, nCols): table = [] for r in range(nRows): table.append(nCols*[None]) return table
```

We start with an empty list of rows table and append a new row to it nRows times. nCols*[None] creates one row: a list of nCols elements, all set to None. *You may want to experiment with the example functions in the space below:*

Now try to write the Python code to answer the following questions in the cells below

1. Write and test a function that returns the index of the largest element in the list (or, if several elements have the largest value, the index of the first one of them). Write your own algorithmn do not use built-in functions max or index or any library functions.

```
def element(value):
  max=value[0]
  max_index=0
  for i in range(len(value)):
    if max<value[i]:
```

```
      max=value[i]
    max_index=i
  return (max,max_index)
value=[]

#num of element as input

x=int(input("Enter the elements: "))

for i in range(0,x):
  elements=int(input())
  value.append(elements)
print(element(value))
```

```
Enter the elements: 4
23
24
45
56
(56, 3)
```

2. Write a python script (ie. does not need to be a function) that swaps the values of three variables x, y, and z, so by the end x gets the value of y, y has the value of z, and z gets the value that was in x. *Note: the swapping can be done is a single line of code if you use good pythonic design.*

```
x=12
y=14
z=16
print(x,y,z)

x,y,z=z,y,x
print(x,y,z)
```

```
12 14 16
16 14 12
```

3. Pretend that list's reverse method does not exist and write your own function reverseList(lst) that reverses the list in place and returns a None. *Hints: proceed from both ends, but not too far..., the swapping (from Q2) should be useful and no need for a return statement.*

```
def reverse(list):
  left_index=0
  right_index=len(list)-1
  while left_index<right_index:
    list[left_index],list[right_index]=list[right_index],list[left_index]
    left_index+=1
    right_index-=1
mylist=[3,5,7,9,12,45,7]
reverse(mylist)
print(mylist)
```

```
[7, 45, 12, 9, 7, 5, 3]
```

4. Write and test a function that takes a list with four or more elements and returns the index of the element that makes the greatest sum with its left and right neighbors. *Hint: a good solution here might be similar to your answer to Q1.*

```
def greatest(list1):
  max_sum=0
  max_index=0
  for index in range(1,len(list1)-1):
    sum=list1[index-1]+list1[index]+list1[index+1]

    if (max_sum<sum):
      max_sum=max_sum
      max_index=max_index
  return index

list1 [2 4 6 8 10 12 14]
```

```
list1=[2,4,6,8,10,12,14]
result=greatest(list1)
print("The greatest sum of index element is " , result)
```

```
The greatest sum of index element is  5
```

5. Write a function called *intersection* that takes 2 lists of integers and returns a third list values that occur in both lists ie. *intersection([2, 3, 4], [1, 2, 5])* should return [2]. Write your solution in pure Python to show the logic in your solution, do not use any library functions.¶

```
def intersection(listA,listB):
  result=[]
  for value in listA:
    if value in listB and value not in result:
      result.append(value)
  return result
intersection([2,3,4,5,6,7],[1,2,5,7])
```

```
[2, 5, 7]
```

6. Let $\bar{x} = (x_1, x_2 \ldots x_n)$ and $\bar{y} = (y_1, y_2 \ldots y_n)$ be two n dimensional vectors. Their *dot product*, denoted $\bar{x} . \bar{y}$, is defined as $\bar{x} . \bar{y} = x_1y_1 + x_2y_2 + x_3y_3 + \ldots + x_ny_n$. Write and test a Python function that returns the dot product of two given vectors (represented as lists of the same length).

```
def dotProduct(x,y):
  result=0
  for i in range(len(x)):
    result+=x[i]*y[i]
  return result
x=[2,3,4,5]
y=[1,3,5,7]
result=dotProduct(x,y)
print(result)
```

```
66
```

7. Write and test a Python function that returns the sum of the elements on the main diagonal (upper left to lower right) of a square matrix (represented as a list of lists *eg. [[9, 8, 7], [6, 5, 4], [3, 2, 1]]*). In linear algebra, this value is called the trace of the matrix. Note: your solution should work for any sized square matrix passed to it.

```
def matrixTrace(x):
  trace=0
  for i in range (len(x)):
    trace+=x[i][i]
  return trace
x=[[1,2,3],[4,5,6],[7,8,9]]
trace=matrixTrace(x)
print(trace)
```

```
15
```

Downloading your workbook

Please note even if you are Jupyter on your computer (ie. not on a network) the Jupyter notebook runs as a server, so you are editing and running a file that is not easily accessed on your filing system. So to obtain your workbook (to use on a different computer or upload as an assignment, etc.) it must be downloaded to your file system. To do this...

Goto the "**File**" menu and select the "**Download as**" item. You can then select "**notebook (ipynb)**" to download.

✓ 0s completed at 2:17 PM