Python exercises

Briefing

The goal of this assignment is for you to get used to python and the advantages of using libraries like numPy and matplotlib. Feel free to experiment with the libraries any way you like(you should have already done this by now). One of the main benefits of using numPy is to reduce the usage of loops, so work on that(minimize using loops as much as you can).

Once you get the hang of doing that, understanding the language and the reducing the complexity of each problem becomes so much easier !

Most of the questions can be completed using functions/classes. This way, you can use different data as inputs and hence get different results. I suggest that you dont just work toward the assignment for the sake of completion, but to test your inquisitivity and expand your knowledge.

I have not imported any of the libraries for you. Some of the libraries that you will be using in this notebook will be math, numpy, matplotlib and csv. Figure out how they are used and the benefits of using these libraries.

There are **ONLY** 30 questions and some of them are super easy to do and some require you think about the question a little. Before you ask me a question for help, research about the your question and then tell me what you found and why you think does not work. I do not think the questions are too complicated to be solved. You just need to spend enough time on them.

Submission for this assignment is on Sunday, 10th September by midnight. To submit, **either print the page** in pdf format or you can also use the **'download as'** option in the **'file'** menu(you may face some errors). If you complete it before that time, let me know and maybe I can give you more work to do :P

Happy programming !

In [640]:
```
import sys
import math
import numpy as np
import random
import matplotlib.pyplot as plt
sys.version
```

Out[640]:
```
'2.7.12 (default, Dec  4 2017, 14:50:18) \n[GCC 5.4.0 20160609]'
```

1

Write a program which will find all such numbers which are divisible by 7 but are not a multiple of 5, between 2000 and 3200 (both included). The numbers obtained should be printed in a comma-separated sequence on a single line.

In [20]:
```
lst1 = [str(x) for x in range(2000,3201) if x%7==0 and x%5!=0]
print ",".join(lst1)
```

```
2002,2009,2016,2023,2037,2044,2051,2058,2072,2079,2086,2093,2107,2114,212
1,2128,2142,2149,2156,2163,2177,2184,2191,2198,2212,2219,2226,2233,2247,2
254,2261,2268,2282,2289,2296,2303,2317,2324,2331,2338,2352,2359,2366,2373
,2387,2394,2401,2408,2422,2429,2436,2443,2457,2464,2471,2478,2492,2499,25
06,2513,2527,2534,2541,2548,2562,2569,2576,2583,2597,2604,2611,2618,2632,
2639,2646,2653,2667,2674,2681,2688,2702,2709,2716,2723,2737,2744,2751,275
8,2772,2779,2786,2793,2807,2814,2821,2828,2842,2849,2856,2863,2877,2884,2
891,2898,2912,2919,2926,2933,2947,2954,2961,2968,2982,2989,2996,3003,3017
,3024,3031,3038,3052,3059,3066,3073,3087,3094,3101,3108,3122,3129,3136,31
43,3157,3164,3171,3178,3192,3199
```

2

Write a program which can compute the factorial of a given numbers. The results should be printed in a comma-separated sequence on a single line. Suppose the following input is supplied to the program: 8 Then, the output should be: 40320

In [41]:
```
lst_numbers = [8, 4, 9]
def factorial(x):
    if x==1:
        return x
    else:
        return x * factorial(x-1)
print "input: {}".format(",".join(map(str,lst_numbers)))
print "output: {}".format(",".join(map(str,map(factorial,lst_numbers))))
```

```
input: 8,4,9
output: 40320,24,362880
```

3

With a given integral number n, write a program to generate a dictionary that contains (i, i*i) such that is an integral number between 1 and n (both included). and then the program should print the dictionary. Suppose the following input is supplied to the program: 8 Then, the output should be: {1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64}

In [44]:
```
dict_len = input("Enter an integer:")
print "output: {}".format({x:x*x for x in range(1,dict_len+1)})
```

```
Enter an integer:8
output: {1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64}
```

4

Define a class which has at least two methods: getString: to get a string from console input printString: to print the string in upper case. Also please include simple test function to test the class methods.

In [65]:
```
class String:
    def __init__(self,content="empty"):
        self.content = content
        return
    def getString(self):
        self.content = raw_input("enter a string :")
        return
    def printString(self):
        print "output : {}".format(self.content)
        return
obj1 = String()
obj1.getString()
obj1.printString()
```

```
enter a string :asd
output : asd
```

5

Write a program that calculates and prints the value according to the given formula: Q = Square root of [(2 *C* D)/H] Following are the fixed values of C and H: C is 50. H is 30. D is the variable whose values should be input to your program in a comma-separated sequence. Example Let us assume the following comma separated input sequence is given to the program: 100,150,180 The output of the program should be: 18,22,24

In [86]:
```
C, H = 50, 30
D = raw_input("Enter values seperated by commas : ")
Q = [str(int(round(math.sqrt(2*C*int(d)/H)))) for d in D.split(",")]
print "output : {}".format(",".join(Q))
```

```
Enter values seperated by commas : 100,150,180
output : 18,22,24
```

6

Write a program which takes 2 digits, X,Y as input and generates a 2-dimensional array. The element value in the i-th row and j-th column of the array should be i*j.

Note: i=0,1.., X-1; j=0,1,¡Y-1. USE NUMPY. Example

Suppose the following inputs are given to the program:

3,5

Then, the output of the program should be:

[[0, 0, 0, 0, 0], [0, 1, 2, 3, 4], [0, 2, 4, 6, 8]]

In [105]:
```
def Create_array(X,Y):
    arr1 = np.arange(X).reshape(X,1)
    arr2 = np.arange(Y).reshape((1,Y))
    arr = arr1 *  arr2
    return arr
Create_array(3,5)
```

Out[105]:
```
array([[0, 0, 0, 0, 0],
       [0, 1, 2, 3, 4],
       [0, 2, 4, 6, 8]])
```

7

Write a program that takes a comma separated sequence of words as input and prints the words in a comma-separated sequence after sorting them alphabetically.

Suppose the following input is supplied to the program(**you do not have to use raw_input**):

without,hello,bag,world

Then, the output should be:

bag,hello,without,world

In [116]:
```
input_string = "without,hello,bag,world"
sorted_array = np.sort(np.array(input_string.split(",")))
print ",".join(sorted_array)
```

```
bag,hello,without,world
```

8

Write a program that accepts a sequence of whitespace separated words as input and prints the words after removing all duplicate words and sorting them alphanumerically.

Use the following input:

hello world and practice makes perfect and hello world again

Then, the output should be:

again and hello makes perfect practice world

In [119]:
```
input_string = "hello world and practice makes perfect and hello world a
gain"
sorted_array = np.sort(np.unique(np.array(input_string.split(" "))))
print " ".join(sorted_array)
```

```
again and hello makes perfect practice world
```

9

Write a program which takes a sequence of comma separated 4 digit binary numbers as an input and then check whether they are divisible by 5 or not. The numbers that are divisible by 5 are to be printed in a comma separated sequence.

Input: 0100,0011,1010,1001

Then the output should be: 1010

In [168]:
```
input_string = "0100,0011,1010,1001,1111"
arr = np.array([int(b,2) for b in input_string.split(",")])
print ",".join(np.array(input_string.split(","))[arr%5==0])
```

```
1010,1111
```

10

Write a program, which will find all such numbers between 1000 and 3000 (both included) such that each digit of the number is an even number. Print all the numbers as a list.

In [198]:
```
arr = np.arange(1000,3000)
def even_digits(number):
    arr_str = np.array([int(x) for x in str(number)])
    return np.array_equal(arr_str[arr_str%2==0],arr_str)
print arr[map(even_digits,arr)]
```

```
[2000 2002 2004 2006 2008 2020 2022 2024 2026 2028 2040 2042 2044 2046
 2048 2060 2062 2064 2066 2068 2080 2082 2084 2086 2088 2200 2202 2204
 2206 2208 2220 2222 2224 2226 2228 2240 2242 2244 2246 2248 2260 2262
 2264 2266 2268 2280 2282 2284 2286 2288 2400 2402 2404 2406 2408 2420
 2422 2424 2426 2428 2440 2442 2444 2446 2448 2460 2462 2464 2466 2468
 2480 2482 2484 2486 2488 2600 2602 2604 2606 2608 2620 2622 2624 2626
 2628 2640 2642 2644 2646 2648 2660 2662 2664 2666 2668 2680 2682 2684
 2686 2688 2800 2802 2804 2806 2808 2820 2822 2824 2826 2828 2840 2842
 2844 2846 2848 2860 2862 2864 2866 2868 2880 2882 2884 2886 2888]
```

11

Write a program that uses a sentence and calculates the number of letters and digits.

Example input: hello world! 123

Then, the output should be:

LETTERS 10

DIGITS 3

In [230]:
```
inp_str = "hello world! 123"
letters = np.array(list(inp_str))[[char.isalpha() for char in list(inp_s
tr)]]
numbers = np.array(list(inp_str))[[char.isdigit() for char in list(inp_s
tr)]]
print "LETTERS: {}   NUMBERS: {}".format(len(letters),len(numbers))
```

```
LETTERS: 10   NUMBERS: 3
```

12

Write a program that is input a sentence and calculates the number of upper case letters and lower case letters.

Sample input: Hello world!

Then, the output should be:

UPPER CASE 1

LOWER CASE 9

In [231]:
```
inp_str = "Hello world!"
lower = np.array(list(inp_str))[[char.islower() for char in list(inp_str
)]]
upper = np.array(list(inp_str))[[char.isupper() for char in list(inp_str
)]]
print "UPPER CASE: {}   LOWER CASE: {}".format(len(upper),len(lower))
```

```
UPPER CASE: 1   LOWER CASE: 9
```

13

Write a program which accepts a string from console and print the characters that have even indexes.

Sample input: H1e2l3l4o5w6o7r8l9d

Then, the output of the program should be:

Helloworld

In [258]:
```
inp_str = raw_input("Enter a set of characters")
print "output: {}".format("".join(np.array(list(inp_str))[np.arange(len(
inp_str),step=2)]))
```

```
Enter a set of charactersH1e2l3l4o5w6o7r8l9d
output: Helloworld
```

14

Write a program which count and print the numbers of each character in a string.

Sample input: abcdefgabc

Then, the output of the program should be:

a,2

c,2

b,2

e,1

d,1

g,1

f,1

In [287]:
```
str1 = "abcdefgabc"
(unq_arr,count) = np.unique(np.array(list(str1)), return_counts=True)
for x in np.dstack((unq_arr,count))[0]:
    print ",".join(x)
```

```
a,2
b,2
c,2
d,1
e,1
f,1
g,1
```

15

Using the list as input

[12,24,35,24,88,120,155,88,120,155]

write a program to print this list after removing all duplicate values with original order reserved

In [300]:
```
inp_lst = [12,24,99,35,24,88,120,155,88,120,155]
(unq_arr,indices) = np.unique(np.array(inp_lst),return_index=True)
print unq_arr[indices.argsort()]
```

```
[ 12  24  99  35  88 120 155]
```

16

Using two lists as input, example, [1,3,6,78,35,55] and [12,24,35,24,88,120,155]

Write a program to make a list whose elements are intersection of the above given lists.

In [303]:
```
lst1 = [1,3,6,78,35,55]
lst2 = [12,24,35,24,88,120,155]
print "common elements: {}".format(np.intersect1d(lst1,lst2))
```

```
common elements: [35]
```

17

By using the built in list functions, write a program to print the list(10 elements) after removing a value from the list. Using the same list, write a program to print the list after removing the 1th,5th,7th numbers(**not indices**).

Example:

List :[12,10,9,24,35,24,88,65,120,155]

Remove : 24

In [323]:
```
arr1 = np.array([12,10,9,24,35,24,88,65,120,155])
num = input("Enter a number to remove:")
arr2 = np.delete(arr,np.where(arr==num))
print "The number {} has been removed: {}".format(num,arr2)
print "The 1st, 5th & 7th numbers have been removed: {}".format(np.delet
e(arr2,[0,4,6]))
```

```
Enter a number to remove:24
The number 24 has been removed: [ 12  10   9  35  88  65 120 155]
The 1st, 5th & 7th numbers have been removed: [ 10   9  35  65 155]
```

18

Write a program to generate all sentences(that contain the subject, verb and object) where subject is in list ["I", "You"] and verb is in ["can code in", "love"] and the object is in ["Foundation Course","Java, C++, Python"].

Example sentence:

I love Foundation Course

In [332]:
```
subjects = ["I", "You"]
verbs = ["can code in", "love"]
objects = ["Foundation Course","Java, C++, Python"]
for snt in [" ".join([x,y,z]) for x in subjects for y in verbs for z in
objects]:
    print snt
```

```
I can code in Foundation Course
I can code in Java, C++, Python
I love Foundation Course
I love Java, C++, Python
You can code in Foundation Course
You can code in Java, C++, Python
You love Foundation Course
You love Java, C++, Python
```

19

Please write a program to shuffle and print the list [2,3,9,6,7,1,5,10,8].

Hint: Check for a library that has the 'shuffle' function or if you can do it by yourself, well and good :D

In [342]:
```
lst = [2,3,9,6,7,1,5,10,8]
random.shuffle(lst)
print lst
```

```
[10, 9, 1, 5, 3, 8, 7, 2, 6]
```

20

Write a program to print the running time of execution of "1+1" for 100 times.

Hint: There are many ways to do this.

In [388]:
```
%%time
for x in range(100):
    add = 1+1
```

```
CPU times: user 525 µs, sys: 23 µs, total: 548 µs
Wall time: 359 µs
```

21

Write a program to randomly print an int between 2 and 20 inclusive of 20.

Run your code, and check if 20 is generated atleast once.

In [571]:
```
count = 0
for x in range(100):
    rand_num = np.random.randint(2,21)
    if rand_num==20:
        count += 1
print "The number '20' was generated {} times out of 100".format(count)
```

```
The number '20' was generated 6 times out of 100
```

22

Write a program to randomly generate a list that contains numbers in a range in such a way that you print n even numbers followed by n odd numbers, in the range.

Example:

Range: 1-10

n:2

Output:

[2,4,1,3,6,8,5,7,10,9]

In [619]:
```
int_range = "1-10"
n = 2
lst_nums = range(int(int_range.split("-")[0]),int(int_range.split("-")[1
])+1)
even_nums = [x for x in lst_nums if x%2==0]
odd_nums = [x for x in lst_nums if x%2==1]

output = [np.random.permutation(even_nums)[0] if math.ceil(x/float(n))%2
==1 else np.random.permutation(odd_nums)[0] for x in range(1,n*5+1)]
print "Output: {}".format(output)
```

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

23

A robot moves in a plane starting from the original point (0,0). The robot can move toward UP, DOWN, LEFT and RIGHT with given number of steps. The trace of robot movement is shown as the following:

UP 5

DOWN 3

LEFT 3

RIGHT 2

The numbers after the direction are the number of steps moved by the robot. Write a program to compute the distance from current position after a sequence of movement and original point. If the distance is a float, then just print the nearest integer. Example:

If the following tuples are given as input to the program:

UP 5

DOWN 3

LEFT 3

RIGHT 2

Then, the output of the program should be:

2

DO NOT USE raw_input

In [ ]:

25

Write a program to sort the (name, age, height) tuples by ascending order where name is string, age and height are numbers. Make provision for the sort criteria, which is:

- Sort based on name
- Then sort based on age;
- Then sort by score.

The priority is that **name > age > score**

If the following tuples are used as input to the program:

Tom,19,80

John,20,90

Jony,17,91

Jony,17,93

Json,21,85

Then, the output of the program should be:

[('John', '20', '90'), ('Jony', '17', '91'), ('Jony', '17', '93'), ('Json', '21', '85'), ('Tom', '19', '80')]

In [622]:
```
lst = [('Tom',19,80),('John',20,90),('Jony',17,91),('Jony',17,93),('Json
',21,85)]
lst.sort()
print lst
```

```
[('John', 20, 90), ('Jony', 17, 91), ('Jony', 17, 93), ('Json', 21, 85),
('Tom', 19, 80)]
```

26

Write a program that squares each odd number in a list.

Suppose the following input is supplied to the program:

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

Then, the output should be:

1,2,9,4,25,6,49,8,81

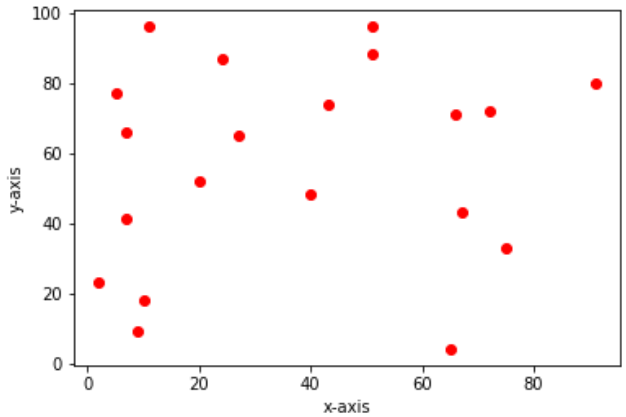
In [638]:
```
inp_lst = [1,2,3,4,5,6,7,8,9]
np_arr = np.array(inp_lst)
np_arr[np.where(np_arr%2==1)] = np_arr[np.where(np_arr%2==1)]**2
print "output: {}".format(",".join(map(str,np_arr)))
```

```
output: 1,2,9,4,25,6,49,8,81
```

27

Create two random arrays of equal length and plot the data as an XY graph(label the axes and try different plot styles?).

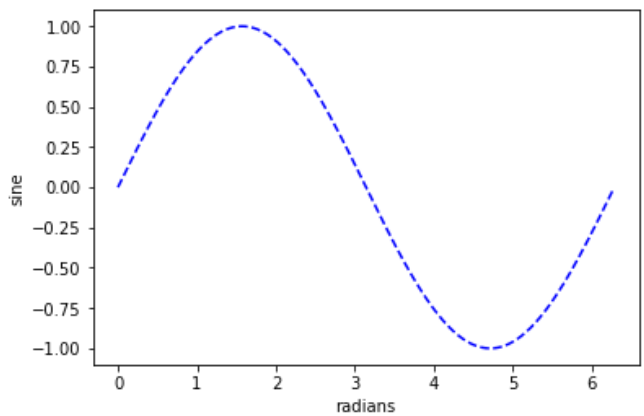
In [655]:
```
arr1 = np.random.randint(0,100,20)
arr2 = np.random.randint(0,100,20)
plt.plot(arr1,arr2,'ro')
plt.xlabel("x-axis")
plt.ylabel("y-axis")
plt.show()
```



28

Plot a sine function with a dashed line

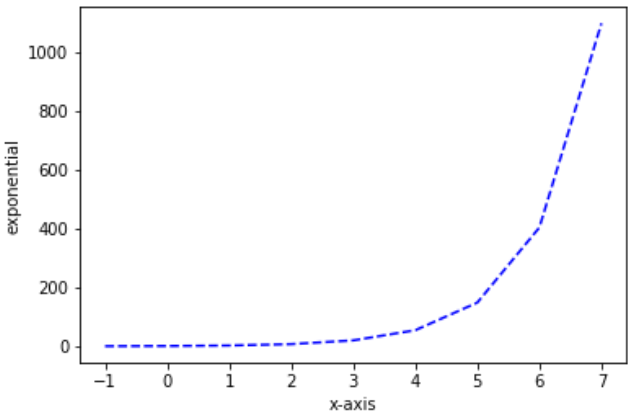
In [674]:
```
radians = np.arange(0,2*math.pi,0.01)
sine = np.array(map(math.sin,radians))
plt.plot(radians,sine,'b--')
plt.xlabel("radians")
plt.ylabel("sine")
plt.show()
```



29

Plot an exp function

In [688]:
```
x_vals = np.arange(-1,8)
exp = np.array(map(math.exp,x_vals))
plt.plot(x_vals,exp,'b--')
plt.xlabel("x-axis")
plt.ylabel("exponential")
plt.show()
```



30

Load the csv file using Python and use the data in the csv file to plot a graph with the x axis containing the day of the month and the y axis containing the time worked out.

The csv file is of the format: "2012, Mar-01", run, 2, 25

"2012, Mar-03", bike, 10, 55

"2012, Mar-06", bike, 5, 20

Lets take the first row. We start with the date of the workout, the kind of workout, how many miles you travelled and how much time you spent in minutes

You need to import the exercise_data.csv (\\:workout.csv)

Hints:

- Import the file and check its content. Note that the content may be different on different rows, adjust your algorithm accordingly(**BETTER HINT ;) -> check for the letter that each row starts with**).
- Extract the data you want from the csv according to the format that you want. Example: "2012, Mar-01" could be of type datetime (https://docs.python.org/2/library/datetime.html)

Your result should look a little like this:

An

In [ ]:
```
```