Dyson School of Design Engineering

DE1-Computing -1, Sample answers for assignment-1

This assignment carries 40% of the final mark.

This is a Python coding assignment. Please submit your answers as one Jupyter notebook file with code cells or a zip file of separate Python codes for each question. Note that you will get marks for your correct attempts even if the code gives errors for some lines.

Question - 1: Use the numpy module to generate two 5 by 1 arrays of integer random numbers between 0 and 15 and display the arrays. You can name the two arrays to be a1 and a2.

In [16]:
```
import numpy as np

a1 = np.random.randint(15,size=(5,1))
a2 = np.random.randint(15,size=(5,1))

print("The two random arrays are a1: {}, and a2: {}" .format(a1,a2))
```

```
The two random arrays are a1: [[ 1]
 [ 8]
 [ 5]
 [13]
 [14]], and a2: [[ 2]
 [11]
 [14]
 [13]
 [ 9]]
```

Question - 2: Display the transpose of the arrays a1 and a2 which should be of size 1 by 5

In [17]:
```
import numpy as np

a1 = np.random.randint(15,size=(5,1))
a2 = np.random.randint(15,size=(5,1))

a1t = np.transpose(a1)
a2t = np.transpose(a2)

print("The two random arrays are a1: {}, and a2: {}" .format(a1t,a2t))
```

```
The two random arrays are a1: [[ 6 13  3  4  0]], and a2: [[ 4  6  8 12
  5]]
```

Question - 3: Store arrays a1 and a2 in a dictionary structure. Choose your own names of keys and the dictionary. Hence print the 4th value of each array.

In [18]:
```
import numpy as np

a1 = np.random.randint(15,size=(5,1))
a2 = np.random.randint(15,size=(5,1))

arrays = {
    'a1' : a1,
    'a2' : a2
}
print("The 4th value of a1 is {} and that of a2 is {}" .format(arrays['a
1'][3],arrays['a2'][3]))
```

```
The 4th value of a1 is [13] and that of a2 is [4]
```

Question - 4: Write a Python code to compare the 4th value of arrays a1 and a2 and display the result of the comparison. i.e. 4th value of a1 is greater than or less than that in a2

In [5]:
```
import numpy as np

a1 = np.random.randint(15,size=(5,1))
a2 = np.random.randint(15,size=(5,1))

#a1 = [1,2,3,4,5]
#a2 = [1,2,3,3,5]
arrays = {
    'a1' : a1,
    'a2' : a2
}

if arrays['a1'][3] > arrays['a2'][3]:
    print("The 4th value of a1 is greater than that of a2")
elif arrays['a1'][3] == arrays['a2'][3]:
    print("The 4th value of a1 is equal to that of a2")
else:
    print("The 4th value of a1 is less than than that of a2")

#This can be done in another way
boolvalue1 = arrays['a1'][3] > arrays['a2'][3]
boolvalue2 = arrays['a1'][3] == arrays['a2'][3]
if boolvalue1:
    print("The 4th value of a1 is greater than that of a2")
elif boolvalue2:
    print("The 4th value of a1 is equal to that of a2")
else:
    print("The 4th value of a1 is less than than that of a2")
```

```
The 4th value of a1 is greater than that of a2
The 4th value of a1 is greater than that of a2
```

Question - 5: Write a Python function that takes a value of x and gives out the value of y, where $y = 0.1x^3 + 0.2x^2 - 1.2x - 2$. Hence, print the value of y at $x = 2$. Comment your code to explain the vital steps.

In [2]:
```
def myfun(x):#Define myfun that takes x as an input
    y = 0.1*x**3 + 0.2*x**2 -1.2*x -2 #Implement the equation
    return y #Return the value of y

x = 2
y1 = myfun(x) #Call myfun by passing x = 2
print("The value of y is {} at x = {}" .format(y1,x))
```

```
The value of y is -2.8 at x = 2
```

Question - 6: Extend the function you developed in answer to question 5 to compute a vector of values for y for a vector input for x. Hence, print the vector of y for vector of 5×1 random values between 0 and 10 for x.

In [7]:
```
import numpy as np

def myfun(x):#Define myfun that takes x as an input
    y = 0.1*x**3 + 0.2*x**2 -1.2*x -2 #Implement the equation
    return y #Return the value of y

vecmyfun = np.vectorize(myfun)

x = np.random.randint(10,size=(5,1))

y1 = myfun(x) #Call myfun by passing x
print("The value of y is {} at x = {}" .format(y1,x))
```

```
The value of y is [[ 19.6]
 [ 33.7]
 [ 19.6]
 [ 76.3]
 [ -2. ]] at x = [[6]
 [7]
 [6]
 [9]
 [0]]
```

Question - 7: Use a dictionary data structure called "func" to store the random value vector of x using the key, "Inputs" and the corresponding output vector in question - 6 using the key, "Output". Hence display the 3rd value of inputs and 2nd value of outputs.

In [11]:
```
import numpy as np

def myfun(x):#Define myfun that takes x as an input
    y = 0.1*x**3 + 0.2*x**2 -1.2*x -2 #Implement the equation
    return y #Return the value of y

vecmyfun = np.vectorize(myfun)

x = np.random.randint(10,size=(5,1))

y1 = myfun(x) #Call myfun by passing x

func = {
    'Inputs' : x,
    'Outputs' : y1
}
print("The value of y is {} at x = {}" .format(y1,x)) # This is just to
 check the next print
print("The 3rd value of inputs is {}, and the 2nd value of the outputs i
s {}" .format(func['Inputs'][2],func['Outputs'][1]))
```

```
The value of y is [[-2. ]
 [ 2.8]
 [-2.8]
 [ 9.5]
 [-2. ]] at x = [[0]
 [4]
 [2]
 [5]
 [0]]
The 3rd value of inputs is [2], and the 2nd value of the outputs is [
2.8]
```

Question - 8: Use symbolic Python to differentiate the function $y = 0.1x^3 + 0.2x^2 - 1.2x - 2$ with respect to x. Hence, plot y and $\frac{dy}{dx}$ in the range $x = [-3, 3]$ in steps of 0.1. You can plot the two curves in tw plots or in the same plot.

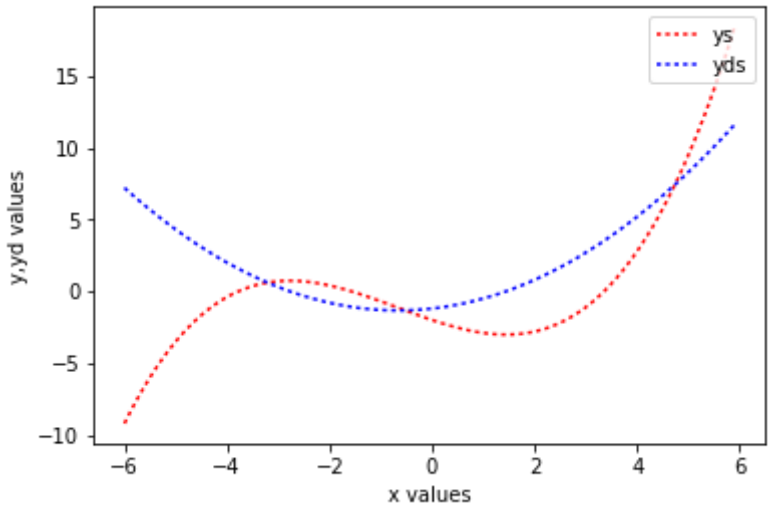
In [14]:
```
import matplotlib.pyplot as plt #Import plot module
import numpy as np #Import numpy for math and array handling
from sympy import * #Import symbolic py module

x,y = symbols('x y') #Declare the symbolic variables

y = 0.1*x**3 + 0.2*x**2 -1.2*x - 2
yd = diff(y,x) #Differentiate y with respect to x

xr = np.arange(-6,6,0.1) #Initialize xr between -6 and 6 in steps of 0.1
ys = [] #Empty arrays to be filled
yds = []
for i in range(len(xr)): #Loop through the length of xr
    ys.append(y.subs([(x,xr[i])])) #Substitute xr values in x and append
 to ys and yds arrays
    yds.append(yd.subs([(x,xr[i])]))


fig, ax = plt.subplots() #open a figure
ax.plot(xr,ys,'r:',label = 'ys') #Use the axis object to plot and add le
gend items
ax.plot(xr,yds,'b:',label = 'yds')
ax.legend(loc = 'upper left') #Specify the position of the legend
plt.xlabel('x values')
plt.ylabel('y,yd values')
plt.show()
```



Question-9: mention an advantage and a disadvantage of an Interpreter compared to a compiler

An intepreter translates high level language codes to machine language line by line when the program is running, whereas a compiler does that for the whole code in one go.

Advantage: Syntax and execution errors can be quickly identified because the program stops at the line that gives the error. Disadvantage: Relatively slow running compared to compiled programs because each line of the program must be translated before it can be run. This adds overheads to the CPU.

Question-10: Give an example for a semantic error in a program

Any example of a logical error is fine. For instance, a program that computes the compounded interest of a bank account should take the account balance at the end of the month, compute the interest, and add it back to the balance before computing the interest in the next month. Computing interest on the origical balance is a semantic error.

In [6]:
```
def add(x,y):
    res=x+y
    return 0
print(add(1,2))
# in this example we are expecting that the function return the addition
 but it will return 0
# all the time. The program can be run so it is a semantic error
```

```
0
```