CSB310: Artificial Intelligence

Instructor : [Dr. Chandra Prakash]

- For more information visit the class website.

LAB_ASSIGNMENT 3 : INTRODUCTION TO PYTHON PROGRAMMING

```
Due Date:  10-Sept-2022

Student Name: Umang Kumar
Roll No. : 201210051
```

Google CoLab Instructions

The following code ensures that Google CoLab is running the correct version of TensorFlow.

In [15]:
```
try:
    from google.colab import drive
    %tensorflow_version 2.x
    COLAB = True
    print("Hello World")
    print("Note: using Google CoLab")
except:
    print("Hello World")
    print("Note: not using Google CoLab")
    COLAB = False
```

```
Hello World
Note: not using Google CoLab
```

In [16]:
```
from datetime import datetime

now = datetime.now()
print("Name: Umang Kumar\nRoll No.: 201210051")
print("Time: ", end = "")
cur_time = now.strftime("%H:%M:%S")
print(cur_time)
```

```
Name: Umang Kumar
Roll No.: 201210051
Time: 16:55:53
```

Exercise 1: Create following list:

```
data = [2,34,23,16,56,45,34,26,78,56,1,16]
```

Write a function to count number of elements in a list . Now delete the duplicate values and print the list in ascending order.

In [17]:
```
data = [2,34,23,16,56,45,34,26,78,56,1,16]

print("Size of the array: ", len(data))

# method 1
newData = list(set(data))
print("Output of Method 1:")
print("Unique Element: ", newData)
newData.sort()
print("Sorted Array: ", newData)
print()

# method 2
dic = {}
for i in data:
    if i not in dic.keys():
        dic[i] = 1

newData = list(dic.keys()) # removed duplicates
print("Output of Method 2:")
print("Unique Element: ", newData)
newData.sort() # sorted in ascending order

print("Sorted Array: ", newData)
```

```
Size of the array:  12
Output of Method 1:
Unique Element:  [1, 2, 34, 45, 78, 16, 23, 56, 26]
Sorted Array:  [1, 2, 16, 23, 26, 34, 45, 56, 78]

Output of Method 2:
Unique Element:  [2, 34, 23, 16, 56, 45, 26, 78, 1]
Sorted Array:  [1, 2, 16, 23, 26, 34, 45, 56, 78]
```

Exercise 2:

Take five numbers as input from the user and save into a list. Find the maximum of the list and sort the data in descending order.

In [18]:
```
# input 5 numbers from user
arr = list(map(int, input().split()))

print("Given Array: ", arr)
print("Maximum: ", max(arr))
arr.sort(reverse=True)
print("Sorted array: ",arr)
print()
```

```
Given Array:  [12, 34, 54, 67, 23]
Maximum:  67
Sorted array:  [67, 54, 34, 23, 12]
```

Exercise 4:

(i) Generate two arrays A1 and A2 of size 5 X 4 and 3 X 4 respectively using np.random()
(ii) Join them and make an array A3 of 8 X 4. Now append random numbers ranging between from 0 to 5 to make the fourth array A4 of size 10 X 10.
(iii) Print all the arrays and their transpose (Transpose of 'A' can be obtained by 'A.T')

In [22]:
```python
import numpy as np

# generate two matrix using np.random
A1 = np.random.rand(5, 4)
A2 = np.random.rand(3, 4)
print("A1:\n",A1); print()
print("A2:\n",A2); print()

# Join them and make an array A3 of 8 X 4.
A3 = np.concatenate((A1, A2))
print("A3:\n",A3); print()

# Now append random numbers ranging between from 0 to 5 to make the fourt
A4 = A3.flatten()
arr = np.random.randint(5, size=(1, 100 - len(A4)))
# print(A4)
A4 = np.concatenate((A4, arr[0]))
A4 = np.reshape(A4, (10,10))
print("A4:\n",A4); print()

print("Transpose: ")
print("A1:\n",A1.transpose()); print()
print("A2:\n",A2.transpose()); print()
print("A3:\n",A3.transpose()); print()
print("A4:\n",A4.transpose()); print()
```

```
A1:
 [[0.45237929 0.96176518 0.43574598 0.40970668]
 [0.4446447  0.92895954 0.32755112 0.31478701]
 [0.85308555 0.30337152 0.7841342  0.77888547]
 [0.96167099 0.1032272  0.35552261 0.59353056]
 [0.60237251 0.42463533 0.96279652 0.75794169]]

A2:
 [[0.6667326  0.0367286  0.84070888 0.47575324]
 [0.31151311 0.80639697 0.72803692 0.77902759]
 [0.44378869 0.29689611 0.89241741 0.46533356]]

A3:
 [[0.45237929 0.96176518 0.43574598 0.40970668]
```

```
 [0.4446447  0.92895954 0.32755112 0.31478701]
 [0.85308555 0.30337152 0.7841342  0.77888547]
 [0.96167099 0.1032272  0.35552261 0.59353056]
 [0.60237251 0.42463533 0.96279652 0.75794169]
 [0.6667326  0.0367286  0.84070888 0.47575324]
 [0.31151311 0.80639697 0.72803692 0.77902759]
 [0.44378869 0.29689611 0.89241741 0.46533356]]

A4:
 [[0.45237929 0.96176518 0.43574598 0.40970668 0.4446447  0.92895954
  0.32755112 0.31478701 0.85308555 0.30337152]
 [0.7841342  0.77888547 0.96167099 0.1032272  0.35552261 0.59353056
  0.60237251 0.42463533 0.96279652 0.75794169]
 [0.6667326  0.0367286  0.84070888 0.47575324 0.31151311 0.80639697
  0.72803692 0.77902759 0.44378869 0.29689611]
 [0.89241741 0.46533356 0.         4.         1.         4.
  0.         2.         2.         3.        ]
 [2.         2.         2.         3.         2.         0.
  2.         2.         4.         4.        ]
 [4.         3.         4.         4.         0.         3.
  4.         1.         0.         0.        ]
 [4.         4.         1.         4.         1.         0.
  1.         3.         0.         3.        ]
 [4.         2.         0.         0.         2.         2.
  3.         4.         3.         0.        ]
 [0.         3.         3.         1.         1.         3.
  3.         0.         1.         3.        ]
 [2.         4.         2.         1.         3.         1.
  2.         2.         0.         0.        ]]

Transpose:
A1:
 [[0.45237929 0.4446447  0.85308555 0.96167099 0.60237251]
 [0.96176518 0.92895954 0.30337152 0.1032272  0.42463533]
 [0.43574598 0.32755112 0.7841342  0.35552261 0.96279652]
 [0.40970668 0.31478701 0.77888547 0.59353056 0.75794169]]

A2:
 [[0.6667326  0.31151311 0.44378869]
 [0.0367286  0.80639697 0.29689611]
 [0.84070888 0.72803692 0.89241741]
 [0.47575324 0.77902759 0.46533356]]

A3:
 [[0.45237929 0.4446447  0.85308555 0.96167099 0.60237251 0.6667326
  0.31151311 0.44378869]
 [0.96176518 0.92895954 0.30337152 0.1032272  0.42463533 0.0367286
  0.80639697 0.29689611]
 [0.43574598 0.32755112 0.7841342  0.35552261 0.96279652 0.84070888
  0.72803692 0.89241741]
 [0.40970668 0.31478701 0.77888547 0.59353056 0.75794169 0.47575324
  0.77902759 0.46533356]]

A4:
 [[0.45237929 0.7841342  0.6667326  0.89241741 2.         4.
  4.         4.         0.         2.        ]
 [0.96176518 0.77888547 0.0367286  0.46533356 2.         3.
```

```
 4.         2.         3.         4.        ]
[0.43574598 0.96167099 0.84070888 0.         2.         4.
 1.         0.         3.         2.        ]
[0.40970668 0.1032272  0.47575324 4.         3.         4.
 4.         0.         1.         1.        ]
[0.4446447  0.35552261 0.31151311 1.         2.         0.
 1.         2.         1.         3.        ]
[0.92895954 0.59353056 0.80639697 4.         0.         3.
 0.         2.         3.         1.        ]
[0.32755112 0.60237251 0.72803692 0.         2.         4.
 1.         3.         3.         2.        ]
[0.31478701 0.42463533 0.77902759 2.         2.         1.
 3.         4.         0.         2.        ]
[0.85308555 0.96279652 0.44378869 2.         4.         0.
 0.         3.         1.         0.        ]
[0.30337152 0.75794169 0.29689611 3.         4.         0.
 3.         0.         3.         0.        ]]
```

Exercise 5: Create two dictionaries.

The first dictionary ‘name’ will contain first name(key) of a person and its hash value(value). The Second will contain hash value(key) and mobile no(value).

i) Add 5 entries.

ii) Delete two entries by taking the input from user as the first name.

iii) Add two entries by taking the input as the first name and mobile no.

Hint: You can use remainder (%) to obtain hash value.

In [3]:
```
# code
dict1={}
dict2={}

dict1["mohit"]=hash("mohit")
dict1["aditya"]=hash("aditya")
dict1["umang"]=hash("umang")
dict1["sanskar"]=hash("sanskar")
dict1["sachin"]=hash("sachin")

print( dict1 )

for x in range(1,3):
    a=input("Enter first name: ")
    del dict1[a]

print( dict1 )

for x in range(1,3):
    a=input("Enter first name: ")
    b=input("Enter Number: ")
    dict2[a]=b

print( dict2 )
```

```
{'mohit': 3269859731064344992, 'aditya': -889627898511335941, 'umang': 4
686701392842812356, 'sanskar': 3846248293541280279, 'sachin': 7269661652
97133106}
{'aditya': -889627898511335941, 'sanskar': 3846248293541280279, 'sachin'
: 726966165297133106}
{'Rahul': '9634916827', 'Sachin': '1234123456'}
```

PART 2: Introduction to NumPy

Exercise 6:

Write a NumPy program to create an element-wise comparison (greater, greater_equal, less and less_equal) of two given arrays

In [7]:
```
# code
data1 = [1, 3, 54, 6, 78, 9]
data2 = [10, 7, 56, 6, 58, 6]

Greater = []
Greater_eq = []
less = []
less_eq = []

for x in range(0, len(data1)-1):
    Greater.append(data1[x] > data2[x])
    Greater_eq.append(data1[x] >= data2[x])
    less.append(data1[x] < data2[x])
    less_eq.append(data1[x] <= data2[x])


print("Greater: \t", Greater)
print("Greater Equal: \t", Greater_eq)
print("Less: \t\t", less)
print("Less Equal: \t", less_eq)
```

```
Greater:         [False, False, False, False, True]
Greater Equal:   [False, False, False, True, True]
Less:            [True, True, True, False, False]
Less Equal:      [True, True, True, True, False]
```

Exercise 8:

Write a NumPy program to get the powers (x ^ 3) of an array values element-wise

Expected Output: Original array `[1 2 3 4 5 ]`

Output array: `[ 1 8 27 64 125 ]`

In [10]:
```
import numpy as np
data =[ 1,2,3,4,5]
np.power(data,3)
```

Out[10]:
```
array([  1,   8,  27,  64, 125])
```

Exercise 9:

Write a NumPy program to get the floor, ceiling and truncated values of the elements of a numpy array.

Sample Output:

Original array:

```
[-1.3, -1.15, -0.1, 0.12, 1.7, 0.9, 1.1]
```

In [12]:
```
# code
import numpy as np
data1 = [-1.3, -1.15, -0.1, 0.12, 1.7, 0.9, 1.1]

floor = np.floor(data1)
ceill = np.ceil(data1)
truncated = np.trunc(data1)

print("Floor: ", floor)
print("Ceil: ", ceill)
print("Truncated: ", truncated)
```

```
Floor:  [-2. -2. -1.  0.  1.  0.  1.]
Ceil:  [-1. -1. -0.  1.  2.  1.  2.]
Truncated:  [-1. -1. -0.  0.  1.  0.  1.]
```

Exercise 10:

Write a program in python to display prime numbers from x to y (here x and y are user given values).

In [18]:
```
import sympy as sy

x = int(input("Enter first number :"))
y = int(input("Enter second number :"))
print(f"\nPrime numbers in the range {x} - {y}")

for i in range(x, y+1):
    if(sy.isprime(i)):
        print(i,end=" ")
```

```
Prime numbers in the range 10 - 100
11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97
```

Observations

1. Python is one of the simplest Programming language.
2. Python has vast libraries designed for specific tasks.
3. Python is really convenient for Machine Learning and Artificial Intelligent.
4. Python is easy to understand and to code.
5. Python is little bit slower than C, C++, etc.
6. We learn about basics syntax of Python for writing code and taking input.