Basic Numpy Assignment

Yash Shah - J060

```
import numpy as np
from time import process_time
```

Proving matrix multiplication properties

```
a=np.array([[2,4],[1,2]])
b=np.array([[-5,11],[4,2]])
c=np.array([[3,-2],[4,5]])
```

Commutive property

```
## AB!=BA

print(f'A.B: \n\n{np.matmul(a,b)} \n')
print(f'B.A: \n\n{np.matmul(b,a)}')
```

```
A.B: 

[[ 6 30]
 [ 3 15]] 

B.A: 

[[ 1  2]
 [10 20]]
```

Associative property of multiplication

```
## A(BC)= (AB)C

print(f'A(BC): \n\n {np.matmul(a,np.matmul(b,c))} \n')
print(f'(AB)C): \n\n {np.matmul(np.matmul(a,b),c)}')
```

```
A(BC): 

 [[138 138]
 [ 69  69]] 

(AB)C): 

 [[138 138]
 [ 69  69]]
```

Distributive properties

```
#A(B+C) = AB + AC
print(f'A(B+C): \n {np.matmul(a, b+c)}')
print(f'AB + AC: \n {np.matmul(a,b) + np.matmul(a,c)}')
```

```
A(B+C): 
 [[28 46]
 [14 23]]
AB + AC:
 [[28 46]
 [14 23]]
```

Multiplicative identity property

```
## IA = A and AI=A

print(f'A \n{a}')
print(f'I.A \n {np.matmul(np.identity(2),a)}')
```

```
A
[[2 4]
 [1 2]]
I.A
 [[2. 4.]
 [1. 2.]]
```

Multiplicative property of zero

```
## 0A= 0 and A0=0

print(f'A0= 0:  \n {np.matmul(a,np.zeros_like(a))}')
```

```
A0= 0:
 [[0 0]
 [0 0]]
```

Dimension property

```
#mxn and nxp = mxp

print(f'mxn and nxp: \n {np.matmul(a,b)}')
```

```
mxn and nxp:
 [[ 6 30]
 [ 3 15]]
```

Calculating inverse of a matrix

Matrix

```
a= np.array([[4,3,2],[1,3,23],[10,12,1]])
a
```

```
array([[ 4,  3,  2],
       [ 1,  3, 23],
       [10, 12,  1]])
```

Inverse Calculation

```
np.matmul(np.linalg.inv(a), a)
```

```
array([[ 1.00000000e+00,  4.44089210e-16,  4.16333634e-16],
       [ 3.33066907e-16,  1.00000000e+00, -1.66533454e-16],
       [-2.08166817e-17,  1.38777878e-17,  1.00000000e+00]])
```

Prove Numpy is faster than traditional loop

```
l1= [i for i in range(10000)]
l2= [i for i in range(10000)]

start=process_time()

dot= 0

for i,j in zip(l1,l2):
    dot += i*j
end=process_time()

print(end-start)
```

```
0.003667100999999562
```

```
arr1= np.array([i for i in range(10000)])
arr2= np.array([i for i in range(10000)])

start=process_time()

print(np.dot(arr1,arr2))

end= process_time()
print(end-start)
```

```
333283335000
0.0007434399999999286
```

✓ 0s completed at 9:40 AM