B.TECH. (2020-24)
Artificial Intelligence

Lab File
on
FUNDAMENTALS OF MACHINE LEARNING
[CSE313]

Submitted To
Dr Monika Arora

Submitted By
HITESH
A023119820027
5AI 1

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
AMITY SCHOOL OF ENGINEERING AND TECHNOLOGY
AMITY UNIVERSITY UTTAR PRADESH
NOIDA (U.P)

EXPERIMENT- 1

AIM

Perform Basic operations on matrices (like addition, subtraction, multiplication) and display specific rows or columns of a matrix.

SOFTWARE USED

Jupyter Platform - Python Programming Language

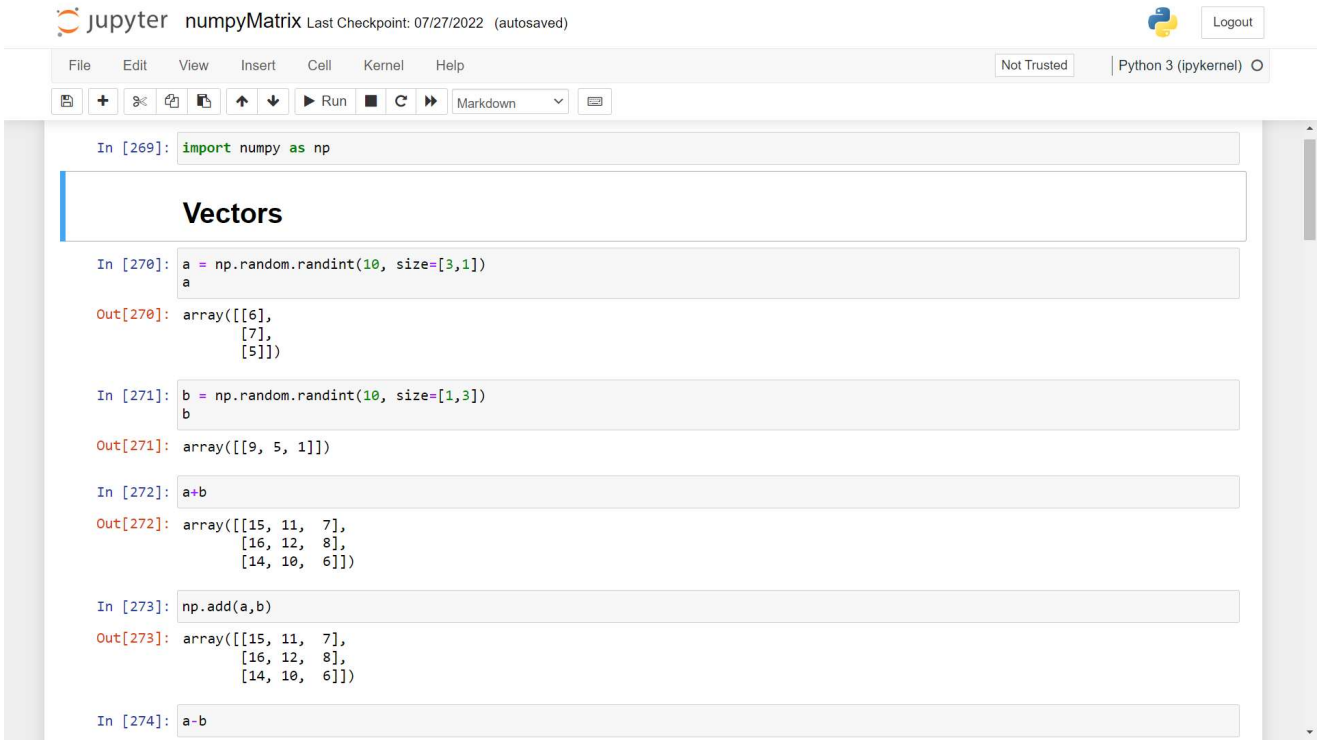
PROGRAM CODE

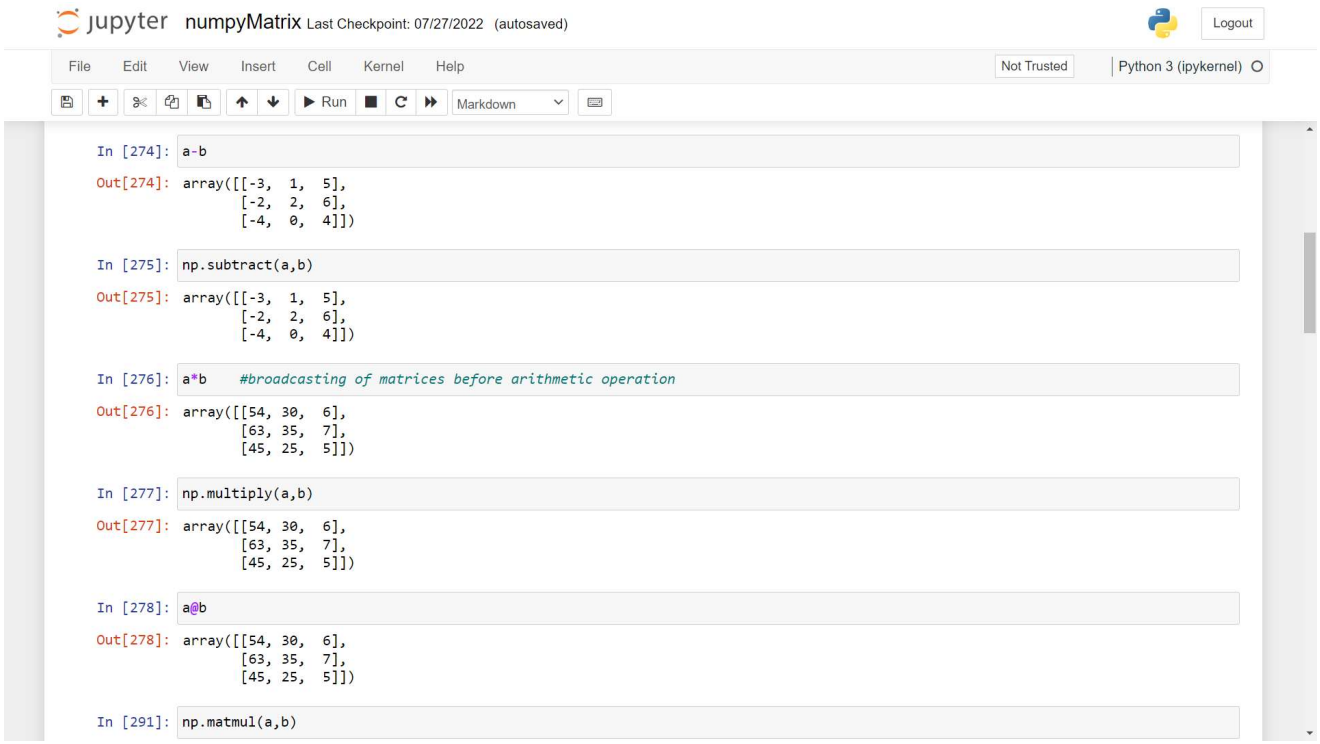
```
import numpy as np

a = np.random.randint(10, size=[3,1])
b = np.random.randint(10, size=[1,3])

print(a+b)
print(np.add(a,b))

print(a-b)
print(np.subtract(a,b))

print(a*b)
print(np.multiply(a,b))

print(a@b)
print(np.matmul(a,b))

#Indexing
print(a[1])
print(a[0:2])
print(a[:])
```

```
import numpy as np

p = np.random.randint(10, size = [3,3])
q = np.random.randint(10, size = [3,3])

print(p+q)
print(np.add(p,q))

print(p-q)
print(np.subtract(p,q))

print(p*q)
print(np.multiply(p,q))

print(p@q)
print(np.matmul(p,q))

#Indexing
print(p[2])
```

```
print(p[0:2])
print(p[1:3,1:3])
print(np.concatenate([p,q],))
print(np.concatenate([p,q],axis=1))
```

OUTPUT





jupyter numpyMatrix Last Checkpoint: 07/27/2022 (autosaved)

Logout

File Edit View Insert Cell Kernel Help

Not Trusted | Python 3 (ipykernel)

Run Markdown

```
In [291]: np.matmul(a,b)
Out[291]: array([[54, 30,  6],
                 [63, 35,  7],
                 [45, 25,  5]])

In [292]: a[1]
Out[292]: array([7])

In [295]: a[0:2]
Out[295]: array([[6],
                 [7]])

In [296]: a[:]
Out[296]: array([[6],
                 [7],
                 [5]])
```

Matrices

```
In [280]: p = np.random.randint(10, size = [3,3])
          p
Out[280]: array([[6, 3, 1],
                 [6, 9, 1],
                 [5, 1, 6]])
```

jupyter numpyMatrix Last Checkpoint: 07/27/2022 (autosaved)

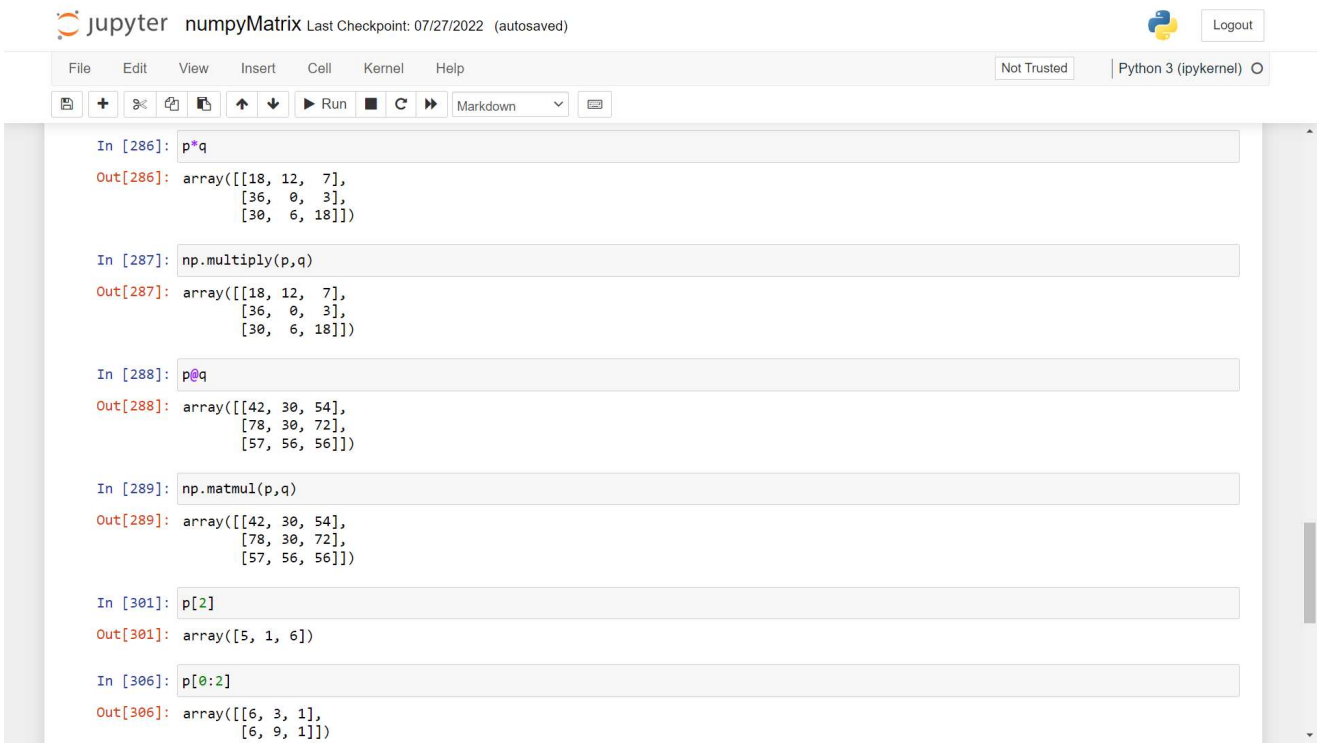
Logout

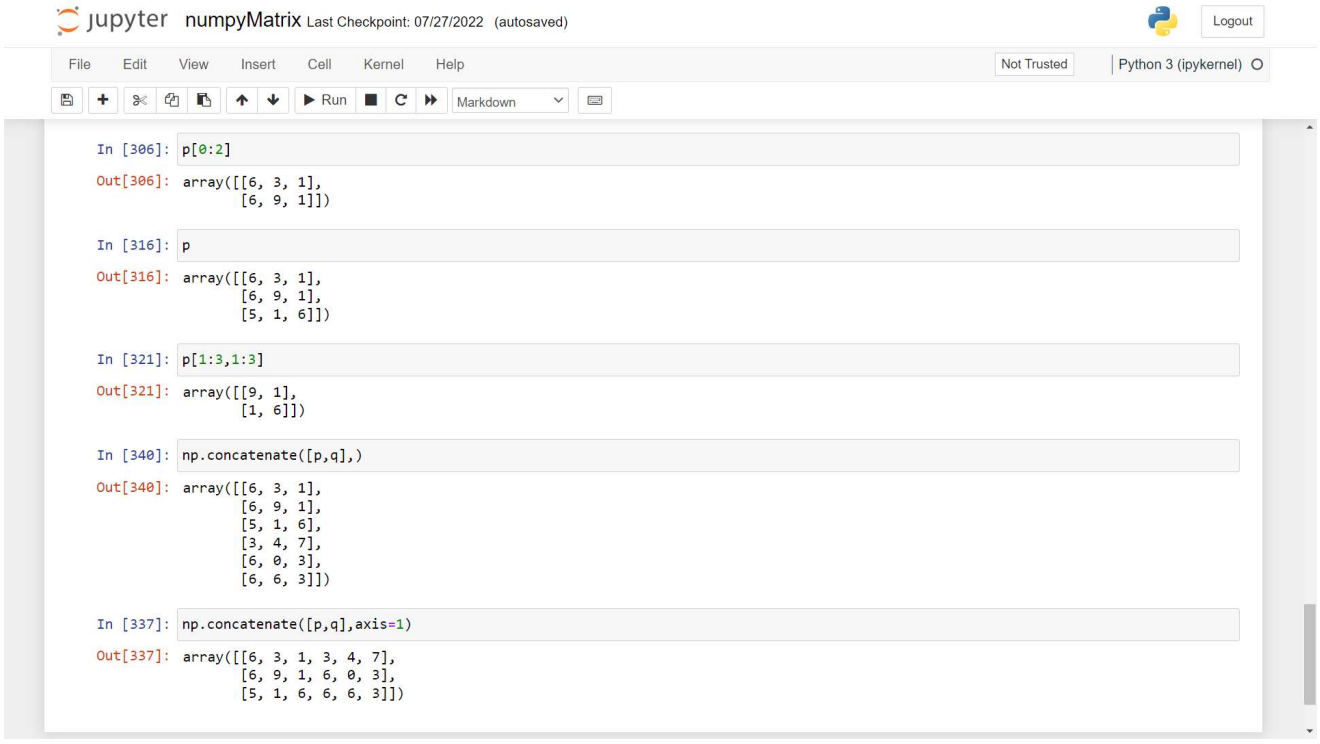
File Edit View Insert Cell Kernel Help

Not Trusted | Python 3 (ipykernel)

Run Markdown

```
In [281]: q = np.random.randint(10, size = [3,3])
          q
Out[281]: array([[3, 4, 7],
                 [6, 0, 3],
                 [6, 6, 3]])

In [282]: p+q
Out[282]: array([[ 9,  7,  8],
                 [12,  9,  4],
                 [11,  7,  9]])

In [283]: np.add(p,q)
Out[283]: array([[ 9,  7,  8],
                 [12,  9,  4],
                 [11,  7,  9]])

In [284]: p-q
Out[284]: array([[ 3, -1, -6],
                 [ 0,  9, -2],
                 [-1, -5,  3]])

In [285]: np.subtract(p,q)
Out[285]: array([[ 3, -1, -6],
                 [ 0,  9, -2],
                 [-1, -5,  3]])

In [286]: p*q
```





DISCUSSION and CONCLUSION

The basic matrix operations (like addition, subtraction, multiplication) and matrix slicing have been studied and performed in the Jupyter platform.

CRITERIA	TOTAL MARKS	MARKS OBTAINED	COMMENTS
Concept (A)	2		
Implementation (B)	2		
Performance (C)	2		
Total	6		

EXPERIMENT- 2

AIM

Perform other matrix operations like converting matrix data to absolute values, taking the negative of matrix values, adding/removing rows/columns, finding maximum or minimum values of a matrix in a row /column, and finding the sum of all the elements in a matrix.

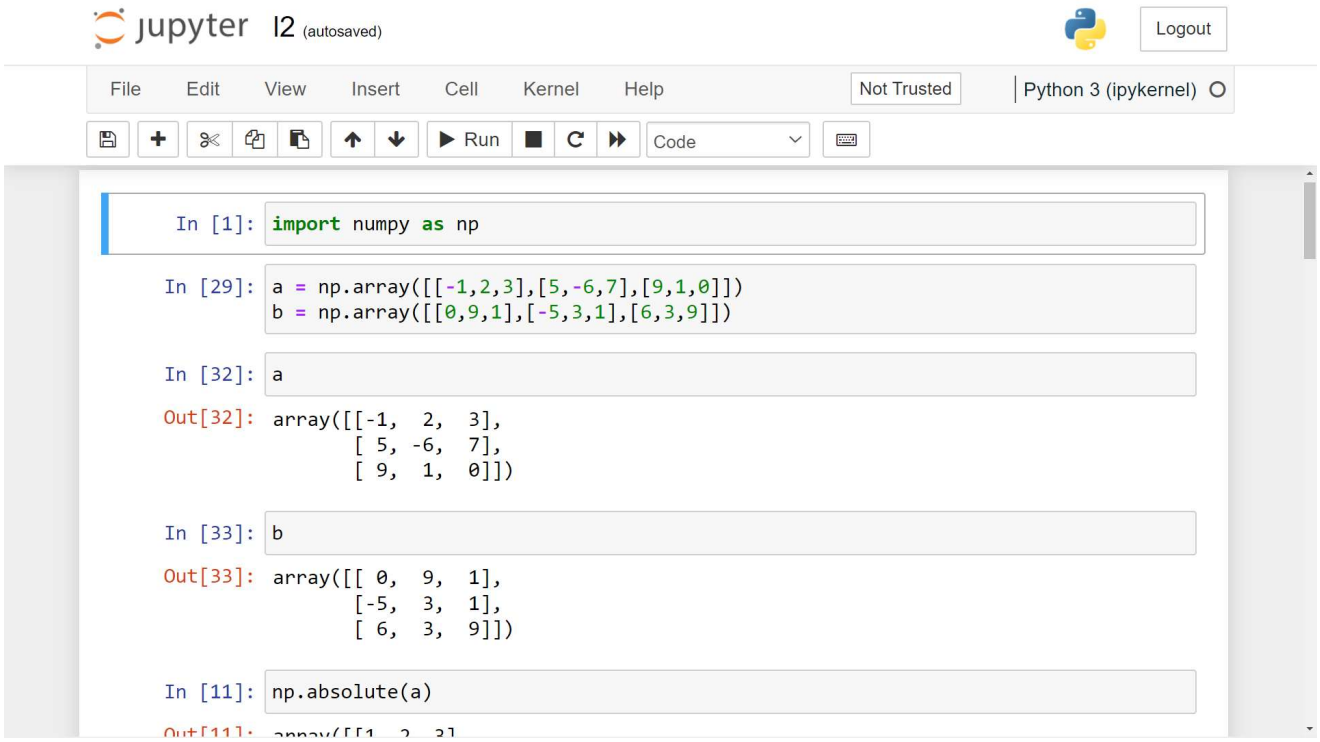
SOFTWARE USED

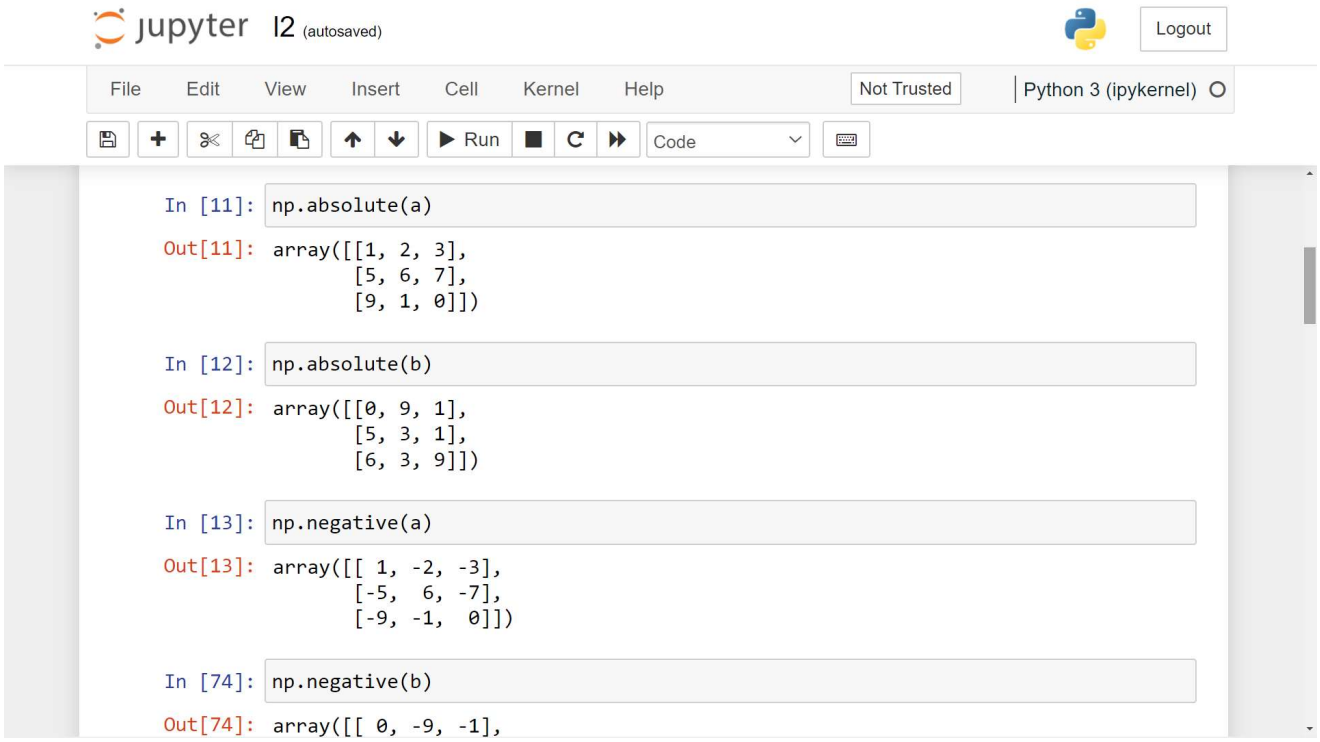
Jupyter Platform - Python Programming Language

PROGRAM CODE

```
import numpy as np

a = np.array([[-1,2,3],[5,-6,7],[9,1,0]])
b = np.array([[0,9,1],[-5,3,1],[6,3,9]])

print(a)
print(b)

print(np.absolute(a))
print(np.absolute(b))

print(np.negative(a))
print(np.negative(b))

print(np.append(a,[[1,1,1]], axis = 0))
print(np.append(a,[[1],[1],[1]],axis = 1))

print(np.delete(a,1,0))
print(np.delete(a,1,1))

c = np.concatenate([a,b],axis = 0)
d = np.concatenate([a,b],axis = 1)

print(c)
print(d)

print(d[2,:])
print(c[:,1])
print(d[0:2,0:2])

print(np.max(a))
print(np.max(c,axis=0))
print(np.min(c,axis=0))
print(np.min(c,axis=1))

print(np.sum(a))
```

OUTPUT


jupyter l2 (autosaved)
Logout
File Edit View Insert Cell Kernel Help
Not Trusted
Python 3 (ipykernel)
Run
Code
In [1]: import numpy as np
In [29]: a = np.array([[-1,2,3],[5,-6,7],[9,1,0]])
b = np.array([[0,9,1],[-5,3,1],[6,3,9]])
In [32]: a
Out[32]: array([[-1, 2, 3],
[5, -6, 7],
[9, 1, 0]])
In [33]: b
Out[33]: array([[0, 9, 1],
[-5, 3, 1],
[6, 3, 9]])
In [11]: np.absolute(a)


jupyter l2 (autosaved)
Logout
File Edit View Insert Cell Kernel Help
Not Trusted
Python 3 (ipykernel)
Run
Code
In [11]: np.absolute(a)
Out[11]: array([[1, 2, 3],
[5, 6, 7],
[9, 1, 0]])
In [12]: np.absolute(b)
Out[12]: array([[0, 9, 1],
[5, 3, 1],
[6, 3, 9]])
In [13]: np.negative(a)
Out[13]: array([[1, -2, -3],
[-5, 6, -7],
[-9, -1, 0]])
In [74]: np.negative(b)
Out[74]: array([[0, -9, -1],

jupyter l2 (autosaved) Logout

File Edit View Insert Cell Kernel Help Not Trusted Python 3 (ipykernel)

Run Code

In [74]:
```
np.negative(b)
```

Out[74]:
```
array([[ 0, -9, -1],
       [ 5, -3, -1],
       [-6, -3, -9]])
```

In [69]:
```
np.append(a,[[1,1,1]], axis = 0)
```

Out[69]:
```
array([[-1,  2,  3],
       [ 5, -6,  7],
       [ 9,  1,  0],
       [ 1,  1,  1]])
```

In [70]:
```
np.append(a,[[1],[1],[1]],axis = 1)
```

Out[70]:
```
array([[-1,  2,  3,  1],
       [ 5, -6,  7,  1],
       [ 9,  1,  0,  1]])
```

In [72]:
```
np.delete(a,1,0)
```

jupyter l2 (autosaved) Logout

File Edit View Insert Cell Kernel Help Not Trusted Python 3 (ipykernel)

Run Code

In [72]:
```
np.delete(a,1,0)
```

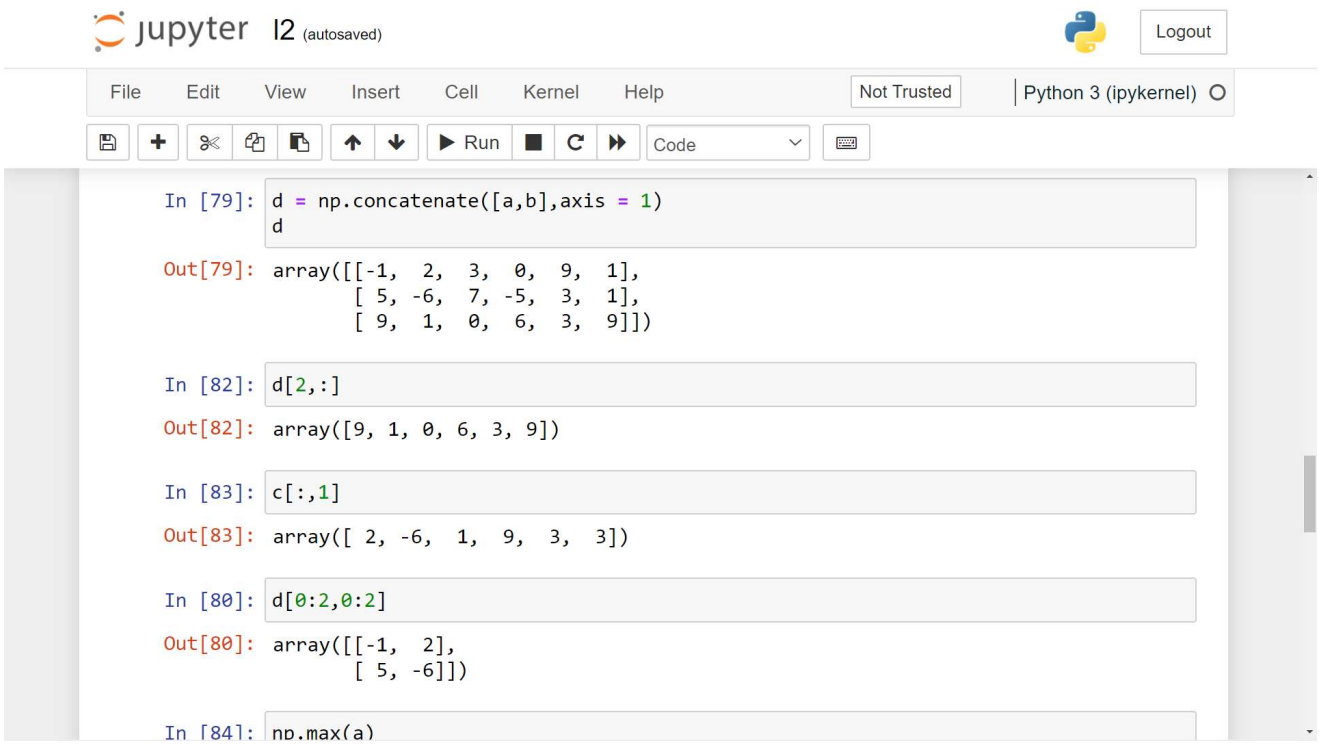
Out[72]:
```
array([[-1,  2,  3],
       [ 9,  1,  0]])
```

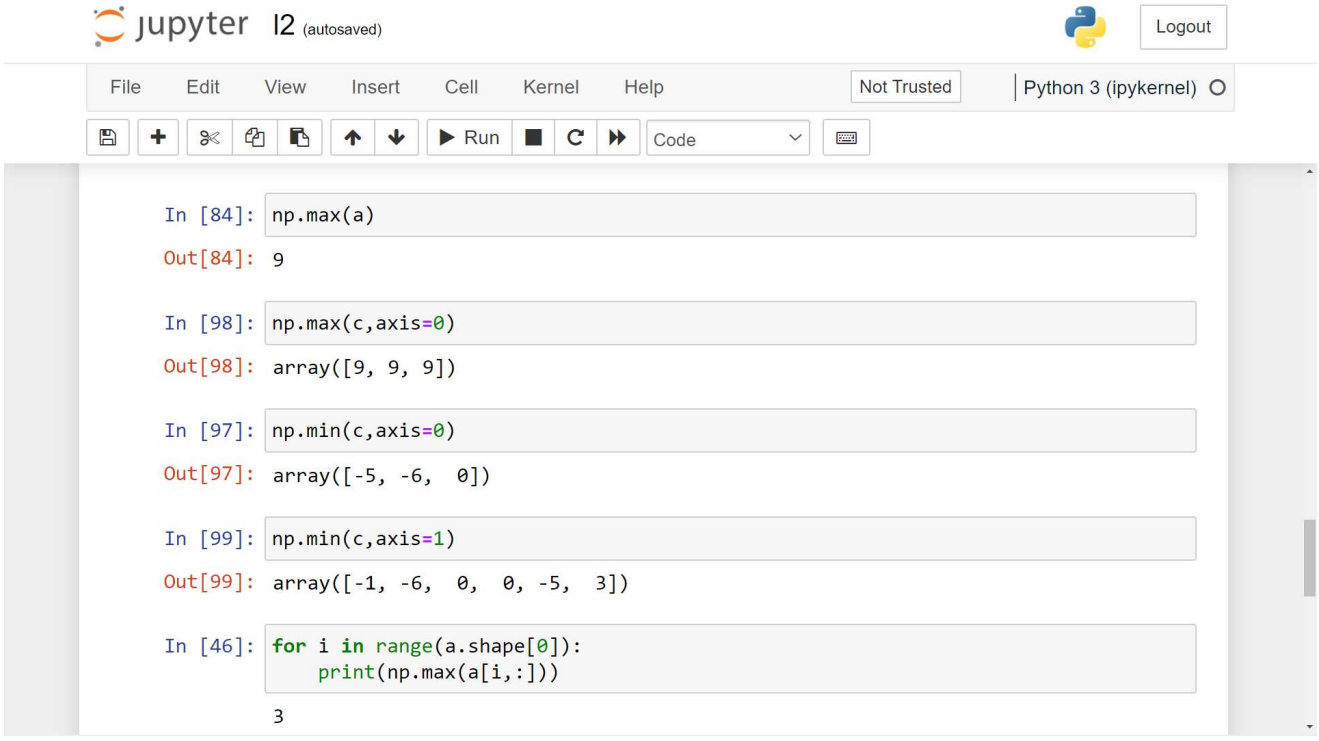
In [73]:
```
np.delete(a,1,1)
```

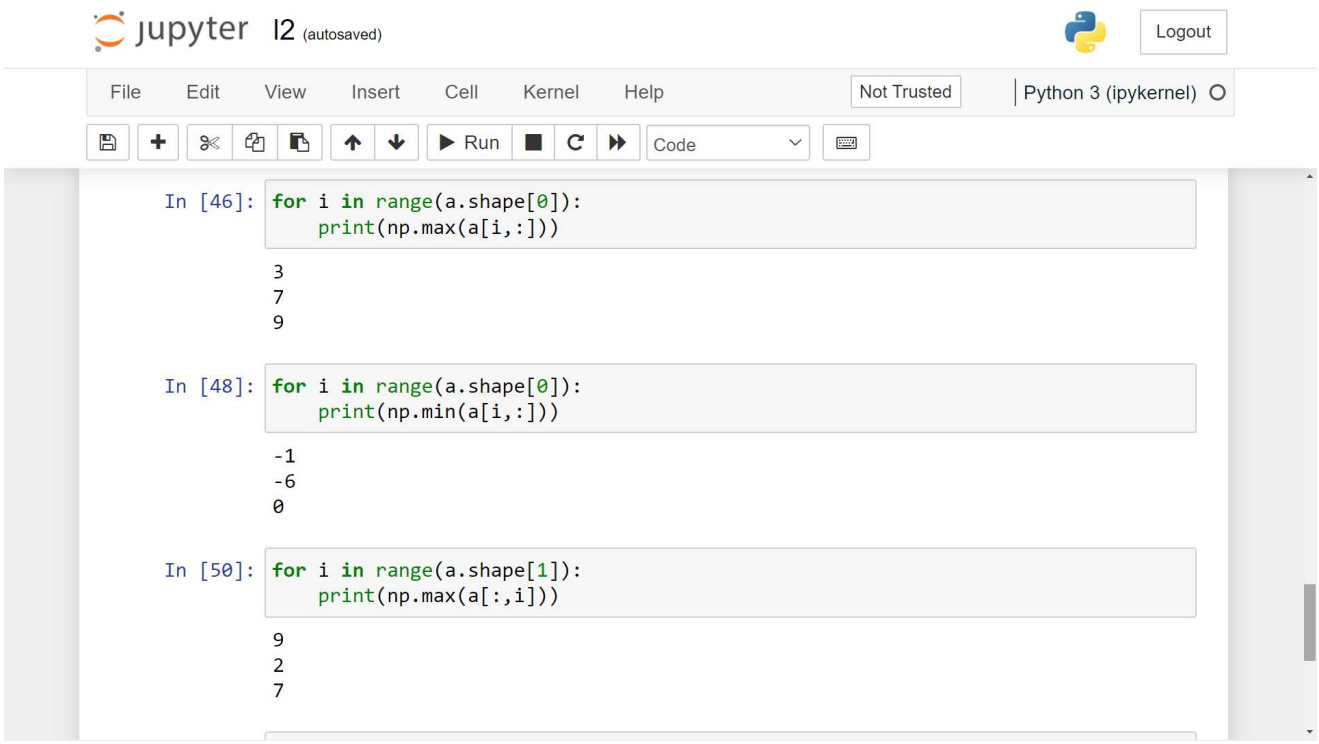
Out[73]:
```
array([[-1,  3],
       [ 5,  7],
       [ 9,  0]])
```

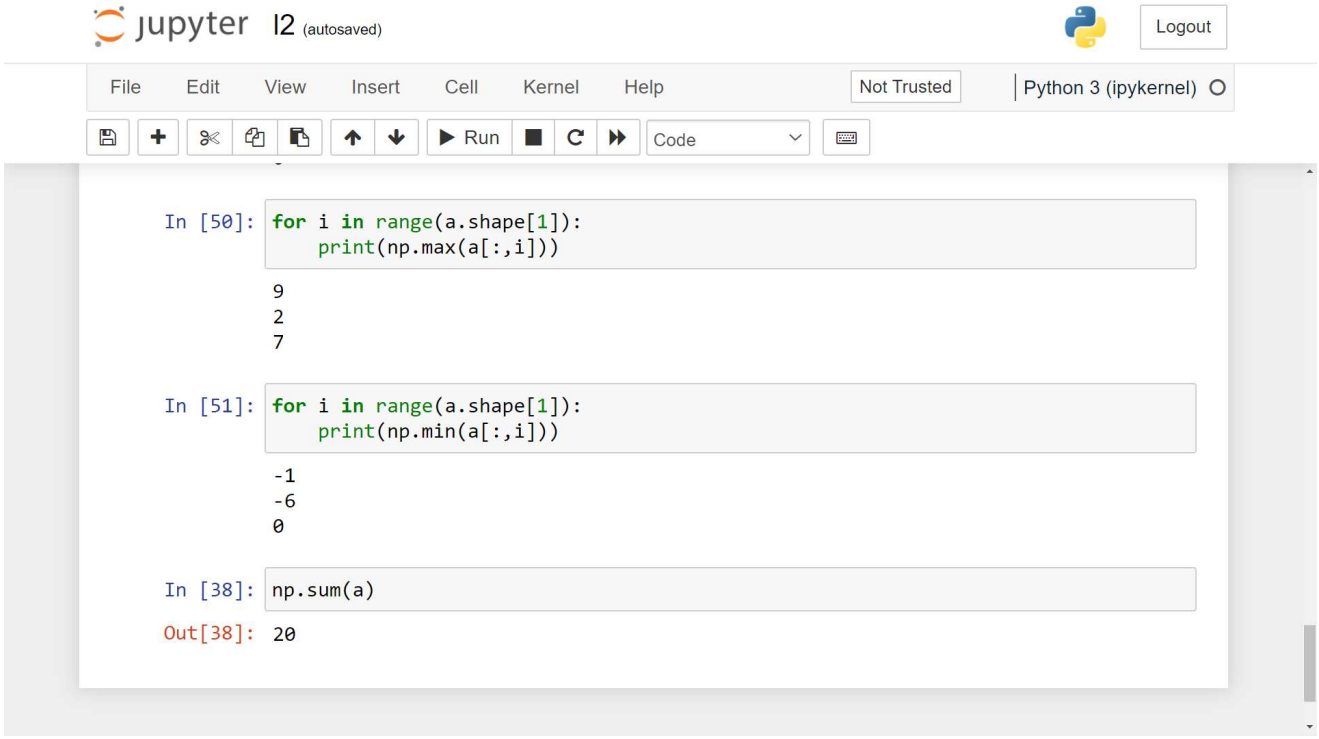
In [78]:
```
c = np.concatenate([a,b],axis = 0)
c
```

Out[78]:
```
array([[-1,  2,  3],
       [ 5, -6,  7],
       [ 9,  1,  0],
       [ 0,  9,  1],
       [-5,  3,  1],
       [ 6,  3,  9]])
```


jupyter l2 (autosaved)
Logout
File Edit View Insert Cell Kernel Help
Not Trusted
Python 3 (ipykernel)
Run
Code
In [79]:
d = np.concatenate([a,b],axis = 1)
d
Out[79]:
array([[-1, 2, 3, 0, 9, 1],
[5, -6, 7, -5, 3, 1],
[9, 1, 0, 6, 3, 9]])
In [82]:
d[2,:]
Out[82]:
array([9, 1, 0, 6, 3, 9])
In [83]:
c[:,1]
Out[83]:
array([2, -6, 1, 9, 3, 3])
In [80]:
d[0:2,0:2]
Out[80]:
array([[-1, 2],
[5, -6]])
In [84]:
np.max(a)


jupyter l2 (autosaved)
Logout
File Edit View Insert Cell Kernel Help
Not Trusted
Python 3 (ipykernel)
Run
Code
In [84]:
np.max(a)
Out[84]:
9
In [98]:
np.max(c,axis=0)
Out[98]:
array([9, 9, 9])
In [97]:
np.min(c,axis=0)
Out[97]:
array([-5, -6, 0])
In [99]:
np.min(c,axis=1)
Out[99]:
array([-1, -6, 0, 0, -5, 3])
In [46]:
for i in range(a.shape[0]):
print(np.max(a[i,:]))
3





DISCUSSION and CONCLUSION

The matrix operations (like absolute, negative, adding or removing rows/columns) have been studied and performed in the Jupyter platform. Also, the maximum and minimum values of a matrix and sum of all elements have been evaluated and displayed.

CRITERIA	TOTAL MARKS	MARKS OBTAINED	COMMENTS
Concept (A)	2		
Implementation (B)	2		
Performance (C)	2		
Total	6		