

Name: Ritesh Rodge
PRN No: 202201070121
Roll No: 580

Source Code:

```
import numpy as np
a = np.loadtxt("Numpy/testmarks1.csv", delimiter=",", dtype=float,
skiprows=1)
print(a)
b = np.loadtxt("Numpy/testmarks2.csv", delimiter=",", dtype=float,
skiprows=1)
print(b)
# matrix operations

print("Transpose of Matrix a is: \n", a.T)
print("\nTranspose of Matrix b is: \n", b.T)
print(a*b)
print("\nTrace of a:\n", a.trace())
print("\nTrace of b:\n", b.trace())
print("\nFlatten a: ", a.flatten())
print("\nFlatten b: ", b.flatten())
# Horizontal stacking
print("Horizontal Stacking")
print(np.hstack((a, b)), end="\n\n")
# Vertical stacking
print("Vertical Stacking")
print(np.vstack((a, b)), end="\n\n")
# Custom sequence generation print("Generating
Custom Sequences:\n")
print(np.arange(0, 10))
print(np.arange(0, 105, 5)) # Arithmetic
and Mathematical Operations
print("Adding a and b:\n", np.add(a, b))
print("Subtracting a and b:\n", np.subtract(a, b))
print("Multiplying a nd b :\n", np.multiply(a, b))
print("Dividing a nd b :\n", np.divide(a, b))
print("Mod of a and b:\n", np.mod(a, b))
print("Remainder of a and b:\n", np.remainder(a, b))
# Statistical Operations

print("Mean of a: ", np.mean(a)) print("Mean
of b: ", np.mean(b))

print("Variance of a: ", np.var(a))
print("Variance of b: ", np.var(b))

print("Standard Deviation of a: ", np.std(a))
print("Standard Deviation of b: ", np.std(b))
```

```

print("Sum of all elements in a: ", np.sum(a))
print("Sum of all elements in b: ", np.sum(b))

# stacking and sorting
print("Broadcasting:\n",
a+5) print("Data Stacking:\n", np.stack((a, b),
axis=2))

print("Sorting a: \n", np.sort(a))
print("Sorting b: \n", np.sort(b))

print("Counting elements in a: ", np.count_nonzero(a))
print("Counting elements in b: ", np.count_nonzero(b))

print("Counting using elements less than 50 in a: ",
np.count_nonzero(a > 4))
print("Counting using elements less than 10 in b: ",
np.count_nonzero(b > 50))

# view and copy
print("\n\nView Method\n")
v = a.view() v[:]=0

print("a=\n", a) print("v=\n",
v)

print("Array created using view method is just shallow copy of
original array\nSO changes made in original array reflects in view
copy or vice versa") print("\n\ncopy method: \n") c = b.copy()
c[:] = 0

print("b=\n", b) print("c=\n",
c)
print("Both b and c has showed different o/p cz they are different
arrays!")
#Bitwise operations

a=15 b=20

print("Binary of a: ",bin(a)) print("Binary
of b:",bin(b))
print("Bitwise a and b: ",np.bitwise_and(a,b))
print("Bitwise a or b: ",np.bitwise_or(a,b))
print("Bitwise a xor b: ",np.bitwise_xor(a,b))

```

Output:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
• (base) kartikeysapkal@Kartikeys-MacBook-Air Python % python -u "/Users/kartikeysapkal/Documents/Python/Numpy/NumpyAssign.py"
a=
[[801.  43.05  27.79  28.7  27.79]
 [802.  43.47  28.52  28.98  27.89]
 [803.  42.24  28.16  28.16  25.63]
 [804.  39.24  26.16  26.16  26.16]
 [805.  40.9  26.03  27.27  25.65]
 [806.  39.47  26.31  26.31  25.21]
 [807.  41.68  25.63  27.79  25.46]
 [808.  42.19  27.61  28.13  26.21]
 [809.  44.75  28.35  29.83  28.21]
 [810.  46.95  28.88  31.3  28.53]]
b=
[[801.  28.48  34.18  30.56  22.23]
 [802.  28.1  33.72  30.68  22.82]
 [803.  26.16  31.39  28.2  22.53]
 [804.  26.16  31.39  28.78  20.93]
 [805.  26.1  31.32  28.22  20.82]
 [806.  25.45  30.54  27.73  21.05]
 [807.  26.16  31.39  28.01  20.51]
 [808.  27.44  32.93  28.83  22.08]
 [809.  28.63  34.35  31.03  22.68]
 [810.  30.35  36.42  31.38  23.1 ]]
Transpose of Matrix a is:
[[801.  802.  803.  804.  805.  806.  807.  808.  809.  810. ]
 [ 43.05  43.47  42.24  39.24  40.9  39.47  41.68  42.19  44.75  46.95]
 [ 27.79  28.52  28.16  26.16  26.03  26.31  25.63  27.61  28.35  28.88]
 [ 28.7  28.98  28.16  26.16  27.27  26.31  27.79  28.13  29.83  31.3 ]
 [ 27.79  27.89  25.63  26.16  25.65  25.21  25.46  26.21  28.21  28.53]]
Transpose of Matrix b is:
[[801.  802.  803.  804.  805.  806.  807.  808.  809.  810. ]
 [ 28.48  28.1  26.16  26.16  26.1  25.45  26.16  27.44  28.63  30.35]
 [ 34.18  33.72  31.39  31.39  31.32  30.54  31.39  32.93  34.35  36.42]
 [ 30.56  30.68  28.2  28.78  28.22  27.73  28.01  28.83  31.03  31.38]
 [ 22.23  22.82  22.53  20.93  20.82  21.05  20.51  22.08  22.68  23.1 ]]
a*b is:
[[6.4160100e+05 1.2260640e+03 9.4986220e+02 8.7707200e+02 6.1777170e+02]
 [6.4320400e+05 1.2215070e+03 9.6169440e+02 8.8910640e+02 6.3644980e+02]]
Ln 13, Col 1 Spaces: 4 UTF-8 LF Python 3.10.10 64-bit Go Live Prettier
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
[[6.4160100e+05 1.2260640e+03 9.4986220e+02 8.7707200e+02 6.1777170e+02]
 [6.4320400e+05 1.2215070e+03 9.6169440e+02 8.8910640e+02 6.3644980e+02]
 [6.4480900e+05 1.1049984e+03 8.8394240e+02 7.9411200e+02 5.7744390e+02]
 [6.4641600e+05 1.0265184e+03 8.2116240e+02 7.5288480e+02 5.4752880e+02]
 [6.4802500e+05 1.0674900e+03 8.1525960e+02 7.6955940e+02 5.3403300e+02]
 [6.4963600e+05 1.0045115e+03 8.0350740e+02 7.2957630e+02 5.3067050e+02]
 [6.5124900e+05 1.0903488e+03 8.0452570e+02 7.7839790e+02 5.2218460e+02]
 [6.5286400e+05 1.1576936e+03 9.0919730e+02 8.1098790e+02 5.7871680e+02]
 [6.5448100e+05 1.2811925e+03 9.7382250e+02 9.2562490e+02 6.3980280e+02]
 [6.5610000e+05 1.4249325e+03 1.0518096e+03 9.8219400e+02 6.5904300e+02]]

Trace of a:
924.4399999999999

Trace of b:
910.09

Flatten a: [801.  43.05  27.79  28.7  27.79 802.  43.47  28.52  28.98  27.89
803.  42.24  28.16  28.16  25.63 804.  39.24  26.16  26.16  26.16
805.  40.9  26.03  27.27  25.65 806.  39.47  26.31  26.31  25.21
807.  41.68  25.63  27.79  25.46 808.  42.19  27.61  28.13  26.21
809.  44.75  28.35  29.83  28.21 810.  46.95  28.88  31.3  28.53]

Flatten b: [801.  28.48  34.18  30.56  22.23 802.  28.1  33.72  30.68  22.82
803.  26.16  31.39  28.2  22.53 804.  26.16  31.39  28.78  20.93
805.  26.1  31.32  28.22  20.82 806.  25.45  30.54  27.73  21.05
807.  26.16  31.39  28.01  20.51 808.  27.44  32.93  28.83  22.08
809.  28.63  34.35  31.03  22.68 810.  30.35  36.42  31.38  23.1 ]
Horizontal Stacking
[[801.  43.05  27.79  28.7  27.79 801.  28.48  34.18  30.56  22.23]
 [802.  43.47  28.52  28.98  27.89 802.  28.1  33.72  30.68  22.82]
 [803.  42.24  28.16  28.16  25.63 803.  26.16  31.39  28.2  22.53]
 [804.  39.24  26.16  26.16  26.16 804.  26.16  31.39  28.78  20.93]
 [805.  40.9  26.03  27.27  25.65 805.  26.1  31.32  28.22  20.82]
 [806.  39.47  26.31  26.31  25.21 806.  25.45  30.54  27.73  21.05]
 [807.  41.68  25.63  27.79  25.46 807.  26.16  31.39  28.01  20.51]
 [808.  42.19  27.61  28.13  26.21 808.  27.44  32.93  28.83  22.08]
 [809.  44.75  28.35  29.83  28.21 809.  28.63  34.35  31.03  22.68]
 [810.  46.95  28.88  31.3  28.53 810.  30.35  36.42  31.38  23.1 ]]
Ln 13, Col 1 Spaces: 4 UTF-8 LF Python 3.10.10 64-bit Go Live Prettier
```

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

Code

+

-

🔍

🗑️

⋮

↶

↷

```
Vertical Stacking
[[801. 43.05 27.79 28.7 27.79]
 [802. 43.47 28.52 28.98 27.89]
 [803. 42.24 28.16 28.16 25.63]
 [804. 39.24 26.16 26.16 26.16]
 [805. 40.9 26.03 27.27 25.65]
 [806. 39.47 26.31 26.31 25.21]
 [807. 41.68 25.63 27.79 25.46]
 [808. 42.19 27.61 28.13 26.21]
 [809. 44.75 28.35 29.83 28.21]
 [810. 46.95 28.88 31.3 28.53]
 [801. 28.48 34.18 30.56 22.23]
 [802. 28.1 33.72 30.68 22.82]
 [803. 26.16 31.39 28.2 22.53]
 [804. 26.16 31.39 28.78 20.93]
 [805. 26.1 31.32 28.22 20.82]
 [806. 25.45 30.54 27.73 21.05]
 [807. 26.16 31.39 28.01 20.51]
 [808. 27.44 32.93 28.83 22.08]
 [809. 28.63 34.35 31.03 22.68]
 [810. 30.35 36.42 31.38 23.1 ]]
```

Generating Custom Sequences:

```
[0 1 2 3 4 5 6 7 8 9]
[ 0 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 85
 90 95 100]
```

Adding a and b:

```
[[1602. 71.53 61.97 59.26 50.02]
 [1604. 71.57 62.24 59.66 50.71]
 [1606. 68.4 59.55 56.36 48.16]
 [1608. 65.4 57.55 54.94 47.09]
 [1610. 67. 57.35 55.49 46.47]
 [1612. 64.92 56.85 54.04 46.26]
 [1614. 67.84 57.02 55.8 45.97]
 [1616. 69.63 60.54 56.96 48.29]
 [1618. 73.38 62.7 60.86 50.89]
 [1620. 77.3 65.3 62.68 51.63]]
```

Ln 13, Col 1

Spaces: 4

UTF-8

LF

{}

Python

3.10.10 64-bit

Go Live

Prettier

🔍

🗑️

⋮

↶

↷

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

Code

+

-

🔍

🗑️

⋮

↶

↷

```
Subtracting a and b:
[[ 0. 14.57 -6.39 -1.86 5.56]
 [ 0. 15.37 -5.2 -1.7 5.07]
 [ 0. 16.08 -3.23 -0.04 3.1 ]
 [ 0. 13.08 -5.23 -2.62 5.23]
 [ 0. 14.8 -5.29 -0.95 4.83]
 [ 0. 14.02 -4.23 -1.42 4.16]
 [ 0. 15.52 -5.76 -0.22 4.95]
 [ 0. 14.75 -5.32 -0.7 4.13]
 [ 0. 16.12 -6. -1.2 5.53]
 [ 0. 16.6 -7.54 -0.08 5.43]]
```

Multiplying a nd b :

```
[[6.4160100e+05 1.2260640e+03 9.4986220e+02 8.7707200e+02 6.1777170e+02]
 [6.4320400e+05 1.2215070e+03 9.6169440e+02 8.8910640e+02 6.3644980e+02]
 [6.4480900e+05 1.1049984e+03 8.8394240e+02 7.9411200e+02 5.7744390e+02]
 [6.4641600e+05 1.0265184e+03 8.2116240e+02 7.5288480e+02 5.4752880e+02]
 [6.4802500e+05 1.0674900e+03 8.1525960e+02 7.6955940e+02 5.3403300e+02]
 [6.4963600e+05 1.0045115e+03 8.0350740e+02 7.2957630e+02 5.3067050e+02]
 [6.5124900e+05 1.0903488e+03 8.0452570e+02 7.7839790e+02 5.2218460e+02]
 [6.5286400e+05 1.1576936e+03 9.0919730e+02 8.1098790e+02 5.7871680e+02]
 [6.5448100e+05 1.2811925e+03 9.7382250e+02 9.2562490e+02 6.3980280e+02]
 [6.5610000e+05 1.4249325e+03 1.0518096e+03 9.8219400e+02 6.5904300e+02]]
```

Dividing a nd b :

```
[[1. 1.51158708 0.81304857 0.93913613 1.25011246]
 [1. 1.54697509 0.84578885 0.94458931 1.22217353]
 [1. 1.6146789 0.89710099 0.99858156 1.13759432]
 [1. 1.5 0.83338643 0.90896456 1.24988055]
 [1. 1.56704981 0.83109834 0.96633593 1.23198847]
 [1. 1.55088409 0.86149312 0.94879192 1.1976247 ]
 [1. 1.59327217 0.81650207 0.99214566 1.24134569]
 [1. 1.53753644 0.83844519 0.97571974 1.1870471 ]
 [1. 1.56304576 0.82532751 0.96132775 1.24382716]
 [1. 1.54695222 0.7929709 0.99745061 1.23506494]]
```

Mod of a and b:

```
[[ 0. 14.57 27.79 28.7 5.56]
 [ 0. 15.37 28.52 28.98 5.07]
 [ 0. 16.08 28.16 28.16 3.1 ]
 [ 0. 13.08 26.16 26.16 5.23]
 [ 0. 14.8 26.03 27.27 4.83]]
```

Ln 13, Col 1

Spaces: 4

UTF-8

LF

{}

Python

3.10.10 64-bit

Go Live

Prettier

🔍

🗑️

⋮

↶

↷

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
[ 0. 15.37 28.52 28.98 5.07]
[ 0. 16.08 28.16 28.16 3.1 ]
[ 0. 13.08 26.16 26.16 5.23]
[ 0. 14.8 26.03 27.27 4.83]
[ 0. 14.02 26.31 26.31 4.16]
[ 0. 15.52 25.63 27.79 4.95]
[ 0. 14.75 27.61 28.13 4.13]
[ 0. 16.12 28.35 29.83 5.53]
[ 0. 16.6 28.88 31.3 5.43]]
Remainder of a and b:
[[ 0. 14.57 27.79 28.7 5.56]
[ 0. 15.37 28.52 28.98 5.07]
[ 0. 16.08 28.16 28.16 3.1 ]
[ 0. 13.08 26.16 26.16 5.23]
[ 0. 14.8 26.03 27.27 4.83]
[ 0. 14.02 26.31 26.31 4.16]
[ 0. 15.52 25.63 27.79 4.95]
[ 0. 14.75 27.61 28.13 4.13]
[ 0. 16.12 28.35 29.83 5.53]
[ 0. 16.6 28.88 31.3 5.43]]
Mean of a: 186.03499999999997
Mean of b: 183.35659999999996
Variance of a: 95971.70073699999
Variance of b: 96781.31228644
Standard Deviation of a: 309.7929965912722
Standard Deviation of b: 311.0969499793272
Sum of all elements in a: 9301.749999999998
Sum of all elements in b: 9167.829999999998
Broadcasting:
[[806. 48.05 32.79 33.7 32.79]
[807. 48.47 33.52 33.98 32.89]
[808. 47.24 33.16 33.16 30.63]
[809. 44.24 31.16 31.16 31.16]
[810. 45.9 31.03 32.27 30.65]
[811. 44.47 31.31 31.31 30.21]
[812. 46.68 30.63 32.79 30.46]
[813. 47.19 32.61 33.13 31.21]
[814. 49.75 33.35 34.83 33.21]
[815. 51.95 33.88 36.3 33.53]]
```

The image shows a VS Code editor window with a Python script and its output in the terminal. The script uses a loop to process data from a CSV file, stacking it into a list of lists. The terminal output shows the resulting data structure, which is a list of lists, each containing a row index and a list of values.

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
```

```
[814. 49.75 33.35 34.83 33.21]
[815. 51.95 33.88 36.3 33.53]]

Data Stacking:
[[[801. 801. ]
  [ 43.05 28.48]
  [ 27.79 34.18]
  [ 28.7 30.56]
  [ 27.79 22.23]]

 [802. 802. ]
  [ 43.47 28.1 ]
  [ 28.52 33.72]
  [ 28.98 30.68]
  [ 27.89 22.82]]

 [803. 803. ]
  [ 42.24 26.16]
  [ 28.16 31.39]
  [ 28.16 28.2 ]
  [ 25.63 22.53]]

 [804. 804. ]
  [ 39.24 26.16]
  [ 26.16 31.39]
  [ 26.16 28.78]
  [ 26.16 20.93]]

 [805. 805. ]
  [ 40.9 26.1 ]
  [ 26.03 31.32]
  [ 27.27 28.22]
  [ 25.65 20.82]]

 [806. 806. ]
  [ 39.47 25.45]
  [ 26.31 30.54]
  [ 26.31 27.73]
  [ 25.21 21.05]]]
```

Ln 13, Col 1 Spaces: 4 UTF-8 LF Python 3.10.10 64-bit Go Live Prettier

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
[ 25.21 21.05]]

[[807. 807. ]
 [ 41.68 26.16]
 [ 25.63 31.39]
 [ 27.79 28.01]
 [ 25.46 20.51]]

[[808. 808. ]
 [ 42.19 27.44]
 [ 27.61 32.93]
 [ 28.13 28.83]
 [ 26.21 22.08]]

[[809. 809. ]
 [ 44.75 28.63]
 [ 28.35 34.35]
 [ 29.83 31.03]
 [ 28.21 22.68]]

[[810. 810. ]
 [ 46.95 30.35]
 [ 28.88 36.42]
 [ 31.3 31.38]
 [ 28.53 23.1 ]]]
Sorting a:
[[ 27.79 27.79 28.7 43.05 801. ]
 [ 27.89 28.52 28.98 43.47 802. ]
 [ 25.63 28.16 28.16 42.24 803. ]
 [ 26.16 26.16 26.16 39.24 804. ]
 [ 25.65 26.03 27.27 40.9 805. ]
 [ 25.21 26.31 26.31 39.47 806. ]
 [ 25.46 25.63 27.79 41.68 807. ]
 [ 26.21 27.61 28.13 42.19 808. ]
 [ 28.21 28.35 29.83 44.75 809. ]
 [ 28.53 28.88 31.3 46.95 810. ]]
Sorting b:
[[ 22.23 28.48 30.56 34.18 801. ]
 [ 22.82 28.1 30.68 33.72 802. ]]
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
[ 20.93 26.16 28.78 31.39 804. ]
[ 20.82 26.1 28.22 31.32 805. ]
[ 21.05 25.45 27.73 30.54 806. ]
[ 20.51 26.16 28.01 31.39 807. ]
[ 22.08 27.44 28.83 32.93 808. ]
[ 22.68 28.63 31.03 34.35 809. ]
[ 23.1 30.35 31.38 36.42 810. ]]
Counting elements in a: 50
Counting elements in b: 50
Counting using elements less than 50 in a: 50
Counting using elements less than 10 in b: 10

View Method

a=
[[0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]]
v=
[[0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]]
Array created using view method is just shallow copy of original array
50 changes made in original array reflects in view copy or vice versa
```


 Code ...

