

TASK-8

Python programs

Consoles of programs:

1.vectors-consecutive zeros:



```
Python 3.9.7 (tags/v3.9.7:1016ef3, Aug 30 2021, 20:19:38) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/PYTHON/1.vector consecutive zero.py =====
start with the n.o
10
stop with the n.o
14
10. 0. 0. 0. 0. 0. 11. 0. 0. 0. 0. 0. 12. 0. 0. 0. 0. 13. 0. 0. 0. 0. 14
>>>
===== RESTART: C:/PYTHON/1.vector consecutive zero.py =====
start with the n.o
1
stop with the n.o
9
1. 0. 0. 0. 0. 0. 2. 0. 0. 0. 0. 0. 3. 0. 0. 0. 0. 0. 4. 0. 0. 0. 0. 5. 0. 0. 0. 0. 6. 0. 0. 0. 0. 7. 0. 0. 0. 0. 8. 0.
0. 0. 0. 0. 9
>>> |
```

2.arrays:

```
IDLE Shell 3.9.7
File Edit Shell Debug Options Window Help
4
Enter the elements of 2nd array
5
6
3
2
The Array's are
[1, 2, 3, 4]
[5, 6, 3, 2]
False
>>>
===== RESTART: C:/PYTHON/2.arrays.py =====
n.o of elements in an array:
6
Enter the elements 1st First array
1
0
0
0
0
Enter the elements of 2nd array
0
0
1
1
0
1
The Array's are
[1, 0, 0, 0, 0, 0]
[0, 0, 1, 1, 0, 1]
False
>>> |
```

3.console for given code:

Given code:

```
print(0 * np.nan)
```

```
print(np.nan != np.nan)
```

```
print(np.inf > np.nan)
```

```
print(np.nan - np.nan)
```

```
print(0.3 == 3 * 0.1)
```

```
OneCompiler
3.result of given program
+ 3xvg3k725
NEW PYTHON RUN
1 import numpy as np
2 print(0 * np.nan)
3 print(np.nan != np.nan)
4 print(np.inf > np.nan)
5 print(np.nan - np.nan)
6 print(0.3 == 3 * 0.1)

STDIN
Input for the program ( Optional )

Output:
nan
True
False
nan
False
```

5)

2. matrix multiplication

OneCompiler

5.b-matrixmultiplication 3xvg4sj3m NEW PYTHON RUN

```
1 # importing the module
2 import numpy as np
3
4 # creating two matrices
5 k = [[2, 2], [8, 3]]
6 u = [[7, 9], [4, 6]]
7 print("Matrix k :")
8 print(k)
9 print("Matrix u :")
10 print(u)
11
12 # computing product
13 result = np.dot(k, u)
14
15 # printing the result
16 print("The matrix multiplication is :")
17 print(result)
```

STDIN
Input for the program (Optional)

Output:

Matrix k :
[[2, 2], [8, 3]]
Matrix u :
[[7, 9], [4, 6]]
The matrix multiplication is :
[[22 38]
[68 90]]

FEEDBACK

3.identity matrix:

OneCompiler

5.3-identity matrix 3xvg5u7ud NEW PYTHON RUN

```
1 import numpy as np
2
3 # 2x2 matrix with 1's on main diagonal
4 b = np.identity(2, dtype = float)
5 print("Matrix b : \n", b)
6
7
8 a = np.identity(4)
9 print("\nMatrix a : \n", a)
10
```

STDIN
Input for the program (Optional)

Output:

Matrix b :
[[1. 0.]
[0. 1.]]

Matrix a :
[[1. 0. 0. 0.]
[0. 1. 0. 0.]
[0. 0. 1. 0.]
[0. 0. 0. 1.]]

FEEDBACK

5.3-identity matrix Show all

6.sequence generation:



5.6-sequence generation



3xvg5487p

NEW

PYTHON

RUN



```
1 import numpy as np
2 def generate(value):
3     return 3**value
4 arr = np.array([])
5 for value in range(0, 20):
6     # Appends the new value to the end of the numpy array
7     arr = np.append(arr, [generate(value)])
8 print(arr)
```

STDIN

Input for the program (Optional)

Output:

```
[1.00000000e+00 3.00000000e+00 9.00000000e+00 2.70000000e+01 8.10000000e+01 2.43000000e+02 7.29000000e+02 6.56100000e+03 1.96830000e+04 5.90490000e+04 5.31441000e+05 1.59432300e+06 4.78296900e+06 4.30467210e+07 1.29140163e+08 3.87420481e+08]
```

FEEDBACK