1. Array Creation

Write a function array_operation which can accept a parameter I and create a numpy array x, using array method of numpy package then print the following values

- The type of array x
- Dimensions of x
- Shape of x
- Size of x

▼ Sample Case

Sample Input For Custom Testing

```
5 10 15 20 25
```

Sample Output

```
<class 'numpy.ndarray'>
1
```

```
import numpy as np
    # Enter your code here. Read input from STDIN. Print ou
    def array_operations(l):
         #Write your code below
         x = np.array(l)
        print(type(x))
 9
         print(x.ndim)
10
11
         print(x.shape)
12
         print(x.size)
13
14
15 > if __name__ == "__main__": ...
```

Test Results

Custom Input

1. Determine Attributes

Write a function array_attributes which can accept a parameter 2d array as **I**, then create a numpy array y using array method of numpy package and print the following values

- The type of array y
- · Dimensions of y
- Shape of y
- Size of y
- Type of each data element of y
- Number of bytes occupied by each data element of y

▼ Sample Case

Sample Input

```
2
-1 -2 -3 -4
-2 -4 -6 -8
```

Sample Output

```
import numpy as np
 2
     # Enter your code here. Read input from STDIN. Print output
     def array_attributes(l):
         #write your code here
 6
 7
         y = np.array(l)
         print(type(y))
 8
         print(y.ndim)
 9
10
         print(y.shape)
11
         print(y.size)
12
         print(y.dtype)
13
         print(y.itemsize)
14
15 > if __name__=="__main__": ...
```

Test Results

Custom Input

Compiled successfully. All available test cases passed

✓ Test case 0

Input (stdin)

1. ndarray

Write a function ndarray to get nd list as a argument **array_input**, then create a numpy array x1 using array method of numpy package and print the following values

- Dimensions of x1
- Shape of x1
- Size of x1

Input:

- The input arrays are already given in the main function
- Totally it has 5 nd arrays
- Each test case will pass a nd array as a argument to the function ndarray(array_input)

▶ Sample Case

```
import numpy as np

import numpy as np

# Enter your code here. Read input from STDIN. Print output to STDOUT

def ndarray(array_input):
    #Write your code here
    x1 = np.array(array_input)
    print(x1.ndim)
    print(x1.shape)
    print(x1.size)

if __name__ == "__main__": ...
```

Test Results

Custom Input

Compiled successfully. All available test cases passed

 \Im Test case 0

Input (stdin)

1. ndarray shape

Write a function ndshape which can accepts two arguments **d** and **shape1**, than write a code to perform the following

- Create a ndarray x1, with the shape d, contains 1's on diagonal and 0's elsewhere
- . Create a ndarray x2, with the shape shape1 and contains all 1's
- Then, print the values of x1 and x2

▶ Sample Case

```
import numpy as np

import numpy as np

# Enter your code here. Read input from STDIN. Print output to STDOUT

def ndshape(d, shape1):
    #Write your code here
    x1 = np.eye(d)
    print(x1)
    x2 = np.ones(shape1)
    print(x2)

if __name__ == "__main__":...
```

Test Results

Custom Input

1. 3D Array

Write a function array_3d which can accept four parameters **n**, **x**, **y** and **z**, then perform the following

- Simulate a random normal distribution for **n**, whose mean is 5 and standard deviation 2.5. Capture the result in **x1**.
 - Find the product of the number of element in the array and the size of one array element in byte. Then store it in the variable A

Note: Set numpy random seed value 100

- Define a random 3-D array x2 of shape (x, y, z) and of numbers between 0 and 1
- Print the value of A and x2

▼ Sample Case

Sample Input For Custom Testing

```
import numpy as np
    # Enter your code here. Read input from STDIN. Print output to STDOUT
    def array_3d(n, x, y, z):
        #Write your code below
        np.random.seed(100)
 6
        x1 = 5 + 2.5*np.random.randn(n)
        A = x1.size*x1.itemsize
        x2 = np.random.rand(x,y,z)
 9
        # x is number of nested array, y is dimension of array, z is number of elements in each dimension
10
11
        print(A)
12
        print(x2)
13
14 > if __name__ == "__main__":...
```

Run (

Test Results

Custom Input

1. More on ndarray

Write a function arr which can accept four parameters a, x, y and z, then perform the following operations

- Define a ndarray x1, having first a even numbers(zero not included)
 - Hint: Make use of arange
- Define a ndarray x2, having equally spaced z numbers between x and y
 - Hint: Make use of linspace
- Print x1 and x2

► Sample Case

```
import numpy as np

# Enter your code here. Read input from STDIN. Print output to STDOUT

def arr(a, x, y, z):
    #Write your code here
    x1 = np.arange(2,a,2)
    print(x1)
    x2 = np.linspace(x,y,z)
    print(x2)

if __name__ == "__main__":...
```

Test Results

Custom Input

1. Array Manipulation

Write a function array_split which can accept three parameters **i**, **r** and **c**, then perform the following operations

- Create a ndarray x having first i natural numbers, using arange method
- Change the shape of x to (c, r) and assign it to new array y
- . Split the array y horizontally in to two arrays, then assign it to a and b
- · Print a and b

▼ Sample Case

Sample Input

20 2 10

Sample Output

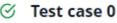
```
[[ 0 1 2 3 4]
[10 11 12 13 14]]
[[ 5 6 7 8 9]
```

```
import numpy as np
     # Enter your code here. Read input from STDIN. Print output to STDOUT
     def array_split(i,r,c):
         #Write your code below
 6
         x = np.arange(i)
         y = x.reshape(r,c)
         a = np.hsplit(y,2)[0]
         b = np.hsplit(y,2)[1]
 9
10
         print(a)
         print(b)
11
12
13 > if __name__ == "__main__ ": ...
```

Test Results

Custom Input

Compiled successfully. All available test cases passed



Input (ctdin)

1. Array Manipulation 2

Array Manipulation 2

Function array_split

Write the functions array_split that accepts three numbers n,n_row,n_col and perform array operations given below

Instructions

- Create a ndarray x having first n natural numbers, using arange method.
- Change the shape of x to (n_row, n_col) and assign it to new array z
- Split the array z vertically in to two arrays
- Hint: Use vsplit
- Print the arrays

```
1 > import ast...
 3
     # Enter your code here. Read input from STDIN. Print output to STDOUT
     def array_split(n,n_row,n_col):
         x = np.arange(n)
         z = x.reshape(n_row,n_col)
         a = np.vsplit(z,2)
 8
         print(a[0])
 9
         print(a[1])
10
11
12
13
14
15 > if __name__ == "__main__":...
```

Test Results

Custom Input

1. Join Arrays

Join Arrays

Function join_array

Write the function join_array which accepts two lists - list1,list2 . Function should print the arrays after joining them horizontally.

Instructions

- Create a 2-D array p, of shape (2,2) with elements of list list1
- Create a 2-D array q, of shape (2,3) with elements of list list2
- Join the two arrays p and q horizontally
- Print the new array

▼ Sample Case 0

Sample Input

Test Results

Custom Input

1. Operations on Arrays 1

Operations on Arrays 1

Function array_oper

Write the function array_oper which accepts two numbers num1,num2 and perform array operations given below.

Instructions

- Create a 2-D array y of shape (2,3) having numbers from num1 to num2
- Square each element of y
- · Add 5 to each element of resulted array.
- Print the new array

▼ Sample Case 0

Sample Input

1

```
import numpy as np

# Enter your code here. Read input from STDIN. Print output to STDOUT

def array_oper(num1,num2):
    y = np.arange(num1,num2+1).reshape(2,3)
    res = y**2 + 5
    print(res)

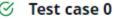
print(res)

if __name__ == "__main__":...
```

Test Results

Custom Input

Compiled successfully. All available test cases passed



Input (ctdin)

1. Operation on Arrays 2

Operation on Arrays 2

Function array_oper

Write the function array_oper which accepts two numbers num1,num2 and performs array operations given below.

Instructions

- Create a array x of shape (5,6), having 30 random integers between num1.num2.
- Print the cumulative sum of x along axis 0.
- Print the cumulative sum of x along axis 1.
- · Note: Set np.random.seed(100).

```
► Sample Case 0
```

```
import numpy as np
    # Enter your code here. Read input from STDIN. Print output to STDOUT
     def array_oper(num1,num2):
         np.random.seed(100)
         x = np.random.randint(num1,num2,30)
         x = x.reshape(5,6)
         a = np.cumsum(x,axis=0)
         b = np.cumsum(x,axis=1)
         print(a[-1][-1]) #to get last element in the last array dimension
10
         print(b[-1][-1])
11
12
13
14
15
16 > if name == " main ": ...
```

Test Results

Custom Input

Compiled successfully. All available test cases passed

Question looking for last element in last array. See below link for examples.

https://numpy.org/doc/stable/reference/generated/numpy.cumsum.html

1. Operation on Arrays 3

Operation on Arrays 3

Function array_oper

Write the function array_oper which accepts three parameters n,m,sd and perform the array operations given below.

Instructions

- Create a array x of shape (n,), having n random numbers from a normal distribution of mean m and standard deviation sd.
- Print mean of x
- Print Standard deviation of x
- Print variance of x
- Note: Set np.random.seed(100)

```
1 > import ast...
 3
     # Enter your code here. Read input from STDIN. Print output to STDOUT
     def array_oper(n,m,sd):
         np.random.seed(100)
         x = m + sd*np.random.randn(n,)
         print(np.mean(x))
         print(np.std(x))
 9
         print(np.var(x))
10
11
12
13
14
15
16 > if __name__ == '__main__':...
```

Test Results

Custom Input

1. Array Indexing

Array Indexing

Function array_index

Write the function array_index which accepts three numbers n,n_row,n_col and performs the array operations given below.

Instructions

- Create a array x of shape (n_row,n_col), having first n natural numbers.
- · Print elements of last row.
- · Print elements of middle column.
- Print elements, overlapping first two rows and last three columns.

▼ Sample Case 0

Sample Input

```
import numpy as np
 2
    # Enter your code here. Read input from STDIN. Print output to STDOUT
    def array_index(n,n_row,n_col):
         x = np.arange(n).reshape(n_row,n_col)
         print(x[-1])
         print(x[:,2])
         print(x[:2,-3:])
 8
 9
10
11
12
13
14
15 > if __name__ == '__main__': ...
```

Test Results

Custom Input



1. Slicing

Slicing

Function array_slice

Write the function array_slice which accepts four numbers n, n_dim, n_row, n_col and performs array operations given below.

Instructions

- Create a array x of shape (n_dim,n_row,n_col), having first n natural numbers.
- Create a boolean array b of shape (2,), having elements True, False.
- · Print the values for following expressions
 - x[b]
 - x[b,:,1:3]

Test Results

Custom Input