In [3]:
```
# 1.Write a NumPy program to test element-wise for NaN of a given array.
import numpy as np
a=np.array([1, np.nan, 3, 4])
print("array :",a)
print("Test element-wise for NaN:",np.isnan(a))
```

```
array : [ 1. nan  3.  4.]
Test element-wise for NaN: [False  True False False]
```

In [5]:
```
# 2.Write a NumPy program to compute the inner product of two given vectors.
a=np.array([4, 5])
b=np.array([7, 10])
print("Given vectors :",a,b)
print("Inner product of two vectors :",np.dot(a,b))
```

```
Given vectors : [4 5] [ 7 10]
Inner product of two vectors : 78
```

In [7]:
```
# 3.Write a NumPy program to create an array with values ranging from 12 to 38.
a=np.arange(12, 39)
print("Values ranging from 12 to 38 :",a)
```

```
Values ranging from 12 to 38 : [12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35
 36 37 38]
```

In [8]:
```
# 4.Write a NumPy program to reverse an array (first element becomes last).
a=np.arange(15,23)
print("array :",a)
a=a[::-1]
print("Reverse array :",a)
```

```
array : [15 16 17 18 19 20 21 22]
Reverse array : [22 21 20 19 18 17 16 15]
```

In [10]:
```
# 5.Write a NumPy program to convert an array to a float type.
a=[4,5,6,7,8,9]
print("array :",a)
a1=np.asfarray(a)
print("Array converted to a float type :",a1)
```

```
array : [4, 5, 6, 7, 8, 9]
Array converted to a float type : [4. 5. 6. 7. 8. 9.]
```

In [14]:
```
# 6.Write a NumPy program to create a 2d array with 1 on the border and 0 inside.
a=np.ones((5,5))
print("array :",a)
a[1:-1,1:-1] = 0
print("1 on the border and 0 inside :",a)
```

```
array : [[1. 1. 1. 1. 1.]
 [1. 1. 1. 1. 1.]
 [1. 1. 1. 1. 1.]
 [1. 1. 1. 1. 1.]
 [1. 1. 1. 1. 1.]]
1 on the border and 0 inside : [[1. 1. 1. 1. 1.]
 [1. 0. 0. 0. 1.]
 [1. 0. 0. 0. 1.]
 [1. 0. 0. 0. 1.]
 [1. 1. 1. 1. 1.]]
```

In [15]:
```
# 7.Write a NumPy program to create a 8x8 matrix and fill it with a checkerboard pattern.
'''Checkerboard pattern:
[[0 1 0 1 0 1 0 1]
 [1 0 1 0 1 0 1 0]
 [0 1 0 1 0 1 0 1]
 [1 0 1 0 1 0 1 0]
 [0 1 0 1 0 1 0 1]
 [1 0 1 0 1 0 1 0]
 [0 1 0 1 0 1 0 1]
 [1 0 1 0 1 0 1 0]]
'''
a=np.ones((3,3))
a=np.zeros((8,8),dtype=int)
a[1::2,::2] = 1
a[::2,1::2] = 1
print("Checkerboard pattern :",a)
```

```
Checkerboard pattern : [[0 1 0 1 0 1 0 1]
 [1 0 1 0 1 0 1 0]
 [0 1 0 1 0 1 0 1]
 [1 0 1 0 1 0 1 0]
 [0 1 0 1 0 1 0 1]
 [1 0 1 0 1 0 1 0]
 [0 1 0 1 0 1 0 1]
 [1 0 1 0 1 0 1 0]]
```

In [18]:
```
# 8.Write a NumPy program to append values to the end of an array.
a=[34,45,12]
print("array :",a)
a=np.append(a, [[45,67,65], [75,23,21]])
print("After append values to the end of an array :",a)
```

```
array : [34, 45, 12]
After append values to the end of an array : [34 45 12 45 67 65 75 23 21]
```

In [21]:
```
# 9.Write a NumPy program to test whether each element of a 1-D array is also present in a second array.
array1=np.array([10,20,30,40,50,60,70,80,90])
print("Array1: ",array1)
array2=[10,50,80,90]
print("Array2: ",array2)
print("Compare each element of array1 and array2")
print(np.in1d(array1, array2))
```

```
Array1:  [10 20 30 40 50 60 70 80 90]
Array2:  [10, 50, 80, 90]
Compare each element of array1 and array2
[ True False False False  True False False  True  True]
```

In [22]:
```
# 10.Write a NumPy program to get the unique elements of an array.
a=np.array([3,4,5,3,4,5,6,7,8])
print("array :",a)
print("Unique elements of the array:",np.unique(a))
a=np.array([[6,6], [7,8]])
print("array :",a)
print("Unique elements of the above array :",np.unique(a))
```

```
array : [3 4 5 3 4 5 6 7 8]
Unique elements of the array: [3 4 5 6 7 8]
array : [[6 6]
 [7 8]]
Unique elements of the above array : [6 7 8]
```

In [25]:
```
# 11.Write a NumPy program to find the indices of the maximum and minimum values along the given axis of an array.
a=np.array([1,2,3,4,5,6,7,8,9])
print("array : ",a)
print("Maximum Values: ",np.argmax(a))
print("Minimum Values: ",np.argmin(a))
```

```
array :  [1 2 3 4 5 6 7 8 9]
Maximum Values:  8
Minimum Values:  0
```

In [27]:
```
# 12.Write a NumPy program to create a contiguous flattened array
a=np.array([[10,20,30,40], [20,30,40,50]])
print("Array :",a)
b=np.ravel(a)
print("Contiguous flattened array:",b)
```

```
Array : [[10 20 30 40]
 [20 30 40 50]]
Contiguous flattened array: [10 20 30 40 20 30 40 50]
```

In [31]:
```
# 13.Write a NumPy program to concatenate two 2-dimensional arrays.
a=np.array([[2, 1, 3], [5, 7, 9]])
b=np.array([[3, 2, 4], [6, 8, 10]])
c=np.concatenate((a, b), 1)
print("Concatenate two dimensional array :",c)
```

```
Concatenate two dimensional array : [[ 2  1  3  3  2  4]
 [ 5  7  9  6  8 10]]
```

In [34]:
```
# 14.Write a NumPy program to convert 1-D arrays as columns into a 2-D array.
a=np.array((4,5,6))
b=np.array((7,8,9))
c=np.column_stack((a, b))
print("Convert 1-D arrays as columns into a 2-D array :")
print(c)
```

```
Convert 1-D arrays as columns into a 2-D array :
[[4 7]
 [5 8]
 [6 9]]
```

In [36]:
```
# 15.Write a NumPy program (using NumPy) to sum of all the multiples of 3 or 5 below 100.
a=np.arange(1, 100)
n= a[(a % 3 == 0) | (a % 5 == 0)]
print("Multiples of 3 or 5 below 100 :",n[:1000])
print("sum of all the multiples of 3 or 5 below 100 :",n.sum())
```

```
Multiples of 3 or 5 below 100 : [ 3  5  6  9 10 12 15 18 20 21 24 25 27 30 33 35 36 39 40 42 45 48 50 51
 54 55 57 60 63 65 66 69 70 72 75 78 80 81 84 85 87 90 93 95 96 99]
sum of all the multiples of 3 or 5 below 100 : 2318
```

In [39]:
```
# 16.Write a NumPy program to get the index of a maximum element in a NumPy array along one axis.
a=np.array([[1,2,3,6],[4,3,1,5]])
print("Array :",a)
i,j = np.unravel_index(a.argmax(), a.shape)
print("Index of a maximum element in a numpy array along one axis:",a[i,j])
```

```
Array : [[1 2 3 6]
 [4 3 1 5]]
Index of a maximum element in a numpy array along one axis: 6
```

In [40]:
```
# 17.Write a NumPy program to stack 1-D arrays as row wise.
a=np.array((1,2,3))
b=np.array((2,3,4))
print("Array-1 :",a)
print("Array-2 :",b)
arr=np.row_stack((a,b))
print("Stack 1-D arrays as rows wise :",arr)
```

```
Array-1 : [1 2 3]
Array-2 : [2 3 4]
Stack 1-D arrays as rows wise : [[1 2 3]
 [2 3 4]]
```

In [48]:
```
# 18.Write a NumPy program to get the minimum and maximum value of a given array along the second axis.
'''Expected Output:
Original array:
[[0 1]
[2 3]]
Maximum value along the second axis:
[1 3]
Minimum value along the second axis:
[0 2]
'''
a=np.arange(4).reshape((2, 2))
print("Original array:",a)
print("Maximum value along the second axis:",np.amax(a, 1))
print("Minimum value along the second axis:",np.amin(a, 1))
```

```
Original array: [[0 1]
 [2 3]]
Maximum value along the second axis: [1 3]
Minimum value along the second axis: [0 2]
```

In [ ]:
```

```