20/07/2023 Daily assessment

1.Create an array with zeros and ones

In [1]:
```
import numpy as np
```

zeros

In [7]:
```
a=np.zeros(6,dtype=np.int64)
print(a)
```

```
[0 0 0 0 0 0]
```

Ones

In [8]:
```
a=np.ones(7,dtype=np.int64)
print(a)
```

```
[1 1 1 1 1 1 1]
```

Both zeros and ones

In [5]:
```
a=np.zeros(4,dtype=np.int64)
b=np.ones(4,dtype=np.int64)
c=np.concatenate((a,b),axis=0)
print(c)
```

```
[0 0 0 0 1 1 1 1]
```

2.Create an array and print the output

In [10]:
```
a=np.array([4,5,6,7,8,9])
print(a)
```

```
[4 5 6 7 8 9]
```

3.Create an array whose initial content is random and print the output

In [13]:
```
a=np.empty(5,dtype=np.int64)
print(a)
```

```
[          4294967296 22236810922950656                788   140716013519616
 20266460320694349]
```

4.Create an array with the range of values with even intervals

In [16]:
```
a=np.linspace(1,100,num=50,dtype=np.int64)
print(a)
```

```
[  1   3   5   7   9  11  13  15  17  19  21  23  25  27  29  31  33  35
  37  39  41  43  45  47  49  51  53  55  57  59  61  63  65  67  69  71
  73  75  77  79  81  83  85  87  89  91  93  95  97 100]
```

In [19]:
```
a=np.arange(2,100,2)
print(a)
```

```
[ 2  4  6  8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48
 50 52 54 56 58 60 62 64 66 68 70 72 74 76 78 80 82 84 86 88 90 92 94 96
 98]
```

5.Create an array with values that are spaced linearly in a specified interval

In [20]:
```
a=np.linspace(1,100,num=25,dtype=np.int64)
print(a)
```

```
[  1   5   9  13  17  21  25  29  34  38  42  46  50  54  58  62  67  71
  75  79  83  87  91  95 100]
```

6.Access and manipulate elements in an array

In [21]:
```
a=np.array([8,7,9,6,3])
print(a)
```

```
[8 7 9 6 3]
```

In [22]:
```
print(a[0])
print(a[4])
print(a[-3])
```

```
8
3
9
```

In [23]:
```
print(a[0])
a[0]=9
print(a[0])
```

```
8
9
```

In [24]:
```
print(a[-2])
a[-2]=2
print(a[-2])
```

```
6
2
```

7.Create a 2d array and check the shape of the array

In [25]:
```
a=np.array([[12,45,6,7],[7,8,9,6]])
print(a)
print(np.shape(a))
```

```
[[12 45  6  7]
 [ 7  8  9  6]]
(2, 4)
```

8.Using the arange() and linspace() function to evenly space values in a specified interval

linspace()

In [26]:
```
a=np.linspace(1,100,num=50,dtype=np.int64)
print(a)
```

```
[  1   3   5   7   9  11  13  15  17  19  21  23  25  27  29  31  33  35
  37  39  41  43  45  47  49  51  53  55  57  59  61  63  65  67  69  71
  73  75  77  79  81  83  85  87  89  91  93  95  97 100]
```

arange()

In [ ]:
```
a=np.arange(2,100,2)
print(a)
```

9.Create an array of random values between 0 and 1 in a given shape

In [ ]:
```
a=np.empty()
```

10.Repeat each element of an array by a specified number of times using repeat() and tile() functions

In [28]:
```
a=np.array([9,8,7,6,5,4])
print(np.repeat(a,3))
```
```
[9 9 9 8 8 8 7 7 7 6 6 6 5 5 5 4 4 4]
```

In [29]:
```
print(np.tile(a,4))
```
```
[9 8 7 6 5 4 9 8 7 6 5 4 9 8 7 6 5 4 9 8 7 6 5 4]
```

11. How do you know the shape and size of an array

To find shape use shape() to find size use size()

In [31]:
```
print(np.shape(a))
```
```
(6,)
```

In [32]:
```
print(np.size(a))
```
```
6
```

12.Create an array that indicates the total number of elements in an array

In [33]:
```
a=np.array([7,89,7,9,5,4,3,2])
print(a)
print(np.size(a))
```
```
[ 7 89  7  9  5  4  3  2]
8
```

13.To find the number of dimensions in an array

In [34]:
```
a=np.array([[3,4,5,6,7],[7,8,9,0,5]])
print(np.ndim(a))
```
```
2
```

14.Create an array and reshape into a new array

In [44]:
```
a=np.array([2,3,4,5,5,6])
print(a)
print(a.reshape(2,3))
```

```
[2 3 4 5 5 6]
[[2 3 4]
 [5 5 6]]
```

15.Create a null array of size 10

In [47]:
```
a=np.zeros(10,dtype=np.int64)
print(a)
```

```
[0 0 0 0 0 0 0 0 0 0]
```

16.Create an array with values ranging from 10 to 49 and print the numbers

In [72]:
```
arr=np.arange(10,50)
print(arr)
```

```
[10 11 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 39 40 41 42 43 44 45 46 47 48 49]
```

17.Create an array and check two conditions and print the output

In [56]:
```
a=np.array([4,5,6,7,8,9])
if a[0]==4:
    print(1)
else:
    print(2)
```

```
1
```

In [64]:
```
if ((a[0]==4)&(a[3]==7)):
    print(1)
else:
    print(2)
```

```
1
```

18.Use arithmetic operator and print the output

In [65]:
```
a=np.array([5,6,7,8,9])
b=np.array([3,5,6,7,1])
print(a+b)
```

```
[ 8 11 13 15 10]
```

In [66]:
```
print(a-b)
```

```
[2 1 1 1 8]
```

In [67]:
```
print(a*b)
```

```
[15 30 42 56  9]
```

In [68]:
```
print(a/b)
```

```
[1.66666667 1.2        1.16666667 1.14285714 9.        ]
```

19.Use relational operators and print the results using array

In [71]:
```
a=np.array([5,6,7,8,9])
if a[0]<=a[4]:
    print(1)
```

```
1
```

20. Difference between python and ipython

IPython is an interactive shell for the Python programming language. It provides a number of features that make it more convenient and efficient to use than the standard Python shell. For example, IPython includes advanced command-line editing and history, as well as interactive widgets and data visualization tools.One of the main advantages of IPython is that it allows you to quickly test and experiment with Python code. You can use the shell to write and execute code one line at a time, rather than having to write a complete script and then run it. This makes it easier to debug and test your code, and also allows you to explore data and libraries more easily.Another advantage of IPython is that it provides a number of built-in magic commands that allow you to perform advanced operations, such as timing code execution, running scripts in parallel, and more.In summary, IPython is a powerful and convenient

In [ ]: