Today Agenda

- continue with Numpy
- Pandas

1. array()-->str/list/tuple/dict/set
2. ones()
3. zeros()
4. diag()
5. full()
6. fill()
7. eye()

arange()

- is the sub module of numpy
- it works similar to the range()
- numpy.arange()

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

In [2]:
```
np.__version__
```

Out[2]:
```
'1.21.5'
```

In [3]:
```
dir(np)
```

```
...
```

In [4]:
```
ar=np.arange(10)
print(ar)
```

```
[0 1 2 3 4 5 6 7 8 9]
```

In [5]:
```
print(np.arange(int(input()),int(input())))
```

```
9
24
[ 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23]
```

In [7]:
```
ar2=np.arange(10,100,8) # step_count=8
print(ar2)
```

```
[10 18 26 34 42 50 58 66 74 82 90 98]
```

In [8]:
```
len(ar2)
```

Out[8]:
```
12
```

In [9]:
```
# 2d array
# reshape()
# (1,12),(2,6),(3,4),(6,2),(12,1),(4,3)
```

In [10]:
```
ar2.reshape(3,4)
```

...

In [11]:
```
ar2.reshape(6,2)
```

...

In [12]:
```
# 501--2d array
# total=6/3
```

In [15]:
```
print(np.arange(501).reshape(3,-1))
```

...

In [18]:
```
# 45 values 9x5
np.arange(45).reshape(9,5)
```

...

In [17]:
```
# 45 values 9x5
np.arange(45).reshape(-1,9)
```

Out[17]:
```
array([[ 0,  1,  2,  3,  4,  5,  6,  7,  8],
       [ 9, 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]])
```

In [25]:
```
# 3d arrays
mul=np.arange(45).reshape(3,5,3)
# (no.of arrays,rows,cols)
print(mul)
```

...

linear space

- linspace() gives 50 equal partitions by default
- both limits in a given are inclusive

In [26]:
```
range(12)
```

Out[26]:
```
range(0, 12)
```

In [27]:
```
ln=np.linspace(0,3)
print(ln)
```

...

In [29]:
```
help(np.linspace)
```

...

In [35]:
```
ln2=np.linspace(10,15,8).reshape(4,2)
print(ln2)
```

...

In [33]:
```
10,11,12,13,14,15

```

Out[33]:
```
(10, 11, 12, 13, 14, 15)
```

In [36]:
```
6/8
```

Out[36]:
```
0.75
```

random()

- generates random values in a range
- random()
- random.randint()
- generates random floting samples with specified range of values
    - random.random()
    - random.rand()
    - random.randn()

In [40]:
```
np.random.randint(11)
```

Out[40]:
```
6
```

In [44]:
```
print(np.random.randint(10,100))
```

...

In [47]:

```python
# n no.of random values\
rn=np.random.randint(10,500,9)
print(rn)
```

...

In [51]:

```python
rad=np.random.random((2,3))
rad # random samples b/w 0 and 1
```

...

In [52]:

```python
rd=np.random.rand(2,3)
rd
```

...

In [55]:

```python
rdn=np.random.randn(2,3)
rdn # -ve random samples
```

...

using index

- formal indexing
  1. positive
     - starts from 0 to len(it)-1
     - travers from left to right
  2. negative
     - starts from -1 to -infinite
     - travers from right to left
- fancy indexing
  - condition based slicing

In [57]:

```python
arn=np.random.randint(10,100,25).reshape(5,5)
print(arn)
```

...

In [58]:

```python
arn[1]
```

Out[58]:

```
array([81, 79, 46, 32, 85])
```

In [59]:

```python
arn[:]
```

Out[59]:

```
array([[34, 57, 94, 18, 67],
       [81, 79, 46, 32, 85],
       [70, 71, 50, 72, 44],
       [45, 78, 70, 86, 51],
       [80, 43, 59, 90, 75]])
```

In [60]:
```
arn[::]
```

Out[60]:
```
array([[34, 57, 94, 18, 67],
       [81, 79, 46, 32, 85],
       [70, 71, 50, 72, 44],
       [45, 78, 70, 86, 51],
       [80, 43, 59, 90, 75]])
```

In [63]:
```
arn[:,2:]
```

Out[63]:
```
array([[94, 18, 67],
       [46, 32, 85],
       [50, 72, 44],
       [70, 86, 51],
       [59, 90, 75]])
```

In [64]:
```
arn[::2,::2] # alternate cols in alternate rows
```

Out[64]:
```
array([[34, 94, 67],
       [70, 50, 44],
       [80, 59, 75]])
```

In [65]:
```
arn[2:4,:]
```

Out[65]:
```
array([[70, 71, 50, 72, 44],
       [45, 78, 70, 86, 51]])
```

In [66]:
```
arn[2][2] #
```

Out[66]:
```
50
```

In [68]:
```
arn[2][3] # 4th value in 3rd row
```

Out[68]:
```
72
```

In [70]:
```
arn[1:4,1:4]
```

Out[70]:
```
array([[79, 46, 32],
       [71, 50, 72],
       [78, 70, 86]])
```

In [71]:
```
arn
```

...

In [72]:
```
# create the new with values >60 present in arn
arn>60
```

...

In [75]:

```
new=arn[arn>60] # fancy indexing
print(new)
len(new)
```

...

In [76]:

```
new.reshape(7,2)
```

...

In [77]:

```
arn
```

...

vectorized functions

scientific computation of Numpy

In [79]:

```
arn
```

...

In [83]:

```
rn=np.random.randint(10,50,25).reshape(5,5)
print(rn)
```

...

In [84]:

```
# arithmetic op'ns
rn+arn
```

...

In [85]:

```
rn-arn
```

...

In [86]:

```
rn*arn
```

...

In [87]:

```
rn/arn
```

...

In [88]:

```
arn.sum()
```

Out[88]:

```
1577
```

In [89]:

```
arn.min()
```

...

```
In [90]: arn.max()
```

...

```
In [91]: arn.mean()
```

...

```
In [92]: arn.std()
```

...

logarithms and exponentials

```
In [93]: np.log(arn)
```

...

```
In [94]: np.log([2,3,9])
```

...

```
In [95]: np.log(2)
```

...

```
In [96]: np.log2(2)
```

...

```
In [97]: np.log(1)

Out[97]: 0.0
```

```
In [98]: np.exp(1)

Out[98]: 2.718281828459045
```

```
In [99]: np.exp(2)

Out[99]: 7.38905609893065
```

```
In [100]: np.exp(rn)
```

...

```
In [101]: np.exp(np.log(arn))
```

...

In [102]:

```
# broad casting
# big change
```

In [103]:

```
arn+4 # applying scalar value on array/vector
```

...

In [107]:

```
# vectorized functions
def greater(a,b):
    if a>b:
        return a
    return b

greater(9,10)
```

...

In [108]:

```
greater(10,4)
```

...

In [109]:

```
greater([1,5,8],[10,6,3])
```

...

In [110]:

```
greater([12,1],[5,7])
```

...

In [111]:

```
gr=np.vectorize(greater)
gr([12,1],[5,7])
```

Out[111]:

```
array([12,  7])
```

pandas

- pandas means Panel Data
- most prominent one in data science modules
- used for data analysis,data manipulation and cleaning
- 2 data structure in Pandas
    1. Series
        - sequential data
    2. DataFrame
        - data is arranged in the form of rows and columns(table)
-

Series

In [112]:
```
# convert str/tuple/list/dict/set into series
```

In [115]:
```
import pandas as pd
```

In [116]:
```
st='srkit located at enikepadu'
```

In [117]:
```
pd.Series(st)
```

...

In [118]:
```
sec=st.split()
pd.Series(sec)
```

...

In [119]:
```
t=(2,5,6,"a",6.9)
pd.Series(t)
```

...

In [120]:
```
li=[2,4,5,4.7,345]
pd.Series(li)
```

...

In [124]:
```
dic={'f':12,'second':89.8,'nums':[1,2,3,4],
    'marks':(90,89,68,78)}
s=pd.Series(dic)
s
```

...

In [126]:
```
s.shape
```

Out[126]:
```
(4,)
```

In [127]:
```
s.index
```

...

In [128]:
```
# you can provide user index
s.index=[8,9,3,4]
```

In [129]:
```
s
```

...

In [130]:
```
# convert numpy array into series
import numpy as np
```

In [131]:

```
ar=np.arange(1,10)
ar
```

...

In [134]:

```
pd.Series(ar,index=[num for num in range(20,29)])
```

...

Data Frame

In [135]:

```
dic
```

...

In [136]:

```
pd.DataFrame(dic) # keys became columns and
# values as rows
```

...

working with .csv file

- csv(comma seperated values)

In [137]:

```
df=pd.read_csv('marks.csv')
df
```

...

In [139]:

```
df.sample() # random sample in df
```

...

In [142]:

```
df.sample(3)
```

...

In [144]:

```
df.head() # generates firs 5 rows by default
```

...

In [145]:

```
df.head(3)
```

...

In [146]:

```
df.tail() # generates last five samples
```

...

In [147]:

```
df.tail(3)
```

...

In [148]:

```
df.isnull()
```

...

In [149]:

```
df.isna()
```

...

In [152]:

```
df.describe() # statistics
```

...

In [151]:

```
df.info() # textual information
```

...

In [153]:

```
marks_df=pd.read_csv('marks.csv')
marks_df
```

...

In [154]:

```
marks_df.isnull()
```

...

In [156]:

```
# removal of NaN values-- cleaning of data
new=marks_df.dropna()
new
```

Out[156]:

	Name	CN	DM	ADU	SET
0	Nagesh	95.0	89.0	83.0	81
2	Siva Narayana	90.0	93.0	83.0	92
3	Anudeep	80.0	95.0	80.0	91
4	Elisha	95.0	81.0	88.0	91
7	Harish	83.0	82.0	82.0	89
8	Siddhartha	87.0	94.0	91.0	89

In [157]:

```
marks_df
```

...

In [160]:

```
# Remove the duplicates from the List
li=  [3,5,1,78,3,5,7]
li1=[]
for i in li:
    if i not in li1:
        li1.append(i)
print(li1)

```

```
[3, 5, 1, 78, 7]
```

In [163]:

```
h = [3,7,8,2,3,5,8,9,1]
s = []
for i in h:
    if h.count(i)==1:
        s.append(i)
print(s)
```

```
[7, 2, 5, 9, 1]
```

In [166]:

```
n = int(input())
d = {}
for i in range(1,n+1):
    d[i]=i**2
print(d)
```

```
50
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64, 9: 81, 10: 100, 11: 121,
12: 144, 13: 169, 14: 196, 15: 225, 16: 256, 17: 289, 18: 324, 19: 361, 20: 40
0, 21: 441, 22: 484, 23: 529, 24: 576, 25: 625, 26: 676, 27: 729, 28: 784, 29:
841, 30: 900, 31: 961, 32: 1024, 33: 1089, 34: 1156, 35: 1225, 36: 1296, 37: 13
69, 38: 1444, 39: 1521, 40: 1600, 41: 1681, 42: 1764, 43: 1849, 44: 1936, 45: 2
025, 46: 2116, 47: 2209, 48: 2304, 49: 2401, 50: 2500}
```

In [168]:

```
s,e=int(input()),int(input())
print("Even numbers are: ",end=' ')
for j in range(s,e+1):
    if(j%2==0):
        print(j,end=' ')
print("\nOdd Numbers are ",end=' ')
for j1 in range(s,e+1):
    if(j1%2!=0):
        print(j1,end=' ')
```

...

In [169]:

```
# Print odd digits in given number
n = int(input())
while(n>0):
    r=n%10
    if(r%2!=0):
        print(r,end=' ')
    n=n//10
```

...

In [175]:

```
n = int(input())
s=0
p=1
print("Even digits are : ",end=' ')
while(n>0):
    r=n%10
    if(r%2==0):
        print(r,end=' ')
        s=s+r
        p=p*r
    n=n//10
print("\nEven digits sum: ",s)
print("\nEven digits product: ",p)
```

```
7283460
Even digits are :  0 6 4 8 2
Even digits sum:  20

Even digits product:  0
```

In [177]:

```
n1 = int(input())
s1=0
for i in range(1,n1+1):
    s1=s1+i
print(s1)
```

```
100
5050
```

In [179]:

```
b = input()
if(b[::-1]==b):
    print("Palindrome")
else:
    print("Not Palindrome")
```

```
level
Palindrome
```

In [180]:

```
k = "SRK INSTITUTE OF TECHNOLOGY"
print(k.replace("S","A"))
```

```
ARK INATITUTE OF TECHNOLOGY
```

In [1]:

```python
c = input().split()
for i in c:
    if(len(i)%2==0):
        print(i)
```

```
this is srk college
this
is
```

In [4]:

```python
num=int(input())
if(num%2==0 and num>20):
    print("Not Weird")
elif(num%2==0 and 2<=num<=5):
    print("Not Weird")
elif(num%2==0 and 6<=num<=10):
    print("Weird")
else:
    print("Weird")

```

```
24
Not Weird
```

In [ ]:

```python

```