program 1

Create a vector with values ranging from 10 to 49

In [5]:

```
import numpy as np
```

In [6]:

```
arr= np.arange(10,49)
arr
```

Out[6]:

```
array([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])
```

#program 2 Create a 3x3 matrix with values ranging from 0 to 8

In [9]:

```
import numpy as np
x =  np.arange(0,9).reshape(3,3)
print(x)
```

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

program 3

Find indices of nonzero elements from [1,2,0,0,4,0]

In [13]:

```
import numpy as np
num = np.array([1,2,0,0,4,0])
print("Original array:")
print(num)
print("Indices of elements equal to zero of the said arr
ay:")
result = np.where(num == 0)[0]
print(result)
```

```
Original array:
[1 2 0 0 4 0]
Indices of elements equal to zero of the said array:
[2 3 5]
```

program 4

Create a random vector of size 10 and sort it

```
In [16]: import numpy as np
         arr = np.random.random(10)
         print("Original array:")
         print(arr)
         arr.sort()
         print("Sorted array:")
         print(arr)
```

```
Original array:
[0.77753239 0.97164915 0.75193122 0.69156803 0.6356655
3 0.85438083
 0.71541587 0.66708381 0.64014395 0.98834965]
Sorted array:
[0.63566553 0.64014395 0.66708381 0.69156803 0.7154158
7 0.75193122
 0.77753239 0.85438083 0.97164915 0.98834965]
```

program 5

Consider the following Python dictionary data and Python list labels : data = {'animal': ['cat', 'cat', 'snake', 'dog', 'dog', 'cat', 'snake' , 'cat', 'dog', 'dog'], 'age': [2.5, 3, 0.5, np.nan, 5, 2, 4.5, np.nan, 7, 3], 'visits': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1], 'priority': ['yes', 'yes', 'no', 'yes', 'no', 'no', 'no', 'ye s', 'no', 'no']} labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

(a) Create a DataFrame df from this dictionary data which has the index labels. (b) Display a summary of the basic information about this DataFrame and its data (c) Return the first 3 rows of the DataFrame df (d) Select just the 'animal' and 'age' columns from the DataFrame df (e) Select the rows where the animal is a cat and the age is less than 3. (f) Calculate the sum of all visits in df (i.e. the total number of visits). (g) Calculate the mean age for each different animal in df.

In [25]:

```
import pandas as pd
pd.__version__
pd.show_versions()
data = {'animal': ['cat', 'cat', 'snake', 'dog', 'dog',
'cat', 'snake' , 'cat', 'dog',
'dog'],
'age': [2.5, 3, 0.5, np.nan, 5, 2, 4.5, np.nan, 7, 3],
'visits': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'priority': ['yes', 'yes', 'no', 'yes', 'no', 'no', 'no
', 'ye s', 'no', 'no']}
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', '
j']
df = pd.DataFrame(data, index=labels)
```

```
INSTALLED VERSIONS
------------------
commit           : None
python           : 3.7.6.final.0
python-bits      : 64
OS               : Windows
OS-release       : 10
machine          : AMD64
processor        : Intel64 Family 6 Model 142 Stepping
10, GenuineIntel
byteorder        : little
LC_ALL           : None
LANG             : None
LOCALE           : None.None

pandas           : 1.0.1
numpy            : 1.18.1
pytz             : 2019.3
dateutil         : 2.8.1
pip              : 20.0.2
setuptools       : 45.2.0.post20200210
Cython           : 0.29.15
pytest           : 5.3.5
hypothesis       : 5.5.4
sphinx           : 2.4.0
blosc            : None
feather          : None
xlsxwriter       : 1.2.7
lxml.etree       : 4.5.0
html5lib         : 1.0.1
pymysql          : None
psycopg2         : None
jinja2           : 2.11.1
IPython          : 7.12.0
pandas_datareader: None
bs4              : 4.8.2
bottleneck       : 1.3.2
fastparquet      : None
gcsfs            : None
lxml.etree       : 4.5.0
matplotlib       : 3.1.3
numexpr          : 2.7.1
odfpy            : None
openpyxl         : 3.0.3
pandas_gbq       : None
pyarrow          : None
pytables         : None
pytest           : 5.3.5
pyxlsb           : None
s3fs             : None
scipy            : 1.4.1
sqlalchemy       : 1.3.13
tables           : 3.6.1
```

```
tabulate         : None
xarray           : None
xlrd             : 1.2.0
xlwt             : 1.3.0
xlsxwriter       : 1.2.7
numba            : 0.48.0
```

```
In [32]: df = pd.DataFrame(data, index=labels)
```

```
In [27]: df.describe()
```

Out[27]:

	age	visits
count	8.000000	10.000000
mean	3.437500	1.900000
std	2.007797	0.875595
min	0.500000	1.000000
25%	2.375000	1.000000
50%	3.000000	2.000000
75%	4.625000	2.750000
max	7.000000	3.000000

```
In [28]: df.iloc[:3]
```

Out[28]:

	animal	age	visits	priority
a	cat	2.5	1	yes
b	cat	3.0	3	yes
c	snake	0.5	2	no

In [29]:

```
df.loc[:, ['animal', 'age']]
```

Out[29]:

	animal	age
a	cat	2.5
b	cat	3.0
c	snake	0.5
d	dog	NaN
e	dog	5.0
f	cat	2.0
g	snake	4.5
h	cat	NaN
i	dog	7.0
j	dog	3.0

In [30]:

```
df.loc[df.index[[3, 4, 8]], ['animal', 'age']]
```

Out[30]:

	animal	age
d	dog	NaN
e	dog	5.0
i	dog	7.0

In [33]:

```
df[(df['animal'] == 'cat') & (df['age'] < 3)]
```

Out[33]:

	animal	age	visits	priority
a	cat	2.5	1	yes
f	cat	2.0	3	no

In [34]:

```
df['visits'].sum()
```

Out[34]:

```
19
```

In [35]:

```
df.groupby('animal')['age'].mean()
```

Out[35]:

```
animal
cat      2.5
dog      5.0
snake    2.5
Name: age, dtype: float64
```

In [ ]: