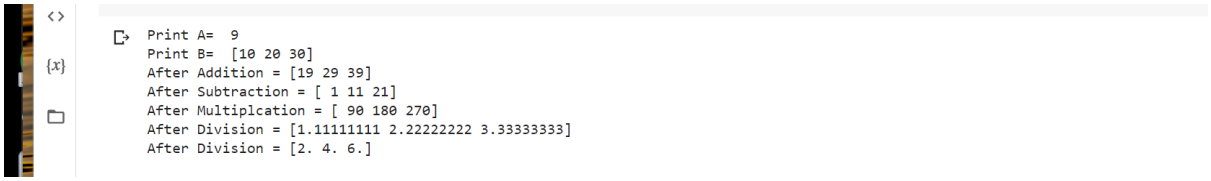
# NUMPY ARITHEMATIC OPERATIONS:

## PROGRAM-1

```
import numpy as np

a=np.array(9)

print("Print A= ",a)
b=np.array([10,20,30])
print("Print B= ", b)
addition= np.add(a,b)
print("After Addition =", addition)
sub=np.subtract(b,a)
print("After Subtraction =",sub)
mul=np.multiply (a,b)
print("After Multiplcation =",mul)
div=np.divide (b,a)
print("After Division =",div)
div1=np.divide (b,5)
print("After Division =",div1)
```

## OUTPUT

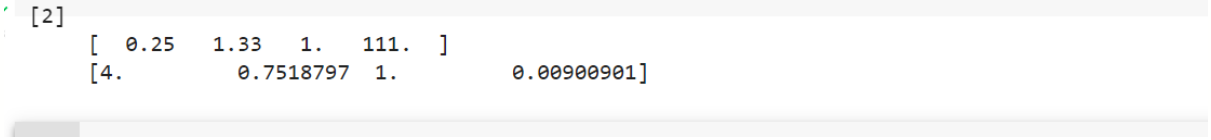


```
Print A=  9
Print B=  [10 20 30]
After Addition = [19 29 39]
After Subtraction = [ 1 11 21]
After Multiplcation = [ 90 180 270]
After Division = [1.11111111 2.22222222 3.33333333]
After Division = [2. 4. 6.]
```

## PROGRAM-2

```
a= np.array([0.25,1.33,1,111])
print(a)
rec=np.reciprocal(a)
print(rec)
```

## OUTPUT



```
[2]
      [  0.25   1.33   1.   111.  ]
      [4.         0.7518797  1.         0.00900901]
```

## PROGRAM-3

```
a=np.array([10,100,1000])

print(a)
pow=np.power(a,2)
print("after ^2 = ", pow)
b=np.array([2,3,1])
pow1=np.power(a,b)
print("after b array elements as ^ = ", pow1)
```

## OUTPUT

```
[  10  100 1000]
after ^2 =  [    100   10000 1000000]
after b array elements as ^ =  [    100 1000000    1000]
```

## PROGRAM-4

```
a= np.array ([10,20,30])

b= np.array ([3,5,7])
print("values of A=", a)
print("values of B=", b)
mm=np.mod(a,b)
rm=np.remainder(a,b)
print("values of MOD=", mm)
print("values of REMAINDER=", rm)
```

## OUTPUT

```
values of A= [10 20 30]
values of B= [3 5 7]
values of MOD= [1 0 2]
values of REMAINDER= [1 0 2]
```

## PROGRAM-5

```
 a=np.array([-5.6j, 0.2j,11, 1+1j])

print(a)
print("real=",np.real(a))
print("imaginary=",np.imag(a))
print("Conjugate=", np.conj(a))
```

## OUTPUT

```
[ ]  a=np.array ([1.0,5.3,123,0.56,25.5,32])

     print(a)

     print("after rounding up=", np.around(a))
```

0s completed at 6:34 PM

## PROGRAM-6

```
a=np.array ([1.0,5.3,123,0.56,25.5,32])

print(a)
print("after rounding up=", np.around(a))
print("after rounding up to 1st decimal value =", np.around(a,decimals=1))
```

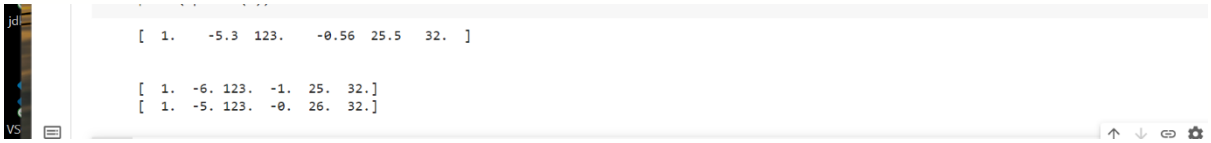
## OUTPUT

```
[  1.    5.3  123.    0.56  25.5   32.  ]
after rounding up= [  1.   5. 123.   1.  26.  32.]
after rounding up to 1st decimal value = [  1.    5.3 123.    0.6  25.5  32. ]
```

# PROGRAM-7

```
a=np.array ([1.0,-5.3,123,-0.56,25.5,32])

print(a)
print('\n')
print(np.floor(a))
print(np.ceil(a))
```

# OUTPUT



# PANDAS:

## PROGRAM-1

```
import pandas as pd

import numpy as np

s = pd.Series(np.random.randn(4))
print (s)
```

## OUTPUT

```
0    0.640962
1    0.448861
2    0.765249
3   -0.058867
dtype: float64
```

## PROGRAM-2

```
import pandas as pd
import numpy as np
s = pd.Series(np.random.randn(4))
print ("The axes are:")
print (s.axes)
```

## OUTPUT

```
The axes are:
[RangeIndex(start=0, stop=4, step=1)]
```

## PROGRAM-3

```
import pandas as pd
import numpy as np

s = pd.Series(np.random.randn(4))
print ("Is the Object empty?")
print (s.empty)
```

## OUTPUT

```
Is the Object empty?
False
```

## PROGRAM-4

```
import pandas as pd
import numpy as np
s = pd.Series(np.random.randn(4))
print (s)

print ("The dimensions of the object:")
print (s.ndim)
```

## OUTPUT

```
0   -0.043565
1    0.139736
2   -0.033096
3   -0.470011
dtype: float64
The dimensions of the object:
1
```

## PROGRAM-5

```
import pandas as pd
import numpy as np
s = pd.Series(np.random.randn(2))
print (s)
print ("The size of the object:")
print (s.size)
```

## OUTPUT

```
0   -0.915215
1    0.250136
dtype: float64
The size of the object:
2
```

## PROGRAM-6

```
import pandas as pd
import numpy as np
s = pd.Series(np.random.randn(4))
print (s)

print ("The actual data series is:")
print (s.values)
```

## OUTPUT

```
0    0.140421
1   -1.173498
2    1.355218
3   -0.133246
dtype: float64
The actual data series is:
[ 0.14042064 -1.17349845  1.35521819 -0.13324604]
```

## PROGRAM-7

```
import pandas as pd
import numpy as np
s = pd.Series(np.random.randn(4))
print ("The original series is:")
print (s)

print ("The first two rows of the data series:")
print (s.head(2))
```

## OUTPUT

```
The original series is:
0   -0.572345
1    1.143168
2    0.547497
3   -0.486110
dtype: float64
The first two rows of the data series:
0   -0.572345
1    1.143168
dtype: float64
```

## PROGRAM-8

```
import pandas as pd
import numpy as np
s = pd.Series(np.random.randn(4))
print ("The original series is:")
print (s)

print ("The last two rows of the data series:")
print (s.tail(2))
```

## OUTPUT

```
The original series is:
0   -1.097691
1   -0.436543
2   -0.535559
3   -0.938054
dtype: float64
The last two rows of the data series:
2   -0.535559
3   -0.938054
dtype: float64
```

## PROGRAM-9

```
import pandas as pd
import numpy as np
d = {'Name':pd.Series(['Tom','James','Ricky','Vin','Steve','Smith','Jack']),
   'Age':pd.Series([25,26,25,23,30,29,23]),
   'Rating':pd.Series([4.23,3.24,3.98,2.56,3.20,4.6,3.8])}
df = pd.DataFrame(d)
print ("Our data series is:")
print (df)
```

## OUTPUT

```
Our data series is:
    Name  Age  Rating
0    Tom   25    4.23
1  James   26    3.24
2  Ricky   25    3.98
3    Vin   23    2.56
4  Steve   30    3.20
5  Smith   29    4.60
6   Jack   23    3.80
```

## PROGRAM-10

```
import pandas as pd
import numpy as np
d = {'Name':pd.Series(['Tom','James','Ricky','Vin','Steve','Smith','Jack']),
   'Age':pd.Series([25,26,25,23,30,29,23]),
   'Rating':pd.Series([4.23,3.24,3.98,2.56,3.20,4.6,3.8])}
df = pd.DataFrame(d)
print ("Row axis labels and column axis labels are:")
print (df.axes)
```

## OUTPUT

```
Row axis labels and column axis labels are:
[RangeIndex(start=0, stop=7, step=1), Index(['Name', 'Age', 'Rating'], dtype='object')]
```

## PROGRAM-11

```
import pandas as pd
import numpy as np
d = {'Name':pd.Series(['Tom','James','Ricky','Vin','Steve','Smith','Jack']),
   'Age':pd.Series([25,26,25,23,30,29,23]),
   'Rating':pd.Series([4.23,3.24,3.98,2.56,3.20,4.6,3.8])}
df = pd.DataFrame(d)
print ("The data types of each column are:")
print (df.dtypes)
```

## OUTPUT

```
The data types of each column are:
Name       object
Age         int64
Rating    float64
dtype: object
```

## PROGRAM-12

```
import pandas as pd
import numpy as np
d = {'Name':pd.Series(['Tom','James','Ricky','Vin','Steve','Smith','Jack']),
   'Age':pd.Series([25,26,25,23,30,29,23]),
   'Rating':pd.Series([4.23,3.24,3.98,2.56,3.20,4.6,3.8])}
df = pd.DataFrame(d)
print ("Is the object empty?")
print (df.empty)
```

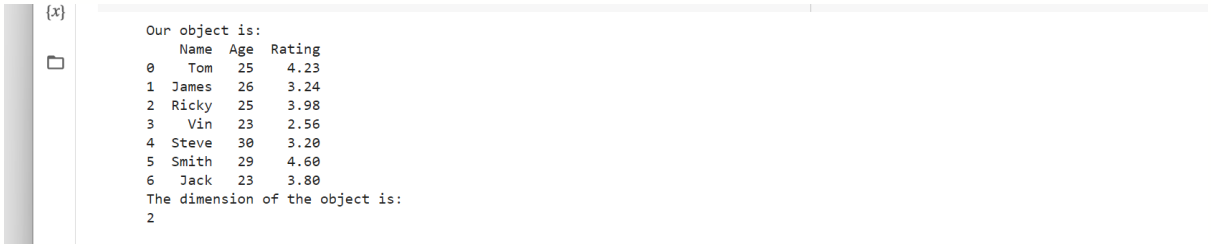
## OUTPUT

```
Is the object empty?
False
```

## PROGRAM-13

```
import pandas as pd
import numpy as np
d = {'Name':pd.Series(['Tom','James','Ricky','Vin','Steve','Smith','Jack']),
   'Age':pd.Series([25,26,25,23,30,29,23]),
   'Rating':pd.Series([4.23,3.24,3.98,2.56,3.20,4.6,3.8])}

df = pd.DataFrame(d)
print ("Our object is:")
print (df)
print ("The dimension of the object is:")
print (df.ndim)
```

## OUTPUT

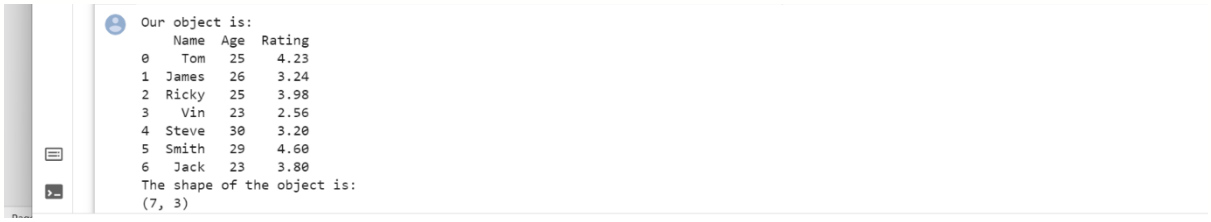


```
Our object is:
    Name  Age  Rating
0    Tom   25    4.23
1  James   26    3.24
2  Ricky   25    3.98
3    Vin   23    2.56
4  Steve   30    3.20
5  Smith   29    4.60
6   Jack   23    3.80
The dimension of the object is:
2
```

## PROGRAM-14

```
import pandas as pd
import numpy as np
d = {'Name':pd.Series(['Tom','James','Ricky','Vin','Steve','Smith','Jack']),
   'Age':pd.Series([25,26,25,23,30,29,23]),
   'Rating':pd.Series([4.23,3.24,3.98,2.56,3.20,4.6,3.8])}

df = pd.DataFrame(d)
print ("Our object is:")
print (df)
print ("The shape of the object is:")
print (df.shape)
```

## OUTPUT

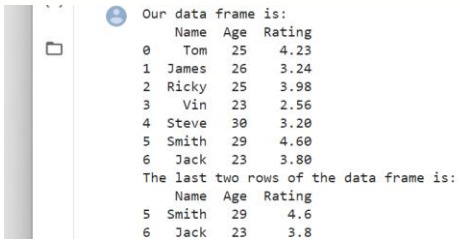


```
Our object is:
    Name  Age  Rating
0    Tom   25    4.23
1  James   26    3.24
2  Ricky   25    3.98
3    Vin   23    2.56
4  Steve   30    3.20
5  Smith   29    4.60
6   Jack   23    3.80
The shape of the object is:
(7, 3)
```

# PROGRAM-15

```
import pandas as pd

import numpy as np
d = {'Name':pd.Series(['Tom','James','Ricky','Vin','Steve','Smith','Jack']),
   'Age':pd.Series([25,26,25,23,30,29,23]),
   'Rating':pd.Series([4.23,3.24,3.98,2.56,3.20,4.6,3.8])}

df = pd.DataFrame(d)
print ("Our data frame is:")
print (df)
print ("The last two rows of the data frame is:")
print (df.tail(2))
```

# OUTPUT



```
Our data frame is:
    Name  Age  Rating
0    Tom   25    4.23
1  James   26    3.24
2  Ricky   25    3.98
3    Vin   23    2.56
4  Steve   30    3.20
5  Smith   29    4.60
6   Jack   23    3.80
The last two rows of the data frame is:
    Name  Age  Rating
5  Smith   29     4.6
6   Jack   23     3.8
```

# DATA FRAMES:

## PROGRAM-1

```
import pandas as pd
data = [1,2,3,4,5]
df = pd.DataFrame(data)
print (df)
```

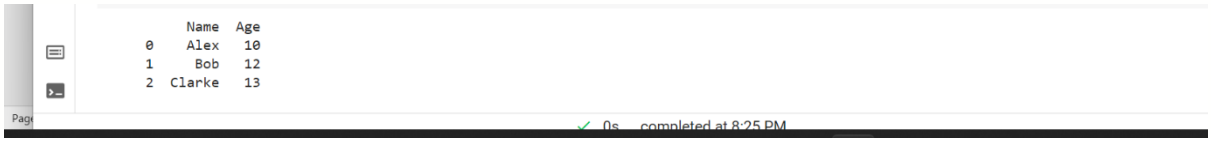
## OUTPUT

```
   0
0  1
1  2
2  3
3  4
4  5
```

## PROGRAM-2

```
import pandas as pd
data = [['Alex',10],['Bob',12],['Clarke',13]]
df = pd.DataFrame(data,columns=['Name','Age'])
print (df)
```

## OUTPUT



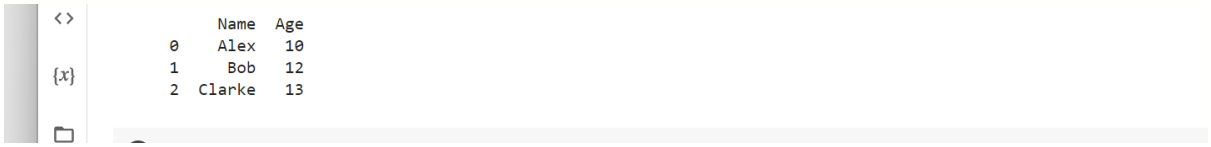
```
     Name  Age
0    Alex   10
1     Bob   12
2  Clarke   13
```

0s completed at 8:25 PM

## PROGRAM-3

```
import pandas as pd

data = [['Alex',10],['Bob',12],['Clarke',13]]
df = pd.DataFrame(data,columns=['Name','Age'])
print (df)
```

## OUTPUT



```
     Name  Age
0    Alex   10
1     Bob   12
2  Clarke   13
```

## PROGRAM-4

```
import pandas as pd
data = {'Name':['Tom', 'Jack', 'Steve', 'Ricky'],
'Age':[28,34,29,42]}
df = pd.DataFrame(data)
print (df)
```

## OUTPUT

```
    Name  Age
0    Tom   28
1   Jack   34
2  Steve   29
3  Ricky   42
```

## PROGRAM-5

```
import pandas as pd
data = {'Name':['Tom', 'Jack', 'Steve', 'Ricky'],'Age':[28,34,29,42]}
df = pd.DataFrame(data, index=['rank1','rank2','rank3','rank4'])
print (df)
```

## OUTPUT

```
          Name  Age
rank1    Tom   28
rank2   Jack   34
rank3  Steve   29
rank4  Ricky   42
```

## PROGRAM-6

```
import pandas as pd
data = [{'a': 1, 'b': 2},{'a': 5, 'b': 10, 'c': 20}]
df = pd.DataFrame(data)
print (df)
```

## OUTPUT

```
[ ] import pandas as pd
    data = [{'a': 1, 'b': 2},{'a': 5, 'b': 10, 'c': 20}]
    df = pd.DataFrame(data)
    print (df)

   a   b     c
0  1   2   NaN
1  5  10  20.0
```

## PROGRAM-7

```
import pandas as pd

data = [{'a': 1, 'b': 2},{'a': 5, 'b': 10, 'c': 20}]

df1 = pd.DataFrame(data, index=['first', 'second'], columns=['a', 'b'])
df2 = pd.DataFrame(data, index=['first', 'second'], columns=['a', 'b1'])
print (df1)
print (df2)
```

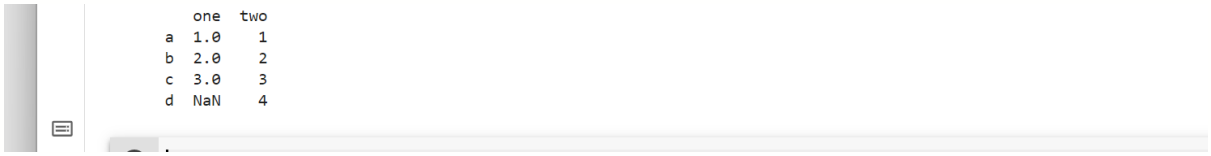
## OUTPUT

```
        a   b
first   1   2
second  5  10
        a  b1
first   1 NaN
second  5 NaN
```

## PROGRAM-8

```
import pandas as pd

d = {'one' : pd.Series([1, 2, 3], index=['a', 'b', 'c']),
   'two' : pd.Series([1, 2, 3, 4], index=['a', 'b', 'c', 'd'])}

df = pd.DataFrame(d)
print (df)
```

## OUTPUT

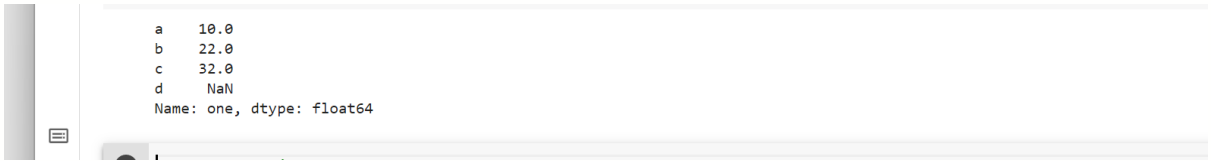


```
   one  two
a  1.0    1
b  2.0    2
c  3.0    3
d  NaN    4
```

## PROGRAM-9

```
import pandas as pd

d = {'one' : pd.Series([10, 22, 32], index=['a', 'b', 'c']),
   'two' : pd.Series([1, 2, 3, 4], index=['a', 'b', 'c', 'd'])}

df = pd.DataFrame(d)
print (df['one'])
```

## OUTPUT



```
a    10.0
b    22.0
c    32.0
d     NaN
Name: one, dtype: float64
```

## PROGRAM-10

```
import pandas as pd

de = {'one' : pd.Series([10, 22, 32,44], index=['a', 'b', 'c','d']),
   'two' : pd.Series([1, 2, 3, 4], index=['a', 'b', 'c', 'd'])}


df = pd.DataFrame(de)
print (df['one'])
```

## OUTPUT

```
a    10
b    22
c    32
d    44
Name: one, dtype: int64
```

## PROGRAM-11

```
import pandas as pd

d = {'one' : pd.Series([1, 2, 3], index=['a', 'b', 'c']),
   'two' : pd.Series([1, 2, 3, 4], index=['a', 'b', 'c', 'd'])}

df = pd.DataFrame(d)

print (df)

print ("Adding a new column by passing as Series:")
df['three']=pd.Series([10,20,30],index=['a','b','c'])
print (df)

print ("Adding a new column using the existing columns in DataFrame:")
df['four']=df['two']+df['three']

print (df)
```

## OUTPUT

```
   one  two
a  1.0    1
b  2.0    2
c  3.0    3
d  NaN    4
Adding a new column by passing as Series:
   one  two  three
a  1.0    1   10.0
b  2.0    2   20.0
c  3.0    3   30.0
d  NaN    4    NaN
Adding a new column using the existing columns in DataFrame:
   one  two  three  four
a  1.0    1   10.0  11.0
b  2.0    2   20.0  22.0
c  3.0    3   30.0  33.0
d  NaN    4    NaN   NaN
```

## PROGRAM-12

```
import pandas as pd

d = {'one' : pd.Series([1, 2, 3], index=['a', 'b', 'c']),
   'two' : pd.Series([1, 2, 3, 4], index=['a', 'b', 'c', 'd']),
   'three' : pd.Series([10,20,30], index=['a','b','c'])}

df = pd.DataFrame(d)
print ("Our dataframe is:")
print (df)
```

```
print ("Deleting the first column using DEL function:")
del df['one']
print (df)
print ("Deleting another column using POP function:")
df.pop('two')
print (df)
```

## OUTPUT

```
Our dataframe is:
   one  two  three
a  1.0    1   10.0
b  2.0    2   20.0
c  3.0    3   30.0
d  NaN    4    NaN
Deleting the first column using DEL function:
   two  three
a    1   10.0
b    2   20.0
c    3   30.0
d    4    NaN
Deleting another column using POP function:
   three
a   10.0
b   20.0
c   30.0
d    NaN
```

## PROGRAM-13

```
import pandas as pd

d = {'one' : pd.Series([1, 2, 3], index=['a', 'b', 'c']),
   'two' : pd.Series([1, 2, 3, 4], index=['a', 'b', 'c', 'd'])}

df = pd.DataFrame(d)
print (df.loc['b'])
```

## PROGRAM-14

```
import pandas as pd

d = {'one' : pd.Series([1, 2, 3], index=['a', 'b', 'c']),
   'two' : pd.Series([1, 2, 3, 4], index=['a', 'b', 'c', 'd'])}

df = pd.DataFrame(d)
print (df.iloc[2])
```

## OUTPUT

```
one    2.0
two    2.0
Name: b, dtype: float64
```

## PROGRAM-15

```
import pandas as pd

d = {'one' : pd.Series([1, 2, 3], index=['a', 'b', 'c']),
   'two' : pd.Series([1, 2, 3, 4], index=['a', 'b', 'c', 'd'])}

df = pd.DataFrame(d)
print (df[2:4])
```

## OUTPUT

```
one    3.0
two    3.0
Name: c, dtype: float64
```

## PROGRAM-16

```
import pandas as pd

df = pd.DataFrame([[1, 2], [3, 4]], columns = ['a','b'])
df2 = pd.DataFrame([[5, 6], [7, 8]], columns = ['a','b'])
print (df)
print("after appending")
df = df.append(df2)
print (df)
```

## OUTPUT

```
   a  b
0  1  2
1  3  4
after appending
   a  b
0  1  2
1  3  4
0  5  6
1  7  8
```

## PROGRAM-17

```
import pandas as pd

df = pd.DataFrame([[1, 2], [3, 4]], columns = ['a','b'])
df2 = pd.DataFrame([[5, 6], [7, 8]], columns = ['a','b'])

df = df.append(df2)

df = df.drop(0)

print (df)
```

## OUTPUT

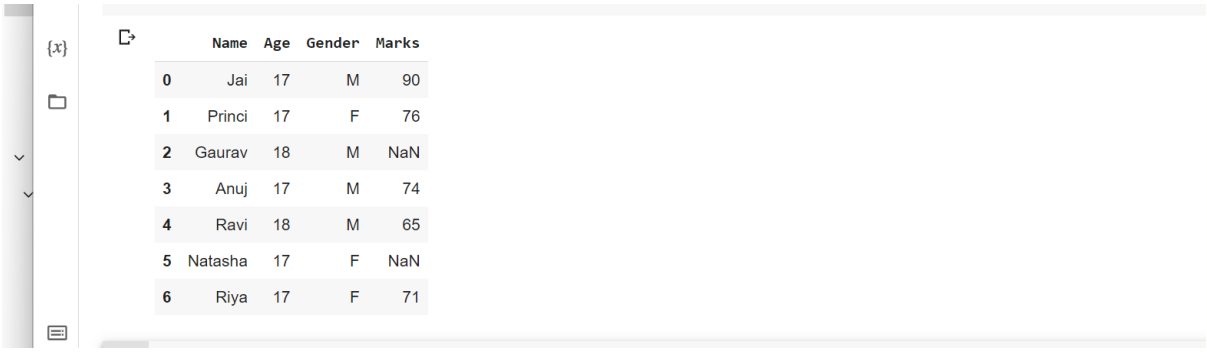
```
   a  b
1  3  4
1  7  8
```

# DATA WRANGLING:

## PROGRAM-1

```
import pandas as pd
data = {'Name': ['Jai', 'Princi', 'Gaurav','Anuj', 'Ravi', 'Natasha', 'Riya'],
   'Age': [17, 17, 18, 17, 18, 17, 17],
   'Gender': ['M', 'F', 'M', 'M', 'M', 'F', 'F'],
   'Marks': [90, 76, 'NaN', 74, 65, 'NaN', 71]}
df = pd.DataFrame(data)
Df
```

## OUTPUT

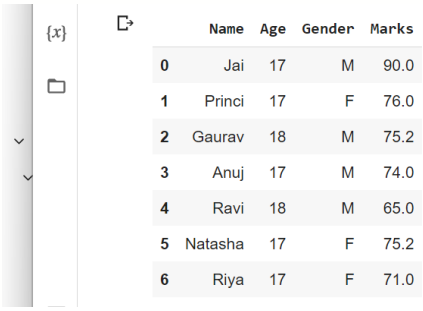


| | Name | Age | Gender | Marks |
|---|---|---|---|---|
| 0 | Jai | 17 | M | 90 |
| 1 | Princi | 17 | F | 76 |
| 2 | Gaurav | 18 | M | NaN |
| 3 | Anuj | 17 | M | 74 |
| 4 | Ravi | 18 | M | 65 |
| 5 | Natasha | 17 | F | NaN |
| 6 | Riya | 17 | F | 71 |

## PROGRAM-2

```
c = avg = 0
for ele in df['Marks']:
  if str(ele).isnumeric():
    c += 1
    avg += ele
avg /= c

df = df.replace(to_replace="NaN",
        value=avg)
Df
```

## OUTPUT

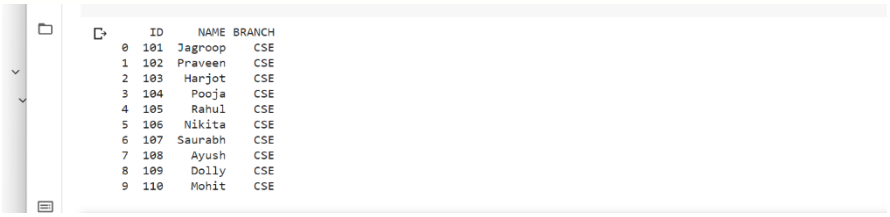


| | Name | Age | Gender | Marks |
|---|---|---|---|---|
| 0 | Jai | 17 | M | 90.0 |
| 1 | Princi | 17 | F | 76.0 |
| 2 | Gaurav | 18 | M | 75.2 |
| 3 | Anuj | 17 | M | 74.0 |
| 4 | Ravi | 18 | M | 65.0 |
| 5 | Natasha | 17 | F | 75.2 |
| 6 | Riya | 17 | F | 71.0 |

## PROGRAM-3

```
import pandas as pd
details = pd.DataFrame({
  'ID': [101, 102, 103, 104, 105, 106,107, 108, 109, 110],
  'NAME': ['Jagroop', 'Praveen', 'Harjot','Pooja', 'Rahul', 'Nikita', 'Saurabh', 'Ayush', 'Dolly', "Mohit"],
  'BRANCH': ['CSE', 'CSE', 'CSE', 'CSE', 'CSE','CSE', 'CSE', 'CSE', 'CSE', 'CSE']})
print(details)
```

## OUTPUT

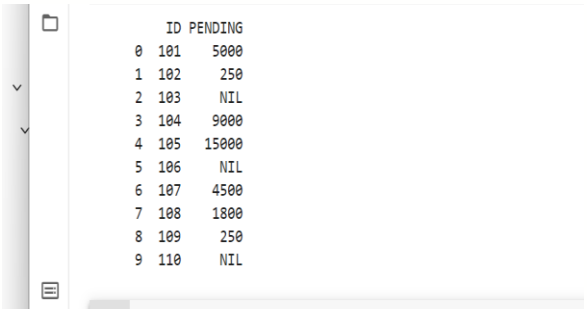


| | ID | NAME | BRANCH |
|---|---|---|---|
| 0 | 101 | Jagroop | CSE |
| 1 | 102 | Praveen | CSE |
| 2 | 103 | Harjot | CSE |
| 3 | 104 | Pooja | CSE |
| 4 | 105 | Rahul | CSE |
| 5 | 106 | Nikita | CSE |
| 6 | 107 | Saurabh | CSE |
| 7 | 108 | Ayush | CSE |
| 8 | 109 | Dolly | CSE |
| 9 | 110 | Mohit | CSE |

## PROGRAM-4

```
import pandas as pd
fees_status = pd.DataFrame(
  {'ID': [101, 102, 103, 104, 105,106, 107, 108, 109, 110],
  'PENDING': ['5000', '250', 'NIL','9000', '15000', 'NIL','4500', '1800', '250', 'NIL']})
print(fees_status)
```

## OUTPUT

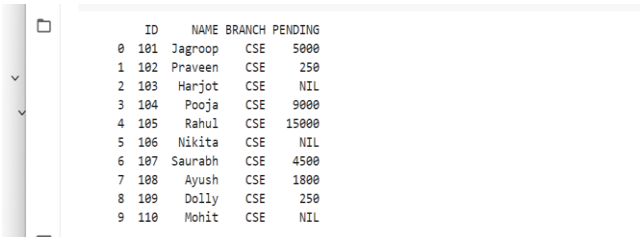


| | ID | PENDING |
|---|---|---|
| 0 | 101 | 5000 |
| 1 | 102 | 250 |
| 2 | 103 | NIL |
| 3 | 104 | 9000 |
| 4 | 105 | 15000 |
| 5 | 106 | NIL |
| 6 | 107 | 4500 |
| 7 | 108 | 1800 |
| 8 | 109 | 250 |
| 9 | 110 | NIL |

## PROGRAM-5

```
import pandas as pd
details = pd.DataFrame({
  'ID': [101, 102, 103, 104, 105,106, 107, 108, 109, 110],
  'NAME': ['Jagroop', 'Praveen', 'Harjot','Pooja', 'Rahul', 'Nikita','Saurabh', 'Ayush', 'Dolly', "Mohit"],
  'BRANCH': ['CSE', 'CSE', 'CSE', 'CSE', 'CSE', 'CSE', 'CSE', 'CSE', 'CSE', 'CSE']})

fees_status = pd.DataFrame(
  {'ID': [101, 102, 103, 104, 105,106, 107, 108, 109, 110],
  'PENDING': ['5000', '250', 'NIL','9000', '15000', 'NIL', '4500', '1800', '250', 'NIL']})

print(pd.merge(details, fees_status, on='ID'))
```

## OUTPUT



| | ID | NAME | BRANCH | PENDING |
|---|---|---|---|---|
| 0 | 101 | Jagroop | CSE | 5000 |
| 1 | 102 | Praveen | CSE | 250 |
| 2 | 103 | Harjot | CSE | NIL |
| 3 | 104 | Pooja | CSE | 9000 |
| 4 | 105 | Rahul | CSE | 15000 |
| 5 | 106 | Nikita | CSE | NIL |
| 6 | 107 | Saurabh | CSE | 4500 |
| 7 | 108 | Ayush | CSE | 1800 |
| 8 | 109 | Dolly | CSE | 250 |
| 9 | 110 | Mohit | CSE | NIL |

## PROGRAM-6

```
import pandas as pd
car_selling_data = {'Brand': ['Maruti', 'Maruti', 'Maruti','Maruti', 'Hyundai', 'Hyundai','Toyota', 'Mahindra', '
Mahindra', 'Ford', 'Toyota', 'Ford'],
        'Year': [2010, 2011, 2009, 2013,2010, 2011, 2011, 2010,2013, 2010, 2010, 2011],
        'Sold': [6, 7, 9, 8, 3, 5,2, 8, 7, 2, 4, 2]}

df = pd.DataFrame(car_selling_data)
print(df)
```

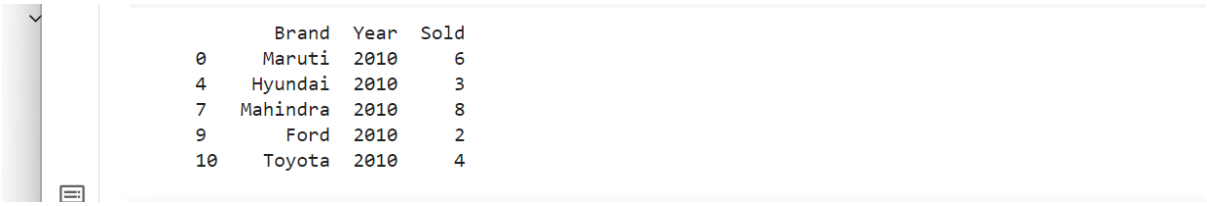
## OUTPUT

| | Brand | Year | Sold |
|---|---|---|---|
| 0 | Maruti | 2010 | 6 |
| 1 | Maruti | 2011 | 7 |
| 2 | Maruti | 2009 | 9 |
| 3 | Maruti | 2013 | 8 |
| 4 | Hyundai | 2010 | 3 |
| 5 | Hyundai | 2011 | 5 |
| 6 | Toyota | 2011 | 2 |
| 7 | Mahindra | 2010 | 8 |
| 8 | Mahindra | 2013 | 7 |
| 9 | Ford | 2010 | 2 |
| 10 | Toyota | 2010 | 4 |
| 11 | Ford | 2011 | 2 |

## PROGRAM-7

```
import pandas as pd
car_selling_data = {'Brand': ['Maruti', 'Maruti', 'Maruti','Maruti', 'Hyundai', 'Hyundai','Toyota', 'Mahindra', '
Mahindra', 'Ford', 'Toyota', 'Ford'],
        'Year': [2010, 2011, 2009, 2013,2010, 2011, 2011, 2010,2013, 2010, 2010, 2011],
        'Sold': [6, 7, 9, 8, 3, 5,2, 8, 7, 2, 4, 2]}

df = pd.DataFrame(car_selling_data)

grouped = df.groupby('Year')
print(grouped.get_group(2010))
```

## OUTPUT



| | Brand | Year | Sold |
|---|---|---|---|
| 0 | Maruti | 2010 | 6 |
| 4 | Hyundai | 2010 | 3 |
| 7 | Mahindra | 2010 | 8 |
| 9 | Ford | 2010 | 2 |
| 10 | Toyota | 2010 | 4 |

# PROGRAM-8

```
import pandas as pd
student_data = {'Name': ['Amit', 'Praveen', 'Jagroop','Rahul', 'Vishal', 'Suraj', 'Rishab', 'Satyapal', 'Amit',
 'Rahul', 'Praveen', 'Amit'],

      'Roll_no': [23, 54, 29, 36, 59, 38,12, 45, 34, 36, 54, 23],

      'Email': ['xxxx@gmail.com', 'xxxxxx@gmail.com', 'xxxxxx@gmail.com', 'xx@gmail.com',
 'xxxx@gmail.com', 'xxxxx@gmail.com', 'xxxxx@gmail.com', 'xxxxx@gmail.com','xxxxx@gmail.com', '
xxxxxx@gmail.com', 'xxxxxxxxxx@gmail.com', 'xxxxxxxxxx@gmail.com']}

df = pd.DataFrame(student_data)

print(df)
```

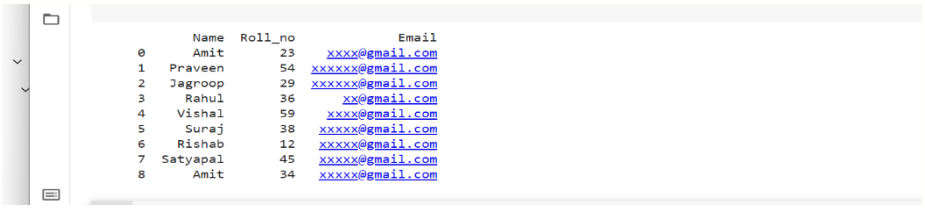
# OUTPUT

| | Name | Roll_no | Email |
|---|---|---|---|
| 0 | Amit | 23 | xxxx@gmail.com |
| 1 | Praveen | 54 | xxxxxx@gmail.com |
| 2 | Jagroop | 29 | xxxxxx@gmail.com |
| 3 | Rahul | 36 | xx@gmail.com |
| 4 | Vishal | 59 | xxxx@gmail.com |
| 5 | Suraj | 38 | xxxxx@gmail.com |
| 6 | Rishab | 12 | xxxxx@gmail.com |
| 7 | Satyapal | 45 | xxxxx@gmail.com |
| 8 | Amit | 34 | xxxxx@gmail.com |
| 9 | Rahul | 36 | xxxxxx@gmail.com |
| 10 | Praveen | 54 | xxxxxxxxxx@gmail.com |
| 11 | Amit | 23 | xxxxxxxxxx@gmail.com |

# PROGRAM-9

```
import pandas as pd
student_data = {'Name': ['Amit', 'Praveen', 'Jagroop','Rahul', 'Vishal', 'Suraj', 'Rishab', 'Satyapal', 'Amit',
 'Rahul', 'Praveen', 'Amit'],

      'Roll_no': [23, 54, 29, 36, 59, 38,12, 45, 34, 36, 54, 23],

      'Email': ['xxxx@gmail.com', 'xxxxxx@gmail.com', 'xxxxxx@gmail.com', 'xx@gmail.com',
 'xxxx@gmail.com', 'xxxxx@gmail.com', 'xxxxx@gmail.com', 'xxxxx@gmail.com','xxxxx@gmail.com', '
xxxxxx@gmail.com', 'xxxxxxxxxx@gmail.com', 'xxxxxxxxxx@gmail.com']}

df = pd.DataFrame(student_data)
non_duplicate = df[~df.duplicated('Roll_no')]
print(non_duplicate)
```

# OUTPUT



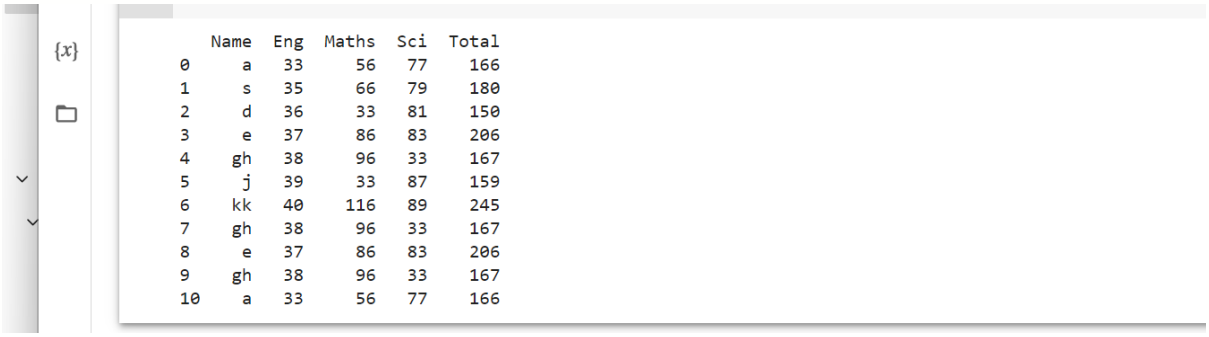
| | Name | Roll_no | Email |
|---|---|---|---|
| 0 | Amit | 23 | xxxx@gmail.com |
| 1 | Praveen | 54 | xxxxxx@gmail.com |
| 2 | Jagroop | 29 | xxxxxx@gmail.com |
| 3 | Rahul | 36 | xx@gmail.com |
| 4 | Vishal | 59 | xxxx@gmail.com |
| 5 | Suraj | 38 | xxxxx@gmail.com |
| 6 | Rishab | 12 | xxxxx@gmail.com |
| 7 | Satyapal | 45 | xxxxx@gmail.com |
| 8 | Amit | 34 | xxxxx@gmail.com |

# PANDAS-DATA FILTERING:

## PROGRAM-1

```
import pandas as pd
d=pd.read_excel("C:\\Users\Shabnam\Desktop\duplicate.xlsx")
df=pd.DataFrame(d)
print(df)
```

## OUTPUT

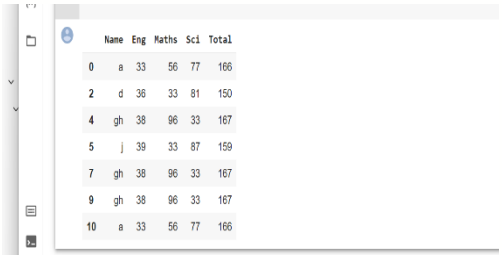


| | Name | Eng | Maths | Sci | Total |
|---|---|---|---|---|---|
| 0 | a | 33 | 56 | 77 | 166 |
| 1 | s | 35 | 66 | 79 | 180 |
| 2 | d | 36 | 33 | 81 | 150 |
| 3 | e | 37 | 86 | 83 | 206 |
| 4 | gh | 38 | 96 | 33 | 167 |
| 5 | j | 39 | 33 | 87 | 159 |
| 6 | kk | 40 | 116 | 89 | 245 |
| 7 | gh | 38 | 96 | 33 | 167 |
| 8 | e | 37 | 86 | 83 | 206 |
| 9 | gh | 38 | 96 | 33 | 167 |
| 10 | a | 33 | 56 | 77 | 166 |

## PROGRAM-2

```
df.loc[df['Total']<170]
```

## OUTPUT

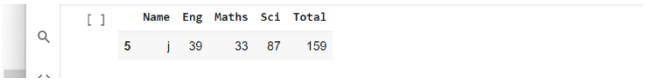


| | Name | Eng | Maths | Sci | Total |
|---|---|---|---|---|---|
| 0 | a | 33 | 56 | 77 | 166 |
| 2 | d | 36 | 33 | 81 | 150 |
| 4 | gh | 38 | 96 | 33 | 167 |
| 5 | j | 39 | 33 | 87 | 159 |
| 7 | gh | 38 | 96 | 33 | 167 |
| 9 | gh | 38 | 96 | 33 | 167 |
| 10 | a | 33 | 56 | 77 | 166 |

## PROGRAM-3

```
df.loc[(df['Eng']> 38) & (df['Maths']<50)]
```

## OUTPUT



| | Name | Eng | Maths | Sci | Total |
|---|---|---|---|---|---|
| 5 | j | 39 | 33 | 87 | 159 |

## PROGRAM-4

```
df.loc[(df['Eng']> 38) | (df['Maths']<50)]
```

## OUTPUT

| | Name | Eng | Maths | Sci | Total |
|---|---|---|---|---|---|
| 2 | d | 36 | 33 | 81 | 150 |
| 5 | j | 39 | 33 | 87 | 159 |
| 6 | kk | 40 | 116 | 89 | 245 |

# PROGRAM-5

```
df.loc[df['Name'].str.contains("k")]
```

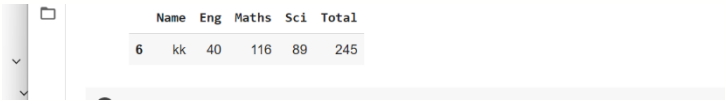
# OUTPUT

| | Name | Eng | Maths | Sci | Total |
|---|---|---|---|---|---|
| 6 | kk | 40 | 116 | 89 | 245 |

# PROGRAM-6

```
df.loc[df['Name'].str.startswith("k")]
```

# OUTPUT

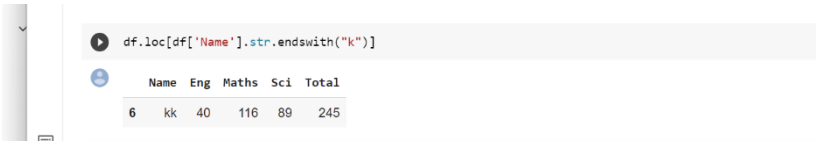


| | Name | Eng | Maths | Sci | Total |
|---|---|---|---|---|---|
| 6 | kk | 40 | 116 | 89 | 245 |

# PROGRAM-7

```
df.loc[df['Name'].str.endswith("k")]
```

# OUTPUT



```
df.loc[df['Name'].str.endswith("k")]
```

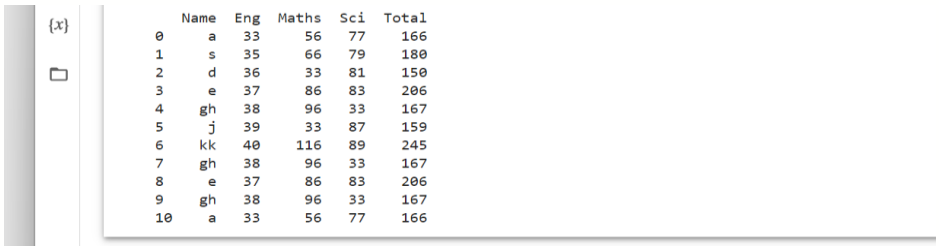
| | Name | Eng | Maths | Sci | Total |
|---|---|---|---|---|---|
| 6 | kk | 40 | 116 | 89 | 245 |

# PANDAS-DUPLICATE

## PROGRAM-1

```
import pandas as pd
d=pd.read_excel("C:\\Users\Shabnam\Desktop\duplicate.xlsx")
df=pd.DataFrame(d)
print(df)
```

## OUTPUT

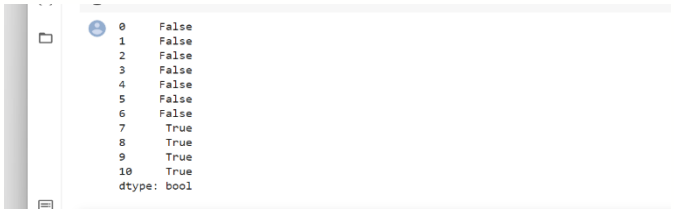


| | Name | Eng | Maths | Sci | Total |
|---|---|---|---|---|---|
| 0 | a | 33 | 56 | 77 | 166 |
| 1 | s | 35 | 66 | 79 | 180 |
| 2 | d | 36 | 33 | 81 | 150 |
| 3 | e | 37 | 86 | 83 | 206 |
| 4 | gh | 38 | 96 | 33 | 167 |
| 5 | j | 39 | 33 | 87 | 159 |
| 6 | kk | 40 | 116 | 89 | 245 |
| 7 | gh | 38 | 96 | 33 | 167 |
| 8 | e | 37 | 86 | 83 | 206 |
| 9 | gh | 38 | 96 | 33 | 167 |
| 10 | a | 33 | 56 | 77 | 166 |

## PROGRAM-2

```
df.duplicated()
```

## OUTPUT

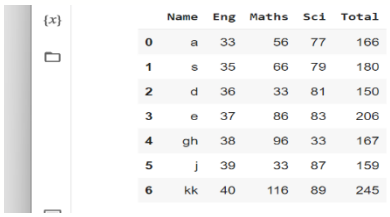


```
0    False
1    False
2    False
3    False
4    False
5    False
6    False
7     True
8     True
9     True
10    True
dtype: bool
```

## PROGRAM-3

```
df.drop_duplicates()
```

## OUTPUT

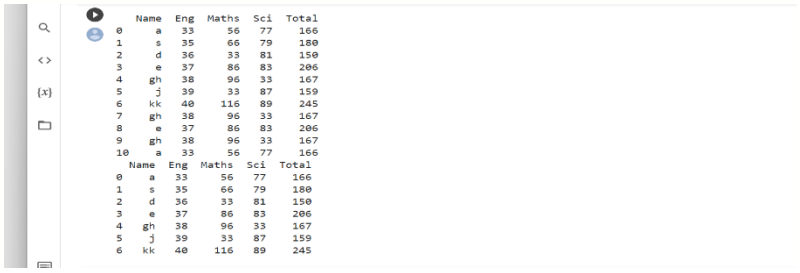


| | Name | Eng | Maths | Sci | Total |
|---|---|---|---|---|---|
| 0 | a | 33 | 56 | 77 | 166 |
| 1 | s | 35 | 66 | 79 | 180 |
| 2 | d | 36 | 33 | 81 | 150 |
| 3 | e | 37 | 86 | 83 | 206 |
| 4 | gh | 38 | 96 | 33 | 167 |
| 5 | j | 39 | 33 | 87 | 159 |
| 6 | kk | 40 | 116 | 89 | 245 |

# PROGRAM-4

```
import pandas as pd
d=pd.read_excel("C:\\Users\Shabnam\Desktop\duplicate.xlsx")
df=pd.DataFrame(d)
print(df)
df.drop_duplicates(inplace=True)
print(df)
```

# OUTPUT



```
   Name  Eng  Maths  Sci  Total
0     a   33     56   77    166
1     s   35     66   79    180
2     d   36     33   81    150
3     e   37     86   83    206
4    gh   38     96   33    167
5     j   39     33   87    159
6    kk   40    116   89    245
7    gh   38     96   33    167
8     e   37     86   83    206
9    gh   38     96   33    167
10    a   33     56   77    166
  Name  Eng  Maths  Sci  Total
0    a   33     56   77    166
1    s   35     66   79    180
2    d   36     33   81    150
3    e   37     86   83    206
4   gh   38     96   33    167
5    j   39     33   87    159
6   kk   40    116   89    245
```

# DESCRIPTIVE STATS:

## PROGRAM-1

```
import pandas as pd
import numpy as np

d = {'Name':pd.Series(['asritha','asri','pandu','geetha','Srinu','avi','avinash' 'Lee','David','Gasper','Betina',
'Andres']),
   'Age':pd.Series([25,26,25,23,30,29,23,34,40,30,51,46]),
   'Rating':pd.Series([4.23,3.24,3.98,2.56,3.20,4.6,3.8,3.78,2.98,4.80,4.10,3.65])
}

df = pd.DataFrame(d)
print (df)
```

## OUTPUT

```
        Name  Age  Rating
0   asritha   25    4.23
1      asri   26    3.24
2     pandu   25    3.98
3    geetha   23    2.56
4     Srinu   30    3.20
5       avi   29    4.60
6   avinash   23    3.80
7       Lee   34    3.78
8     David   40    2.98
9    Gasper   30    4.80
10   Betina   51    4.10
11   Andres   46    3.65
```

## PROGRAM-2

```
import pandas as pd
import numpy as np

d = {'Name':pd.Series(['asritha','asri','pandu','geetha','Srinu','avi','avinash',
   'Lee','David','Gasper','Betina','Andres']),
   'Age':pd.Series([25,26,25,23,30,29,23,34,40,30,51,46]),
   'Rating':pd.Series([4.23,3.24,3.98,2.56,3.20,4.6,3.8,3.78,2.98,4.80,4.10,3.65])
}

df = pd.DataFrame(d)
print (df.sum())
```

## OUTPUT

```
Name      asrithaasripandugeethaSrinuaviavinashLeeDavidG...
Age                                                     382
Rating                                                44.92
dtype: object
```

## PROGRAM-3

```
import pandas as pd
import numpy as np

d = {'Name':pd.Series(['asritha','asri','pandu','geetha','Srinu','avinash','avi',
   'Lee','David','Gasper','Betina','Andres']),
   'Age':pd.Series([25,26,25,23,30,29,23,34,40,30,51,46]),
   'Rating':pd.Series([4.23,3.24,3.98,2.56,3.20,4.6,3.8,3.78,2.98,4.80,4.10,3.65])
}

df = pd.DataFrame(d)
print (df.sum(1))
```

## OUTPUT

```
0     29.23
1     29.24
2     28.98
3     25.56
4     33.20
5     33.60
6     26.80
7     37.78
8     42.98
9     34.80
10    55.10
11    49.65
dtype: float64
```

## PROGRAM-4

```
import pandas as pd
import numpy as np

d = {'Name':pd.Series(['asritha','asri','pandu','geetha','Srinu','avinash','avi',
   'Lee','David','Gasper','Betina','Andres']),
   'Age':pd.Series([25,26,25,23,30,29,23,34,40,30,51,46]),
   'Rating':pd.Series([4.23,3.24,3.98,2.56,3.20,4.6,3.8,3.78,2.98,4.80,4.10,3.65])
}

df = pd.DataFrame(d)
print (df.sum(1))
```

## OUTPUT

```
print (df.mean())

Age       31.833333
Rating     3.743333
dtype: float64
```

# PROGRAM-5

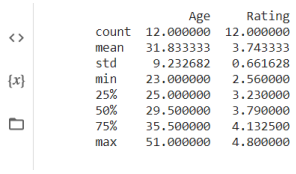
```
import pandas as pd
import numpy as np

d = {'Name':pd.Series(['asritha','asri','pandu','geetha','srinu','avinash','avi',
   'Lee','David','Gasper','Betina','Andres']),
   'Age':pd.Series([25,26,25,23,30,29,23,34,40,30,51,46]),
   'Rating':pd.Series([4.23,3.24,3.98,2.56,3.20,4.6,3.8,3.78,2.98,4.80,4.10,3.65])
}

df = pd.DataFrame(d)
print (df.mean())
```

# OUTPUT


# PROGRAM-6

```
import pandas as pd
import numpy as np

d = {'Name':pd.Series(['asritha','asri','pandu','geetha','Srinu','avi','avinash',
   'Lee','David','Gasper','Betina','Andres']),
   'Age':pd.Series([25,26,25,23,30,29,23,34,40,30,51,46]),
   'Rating':pd.Series([4.23,3.24,3.98,2.56,3.20,4.6,3.8,3.78,2.98,4.80,4.10,3.65])
}

df = pd.DataFrame(d)
print (df.std())
```

# OUTPUT



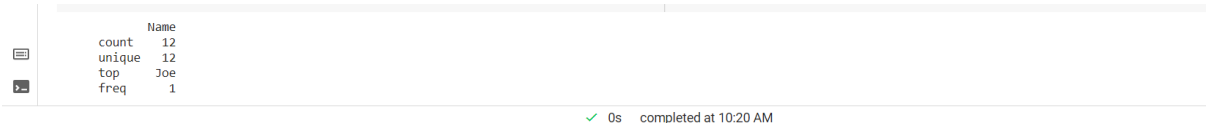
| | Age | Rating |
|---|---|---|
| count | 12.000000 | 12.000000 |
| mean | 31.833333 | 3.743333 |
| std | 9.232682 | 0.661628 |
| min | 23.000000 | 2.560000 |
| 25% | 25.000000 | 3.230000 |
| 50% | 29.500000 | 3.790000 |
| 75% | 35.500000 | 4.132500 |
| max | 51.000000 | 4.800000 |

# PROGRAM-7

```
import pandas as pd
import numpy as np

d = {'Name':pd.Series(['asritha','asri','pandu','srinu','geetha','avinash','avi',
   'Lee','David','Gasper','Betina','Andres']),
   'Age':pd.Series([25,26,25,23,30,29,23,34,40,30,51,46]),
   'Rating':pd.Series([4.23,3.24,3.98,2.56,3.20,4.6,3.8,3.78,2.98,4.80,4.10,3.65])
}
```

```
df = pd.DataFrame(d)
print (df.describe())
```

## OUTPUT



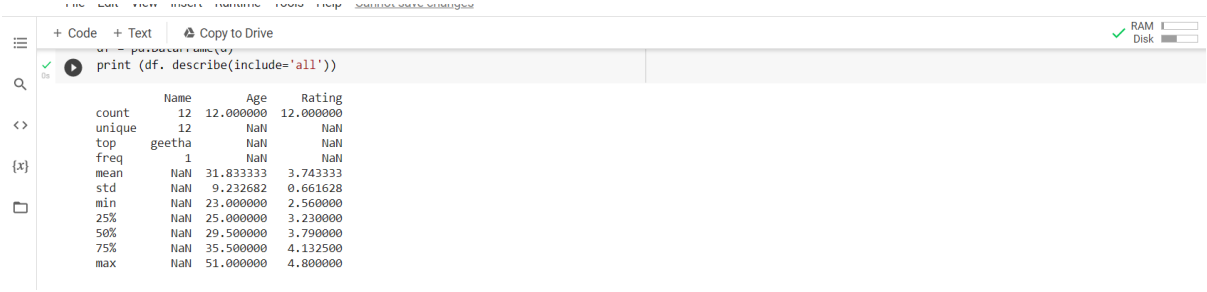
```
        Name
count    12
unique   12
top     Joe
freq     1
```

0s completed at 10:20 AM

## PROGRAM-8

```
import pandas as pd
import numpy as np

d = {'Name':pd.Series(['asritha','asri','pandu','geetha','srinu','avi','avinash',
   'Lee','David','Joe','Tom','Vin']),
   'Age':pd.Series([25,26,25,23,30,29,23,34,40,30,51,46]),
   'Rating':pd.Series([4.23,3.24,3.98,2.56,3.20,4.6,3.8,3.78,2.98,4.80,4.10,3.65])
}

df = pd.DataFrame(d)
print (df.describe(include=['object']))
```

## OUTPUT



+ Code + Text Copy to Drive RAM Disk

```
print (df. describe(include='all'))
```

```
          Name        Age     Rating
count       12  12.000000  12.000000
unique      12        NaN        NaN
top     geetha        NaN        NaN
freq         1        NaN        NaN
mean       NaN  31.833333   3.743333
std        NaN   9.232682   0.661628
min        NaN  23.000000   2.560000
25%        NaN  25.000000   3.230000
50%        NaN  29.500000   3.790000
75%        NaN  35.500000   4.132500
max        NaN  51.000000   4.800000
```

## PROGRAM-9

```
import pandas as pd
import numpy as np

d = {'Name':pd.Series(['asritha','asri','pandu','geetha','Srinu','avinash','avi',
   'Lee','David','Gasper','Betina','Andres']),
   'Age':pd.Series([25,26,25,23,30,29,23,34,40,30,51,46]),
   'Rating':pd.Series([4.23,3.24,3.98,2.56,3.20,4.6,3.8,3.78,2.98,4.80,4.10,3.65])
}

df = pd.DataFrame(d)
print (df. describe(include='all'))
```

## OUTPUT

```
Original DataFrame:
   Product  Price  Year
0  Mobile  20000  2014
1      AC  28000  2015
2  Mobile  22000  2016
3    Sofa  19000  2017
4  Laptop  45000  2018

Descriptive statistics of Price:

count        5.000000
mean     26800.000000
std      10756.393448
min      19000.000000
25%      20000.000000
50%      22000.000000
75%      28000.000000
max      45000.000000
Name: Price, dtype: float64
```

## PROGRAM-10

```
from pandas import DataFrame

# Create DataFrame
cart = {'Product': ['Mobile', 'AC', 'Mobile', 'Sofa', 'Laptop'],
    'Price': [20000, 28000, 22000, 19000, 45000],
    'Year': [2014, 2015, 2016, 2017, 2018]
    }
df = DataFrame(cart, columns = ['Product', 'Price', 'Year'])

# Original DataFrame
print("Original DataFrame:\n", df)

# Describing descriptive statistics of Price
print("\nDescriptive statistics of Price:\n")
stats = df['Price'].describe()
print(stats)
```

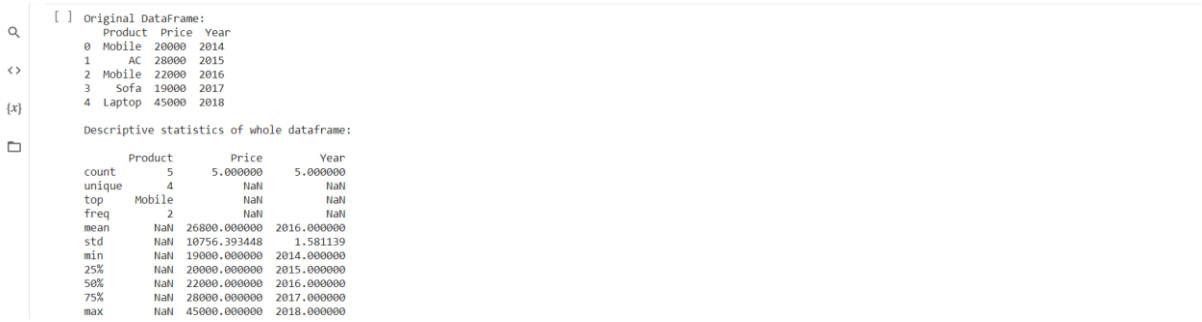
## OUTPUT

```
Original DataFrame:
   Product  Price  Year
0  Mobile  20000  2014
1      AC  28000  2015
2  Mobile  22000  2016
3    Sofa  19000  2017
4  Laptop  45000  2018

Descriptive statistics of year:

count       5.000000
mean     2016.000000
std         1.581139
min      2014.000000
25%      2015.000000
50%      2016.000000
75%      2017.000000
max      2018.000000
Name: Year, dtype: float64
```

## PROGRAM-11

```
cart = {'Product': ['Mobile', 'AC', 'Mobile', 'Sofa', 'Laptop'],
    'Price': [20000, 28000, 22000, 19000, 45000],
    'Year': [2014, 2015, 2016, 2017, 2018]
    }
df = DataFrame(cart, columns = ['Product', 'Price', 'Year'])
print("Original DataFrame:\n", df)
print("\nDescriptive statistics of year:\n")
stats = df['Year'].describe()
print(stats)
```

## OUTPUT

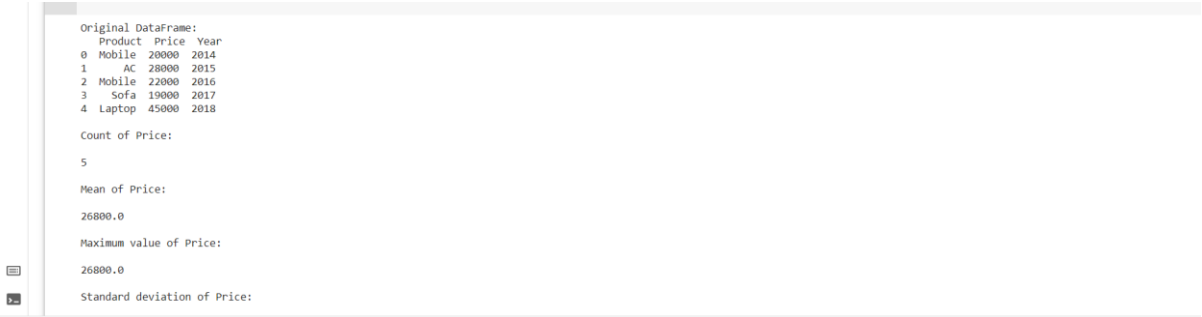


```
[ ] Original DataFrame:
    Product  Price  Year
0  Mobile  20000  2014
1      AC  28000  2015
2  Mobile  22000  2016
3    Sofa  19000  2017
4  Laptop  45000  2018

Descriptive statistics of whole dataframe:

        Product         Price         Year
count         5      5.000000     5.000000
unique        4           NaN          NaN
top      Mobile           NaN          NaN
freq          2           NaN          NaN
mean        NaN  26800.000000  2016.000000
std         NaN  10756.393448     1.581139
min         NaN  19000.000000  2014.000000
25%         NaN  20000.000000  2015.000000
50%         NaN  22000.000000  2016.000000
75%         NaN  28000.000000  2017.000000
max         NaN  45000.000000  2018.000000
```

## PROGRAM-12

```
cart = {'Product': ['Mobile', 'AC', 'Mobile', 'Sofa', 'Laptop'],
   'Price': [20000, 28000, 22000, 19000, 45000],
   'Year': [2014, 2015, 2016, 2017, 2018]
   }
df = DataFrame(cart, columns = ['Product', 'Price', 'Year'])

print("Original DataFrame:\n", df)

print("\nDescriptive statistics of whole dataframe:\n")
stats = df.describe(include = 'all')
print(stats)
```

## OUTPUT



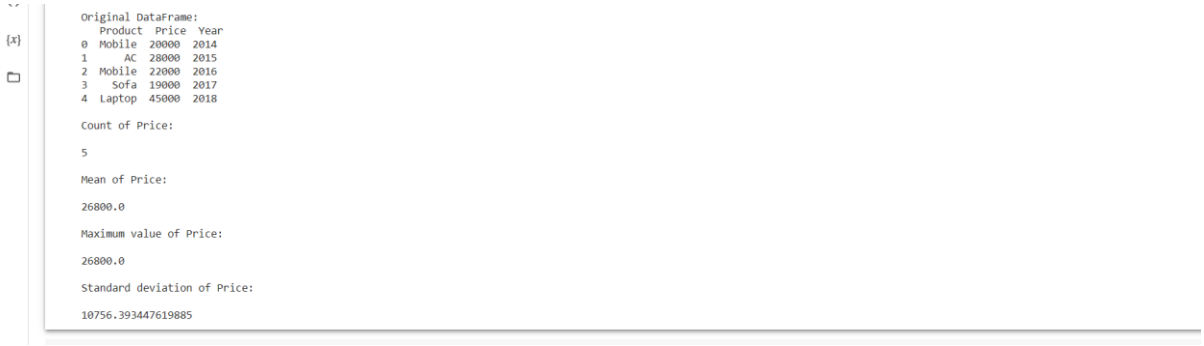
```
Original DataFrame:
   Product  Price  Year
0  Mobile  20000  2014
1      AC  28000  2015
2  Mobile  22000  2016
3    Sofa  19000  2017
4  Laptop  45000  2018

Count of Price:

5

Mean of Price:

26800.0

Maximum value of Price:

26800.0

Standard deviation of Price:
```

## PROGRAM-13

```
cart = {'Product': ['Mobile', 'AC', 'Mobile', 'Sofa', 'Laptop'],
   'Price': [20000, 28000, 22000, 19000, 45000],
   'Year': [2014, 2015, 2016, 2017, 2018]
   }
df = DataFrame(cart, columns = ['Product', 'Price', 'Year'])

print("Original DataFrame:\n", df)

print("\nCount of Price:\n")
counts = df['Price'].count()
print(counts)

print("\nMean of Price:\n")
m = df['Price'].mean()
```

```
print(m)

print("\nMaximum value of Price:\n")
mx = df['Price'].max()
print(m)

print("\nStandard deviation of Price:\n")
sd = df['Price'].std()
print(sd)
```

## OUTPUT



```
Original DataFrame:
   Product  Price  Year
0  Mobile  20000  2014
1      AC  28000  2015
2  Mobile  22000  2016
3    Sofa  19000  2017
4  Laptop  45000  2018

Count of Price:

5

Mean of Price:

26800.0

Maximum value of Price:

26800.0

Standard deviation of Price:

10756.393447619885
```

# PANDAS FUNCTIONS:

## PROGRAM-1

```
import pandas as pd
import numpy as np

def adder(ele1,ele2):
  return ele1+ele2

df = pd.DataFrame(np.random.randn(5,3),columns=['col1','col2','col3'])
print(df)
print("printing the results of pipe")
print(df.pipe(adder,2))
```

## OUTPUT

```
       col1      col2      col3
0  0.650209  0.279272 -0.874434
1 -1.159792  0.406474 -0.532977
2 -1.089769 -0.031652 -0.457861
3  1.006854  0.833344 -0.878184
4 -0.508686 -0.488085 -0.991928
printing the results of pipe
       col1      col2      col3
0  2.650209  2.279272  1.125566
1  0.840208  2.406474  1.467023
2  0.910231  1.968348  1.542139
3  3.006854  2.833344  1.121816
4  1.491314  1.511915  1.008072
```

## PROGRAM-2

```
import pandas as pd
import numpy as np

def adder(ele1,ele2):
  return ele1+ele2

df = pd.DataFrame(np.random.randn(5,3),columns=['col1','col2','col3'])
print(df)

print (df.apply(np.mean))
```

## OUTPUT

```
       col1      col2      col3
0 -0.221325 -0.379971 -0.517350
1 -0.795384  0.654277  0.652740
2  0.141925 -1.009026  0.990916
3 -0.335229 -0.174160 -0.149690
4  0.259244 -0.533985  0.176252
col1   -0.190154
col2   -0.288573
col3    0.230574
dtype: float64
```

## PROGRAM-3

```
import pandas as pd
import numpy as np

df = pd.DataFrame(np.random.randn(5,3),columns=['col1','col2','col3'])
print(df)

print (df.apply(np.mean,axis=1))
```

## OUTPUT

```
       col1      col2      col3
0 -0.925125  0.541934 -0.982347
1 -1.018046  0.082642 -3.191830
2  0.278056 -1.534515 -0.154953
3 -1.916473 -1.760331  0.152299
4  0.145001 -0.521489 -2.528679
0   -0.455179
1   -1.375745
2   -0.470471
3   -1.174835
4   -0.968389
dtype: float64
```

## PROGRAM-4

```
import pandas as pd
import numpy as np
df = pd.DataFrame(np.random.randn(5,3),columns=['col1','col2','col3'])
print(df)

print(df['col1'].map(lambda x:x*100))
```

## OUTPUT

```
[ ]       col1      col2      col3
    0 -0.866337  0.237423  0.274658
    1  1.133911 -2.230867 -0.523681
    2  1.003188  0.029264 -0.024589
    3  0.008908 -1.562550 -1.783723
    4 -0.146589 -0.450539 -1.310938
    0    -86.633711
    1    113.391137
    2    100.318824
    3      0.890847
    4    -14.658866
    Name: col1, dtype: float64
```

## PROGRAM-5

```
df=pd.DataFrame({'id':[1,2,3,4,5],'name':['abc','defg','ghs','eeee','wwww'],'age':[11,22,33,44,55], 'income':[
9999,8888,7777,6666,5555]})
print(df)
```

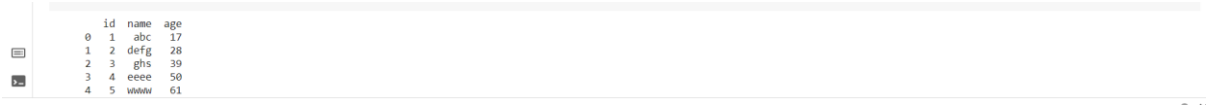
## OUTPUT

```
   id  name  age  income
0   1   abc   11    9999
1   2  defg   22    8888
2   3   ghs   33    7777
3   4  eeee   44    6666
4   5  wwww   55    5555
```

# PROGRAM-6

```
df['age']=df.apply(lambda x: x['age']+3, axis=1)
print(df)
```

# OUTPUT

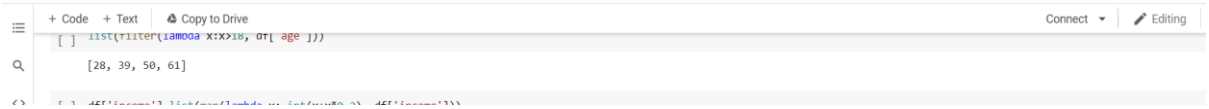


```
   id  name  age
0   1   abc   17
1   2  defg   28
2   3   ghs   39
3   4  eeee   50
4   5  wwww   61
```

# PROGRAM-7

```
list(filter(lambda x:x>18, df['age']))
```

# OUTPUT

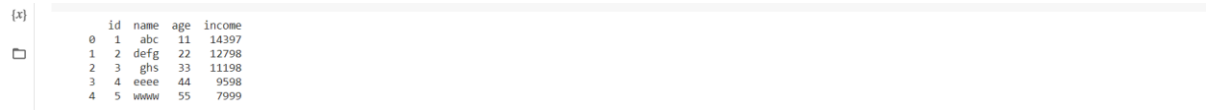


```
[28, 39, 50, 61]
```

# PROGRAM-8

```
df['income']=list(map(lambda x: int(x+x*0.2), df['income']))
print(df)
```

# OUTPUT



```
   id  name  age  income
0   1   abc   11   14397
1   2  defg   22   12798
2   3   ghs   33   11198
3   4  eeee   44    9598
4   5  wwww   55    7999
```

# PROGRAM-9

```
import functools

functools.reduce(lambda a,b: a+b, df['income'])
```

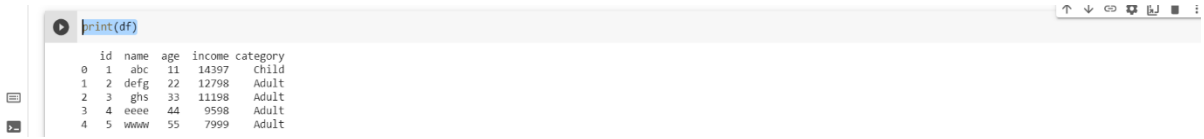
# OUTPUT

```
import functools

functools.reduce(lambda a,b: a+b, df['income'])

55990
```

# PROGRAM-10

```
df['category']=df['age'].apply(lambda x: 'Adult' if x>=18 else 'Child')
print(df)
```

# OUTPUT



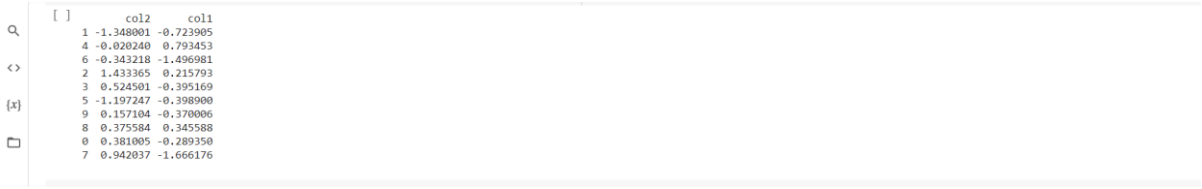
```
print(df)

   id  name  age  income category
0   1   abc   11   14397    Child
1   2  defg   22   12798    Adult
2   3   ghs   33   11198    Adult
3   4  eeee   44    9598    Adult
4   5  wwww   55    7999    Adult
```

# PANDAS SORTING:

## PROGRAM-1

```
import pandas as pd
import numpy as np

unsorted_df=pd.DataFrame(np.random.randn(10,2),index=[1,4,6,2,3,5,9,8,0,7],columns=['col2','col1'])
print (unsorted_df)
```

## OUTPUT

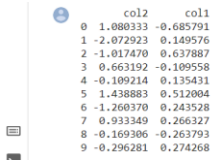


```
[ ]       col2      col1
    1 -1.348001 -0.723905
    4 -0.020240  0.793453
    6 -0.343218 -1.496981
    2  1.433365  0.215793
    3  0.524501 -0.395169
    5 -1.197247 -0.398900
    9  0.157104 -0.370006
    8  0.375584  0.345588
    0  0.381005 -0.289350
    7  0.942037 -1.666176
```

## PROGRAM-2

```
import pandas as pd
import numpy as np

unsorted_df = pd.DataFrame(np.random.randn(10,2),index=[1,4,6,2,3,5,9,8,0,7],columns = ['col2','col1'])

sorted_df=unsorted_df.sort_index()
print (sorted_df)
```

## OUTPUT

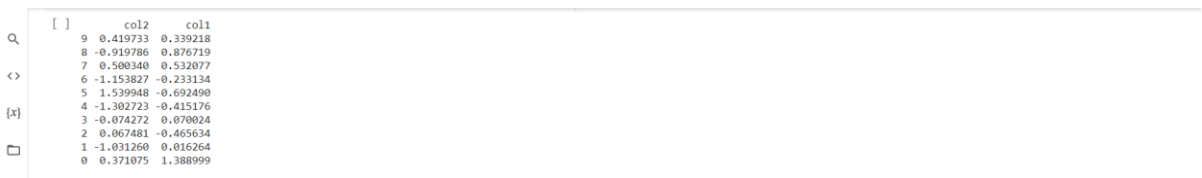


```
      col2      col1
0  1.080333 -0.685791
1 -2.072923  0.149576
2 -1.017470  0.637887
3  0.663192 -0.109558
4 -0.109214  0.135431
5  1.438883  0.512004
6 -1.260370  0.243528
7  0.933349  0.266327
8 -0.169306 -0.263793
9 -0.296281  0.274268
```

## PROGRAM-3

```
import pandas as pd
import numpy as np

unsorted_df = pd.DataFrame(np.random.randn(10,2),index=[1,4,6,2,3,5,9,8,0,7],columns = ['col2','col1'])

sorted_df = unsorted_df.sort_index(ascending=False)
print (sorted_df)
```

## OUTPUT



```
[ ]       col2      col1
    9  0.419733  0.339218
    8 -0.919786  0.876719
    7  0.500340  0.532077
    6 -1.153827 -0.233134
    5  1.539948 -0.692490
    4 -1.302723 -0.415176
    3 -0.074272  0.070024
    2  0.067481 -0.465634
    1 -1.031260  0.016264
    0  0.371075  1.388999
```

# PROGRAM-4

```
import pandas as pd
import numpy as np

unsorted_df = pd.DataFrame(np.random.randn(10,2),index=[1,4,6,2,3,5,9,8,0,7],columns = ['col2','col1'])

sorted_df=unsorted_df.sort_index(axis=1)

print (sorted_df)
```

# OUTPUT

```
       col1      col2
1 -0.154511  0.476389
4  0.269849  0.085632
6 -0.131993 -0.022483
2  0.616650  0.720580
3  0.965333  3.320490
5  0.864344  0.080299
9  0.108683  0.717751
8 -1.214840 -0.267902
0 -0.631016 -0.213998
7  0.293938  1.788917
```

# PROGRAM-5

```
import pandas as pd
import numpy as np

unsorted_df = pd.DataFrame({'col1':[2,1,1,1],'col2':[1,3,2,4]})
sorted_df = unsorted_df.sort_values(by='col1')

print (sorted_df)
```

# OUTPUT

```
print (sorted_df)

   col1  col2
1     1     3
2     1     2
3     1     4
0     2     1
```

# PROGRAM-6

```
import pandas as pd
import numpy as np

unsorted_df = pd.DataFrame({'col1':[2,1,1,1],'col2':[1,3,2,4]})
sorted_df = unsorted_df.sort_values(by=['col1','col2'])

print (sorted_df)
```

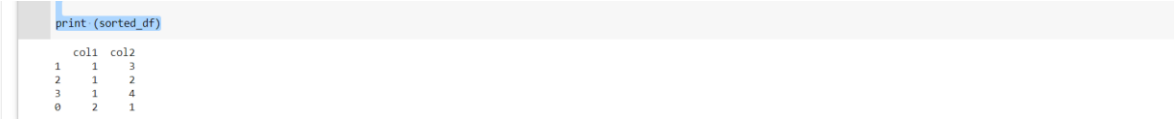
# OUTPUT

```
print (sorted_df)

   col1  col2
2     1     2
1     1     3
3     1     4
0     2     1
```

# PROGRAM-7

```
import pandas as pd
import numpy as np

unsorted_df = pd.DataFrame({'col1':[2,1,1,1],'col2':[1,3,2,4]})
sorted_df = unsorted_df.sort_values(by='col1' ,kind='mergesort')

print (sorted_df)
```

# OUTPUT



```
print (sorted_df)

   col1  col2
1     1     3
2     1     2
3     1     4
0     2     1
```

# PANDAS DATALAMBDA

## PROGRAM-1

```
import pandas as pd
values= [['Rohan',455],['Elvish',250],['Deepak',495],
   ['Soni',400],['Radhika',350],['Vansh',450]]
df = pd.DataFrame(values,columns=['Name','Total_Marks'])
df = df.assign(Percentage = lambda x: (x['Total_Marks'] /500 * 100))
print(df)
```

## OUTPUT

```
      Name  Total_Marks  Percentage
0    Rohan          455        91.0
1   Elvish          250        50.0
2   Deepak          495        99.0
3     Soni          400        80.0
4  Radhika          350        70.0
5    Vansh          450        90.0
```

## PROGRAM-2

```
import pandas as pd
values_list = [[15, 2.5, 100], [20, 4.5, 50], [25, 5.2, 80],
     [45, 5.8, 48], [40, 6.3, 70], [41, 6.4, 90],
     [51, 2.3, 111]]

df = pd.DataFrame(values_list, columns=['Field_1', 'Field_2', 'Field_3'])

df = df.assign(Product=lambda x: (x['Field_1'] * x['Field_2'] * x['Field_3']))

print(df)
```

## OUTPUT

```
   Field_1  Field_2  Field_3  Product
0       15      2.5      100   3750.0
1       20      4.5       50   4500.0
2       25      5.2       80  10400.0
3       45      5.8       48  12528.0
4       40      6.3       70  17640.0
5       41      6.4       90  23616.0
6       51      2.3      111  13020.3
```

## PROGRAM-3

```
import pandas as pd
import numpy as np

values_list = [[15, 2.5, 100], [20, 4.5, 50], [25, 5.2, 80],
     [45, 5.8, 48], [40, 6.3, 70], [41, 6.4, 90],
     [51, 2.3, 111]]

df = pd.DataFrame(values_list, columns=['Field_1', 'Field_2', 'Field_3'],
       index=['a', 'b', 'c', 'd', 'e', 'f', 'g'])
df = df.apply(lambda x: np.square(x) if x.name == 'd' else x, axis=1)
print(df)
```

## OUTPUT

| | Field_1 | Field_2 | Field_3 |
|---|---|---|---|
| a | 15.0 | 2.50 | 100.0 |
| b | 20.0 | 4.50 | 50.0 |
| c | 25.0 | 5.20 | 80.0 |
| d | 2025.0 | 33.64 | 2304.0 |
| e | 40.0 | 6.30 | 70.0 |
| f | 41.0 | 6.40 | 90.0 |
| g | 51.0 | 2.30 | 111.0 |

## PROGRAM-4

```
import pandas as pd
import numpy as np
values_list = [[15, 2.5, 100], [20, 4.5, 50], [25, 5.2, 80],
     [45, 5.8, 48], [40, 6.3, 70], [41, 6.4, 90],
     [51, 2.3, 111]]
df = pd.DataFrame(values_list, columns=['Field_1', 'Field_2', 'Field_3'],
       index=['a', 'b', 'c', 'd', 'e', 'f', 'g'])
df = df.apply(lambda x: np.square(x) if x.name in [
     'a', 'e', 'g'] else x, axis=1)
df
```

## OUTPUT

| | Field_1 | Field_2 | Field_3 |
|---|---|---|---|
| a | 225.0 | 6.25 | 10000.0 |
| b | 20.0 | 4.50 | 50.0 |
| c | 25.0 | 5.20 | 80.0 |
| d | 45.0 | 5.80 | 48.0 |
| e | 1600.0 | 39.69 | 4900.0 |
| f | 41.0 | 6.40 | 90.0 |
| g | 2601.0 | 5.29 | 12321.0 |

## PROGRAM-5

```
import pandas as pd
import numpy as np
values_list = [[1.5, 2.5, 10.0], [2.0, 4.5, 5.0], [2.5, 5.2, 8.0],
     [4.5, 5.8, 4.8], [4.0, 6.3, 70], [4.1, 6.4, 9.0],
     [5.1, 2.3, 11.1]]
df = pd.DataFrame(values_list, columns=['Field_1', 'Field_2', 'Field_3'],
       index=['a', 'b', 'c', 'd', 'e', 'f', 'g'])
df = df.apply(lambda x: np.square(x) if x.name in ['b', 'f'] else x, axis=1)
df = df.assign(Product=lambda x: (x['Field_1'] * x['Field_2'] * x['Field_3']))
df
```

## OUTPUT

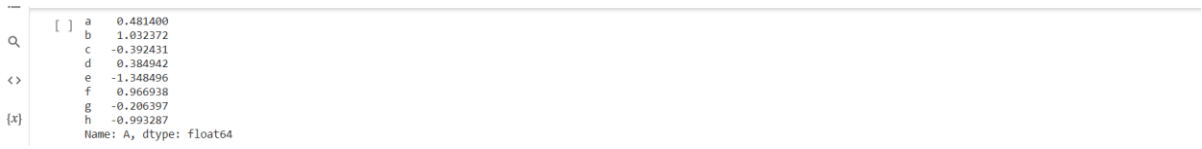
| | Field_1 | Field_2 | Field_3 | Product |
|---|---|---|---|---|
| a | 1.50 | 2.50 | 10.0 | 37.5000 |
| b | 4.00 | 20.25 | 25.0 | 2025.0000 |
| c | 2.50 | 5.20 | 8.0 | 104.0000 |
| d | 4.50 | 5.80 | 4.8 | 125.2800 |
| e | 4.00 | 6.30 | 70.0 | 1764.0000 |
| f | 16.81 | 40.96 | 81.0 | 55771.5456 |
| g | 5.10 | 2.30 | 11.1 | 130.2030 |

# INDEX AND SELECTDATA

## PROGRAM-1

```
import pandas as pd
import numpy as np

df = pd.DataFrame(np.random.randn(8, 4),

index = ['a','b','c','d','e','f','g','h'], columns = ['A', 'B', 'C', 'D'])
print (df.loc[:,'A'])
```

## OUTPUT

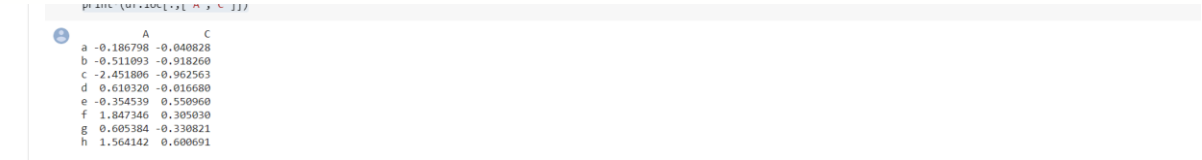


```
a    0.481400
b    1.032372
c   -0.392431
d    0.384942
e   -1.348496
f    0.966938
g   -0.206397
h   -0.993287
Name: A, dtype: float64
```

## PROGRAM-2

```
import pandas as pd
import numpy as np

df = pd.DataFrame(np.random.randn(8, 4),
index = ['a','b','c','d','e','f','g','h'], columns = ['A', 'B', 'C', 'D'])
print (df.loc[:,['A','C']])
```

## OUTPUT

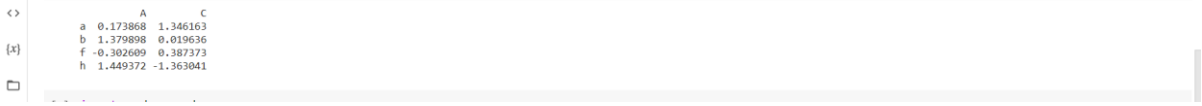


```
          A         C
a -0.186798 -0.040828
b -0.511093 -0.918260
c -2.451806 -0.962563
d  0.610320 -0.016680
e -0.354539  0.550960
f  1.847346  0.305030
g  0.605384 -0.330821
h  1.564142  0.600691
```

## PROGRAM-3

```
import pandas as pd
import numpy as np

df = pd.DataFrame(np.random.randn(8, 4),
index = ['a','b','c','d','e','f','g','h'], columns = ['A', 'B', 'C', 'D'])

print (df.loc[['a','b','f','h'],['A','C']])
```

## OUTPUT



```
          A         C
a  0.173868  1.346163
b  1.379898  0.019636
f -0.302609  0.387373
h  1.449372 -1.363041
```

# PROGRAM-4

```
import pandas as pd
import numpy as np

df = pd.DataFrame(np.random.randn(8, 4),
index = ['a','b','c','d','e','f','g','h'], columns = ['A', 'B', 'C', 'D'])

print (df.loc['a':'d'])
```

# OUTPUT

```
          A         B         C         D
a  1.159802 -0.028690 -0.776045  0.827534
b -1.041473  0.204152 -0.643117  0.396520
c -0.513027  2.195045  1.354764 -0.352770
d  0.951589  1.871177 -0.026550 -0.026447
```

# PROGRAM-5

```
import pandas as pd
import numpy as np

df = pd.DataFrame(np.random.randn(8, 4),
index = ['a','b','c','d','e','f','g','h'], columns = ['A', 'B', 'C', 'D'])
print(df)

print (df.loc['a']>0)
```

# OUTPUT

```
          A         B         C         D
a -1.245752 -0.495460  0.706293  1.770178
b  0.059009  0.329564  1.707307  0.083275
c -1.344858 -2.090786 -1.289896  1.036774
d -1.670576  1.765127 -0.259284  0.664807
e -1.077589  1.166700  0.496885  0.883307
f -1.172176  0.122853 -1.022873 -1.318883
g -1.507572 -1.750927  0.616773  1.320749
h  0.384835 -0.100205 -1.564131 -0.756418
A    False
B    False
C     True
D     True
Name: a, dtype: bool
```

# PROGRAM-6

```
import pandas as pd
import numpy as np

df = pd.DataFrame(np.random.randn(8, 4), columns = ['A', 'B', 'C', 'D'])
print(df)
print (df.iloc[:4])
```

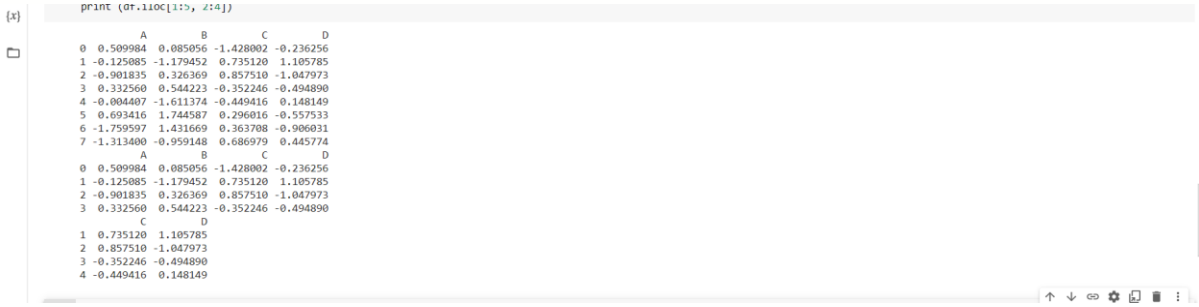
# OUTPUT

```
print (df.iloc[1:5, 2:4])

          A         B         C         D
0  0.509984  0.085056 -1.428002 -0.236256
1 -0.125085 -1.179452  0.735120  1.105785
2 -0.901835  0.326369  0.857510 -1.047973
3  0.332560  0.544223 -0.352246 -0.494890
4 -0.004407 -1.611374 -0.449416  0.148149
5  0.693416  1.744587  0.296016 -0.557533
6 -1.759597  1.431669  0.363708 -0.906031
7 -1.313400 -0.959148  0.686979  0.445774
          A         B         C         D
0  0.509984  0.085056 -1.428002 -0.236256
1 -0.125085 -1.179452  0.735120  1.105785
2 -0.901835  0.326369  0.857510 -1.047973
3  0.332560  0.544223 -0.352246 -0.494890
          C         D
1  0.735120  1.105785
2  0.857510 -1.047973
3 -0.352246 -0.494890
4 -0.449416  0.148149
```

# PROGRAM-7

```
import pandas as pd
import numpy as np

df = pd.DataFrame(np.random.randn(8, 4), columns = ['A', 'B', 'C', 'D'])
print(df)
print (df.iloc[:4])
print (df.iloc[1:5, 2:4])
```

# OUTPUT

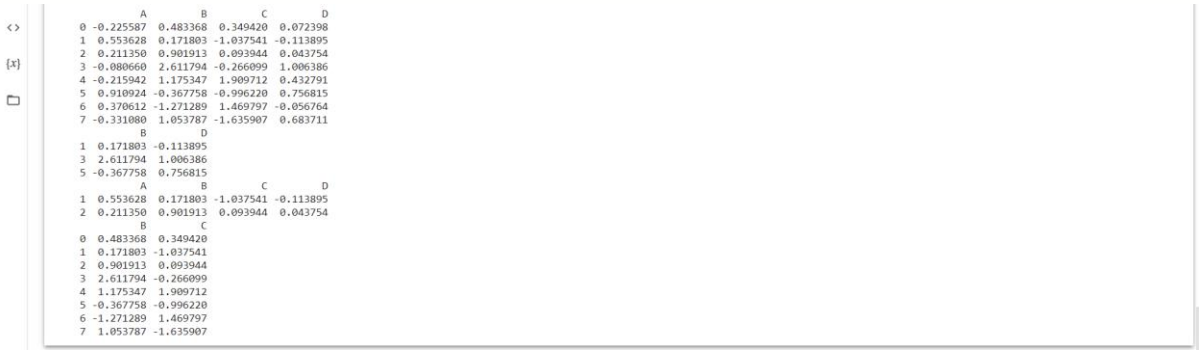


```
print (df.iloc[1:5, 2:4])
          A         B         C         D
0  0.509984  0.085056 -1.428002 -0.236256
1 -0.125085 -1.179452  0.735120  1.105785
2 -0.901835  0.326369  0.857510 -1.047973
3  0.332560  0.544223 -0.352246 -0.494890
4 -0.004407 -1.611374 -0.449416  0.148149
5  0.693416  1.744587  0.296016 -0.557533
6 -1.759597  1.431669  0.363708 -0.906031
7 -1.313400 -0.959148  0.686979  0.445774
          A         B         C         D
0  0.509984  0.085056 -1.428002 -0.236256
1 -0.125085 -1.179452  0.735120  1.105785
2 -0.901835  0.326369  0.857510 -1.047973
3  0.332560  0.544223 -0.352246 -0.494890
          C         D
1  0.735120  1.105785
2  0.857510 -1.047973
3 -0.352246 -0.494890
4 -0.449416  0.148149
```

# PROGRAM-8

```
import pandas as pd
import numpy as np

df = pd.DataFrame(np.random.randn(8, 4), columns = ['A', 'B', 'C', 'D'])
print(df)
print (df.iloc[[1, 3, 5], [1, 3]])
print (df.iloc[1:3, :])
print (df.iloc[:,1:3])
```

# OUTPUT



```
          A         B         C         D
0 -0.225587  0.483368  0.349420  0.072398
1  0.553628  0.171803 -1.037541 -0.113895
2  0.211350  0.901913  0.093944  0.043754
3 -0.080660  2.611794 -0.266099  1.006386
4 -0.215942  1.175347  1.909712  0.432791
5  0.910924 -0.367758 -0.996220  0.756815
6  0.370612 -1.271289  1.469797 -0.056764
7 -0.331080  1.053787 -1.635907  0.683711
          B         D
1  0.171803 -0.113895
3  2.611794  1.006386
5 -0.367758  0.756815
          A         B         C         D
1  0.553628  0.171803 -1.037541 -0.113895
2  0.211350  0.901913  0.093944  0.043754
          B         C
0  0.483368  0.349420
1  0.171803 -1.037541
2  0.901913  0.093944
3  2.611794 -0.266099
4  1.175347  1.909712
5 -0.367758 -0.996220
6 -1.271289  1.469797
7  1.053787 -1.635907
```

# MISSING DATA

## PROGRAM-1

```
import pandas as pd
d=pd.read_excel("C:\\Users\Shabnam\Desktop\missing.xlsx")
df=pd.DataFrame(d)
print(df)
```

## OUTPUT

| | S.No. | Name | Eng | Maths | Sci | Total |
|---|---|---|---|---|---|---|
| 0 | 1 | a | NaN | 56.0 | 77.0 | 133 |
| 1 | 2 | s | 35.0 | 66.0 | 79.0 | 180 |
| 2 | 3 | d | 36.0 | NaN | 81.0 | 117 |
| 3 | 4 | NaN | 37.0 | 86.0 | 83.0 | 206 |
| 4 | 5 | gh | 38.0 | 96.0 | NaN | 134 |
| 5 | 6 | j | 39.0 | NaN | 87.0 | 126 |
| 6 | 7 | kk | 40.0 | 116.0 | 89.0 | 245 |

## PROGRAM- 2

```
df.dropna()
```

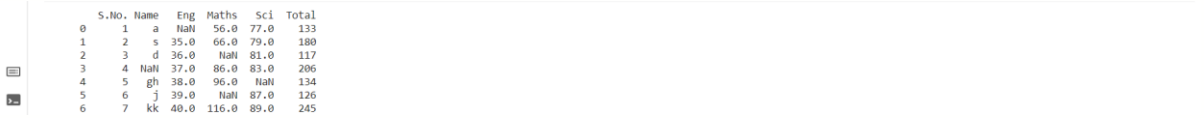
## OUTPUT

| | S.No. | Name | Eng | Maths | Sci | Total |
|---|---|---|---|---|---|---|
| 1 | 2 | s | 35.0 | 66.0 | 79.0 | 180 |
| 6 | 7 | kk | 40.0 | 116.0 | 89.0 | 245 |

## PROGRAM-3

```
import pandas as pd
d=pd.read_excel("C:\\Users\Shabnam\Desktop\missing.xlsx")
df=pd.DataFrame(d)
print(df)
```

## OUTPUT



| | S.No. | Name | Eng | Maths | Sci | Total |
|---|---|---|---|---|---|---|
| 0 | 1 | a | NaN | 56.0 | 77.0 | 133 |
| 1 | 2 | s | 35.0 | 66.0 | 79.0 | 180 |
| 2 | 3 | d | 36.0 | NaN | 81.0 | 117 |
| 3 | 4 | NaN | 37.0 | 86.0 | 83.0 | 206 |
| 4 | 5 | gh | 38.0 | 96.0 | NaN | 134 |
| 5 | 6 | j | 39.0 | NaN | 87.0 | 126 |
| 6 | 7 | kk | 40.0 | 116.0 | 89.0 | 245 |

## PROGRAM-4

```
df['Name'].dropna()
```

## OUTPUT

```
0    a
1    s
2    d
4   gh
5    j
6   kk
Name: Name, dtype: object
```

# PROGRAM-5

```
import pandas as pd
d=pd.read_excel("C:\\Users\Shabnam\Desktop\missing.xlsx")
df=pd.DataFrame(d)
print(df)
```

# OUTPUT

```
   S.No. Name   Eng  Maths   Sci  Total
0      1    a   NaN   56.0  77.0    133
1      2    s  35.0   66.0  79.0    180
2      3    d  36.0    NaN  81.0    117
3      4  NaN  37.0   86.0  83.0    206
4      5   gh  38.0   96.0   NaN    134
5      6    j  39.0    NaN  87.0    126
6      7   kk  40.0  116.0  89.0    245
```

# PROGRAM-6

```
df.loc[:,['Name','Eng']].dropna()
```

# OUTPUT

| | Name | Eng |
|---|---|---|
| 1 | s | 35.0 |
| 2 | d | 36.0 |
| 4 | gh | 38.0 |
| 5 | j | 39.0 |
| 6 | kk | 40.0 |

# PROGRAM-7

```
import pandas as pd
d=pd.read_excel("C:\\Users\Shabnam\Desktop\missing.xlsx")
df=pd.DataFrame(d)
print(df)
```

# OUTPUT

```
   S.No. Name   Eng  Maths   Sci  Total
0      1    a   NaN   56.0  77.0    133
1      2    s  35.0   66.0  79.0    180
2      3    d  36.0    NaN  81.0    117
3      4  NaN  37.0   86.0  83.0    206
4      5   gh  38.0   96.0   NaN    134
5      6    j  39.0    NaN  87.0    126
6      7   kk  40.0  116.0  89.0    245
```

# PROGRAM-8

```
df.fillna("*")
```

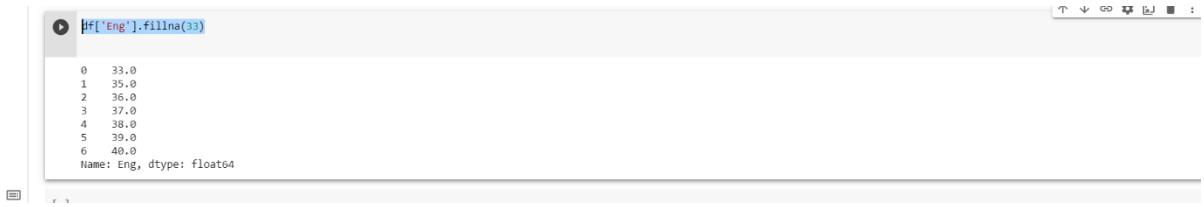
# OUTPUT

| | S.No. | Name | Eng | Maths | Sci | Total |
|---|---|---|---|---|---|---|
| 0 | 1 | a | * | 56.0 | 77.0 | 133 |
| 1 | 2 | s | 35.0 | 66.0 | 79.0 | 180 |
| 2 | 3 | d | 36.0 | * | 81.0 | 117 |
| 3 | 4 | * | 37.0 | 86.0 | 83.0 | 206 |
| 4 | 5 | gh | 38.0 | 96.0 | * | 134 |
| 5 | 6 | j | 39.0 | * | 87.0 | 126 |
| 6 | 7 | kk | 40.0 | 116.0 | 89.0 | 245 |

# PROGRAM-9

```
df['Eng'].fillna(33)
```

# OUTPUT



# IAT LAB-2

US Baby Names 1880–2010: The United States Social Security Administration (SSA) has made available data on the frequency of baby names from 1880 to 2010

i. Use Data Wrangling to load this dataset

```
import pandas as pd
import numpy as np


names1888 = pd.read_csv('yob1888.txt',
                        names=['name', 'sex', 'births'])
names1888.head()
```

## OUTPUT

| | name | sex | births |
|---|---|---|---|
| 0 | Mary | F | 11754 |
| 1 | Anna | F | 4982 |
| 2 | Elizabeth | F | 3224 |
| 3 | Emma | F | 3087 |
| 4 | Margaret | F | 2904 |

ii. Find sum of the birth's column by sex as the total number of births in that year

```
years = [1888,1889,1900,1910,1928,1940,1969,1980,2000,2010]
pieces = []
columns = ['name', 'sex', 'births']
for year in years:
    path = 'yob%d.txt' % year
    frame = pd.read_csv(path, names=columns)

    frame['year'] = year
    pieces.append(frame)

result = pd.concat(pieces, ignore_index=True)
result
```

## OUTPUT

| | name | sex | births | year |
|---|---|---|---|---|
| 0 | Mary | F | 11754 | 1888 |
| 1 | Anna | F | 4982 | 1888 |
| 2 | Elizabeth | F | 3224 | 1888 |
| 3 | Emma | F | 3087 | 1888 |
| 4 | Margaret | F | 2904 | 1888 |
| ... | ... | ... | ... | ... |
| 129745 | Zymaire | M | 5 | 2010 |
| 129746 | Zyonne | M | 5 | 2010 |
| 129747 | Zyquarius | M | 5 | 2010 |
| 129748 | Zyran | M | 5 | 2010 |
| 129749 | Zzyzx | M | 5 | 2010 |

129750 rows × 4 columns

iii. Assemble all of the data into a single Data Frame and further add a year field

```
total_births = result.pivot_table('births', index='year',
                                  columns='sex', aggfunc='sum')
total_births
```

## OUTPUT

| sex | F | M |
|---|---|---|
| year | | |
| 1888 | 178622 | 120851 |
| 1889 | 178366 | 110580 |
| 1900 | 299800 | 150483 |
| 1910 | 396502 | 194213 |
| 1928 | 1153221 | 1107618 |
| 1940 | 1143393 | 1158985 |
| 1969 | 1686999 | 1789216 |
| 1980 | 1660147 | 1784390 |
| 2000 | 1815110 | 1962969 |
| 2010 | 1774758 | 1915942 |

iv. Visualize total births by sex and year

```
line_graph = total_births.plot(title='Total births by sex and year')
print(line_graph)
```

## OUTPUT

```
AxesSubplot(0.125,0.125;0.775x0.755)
```