In [86]:
```
import pandas as pd
```

One Dimention Data Set:

straight line is one dimentional on two dimentional graph (x,y). square is two dimentional. Moreover, a box is three dimentional. An empty serires is zero dimentional. A series of single column is one dimentional.A series with two column is two dimentional. This two dimentional data is called data frame.

In [87]:
```
#one dimentional series
mobile = pd.Series(['samsung','iphone','nokia'])
mobile
```

Out[87]:
```
0    samsung
1     iphone
2      nokia
dtype: object
```

In [88]:
```
bag = pd.Series (['pen', 'paper','notebook'])
bag
```

Out[88]:
```
0         pen
1       paper
2    notebook
dtype: object
```

In [89]:
```
#two dimentionl data:In Data Frame dictionary dataset is usually taken.
dataframe = pd.DataFrame({"electronic":mobile, "item" : bag})
dataframe
```

Out[89]:

	electronic	item
0	samsung	pen
1	iphone	paper
2	nokia	notebook

In [90]:
```
#How to import data:
#mosly used file format is CSV:COMMA SPERATED FILE.
data_csv = pd.read_csv("annual.csv")
```

In [91]:
```
data_csv
```

Out[91]:

	year	industry_code_ANZSIC	industry_name_ANZSIC	rme_size_grp	variable	value	
0	2011	A	Agriculture, Forestry and Fishing	a_0	Activity unit	46134	
1	2011	A	Agriculture, Forestry and Fishing	a_0	Rolling mean employees	0	
2	2011	A	Agriculture, Forestry and Fishing	a_0	Salaries and wages paid	279	DOLLAR
3	2011	A	Agriculture, Forestry and Fishing	a_0	Sales, government funding, grants and subsidies	8187	DOLLAR
4	2011	A	Agriculture, Forestry and Fishing	a_0	Total income	8866	DOLLAR
5	2011	A	Agriculture, Forestry and Fishing	a_0	Total expenditure	7618	DOLLAR
6	2011	A	Agriculture, Forestry and Fishing	a_0	Operating profit before tax	770	DOLLAR
7	2011	A	Agriculture, Forestry and Fishing	a_0	Total assets	55700	DOLLAR
8	2011	A	Agriculture, Forestry and Fishing	a_0	Fixed tangible assets	32155	DOLLAR

In [92]:
```
#every data has two basic components:column, rows. to respresent column: axis=1,
```

In [93]:
```
#to export data
data_csv.to_csv("modified_data.csv")
```

In [94]:
```
data_csv = pd.read_csv("modified_data.csv")
```

In [95]:
```
data_csv
```

Out[95]:

	Unnamed: 0	year	industry_code_ANZSIC	industry_name_ANZSIC	rme_size_grp	variable	valu
0	0	2011	A	Agriculture, Forestry and Fishing	a_0	Activity unit	4613
1	1	2011	A	Agriculture, Forestry and Fishing	a_0	Rolling mean employees	
2	2	2011	A	Agriculture, Forestry and Fishing	a_0	Salaries and wages paid	27
3	3	2011	A	Agriculture, Forestry and Fishing	a_0	Sales, government funding, grants and subsidies	818
4	4	2011	A	Agriculture, Forestry and Fishing	a_0	Total income	886
5	5	2011	A	Agriculture, Forestry and Fishing	a_0	Total expenditure	761
6	6	2011	A	Agriculture, Forestry and Fishing	a_0	Operating profit before tax	77
7	7	2011	A	Agriculture, Forestry and Fishing	a_0	Total assets	5570
8	8	2011	A	Agriculture, Forestry and Fishing	a_0	Fixed tangible assets	3215

In [101]:
```
# in above result extra indexing column is added. In order to remove it, command
Ndata_csv = pd.read_csv("annual.csv")
```

In [102]:
```
Ndata_csv
```

Out[102]:

	year	industry_code_ANZSIC	industry_name_ANZSIC	rme_size_grp	variable	value	
0	2011	A	Agriculture, Forestry and Fishing	a_0	Activity unit	46134	
1	2011	A	Agriculture, Forestry and Fishing	a_0	Rolling mean employees	0	
2	2011	A	Agriculture, Forestry and Fishing	a_0	Salaries and wages paid	279	DOLLAR
3	2011	A	Agriculture, Forestry and Fishing	a_0	Sales, government funding, grants and subsidies	8187	DOLLAR
4	2011	A	Agriculture, Forestry and Fishing	a_0	Total income	8866	DOLLAR
5	2011	A	Agriculture, Forestry and Fishing	a_0	Total expenditure	7618	DOLLAR
6	2011	A	Agriculture, Forestry and Fishing	a_0	Operating profit before tax	770	DOLLAR
7	2011	A	Agriculture, Forestry and Fishing	a_0	Total assets	55700	DOLLAR
8	2011	A	Agriculture, Forestry and Fishing	a_0	Fixed tangible assets	32155	DOLLAR

In [103]:
```
Ndata_csv.to_csv("modified_v1.csv", index= False)
```

In [104]:
```
Ndata_csv = pd.read_csv("modified_v1.csv")
```

In [105]:
```
Ndata_csv
```

Out[105]:

	year	industry_code_ANZSIC	industry_name_ANZSIC	rme_size_grp	variable	value	
0	2011	A	Agriculture, Forestry and Fishing	a_0	Activity unit	46134	
1	2011	A	Agriculture, Forestry and Fishing	a_0	Rolling mean employees	0	
2	2011	A	Agriculture, Forestry and Fishing	a_0	Salaries and wages paid	279	DOLLAR
3	2011	A	Agriculture, Forestry and Fishing	a_0	Sales, government funding, grants and subsidies	8187	DOLLAR
4	2011	A	Agriculture, Forestry and Fishing	a_0	Total income	8866	DOLLAR
5	2011	A	Agriculture, Forestry and Fishing	a_0	Total expenditure	7618	DOLLAR
6	2011	A	Agriculture, Forestry and Fishing	a_0	Operating profit before tax	770	DOLLAR
7	2011	A	Agriculture, Forestry and Fishing	a_0	Total assets	55700	DOLLAR
8	2011	A	Agriculture, Forestry and Fishing	a_0	Fixed tangible assets	32155	DOLLAR

In [106]:
```
#Now, here index is totally removed while exporting.
```

How to describe data?

to describe data we use #Attributes. The most similar thing to attribute is Function(). Function is normally adjoined with round brackets such as read.csv() & to.csv(). However, Attribites donot have round brackets with it. For instacnce, Ndata_csv.dtype.

In [107]:
```
Ndata_csv.dtypes
```

Out[107]:
```
year                    int64
industry_code_ANZSIC    object
industry_name_ANZSIC    object
rme_size_grp            object
variable                object
value                    int64
unit                    object
dtype: object
```

In [108]:
```
#Attribute to know column.
Ndata_csv.columns
```

Out[108]:
```
Index(['year', 'industry_code_ANZSIC', 'industry_name_ANZSIC', 'rme_size_grp',
       'variable', 'value', 'unit'],
      dtype='object')
```

In [109]:
```
#assigning column to another variable
col = Ndata_csv.columns
```

In [110]:
```
col
```

Out[110]:
```
Index(['year', 'industry_code_ANZSIC', 'industry_name_ANZSIC', 'rme_size_grp',
       'variable', 'value', 'unit'],
      dtype='object')
```

In [111]:
```
# .Index attribute
Ndata_csv.index
```

Out[111]:
```
RangeIndex(start=0, stop=9, step=1)
```

In [112]:
```
# Function to know statistical values:describe(). It will show only data of integ
Ndata_csv.describe()
```

Out[112]:

	year	value
count	9.0	9.000000
mean	2011.0	17745.444444
std	0.0	21316.104194
min	2011.0	0.000000
25%	2011.0	770.000000
50%	2011.0	8187.000000
75%	2011.0	32155.000000
max	2011.0	55700.000000

In [113]:
```
#info() function to know the information.
Ndata_csv.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9 entries, 0 to 8
Data columns (total 7 columns):
 #   Column                Non-Null Count  Dtype 
---  ------                --------------  ----- 
 0   year                  9 non-null      int64 
 1   industry_code_ANZSIC  9 non-null      object
 2   industry_name_ANZSIC  9 non-null      object
 3   rme_size_grp          9 non-null      object
 4   variable              9 non-null      object
 5   value                 9 non-null      int64 
 6   unit                  9 non-null      object
dtypes: int64(2), object(5)
memory usage: 632.0+ bytes
```

In [114]:
```
# mean(): to know mean value of int funtion.
Ndata_csv.mean()
```

Out[114]:
```
year      2011.000000
value    17745.444444
dtype: float64
```

In [115]:
```
#funtions can also be called upon particular columns using square brackets.
```

In [116]:
```
#
Ndata_csv
```

Out[116]:

	year	industry_code_ANZSIC	industry_name_ANZSIC	rme_size_grp	variable	value	
0	2011	A	Agriculture, Forestry and Fishing	a_0	Activity unit	46134	
1	2011	A	Agriculture, Forestry and Fishing	a_0	Rolling mean employees	0	
2	2011	A	Agriculture, Forestry and Fishing	a_0	Salaries and wages paid	279	DOLLAR
3	2011	A	Agriculture, Forestry and Fishing	a_0	Sales, government funding, grants and subsidies	8187	DOLLAR
4	2011	A	Agriculture, Forestry and Fishing	a_0	Total income	8866	DOLLAR
5	2011	A	Agriculture, Forestry and Fishing	a_0	Total expenditure	7618	DOLLAR
6	2011	A	Agriculture, Forestry and Fishing	a_0	Operating profit before tax	770	DOLLAR
7	2011	A	Agriculture, Forestry and Fishing	a_0	Total assets	55700	DOLLAR
8	2011	A	Agriculture, Forestry and Fishing	a_0	Fixed tangible assets	32155	DOLLAR

In [117]:
```
Ndata_csv['value']
```

Out[117]:
```
0    46134
1        0
2      279
3     8187
4     8866
5     7618
6      770
7    55700
8    32155
Name: value, dtype: int64
```

In [118]:
```
#Now summing up all the values.
Ndata_csv['value'].sum()
```

Out[118]:
```
159709
```

Data Selection/View:

Use head('add value') to show as number of rows as one wants. Without entering a value, it will show only first five values. Use tail('add value') to show the number of rows from bottom as per the given value. Mo

In [119]:
```
Ndata_csv
```

Out[119]:

	year	industry_code_ANZSIC	industry_name_ANZSIC	rme_size_grp	variable	value	
0	2011	A	Agriculture, Forestry and Fishing	a_0	Activity unit	46134	
1	2011	A	Agriculture, Forestry and Fishing	a_0	Rolling mean employees	0	
2	2011	A	Agriculture, Forestry and Fishing	a_0	Salaries and wages paid	279	DOLLAR
3	2011	A	Agriculture, Forestry and Fishing	a_0	Sales, government funding, grants and subsidies	8187	DOLLAR
4	2011	A	Agriculture, Forestry and Fishing	a_0	Total income	8866	DOLLAR
5	2011	A	Agriculture, Forestry and Fishing	a_0	Total expenditure	7618	DOLLAR
6	2011	A	Agriculture, Forestry and Fishing	a_0	Operating profit before tax	770	DOLLAR
7	2011	A	Agriculture, Forestry and Fishing	a_0	Total assets	55700	DOLLAR
8	2011	A	Agriculture, Forestry and Fishing	a_0	Fixed tangible assets	32155	DOLLAR

In [120]:
```
Ndata_csv.head()
```

Out[120]:

	year	industry_code_ANZSIC	industry_name_ANZSIC	rme_size_grp	variable	value	
0	2011	A	Agriculture, Forestry and Fishing	a_0	Activity unit	46134	
1	2011	A	Agriculture, Forestry and Fishing	a_0	Rolling mean employees	0	
2	2011	A	Agriculture, Forestry and Fishing	a_0	Salaries and wages paid	279	DOLLAR
3	2011	A	Agriculture, Forestry and Fishing	a_0	Sales, government funding, grants and subsidies	8187	DOLLAR
4	2011	A	Agriculture, Forestry and Fishing	a_0	Total income	8866	DOLLAR

In [121]:
```
Ndata_csv.head(7)
```

Out[121]:

	year	industry_code_ANZSIC	industry_name_ANZSIC	rme_size_grp	variable	value	
0	2011	A	Agriculture, Forestry and Fishing	a_0	Activity unit	46134	
1	2011	A	Agriculture, Forestry and Fishing	a_0	Rolling mean employees	0	
2	2011	A	Agriculture, Forestry and Fishing	a_0	Salaries and wages paid	279	DOLLAR
3	2011	A	Agriculture, Forestry and Fishing	a_0	Sales, government funding, grants and subsidies	8187	DOLLAR
4	2011	A	Agriculture, Forestry and Fishing	a_0	Total income	8866	DOLLAR
5	2011	A	Agriculture, Forestry and Fishing	a_0	Total expenditure	7618	DOLLAR
6	2011	A	Agriculture, Forestry and Fishing	a_0	Operating profit before tax	770	DOLLAR

In [122]:
```
Ndata_csv.tail()
```

Out[122]:

	year	industry_code_ANZSIC	industry_name_ANZSIC	rme_size_grp	variable	value	
4	2011	A	Agriculture, Forestry and Fishing	a_0	Total income	8866	DOLLARS
5	2011	A	Agriculture, Forestry and Fishing	a_0	Total expenditure	7618	DOLLARS
6	2011	A	Agriculture, Forestry and Fishing	a_0	Operating profit before tax	770	DOLLARS
7	2011	A	Agriculture, Forestry and Fishing	a_0	Total assets	55700	DOLLARS
8	2011	A	Agriculture, Forestry and Fishing	a_0	Fixed tangible assets	32155	DOLLARS

In [123]:
```
Ndata_csv.tail(7)
```

Out[123]:

	year	industry_code_ANZSIC	industry_name_ANZSIC	rme_size_grp	variable	value	
2	2011	A	Agriculture, Forestry and Fishing	a_0	Salaries and wages paid	279	DOLLAR
3	2011	A	Agriculture, Forestry and Fishing	a_0	Sales, government funding, grants and subsidies	8187	DOLLAR
4	2011	A	Agriculture, Forestry and Fishing	a_0	Total income	8866	DOLLAR
5	2011	A	Agriculture, Forestry and Fishing	a_0	Total expenditure	7618	DOLLAR
6	2011	A	Agriculture, Forestry and Fishing	a_0	Operating profit before tax	770	DOLLAR
7	2011	A	Agriculture, Forestry and Fishing	a_0	Total assets	55700	DOLLAR
8	2011	A	Agriculture, Forestry and Fishing	a_0	Fixed tangible assets	32155	DOLLAR

Index (.loc) or Location(.iloc):

How an individual data can be accessed using index or location? .loc used to carry index while .iloc is used to carry the value at that partiular position

In [124]:
```
birds = pd.Series(['crow','pegion','parrot','sparrow','eagle'],index = [0,3,9,8,3
```

In [125]:
```
birds
```

Out[125]:
```
0       crow
3     pegion
9     parrot
8    sparrow
3      eagle
dtype: object
```

In [126]:
```
birds.loc[8]
```

Out[126]:
```
'sparrow'
```

In [127]:
```
birds.iloc[3]
```

Out[127]:
```
'sparrow'
```

In [128]:
```
Ndata_csv
```

Out[128]:

	year	industry_code_ANZSIC	industry_name_ANZSIC	rme_size_grp	variable	value	
0	2011	A	Agriculture, Forestry and Fishing	a_0	Activity unit	46134	
1	2011	A	Agriculture, Forestry and Fishing	a_0	Rolling mean employees	0	
2	2011	A	Agriculture, Forestry and Fishing	a_0	Salaries and wages paid	279	DOLLAR
3	2011	A	Agriculture, Forestry and Fishing	a_0	Sales, government funding, grants and subsidies	8187	DOLLAR
4	2011	A	Agriculture, Forestry and Fishing	a_0	Total income	8866	DOLLAR
5	2011	A	Agriculture, Forestry and Fishing	a_0	Total expenditure	7618	DOLLAR
6	2011	A	Agriculture, Forestry and Fishing	a_0	Operating profit before tax	770	DOLLAR
7	2011	A	Agriculture, Forestry and Fishing	a_0	Total assets	55700	DOLLAR
8	2011	A	Agriculture, Forestry and Fishing	a_0	Fixed tangible assets	32155	DOLLAR

In [129]:
```
Ndata_csv.loc[3]
```

Out[129]:
```
year                                                    2011
industry_code_ANZSIC                                       A
industry_name_ANZSIC                  Agriculture, Forestry and Fishing
rme_size_grp                                             a_0
variable                Sales, government funding, grants and subsidies
value                                                   8187
unit                                          DOLLARS(millions)
Name: 3, dtype: object
```

In [130]:
```
Ndata_csv.iloc[3]
```

Out[130]:
```
year                                                    2011
industry_code_ANZSIC                                       A
industry_name_ANZSIC                  Agriculture, Forestry and Fishing
rme_size_grp                                             a_0
variable                Sales, government funding, grants and subsidies
value                                                   8187
unit                                          DOLLARS(millions)
Name: 3, dtype: object
```

In [131]:
```
#to show the data of multiple indices using iloc[] and loc[] method.
Ndata_csv.loc[0:3]
```

Out[131]:

	year	industry_code_ANZSIC	industry_name_ANZSIC	rme_size_grp	variable	value	
0	2011	A	Agriculture, Forestry and Fishing	a_0	Activity unit	46134	
1	2011	A	Agriculture, Forestry and Fishing	a_0	Rolling mean employees	0	
2	2011	A	Agriculture, Forestry and Fishing	a_0	Salaries and wages paid	279	DOLLAR
3	2011	A	Agriculture, Forestry and Fishing	a_0	Sales, government funding, grants and subsidies	8187	DOLLAR

In [132]:
```
Ndata_csv.iloc[0:3]
#here iloc is starting exiting before 3. This is the diference with loc
```

Out[132]:

	year	industry_code_ANZSIC	industry_name_ANZSIC	rme_size_grp	variable	value	
0	2011	A	Agriculture, Forestry and Fishing	a_0	Activity unit	46134	
1	2011	A	Agriculture, Forestry and Fishing	a_0	Rolling mean employees	0	
2	2011	A	Agriculture, Forestry and Fishing	a_0	Salaries and wages paid	279	DOLLARS

In [133]:
```
Ndata_csv.head(3)
#similar procedure can be performed with .head() function.
```

Out[133]:

	year	industry_code_ANZSIC	industry_name_ANZSIC	rme_size_grp	variable	value	
0	2011	A	Agriculture, Forestry and Fishing	a_0	Activity unit	46134	
1	2011	A	Agriculture, Forestry and Fishing	a_0	Rolling mean employees	0	
2	2011	A	Agriculture, Forestry and Fishing	a_0	Salaries and wages paid	279	DOLLARS

In [134]:
```
#to show particular column.
Ndata_csv['variable']
```

Out[134]:
```
0                                    Activity unit
1                            Rolling mean employees
2                           Salaries and wages paid
3    Sales, government funding, grants and subsidies
4                                     Total income
5                                Total expenditure
6                       Operating profit before tax
7                                     Total assets
8                              Fixed tangible assets
Name: variable, dtype: object
```

In [135]:
```
Ndata_csv.variable
#note if there is space in the name of column, this method would not be performed
```

Out[135]:
```
0                                    Activity unit
1                              Rolling mean employees
2                             Salaries and wages paid
3    Sales, government funding, grants and subsidies
4                                       Total income
5                                  Total expenditure
6                          Operating profit before tax
7                                       Total assets
8                                Fixed tangible assets
Name: variable, dtype: object
```

In [136]:
```
Ndata_csv
```

Out[136]:

	year	industry_code_ANZSIC	industry_name_ANZSIC	rme_size_grp	variable	value	
0	2011	A	Agriculture, Forestry and Fishing	a_0	Activity unit	46134	
1	2011	A	Agriculture, Forestry and Fishing	a_0	Rolling mean employees	0	
2	2011	A	Agriculture, Forestry and Fishing	a_0	Salaries and wages paid	279	DOLLAR
3	2011	A	Agriculture, Forestry and Fishing	a_0	Sales, government funding, grants and subsidies	8187	DOLLAR
4	2011	A	Agriculture, Forestry and Fishing	a_0	Total income	8866	DOLLAR
5	2011	A	Agriculture, Forestry and Fishing	a_0	Total expenditure	7618	DOLLAR
6	2011	A	Agriculture, Forestry and Fishing	a_0	Operating profit before tax	770	DOLLAR
7	2011	A	Agriculture, Forestry and Fishing	a_0	Total assets	55700	DOLLAR
8	2011	A	Agriculture, Forestry and Fishing	a_0	Fixed tangible assets	32155	DOLLAR

In [137]:
```
#How to select data data conditionally? data-name [ Data-name[column within dataf
Ndata_csv[Ndata_csv['variable']== 'Total income']
```

Out[137]:

	year	industry_code_ANZSIC	industry_name_ANZSIC	rme_size_grp	variable	value	
4	2011	A	Agriculture, Forestry and Fishing	a_0	Total income	8866	DOLLARS(m

In [138]:
```
Ndata_csv[Ndata_csv['value']>=7618]
```

Out[138]:

	year	industry_code_ANZSIC	industry_name_ANZSIC	rme_size_grp	variable	value	
0	2011	A	Agriculture, Forestry and Fishing	a_0	Activity unit	46134	
3	2011	A	Agriculture, Forestry and Fishing	a_0	Sales, government funding, grants and subsidies	8187	DOLLAR
4	2011	A	Agriculture, Forestry and Fishing	a_0	Total income	8866	DOLLAR
5	2011	A	Agriculture, Forestry and Fishing	a_0	Total expenditure	7618	DOLLAR
7	2011	A	Agriculture, Forestry and Fishing	a_0	Total assets	55700	DOLLAR
8	2011	A	Agriculture, Forestry and Fishing	a_0	Fixed tangible assets	32155	DOLLAR

Data Selection Section 2

In [139]:
```
#How to comapre two column? Cross-Tab which is called upon panda/pd as .pdcrosstc
pd.crosstab(Ndata_csv['value'], Ndata_csv['variable'])
```

Out[139]:

variable	Activity unit	Fixed tangible assets	Operating profit before tax	Rolling mean employees	Salaries and wages paid	Sales, government funding, grants and subsidies	Total assets	Total expenditure	To inco
value									
0	0	0	0	1	0	0	0	0	
279	0	0	0	0	1	0	0	0	
770	0	0	1	0	0	0	0	0	
7618	0	0	0	0	0	0	0	1	
8187	0	0	0	0	0	1	0	0	
8866	0	0	0	0	0	0	0	0	
32155	0	1	0	0	0	0	0	0	
46134	1	0	0	0	0	0	0	0	
55700	0	0	0	0	0	0	1	0	

1 Indicate that both particular-index and column head exists in same row. What if one column has to be compared with all? Use groupby.([column head]).mean(): it will compare the given column with all other columns of integer type.

In [140]:
```
Ndata_csv.groupby(['variable']).mean()
```

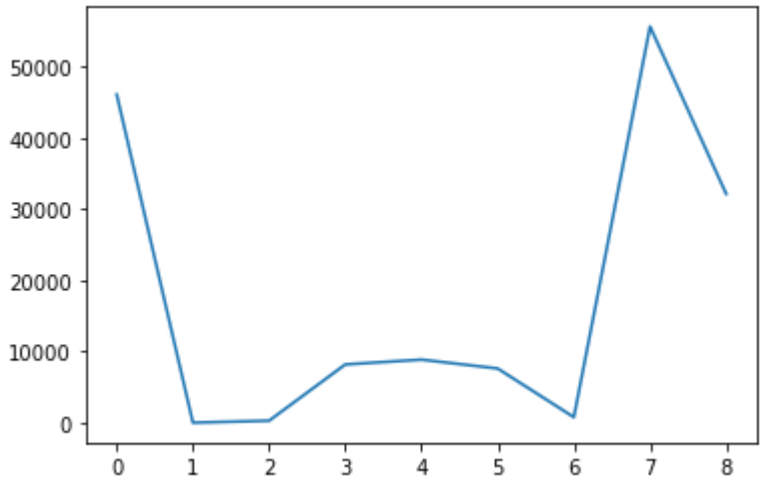
Out[140]:

variable	year	value
Activity unit	2011	46134
Fixed tangible assets	2011	32155
Operating profit before tax	2011	770
Rolling mean employees	2011	0
Salaries and wages paid	2011	279
Sales, government funding, grants and subsidies	2011	8187
Total assets	2011	55700
Total expenditure	2011	7618
Total income	2011	8866

DATA PLOTING

In [141]:
```
Ndata_csv['value'].plot()
```

Out[141]:
```
<AxesSubplot:>
```

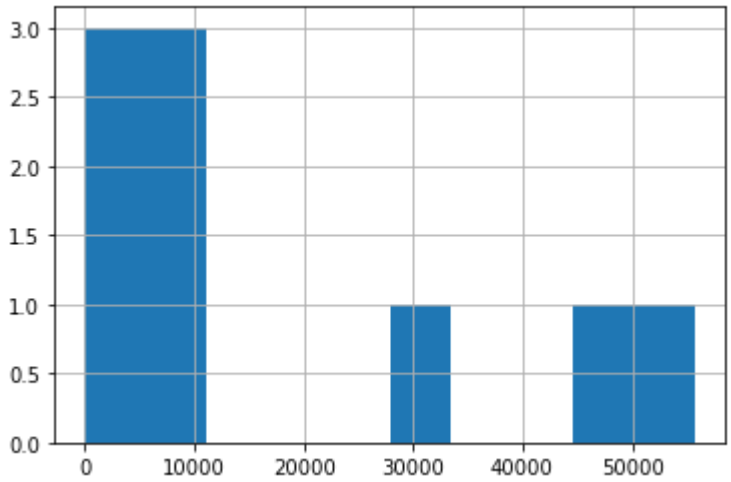


in backgroung panda import matplotlib. if not working use, matplotlib inline import matplotlib.pyplot as plt

In [142]:
```
# Histogram plot: It tells about the spread of data.

Ndata_csv['value'].hist()
```

Out[142]: <AxesSubplot:>



It shows values from 0-20000 are 3, 20000-40000 is 1 and 40000-50000(and above) is also 1.

Reformating

Reformating: if values are given in string type, to convert these given values into integer reformating is used

In [143]:
```
#before that it was not integer, now it coverted into integer, thats why showing
Ndata_csv['rme_size_grp']
```

Out[143]:
```
0    a_0
1    a_0
2    a_0
3    a_0
4    a_0
5    a_0
6    a_0
7    a_0
8    a_0
Name: rme_size_grp, dtype: object
```

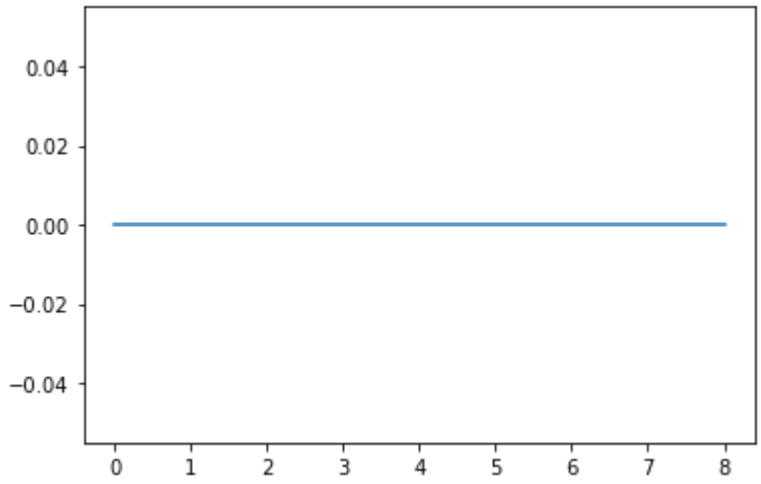
In [144]:
```
#to replace string character use: .str.replace('[to be replaced]','empty space')
#to covert string type to int type use:astype(int)
Ndata_csv['rme_size_grp'].str.replace('[a,_]','').astype(int)
```

Out[144]:
```
0    0
1    0
2    0
3    0
4    0
5    0
6    0
7    0
8    0
Name: rme_size_grp, dtype: int32
```

In [145]:
```
#Ploting rme_size_grp which was string type initially, now has been converted to
Ndata_csv['rme_size_grp'] = Ndata_csv['rme_size_grp'].str.replace('[a,_]','').ast
```

In [146]:
```
Ndata_csv['rme_size_grp'].plot()
```

Out[146]: <AxesSubplot:>



In [147]:
```
#it can also be plot by calling plot() right after converting file str into int.
```

Changing Data

In [148]:
```
#In order to change data from capital to small we use str.lower
Ndata_csv['unit'].str.lower()
```

Out[148]:
```
0                 count
1                 count
2    dollars(millions)
3    dollars(millions)
4    dollars(millions)
5    dollars(millions)
6    dollars(millions)
7    dollars(millions)
8    dollars(millions)
Name: unit, dtype: object
```

In [149]:
```
Ndata_missing = pd.read_csv('marine-economy.csv')
```

In [150]:
```
Ndata_missing
```

Out[150]:

	year	category	variable	units	magnitude	source	data_value	flag
0	2007	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	43.1	R
1	2007	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	NaN	F
2	2007	Fisheries and aquaculture	Contribution to marine economy earnings	Proportion	Actual	LEED	42.7	R
3	2007	Fisheries and aquaculture	Contribution to total GDP	Proportion	Actual	Environmental Accounts	NaN	F
4	2007	Fisheries and aquaculture	GDP	Dollars	Thousands	Environmental Accounts	715722.0	F
5	2007	Fisheries and aquaculture	Gross earnings	Dollars	Thousands	LEED	582377.0	F
6	2007	Fisheries and aquaculture	Wage and salary earners	Number	Actual	LEED	NaN	F
7	2008	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	39.9	R
8	2008	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	14.2	F

In [151]:
```
#column with empty values.
Ndata_missing['data_value']
```

Out[151]:
```
0        43.1
1         NaN
2        42.7
3         NaN
4    715722.0
5    582377.0
6         NaN
7        39.9
8        14.2
Name: data_value, dtype: float64
```

In [152]:
```
#Lets show mean values at empty places:NaN. Firstly, we find out the mean.
Ndata_missing['data_value'].mean()
```

Out[152]:
```
216373.15
```

In [153]:
```
#In order to place this mean value in column we use ' Data-name['column'].fillna(
Ndata_missing['data_value'].fillna(Ndata_missing['data_value'].mean())
```

Out[153]:
```
0        43.10
1    216373.15
2        42.70
3    216373.15
4    715722.00
5    582377.00
6    216373.15
7        39.90
8        14.20
Name: data_value, dtype: float64
```

The fed mean value at NaN places is not stored but apparently shown. To store this too, it can also be done using 'inplace=true

In [154]:
```
Ndata_missing['data_value'].fillna(Ndata_missing['data_value'].mean(), inplace=Tr
```

In [155]:
```
Ndata_missing
```

Out[155]:

	year	category	variable	units	magnitude	source	data_value	flag
0	2007	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	43.10	R
1	2007	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	216373.15	F
2	2007	Fisheries and aquaculture	Contribution to marine economy earnings	Proportion	Actual	LEED	42.70	R
3	2007	Fisheries and aquaculture	Contribution to total GDP	Proportion	Actual	Environmental Accounts	216373.15	F
4	2007	Fisheries and aquaculture	GDP	Dollars	Thousands	Environmental Accounts	715722.00	F
5	2007	Fisheries and aquaculture	Gross earnings	Dollars	Thousands	LEED	582377.00	F
6	2007	Fisheries and aquaculture	Wage and salary earners	Number	Actual	LEED	216373.15	F
7	2008	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	39.90	R
8	2008	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	14.20	F

See the data is stored and change in original data is occured. This task can also be performed by assigning the values to that particular column. In this inplace will not be used.

In [156]:
```
Ndata_missing['data_value'] = Ndata_missing['data_value'].fillna(Ndata_missing['d
```

In [157]:
```
Ndata_missing
```

Out[157]:

	year	category	variable	units	magnitude	source	data_value	flag
0	2007	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	43.10	R
1	2007	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	216373.15	F
2	2007	Fisheries and aquaculture	Contribution to marine economy earnings	Proportion	Actual	LEED	42.70	R
3	2007	Fisheries and aquaculture	Contribution to total GDP	Proportion	Actual	Environmental Accounts	216373.15	F
4	2007	Fisheries and aquaculture	GDP	Dollars	Thousands	Environmental Accounts	715722.00	F
5	2007	Fisheries and aquaculture	Gross earnings	Dollars	Thousands	LEED	582377.00	F
6	2007	Fisheries and aquaculture	Wage and salary earners	Number	Actual	LEED	216373.15	F
7	2008	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	39.90	R
8	2008	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	14.20	F

NaN value can also be removed by using .dropna(). To bring changes in original data .dropna(inplace=true).

In [158]:
```
#again to have original data with Nan value we import data file.
Ndata_missing = pd.read_csv('marine-economy.csv')
```

In [159]:
```
Ndata_missing
```

Out[159]:

	year	category	variable	units	magnitude	source	data_value	flag
0	2007	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	43.1	R
1	2007	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	NaN	F
2	2007	Fisheries and aquaculture	Contribution to marine economy earnings	Proportion	Actual	LEED	42.7	R
3	2007	Fisheries and aquaculture	Contribution to total GDP	Proportion	Actual	Environmental Accounts	NaN	F
4	2007	Fisheries and aquaculture	GDP	Dollars	Thousands	Environmental Accounts	715722.0	F
5	2007	Fisheries and aquaculture	Gross earnings	Dollars	Thousands	LEED	582377.0	F
6	2007	Fisheries and aquaculture	Wage and salary earners	Number	Actual	LEED	NaN	F
7	2008	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	39.9	R
8	2008	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	14.2	F

In [160]:
```
#Changing in original data can also be performed by assinging this to variable i.
Ndata_missing.dropna(inplace = True)
```

In [161]:
```
Ndata_missing
```

Out[161]:

	year	category	variable	units	magnitude	source	data_value	flag
0	2007	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	43.1	R
2	2007	Fisheries and aquaculture	Contribution to marine economy earnings	Proportion	Actual	LEED	42.7	R
4	2007	Fisheries and aquaculture	GDP	Dollars	Thousands	Environmental Accounts	715722.0	F
5	2007	Fisheries and aquaculture	Gross earnings	Dollars	Thousands	LEED	582377.0	F
7	2008	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	39.9	R
8	2008	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	14.2	F

Now we can extract this file too which is cleaned from NaN values. Here the method it

In [162]:
```
Ndata_missing.to_csv('cleaned-maritime-economy', index=False)
```

Add, Remove &Change Column:

In [163]:
```
cleaned_data = pd.read_csv('cleaned-maritime-economy')
```

In [164]:
```
cleaned_data
```

Out[164]:

	year	category	variable	units	magnitude	source	data_value	flag
0	2007	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	43.1	R
1	2007	Fisheries and aquaculture	Contribution to marine economy earnings	Proportion	Actual	LEED	42.7	R
2	2007	Fisheries and aquaculture	GDP	Dollars	Thousands	Environmental Accounts	715722.0	F
3	2007	Fisheries and aquaculture	Gross earnings	Dollars	Thousands	LEED	582377.0	F
4	2008	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	39.9	R
5	2008	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	14.2	F

In [165]:
```
earned_value = pd.Series([3,4,5,2,4])
cleaned_data['earned_value']= earned_value
```

In [166]:
```
cleaned_data
```

Out[166]:

	year	category	variable	units	magnitude	source	data_value	flag	earned_val
0	2007	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	43.1	R	3
1	2007	Fisheries and aquaculture	Contribution to marine economy earnings	Proportion	Actual	LEED	42.7	R	4
2	2007	Fisheries and aquaculture	GDP	Dollars	Thousands	Environmental Accounts	715722.0	F	5
3	2007	Fisheries and aquaculture	Gross earnings	Dollars	Thousands	LEED	582377.0	F	2
4	2008	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	39.9	R	4
5	2008	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	14.2	F	Na

In [167]:
```
cleaned_data['earned_value'].fillna(5, inplace=True)
```

In [168]:
```
cleaned_data
```

Out[168]:

	year	category	variable	units	magnitude	source	data_value	flag	earned_val
0	2007	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	43.1	R	3
1	2007	Fisheries and aquaculture	Contribution to marine economy earnings	Proportion	Actual	LEED	42.7	R	4
2	2007	Fisheries and aquaculture	GDP	Dollars	Thousands	Environmental Accounts	715722.0	F	5
3	2007	Fisheries and aquaculture	Gross earnings	Dollars	Thousands	LEED	582377.0	F	2
4	2008	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	39.9	R	4
5	2008	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	14.2	F	5

In list dimensions of new-added-column must be as equal to the dimention of data otherwise error would be occured. On the other hand, series remain unscathed from these errors or dimensions of new column are not compulsory to be as equal to dimentions of data.

In [169]:
```
status = [1,4,6,7,8]
cleaned_data['status'] = status
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-169-394580ed3f5c> in <module>
      1 status = [1,4,6,7,8]
----> 2 cleaned_data['status'] = status

C:\Users\AI.khan\desktop\f_project\venv\lib\site-packages\pandas\core\frame.py
 in __setitem__(self, key, value)
   3038         else:
   3039             # set column
-> 3040             self._set_item(key, value)
   3041
   3042     def _setitem_slice(self, key: slice, value):

C:\Users\AI.khan\desktop\f_project\venv\lib\site-packages\pandas\core\frame.py
 in _set_item(self, key, value)
   3114         """
   3115         self._ensure_valid_index(value)
-> 3116         value = self._sanitize_column(key, value)
   3117         NDFrame._set_item(self, key, value)
   3118

C:\Users\AI.khan\desktop\f_project\venv\lib\site-packages\pandas\core\frame.py
 in _sanitize_column(self, key, value, broadcast)
   3762
   3763             # turn me into an ndarray
-> 3764             value = sanitize_index(value, self.index)
   3765             if not isinstance(value, (np.ndarray, Index)):
   3766                 if isinstance(value, list) and len(value) > 0:

C:\Users\AI.khan\desktop\f_project\venv\lib\site-packages\pandas\core\internals
\construction.py in sanitize_index(data, index)
    745     """
    746     if len(data) != len(index):
--> 747         raise ValueError(
    748             "Length of values "
    749             f"({len(data)}) "

ValueError: Length of values (5) does not match length of index (6)
```

In [170]:
```
status = [1,4,6,7,8]
cleaned_data['status'] = status
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-170-394580ed3f5c> in <module>
      1 status = [1,4,6,7,8]
----> 2 cleaned_data['status'] = status

C:\Users\AI.khan\desktop\f_project\venv\lib\site-packages\pandas\core\frame.py
 in __setitem__(self, key, value)
   3038         else:
   3039             # set column
-> 3040             self._set_item(key, value)
   3041
   3042     def _setitem_slice(self, key: slice, value):

C:\Users\AI.khan\desktop\f_project\venv\lib\site-packages\pandas\core\frame.py
 in _set_item(self, key, value)
   3114         """
   3115         self._ensure_valid_index(value)
-> 3116         value = self._sanitize_column(key, value)
   3117         NDFrame._set_item(self, key, value)
   3118

C:\Users\AI.khan\desktop\f_project\venv\lib\site-packages\pandas\core\frame.py
 in _sanitize_column(self, key, value, broadcast)
   3762
   3763             # turn me into an ndarray
-> 3764             value = sanitize_index(value, self.index)
   3765             if not isinstance(value, (np.ndarray, Index)):
   3766                 if isinstance(value, list) and len(value) > 0:

C:\Users\AI.khan\desktop\f_project\venv\lib\site-packages\pandas\core\internals
\construction.py in sanitize_index(data, index)
    745     """
    746     if len(data) != len(index):
--> 747         raise ValueError(
    748             "Length of values "
    749             f"({len(data)}) "

ValueError: Length of values (5) does not match length of index (6)
```

In [171]:
```
cleaned_data
```

Out[171]:

	year	category	variable	units	magnitude	source	data_value	flag	earned_val
0	2007	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	43.1	R	3
1	2007	Fisheries and aquaculture	Contribution to marine economy earnings	Proportion	Actual	LEED	42.7	R	4
2	2007	Fisheries and aquaculture	GDP	Dollars	Thousands	Environmental Accounts	715722.0	F	5
3	2007	Fisheries and aquaculture	Gross earnings	Dollars	Thousands	LEED	582377.0	F	2
4	2008	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	39.9	R	4
5	2008	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	14.2	F	5

In [172]:
```
#new column after performing operation on other two columns
cleaned_data['product'] = cleaned_data['earned_value']*cleaned_data['status']
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
C:\Users\AI.khan\desktop\f_project\venv\lib\site-packages\pandas\core\indexes\base.py in get_loc(self, key, method, tolerance)
   2894             try:
-> 2895                 return self._engine.get_loc(casted_key)
   2896             except KeyError as err:

pandas\_libs\index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas\_libs\index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas\_libs\hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.get_item()

pandas\_libs\hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.get_item()

KeyError: 'status'

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call last)
<ipython-input-172-49366b8848b8> in <module>
      1 #new column after performing operation on other two columns
----> 2 cleaned_data['product'] = cleaned_data['earned_value']*cleaned_data['status']

C:\Users\AI.khan\desktop\f_project\venv\lib\site-packages\pandas\core\frame.py in __getitem__(self, key)
   2900             if self.columns.nlevels > 1:
   2901                 return self._getitem_multilevel(key)
-> 2902             indexer = self.columns.get_loc(key)
   2903             if is_integer(indexer):
   2904                 indexer = [indexer]

C:\Users\AI.khan\desktop\f_project\venv\lib\site-packages\pandas\core\indexes\base.py in get_loc(self, key, method, tolerance)
   2895                 return self._engine.get_loc(casted_key)
   2896             except KeyError as err:
-> 2897                 raise KeyError(key) from err
   2898
   2899         if tolerance is not None:

KeyError: 'status'
```

In [ ]:
```
cleaned_data
```

In [173]:
```
#Generating column on bolean
cleaned_data['w-status'] = True
```

In [174]:
```
cleaned_data
```

Out[174]:

	year	category	variable	units	magnitude	source	data_value	flag	earned_valu
0	2007	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	43.1	R	3
1	2007	Fisheries and aquaculture	Contribution to marine economy earnings	Proportion	Actual	LEED	42.7	R	4
2	2007	Fisheries and aquaculture	GDP	Dollars	Thousands	Environmental Accounts	715722.0	F	5
3	2007	Fisheries and aquaculture	Gross earnings	Dollars	Thousands	LEED	582377.0	F	2
4	2008	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	39.9	R	4
5	2008	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	14.2	F	5

.drop('column name', axis=1): to remove any column. axis=1 is mentioned because column is represented by axis = 1 while row with axis = 0.

In [175]:
```
cleaned_data.drop('w-status',axis=1)
```

Out[175]:

	year	category	variable	units	magnitude	source	data_value	flag	earned_val
0	2007	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	43.1	R	3
1	2007	Fisheries and aquaculture	Contribution to marine economy earnings	Proportion	Actual	LEED	42.7	R	4
2	2007	Fisheries and aquaculture	GDP	Dollars	Thousands	Environmental Accounts	715722.0	F	5
3	2007	Fisheries and aquaculture	Gross earnings	Dollars	Thousands	LEED	582377.0	F	2
4	2008	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	39.9	R	4
5	2008	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	14.2	F	5

Manipulating/Shuffling of Data

In order to shuffle data .sample(frac=0-1) is used. 0 for not to shuffle any data. 1 to shuffle complete data. 0.5 to shuffle 50 percent data.

In [176]:
```
cleaned_data.sample(frac=1)
```

Out[176]:

	year	category	variable	units	magnitude	source	data_value	flag	earned_val
3	2007	Fisheries and aquaculture	Gross earnings	Dollars	Thousands	LEED	582377.0	F	2
5	2008	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	14.2	F	5
4	2008	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	39.9	R	4
0	2007	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	43.1	R	3
2	2007	Fisheries and aquaculture	GDP	Dollars	Thousands	Environmental Accounts	715722.0	F	5
1	2007	Fisheries and aquaculture	Contribution to marine economy earnings	Proportion	Actual	LEED	42.7	R	4

In order to reset data .reset_index(drop=true, inplace=true). drop=true is used to remove the appearance of extra index and inplace=true is also used to bring changes in original data.

In [191]:
```
cleaned_data.reset_index(drop=True, inplace=True)
```

In [192]:
```
cleaned_data
```

Out[192]:

	year	category	variable	units	magnitude	source	data_value	flag	earned_val
0	2007	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	43.1	R	3
1	2007	Fisheries and aquaculture	Contribution to marine economy earnings	Proportion	Actual	LEED	42.7	R	4
2	2007	Fisheries and aquaculture	GDP	Dollars	Thousands	Environmental Accounts	715722.0	F	5
3	2007	Fisheries and aquaculture	Gross earnings	Dollars	Thousands	LEED	582377.0	F	2
4	2008	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	39.9	R	4
5	2008	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	14.2	F	5

Applying Function Upon Column

.apply(lamda x:x/1000) is called. lamda is a anonymous function we use this keyword to define our function.

In [194]:
```
#performing function upon earned-data and dividing it by 2.
cleaned_data['earned_value']= cleaned_data['earned_value'].apply(lambda x:x/2)
```

In [195]:
```
cleaned_data
```

Out[195]:

	year	category	variable	units	magnitude	source	data_value	flag	earned_val
0	2007	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	43.1	R	1
1	2007	Fisheries and aquaculture	Contribution to marine economy earnings	Proportion	Actual	LEED	42.7	R	2
2	2007	Fisheries and aquaculture	GDP	Dollars	Thousands	Environmental Accounts	715722.0	F	2
3	2007	Fisheries and aquaculture	Gross earnings	Dollars	Thousands	LEED	582377.0	F	1
4	2008	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	39.9	R	2
5	2008	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	14.2	F	2

In [ ]:
```

```