# 1 第一章：Pandas的基础知识

数据小鱼Rexa
CSDN：https://blog.csdn.net/qq_38395376?spm=1011.2124.3001.5343
Blibli：https://space.bilibili.com/283181288
Github：https://github.com/Rexa-Yu

## 1.1 加载数据集

In [1]:

```
import pandas as pd
data=pd.read_csv(r'E:\jupyter notebook storage\Practice in Pandas\data\gapminder.tsv
data.head(5)
```

Out[1]:

| | country | continent | year | lifeExp | pop | gdpPercap |
|---|---|---|---|---|---|---|
| 0 | Afghanistan | Asia | 1952 | 28.801 | 8425333 | 779.445314 |
| 1 | Afghanistan | Asia | 1957 | 30.332 | 9240934 | 820.853030 |
| 2 | Afghanistan | Asia | 1962 | 31.997 | 10267083 | 853.100710 |
| 3 | Afghanistan | Asia | 1967 | 34.020 | 11537966 | 836.197138 |
| 4 | Afghanistan | Asia | 1972 | 36.088 | 13079460 | 739.981106 |

## 1.2 数据集的描述性分析函数describe()

In [2]:

```
data.describe()
```

Out[2]:

| | year | lifeExp | pop | gdpPercap |
|---|---|---|---|---|
| count | 1704.00000 | 1704.000000 | 1.704000e+03 | 1704.000000 |
| mean | 1979.50000 | 59.474439 | 2.960121e+07 | 7215.327081 |
| std | 17.26533 | 12.917107 | 1.061579e+08 | 9857.454543 |
| min | 1952.00000 | 23.599000 | 6.001100e+04 | 241.165877 |
| 25% | 1965.75000 | 48.198000 | 2.793664e+06 | 1202.060309 |
| 50% | 1979.50000 | 60.712500 | 7.023596e+06 | 3531.846989 |
| 75% | 1993.25000 | 70.845500 | 1.958522e+07 | 9325.462346 |
| max | 2007.00000 | 82.603000 | 1.318683e+09 | 113523.132900 |

### 1.2.1 查看数据类别type和强制转化格式DataFrame

In [3]:

```
print(type(data))
data=pd.DataFrame(data)
```

```
<class 'pandas.core.frame.DataFrame'>
```

### 1.2.2 获取行数和列数shape(shape函数后面无括号）

In [4]:

```
data.shape
```

Out[4]:

```
(1704, 6)
```

### 1.2.3 获取列名（与shape一样不需要加括号）

In [5]:

```
data.columns
```

Out[5]:

```
Index(['country', 'continent', 'year', 'lifeExp', 'pop', 'gdpPercap'],
dtype='object')
```

### 1.2.4 查看更多的数据信息可以用info()或者dtypes

In [6]:

```
print(data.info())
data.dtypes
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1704 entries, 0 to 1703
Data columns (total 6 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   country    1704 non-null   object 
 1   continent  1704 non-null   object 
 2   year       1704 non-null   int64  
 3   lifeExp    1704 non-null   float64
 4   pop        1704 non-null   int64  
 5   gdpPercap  1704 non-null   float64
dtypes: float64(2), int64(2), object(2)
memory usage: 80.0+ KB
None
```

Out[6]:

```
country       object
continent     object
year           int64
lifeExp      float64
pop            int64
gdpPercap    float64
dtype: object
```

# 1.3 查看列和单元格

## 1.3.1 显示前5行和后5行

In [7]:

```
print(data.head())
print(data.tail())
```

```
       country continent  year  lifeExp       pop   gdpPercap
0  Afghanistan      Asia  1952   28.801   8425333  779.445314
1  Afghanistan      Asia  1957   30.332   9240934  820.853030
2  Afghanistan      Asia  1962   31.997  10267083  853.100710
3  Afghanistan      Asia  1967   34.020  11537966  836.197138
4  Afghanistan      Asia  1972   36.088  13079460  739.981106
       country continent  year  lifeExp       pop   gdpPercap
1699  Zimbabwe    Africa  1987   62.351   9216418  706.157306
1700  Zimbabwe    Africa  1992   60.377  10704340  693.420786
1701  Zimbabwe    Africa  1997   46.809  11404948  792.449960
1702  Zimbabwe    Africa  2002   39.989  11926563  672.038623
1703  Zimbabwe    Africa  2007   43.487  12311143  469.709298
```

## 1.3.2 获取子集loc和iloc

In [8]:

```
data.iloc[:,1]
```

Out[8]:

```
0          Asia
1          Asia
2          Asia
3          Asia
4          Asia
          ...
1699     Africa
1700     Africa
1701     Africa
1702     Africa
1703     Africa
Name: continent, Length: 1704, dtype: object
```

In [9]:

```
data.loc[1]
```

Out[9]:

```
country      Afghanistan
continent           Asia
year                1957
lifeExp           30.332
pop              9240934
gdpPercap        820.853
Name: 1, dtype: object
```

### 1.3.3 获取子集列 使用loc和iloc

In [10]:

```
subset=data.loc[:,['year','pop']]
subset.head(5)
```

Out[10]:

| | year | pop |
|---|---|---|
| 0 | 1952 | 8425333 |
| 1 | 1957 | 9240934 |
| 2 | 1962 | 10267083 |
| 3 | 1967 | 11537966 |
| 4 | 1972 | 13079460 |

In [11]:

```
subset1=data.iloc[:,[2,4,-1]]
subset1.head()
```

Out[11]:

| | year | pop | gdpPercap |
|---|---|---|---|
| 0 | 1952 | 8425333 | 779.445314 |
| 1 | 1957 | 9240934 | 820.853030 |
| 2 | 1962 | 10267083 | 853.100710 |
| 3 | 1967 | 11537966 | 836.197138 |
| 4 | 1972 | 13079460 | 739.981106 |

## 1.3.4 Python 切片法使用

In [12]:

```
subset2=data.iloc[:,0:3:1]
subset2.head()
```

Out[12]:

| | country | continent | year |
|---|---|---|---|
| 0 | Afghanistan | Asia | 1952 |
| 1 | Afghanistan | Asia | 1957 |
| 2 | Afghanistan | Asia | 1962 |
| 3 | Afghanistan | Asia | 1967 |
| 4 | Afghanistan | Asia | 1972 |

In [13]:

```
subset3=data.iloc[:,0::2]
subset3.head()
```

Out[13]:

| | country | year | pop |
|---|---|---|---|
| 0 | Afghanistan | 1952 | 8425333 |
| 1 | Afghanistan | 1957 | 9240934 |
| 2 | Afghanistan | 1962 | 10267083 |
| 3 | Afghanistan | 1967 | 11537966 |
| 4 | Afghanistan | 1972 | 13079460 |

In [14]:

```
subset4=data.iloc[:,0:6:]
subset4.head()
```

Out[14]:

| | country | continent | year | lifeExp | pop | gdpPercap |
|---|---|---|---|---|---|---|
| 0 | Afghanistan | Asia | 1952 | 28.801 | 8425333 | 779.445314 |
| 1 | Afghanistan | Asia | 1957 | 30.332 | 9240934 | 820.853030 |
| 2 | Afghanistan | Asia | 1962 | 31.997 | 10267083 | 853.100710 |
| 3 | Afghanistan | Asia | 1967 | 34.020 | 11537966 | 836.197138 |
| 4 | Afghanistan | Asia | 1972 | 36.088 | 13079460 | 739.981106 |

# 1.4 分组与聚合

## 1.4.1 分组方式groupby函数

In [15]:

```
group=data.groupby('year')['lifeExp'].mean()
group
```

Out[15]:

```
year
1952    49.057620
1957    51.507401
1962    53.609249
1967    55.678290
1972    57.647386
1977    59.570157
1982    61.533197
1987    63.212613
1992    64.160338
1997    65.014676
2002    65.694923
2007    67.007423
Name: lifeExp, dtype: float64
```

In [16]:

```
group1=data.groupby(['year','continent'])[['lifeExp','gdpPercap']].mean()
group1
```

Out[16]:

| | | lifeExp | gdpPercap |
|---|---|---|---|
| year | continent | | |
| | Africa | 39.135500 | 1252.572466 |
| | Americas | 53.279840 | 4079.062552 |
| 1952 | Asia | 46.314394 | 5195.484004 |
| | Europe | 64.408500 | 5661.057435 |
| | Oceania | 69.255000 | 10298.085650 |
| | Africa | 41.266346 | 1385.236062 |
| | Americas | 55.960280 | 4616.043733 |
| 1957 | Asia | 49.318544 | 5787.732940 |
| | Europe | 66.703067 | 6963.012816 |
| | Oceania | 70.295000 | 11598.522455 |
| | Africa | 43.319442 | 1598.078825 |
| | Americas | 58.398760 | 4901.541870 |
| 1962 | Asia | 51.563223 | 5729.369625 |
| | Europe | 68.539233 | 8365.486814 |
| | Oceania | 71.085000 | 12696.452430 |
| | Africa | 45.334538 | 2050.363801 |
| | Americas | 60.410920 | 5668.253496 |
| 1967 | Asia | 54.663640 | 5971.173374 |
| | Europe | 69.737600 | 10143.823757 |
| | Oceania | 71.310000 | 14495.021790 |
| | Africa | 47.450942 | 2339.615674 |
| | Americas | 62.394920 | 6491.334139 |
| 1972 | Asia | 57.319269 | 8187.468699 |
| | Europe | 70.775033 | 12479.575246 |
| | Oceania | 71.910000 | 16417.333380 |
| | Africa | 49.580423 | 2585.938508 |
| | Americas | 64.391560 | 7352.007126 |
| 1977 | Asia | 59.610556 | 7791.314020 |
| | Europe | 71.937767 | 14283.979110 |
| | Oceania | 72.855000 | 17283.957605 |
| 1982 | Africa | 51.592865 | 2481.592960 |
| | Americas | 66.228840 | 7506.737088 |

| year | continent | lifeExp | gdpPercap |
|---|---|---|---|
| | Asia | 62.617939 | 7434.135157 |
| | Europe | 72.806400 | 15617.896551 |
| | Oceania | 74.290000 | 18554.709840 |
| | Africa | 53.344788 | 2282.668991 |
| | Americas | 68.090720 | 7793.400261 |
| 1987 | Asia | 64.851182 | 7608.226508 |
| | Europe | 73.642167 | 17214.310727 |
| | Oceania | 75.320000 | 20448.040160 |
| | Africa | 53.629577 | 2281.810333 |
| | Americas | 69.568360 | 8044.934406 |
| 1992 | Asia | 66.537212 | 8639.690248 |
| | Europe | 74.440100 | 17061.568084 |
| | Oceania | 76.945000 | 20894.045885 |
| | Africa | 53.598269 | 2378.759555 |
| | Americas | 71.150480 | 8889.300863 |
| 1997 | Asia | 68.020515 | 9834.093295 |
| | Europe | 75.505167 | 19076.781802 |
| | Oceania | 78.190000 | 24024.175170 |
| | Africa | 53.325231 | 2599.385159 |
| | Americas | 72.422040 | 9287.677107 |
| 2002 | Asia | 69.233879 | 10174.090397 |
| | Europe | 76.700600 | 21711.732422 |
| | Oceania | 79.740000 | 26938.778040 |
| | Africa | 54.806038 | 3089.032605 |
| | Americas | 73.608120 | 11003.031625 |
| 2007 | Asia | 70.728485 | 12473.026870 |
| | Europe | 77.648600 | 25054.481636 |
| | Oceania | 80.719500 | 29810.188275 |

## 1.4.2 重置index使用reset_index()函数

In [17]:

```
group1.index.values
```

Out[17]:

```
array([(1952, 'Africa'), (1952, 'Americas'), (1952, 'Asia'),
       (1952, 'Europe'), (1952, 'Oceania'), (1957, 'Africa'),
       (1957, 'Americas'), (1957, 'Asia'), (1957, 'Europe'),
       (1957, 'Oceania'), (1962, 'Africa'), (1962, 'Americas'),
       (1962, 'Asia'), (1962, 'Europe'), (1962, 'Oceania'),
       (1967, 'Africa'), (1967, 'Americas'), (1967, 'Asia'),
       (1967, 'Europe'), (1967, 'Oceania'), (1972, 'Africa'),
       (1972, 'Americas'), (1972, 'Asia'), (1972, 'Europe'),
       (1972, 'Oceania'), (1977, 'Africa'), (1977, 'Americas'),
       (1977, 'Asia'), (1977, 'Europe'), (1977, 'Oceania'),
       (1982, 'Africa'), (1982, 'Americas'), (1982, 'Asia'),
       (1982, 'Europe'), (1982, 'Oceania'), (1987, 'Africa'),
       (1987, 'Americas'), (1987, 'Asia'), (1987, 'Europe'),
       (1987, 'Oceania'), (1992, 'Africa'), (1992, 'Americas'),
       (1992, 'Asia'), (1992, 'Europe'), (1992, 'Oceania'),
       (1997, 'Africa'), (1997, 'Americas'), (1997, 'Asia'),
       (1997, 'Europe'), (1997, 'Oceania'), (2002, 'Africa'),
       (2002, 'Americas'), (2002, 'Asia'), (2002, 'Europe'),
       (2002, 'Oceania'), (2007, 'Africa'), (2007, 'Americas'),
       (2007, 'Asia'), (2007, 'Europe'), (2007, 'Oceania')], dtype=obj
ect)
```

In [18]:

```
group1.reset_index(inplace=True)
group1
```

Out[18]:

| | year | continent | lifeExp | gdpPercap |
|---|---|---|---|---|
| 0 | 1952 | Africa | 39.135500 | 1252.572466 |
| 1 | 1952 | Americas | 53.279840 | 4079.062552 |
| 2 | 1952 | Asia | 46.314394 | 5195.484004 |
| 3 | 1952 | Europe | 64.408500 | 5661.057435 |
| 4 | 1952 | Oceania | 69.255000 | 10298.085650 |
| 5 | 1957 | Africa | 41.266346 | 1385.236062 |
| 6 | 1957 | Americas | 55.960280 | 4616.043733 |
| 7 | 1957 | Asia | 49.318544 | 5787.732940 |
| 8 | 1957 | Europe | 66.703067 | 6963.012816 |
| 9 | 1957 | Oceania | 70.295000 | 11598.522455 |
| 10 | 1962 | Africa | 43.319442 | 1598.078825 |
| 11 | 1962 | Americas | 58.398760 | 4901.541870 |
| 12 | 1962 | Asia | 51.563223 | 5729.369625 |
| 13 | 1962 | Europe | 68.539233 | 8365.486814 |
| 14 | 1962 | Oceania | 71.085000 | 12696.452430 |
| 15 | 1967 | Africa | 45.334538 | 2050.363801 |
| 16 | 1967 | Americas | 60.410920 | 5668.253496 |
| 17 | 1967 | Asia | 54.663640 | 5971.173374 |
| 18 | 1967 | Europe | 69.737600 | 10143.823757 |
| 19 | 1967 | Oceania | 71.310000 | 14495.021790 |
| 20 | 1972 | Africa | 47.450942 | 2339.615674 |
| 21 | 1972 | Americas | 62.394920 | 6491.334139 |
| 22 | 1972 | Asia | 57.319269 | 8187.468699 |
| 23 | 1972 | Europe | 70.775033 | 12479.575246 |
| 24 | 1972 | Oceania | 71.910000 | 16417.333380 |
| 25 | 1977 | Africa | 49.580423 | 2585.938508 |
| 26 | 1977 | Americas | 64.391560 | 7352.007126 |
| 27 | 1977 | Asia | 59.610556 | 7791.314020 |
| 28 | 1977 | Europe | 71.937767 | 14283.979110 |
| 29 | 1977 | Oceania | 72.855000 | 17283.957605 |
| 30 | 1982 | Africa | 51.592865 | 2481.592960 |
| 31 | 1982 | Americas | 66.228840 | 7506.737088 |
| 32 | 1982 | Asia | 62.617939 | 7434.135157 |

| | year | continent | lifeExp | gdpPercap |
|---|---|---|---|---|
| **33** | 1982 | Europe | 72.806400 | 15617.896551 |
| **34** | 1982 | Oceania | 74.290000 | 18554.709840 |
| **35** | 1987 | Africa | 53.344788 | 2282.668991 |
| **36** | 1987 | Americas | 68.090720 | 7793.400261 |
| **37** | 1987 | Asia | 64.851182 | 7608.226508 |
| **38** | 1987 | Europe | 73.642167 | 17214.310727 |
| **39** | 1987 | Oceania | 75.320000 | 20448.040160 |
| **40** | 1992 | Africa | 53.629577 | 2281.810333 |
| **41** | 1992 | Americas | 69.568360 | 8044.934406 |
| **42** | 1992 | Asia | 66.537212 | 8639.690248 |
| **43** | 1992 | Europe | 74.440100 | 17061.568084 |
| **44** | 1992 | Oceania | 76.945000 | 20894.045885 |
| **45** | 1997 | Africa | 53.598269 | 2378.759555 |
| **46** | 1997 | Americas | 71.150480 | 8889.300863 |
| **47** | 1997 | Asia | 68.020515 | 9834.093295 |
| **48** | 1997 | Europe | 75.505167 | 19076.781802 |
| **49** | 1997 | Oceania | 78.190000 | 24024.175170 |
| **50** | 2002 | Africa | 53.325231 | 2599.385159 |
| **51** | 2002 | Americas | 72.422040 | 9287.677107 |
| **52** | 2002 | Asia | 69.233879 | 10174.090397 |
| **53** | 2002 | Europe | 76.700600 | 21711.732422 |
| **54** | 2002 | Oceania | 79.740000 | 26938.778040 |
| **55** | 2007 | Africa | 54.806038 | 3089.032605 |
| **56** | 2007 | Americas | 73.608120 | 11003.031625 |
| **57** | 2007 | Asia | 70.728485 | 12473.026870 |
| **58** | 2007 | Europe | 77.648600 | 25054.481636 |
| **59** | 2007 | Oceania | 80.719500 | 29810.188275 |

## 1.4.3 分组频率统计

In [19]:

```
flt=data.groupby('continent')['country'].nunique()
flt
```

Out[19]:

```
continent
Africa      52
Americas    25
Asia        33
Europe      30
Oceania      2
Name: country, dtype: int64
```

In [20]:

```
flt1=data.groupby('continent')['country'].value_counts()
flt1
```

Out[20]:

```
continent  country
Africa     Algeria           12
           Angola            12
           Benin             12
           Botswana          12
           Burkina Faso      12
                             ..
Europe     Switzerland       12
           Turkey            12
           United Kingdom    12
Oceania    Australia         12
           New Zealand       12
Name: country, Length: 142, dtype: int64
```

总结：nunique会以groupby()后的属性为一类分类，value_counts()会以它之前的属性为细小分类，同时也会根据groupby()进行大类分类

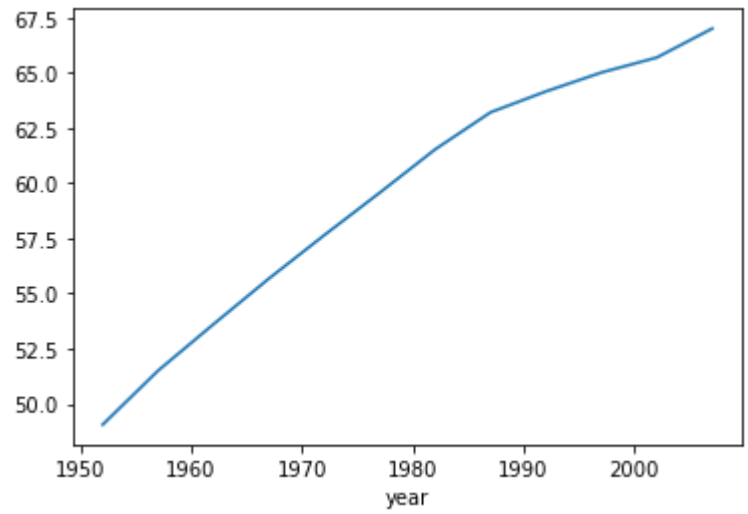
# 1.5 基本绘图

In [21]:

```
group.plot()
```

Out[21]:

```
<AxesSubplot:xlabel='year'>
```




# 2 第二章： Pandas的数据结构

## 2.1 Series和DataFrame 结构建立

In [22]:

```
s=pd.Series(['Banana','42'])
s1=pd.DataFrame(['Banana','42'])
```

## 2.2 Series 的index和keys()具有相同小果果

In [23]:

```
s.index
```

Out[23]:

```
RangeIndex(start=0, stop=2, step=1)
```

In [24]:

```
s.keys()
```

Out[24]:

```
RangeIndex(start=0, stop=2, step=1)
```

## 2.3 类似于ndarray的Series

In [25]:

```
data_scientists = pd.read_csv(r'E:\jupyter notebook storage\Practice in Pandas\data\
data_scientists.head()
```

Out[25]:

| | Name | Born | Died | Age | Occupation |
|---|---|---|---|---|---|
| 0 | Rosaline Franklin | 1920-07-25 | 1958-04-16 | 37 | Chemist |
| 1 | William Gosset | 1876-06-13 | 1937-10-16 | 61 | Statistician |
| 2 | Florence Nightingale | 1820-05-12 | 1910-08-13 | 90 | Nurse |
| 3 | Marie Curie | 1867-11-07 | 1934-07-04 | 66 | Chemist |
| 4 | Rachel Carson | 1907-05-27 | 1964-04-14 | 56 | Biologist |

In [26]:

```
ages=data_scientists['Age']
# 检测ages是不是Series格式
print(type(ages))
```

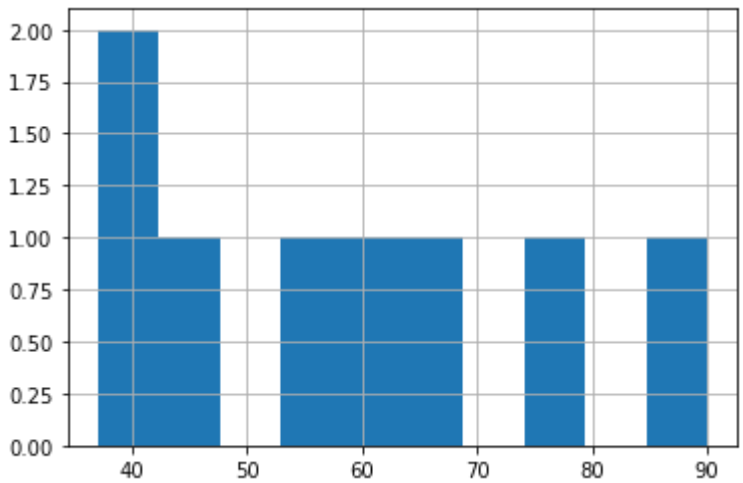
```
<class 'pandas.core.series.Series'>
```

In [27]:

```
ages.hist()
```

Out[27]:

```
<AxesSubplot:>
```



In [28]:

```
ages.describe()
```

Out[28]:

```
count     8.000000
mean     59.125000
std      18.325918
min      37.000000
25%      44.000000
50%      58.500000
75%      68.750000
max      90.000000
Name: Age, dtype: float64
```

In [29]:

```
ages.values
```

Out[29]:

```
array([37, 61, 90, 66, 56, 45, 41, 77], dtype=int64)
```

In [30]:

```
# 返回唯一的值组成一个np数组(去重)
ages.unique()
```

Out[30]:

```
array([37, 61, 90, 66, 56, 45, 41, 77], dtype=int64)
```

In [31]:

```
# 获取大于平均值的年龄
ages[ages>ages.mean()]
```

Out[31]:

```
1    61
2    90
3    66
7    77
Name: Age, dtype: int64
```

## 2.4 操作自动对齐和向量化（广播）

In [32]:

```
# 可以相加相乘
ages+ages
```

Out[32]:

```
0     74
1    122
2    180
3    132
4    112
5     90
6     82
7    154
Name: Age, dtype: int64
```

In [33]:

```
ages*ages
```

Out[33]:

```
0    1369
1    3721
2    8100
3    4356
4    3136
5    2025
6    1681
7    5929
Name: Age, dtype: int64
```

In [34]:

```
# 可以直接数字加减，比如加100
ages+100
```

Out[34]:

```
0    137
1    161
2    190
3    166
4    156
5    145
6    141
7    177
Name: Age, dtype: int64
```

In [35]:

```
# 不同向量之间的计算会出错，前提shape要一样，即长度格式一样
ages+pd.Series([1,100])
```

Out[35]:

```
0      38.0
1     161.0
2       NaN
3       NaN
4       NaN
5       NaN
6       NaN
7       NaN
dtype: float64
```

In [36]:

```
# 带有常见的索引标签的向量（自动对齐）
rev_ages=ages.sort_index(ascending=False)
rev_ages
```

Out[36]:

```
7    77
6    41
5    45
4    56
3    66
2    90
1    61
0    37
Name: Age, dtype: int64
```

In [37]:

```
ages*2
```

Out[37]:

```
0     74
1    122
2    180
3    132
4    112
5     90
6     82
7    154
Name: Age, dtype: int64
```

In [38]:

```
ages+rev_ages
# 所以向量会自己找到合适的索引进行操作
```

Out[38]:

```
0     74
1    122
2    180
3    132
4    112
5     90
6     82
7    154
Name: Age, dtype: int64
```

## 2.5 更改Series和DataFrame

In [39]:

```
# 将字符化的日期改为日期格式
born_datatime=pd.to_datetime(data_scientists['Born'],format='%Y-%m-%d')
# format 即年月日
born_datatime
```

Out[39]:

```
0   1920-07-25
1   1876-06-13
2   1820-05-12
3   1867-11-07
4   1907-05-27
5   1813-03-15
6   1912-06-23
7   1777-04-30
Name: Born, dtype: datetime64[ns]
```

In [40]:

```
died_datatime=pd.to_datetime(data_scientists['Died'])
died_datatime
```

Out[40]:

```
0   1958-04-16
1   1937-10-16
2   1910-08-13
3   1934-07-04
4   1964-04-14
5   1858-06-16
6   1954-06-07
7   1855-02-23
Name: Died, dtype: datetime64[ns]
```

In [41]:

```
# 使用Python的多重赋值法可以操作
data_scientists['Born_dt'],data_scientists['Died_dt']=(born_datatime,died_datatime)
data_scientists
```

Out[41]:

| | Name | Born | Died | Age | Occupation | Born_dt | Died_dt |
|---|---|---|---|---|---|---|---|
| 0 | Rosaline Franklin | 1920-07-25 | 1958-04-16 | 37 | Chemist | 1920-07-25 | 1958-04-16 |
| 1 | William Gosset | 1876-06-13 | 1937-10-16 | 61 | Statistician | 1876-06-13 | 1937-10-16 |
| 2 | Florence Nightingale | 1820-05-12 | 1910-08-13 | 90 | Nurse | 1820-05-12 | 1910-08-13 |
| 3 | Marie Curie | 1867-11-07 | 1934-07-04 | 66 | Chemist | 1867-11-07 | 1934-07-04 |
| 4 | Rachel Carson | 1907-05-27 | 1964-04-14 | 56 | Biologist | 1907-05-27 | 1964-04-14 |
| 5 | John Snow | 1813-03-15 | 1858-06-16 | 45 | Physician | 1813-03-15 | 1858-06-16 |
| 6 | Alan Turing | 1912-06-23 | 1954-06-07 | 41 | Computer Scientist | 1912-06-23 | 1954-06-07 |
| 7 | Johann Gauss | 1777-04-30 | 1855-02-23 | 77 | Mathematician | 1777-04-30 | 1855-02-23 |

## 2.6 打乱顺序函数

In [42]:

```
import random
random.seed(42)
random.shuffle(data_scientists['Age'])
print(data_scientists['Age'])
```

```
0    66
1    56
2    41
3    77
4    90
5    45
6    37
7    61
Name: Age, dtype: int64

E:\anaconda\lib\random.py:307: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas
-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
 (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.htm
l#returning-a-view-versus-a-copy)
  x[i], x[j] = x[j], x[i]
```

In [43]:

```
data_scientists["Age"]=data_scientists["Age"].sample(len(data_scientists["Age"]),rar
data_scientists["Age"]
```

Out[43]:

```
0    61
1    90
2    45
3    41
4    56
5    66
6    37
7    77
Name: Age, dtype: int64
```

## 2.7 转换日期，比如日-月，月-年

In [44]:

```
data_scientists['age_days_dt']=data_scientists['Died_dt']-data_scientists['Born_dt']
data_scientists
```

Out[44]:

| | Name | Born | Died | Age | Occupation | Born_dt | Died_dt | age_days_dt |
|---|---|---|---|---|---|---|---|---|
| **0** | Rosaline Franklin | 1920-07-25 | 1958-04-16 | 61 | Chemist | 1920-07-25 | 1958-04-16 | 13779 days |
| **1** | William Gosset | 1876-06-13 | 1937-10-16 | 90 | Statistician | 1876-06-13 | 1937-10-16 | 22404 days |
| **2** | Florence Nightingale | 1820-05-12 | 1910-08-13 | 45 | Nurse | 1820-05-12 | 1910-08-13 | 32964 days |
| **3** | Marie Curie | 1867-11-07 | 1934-07-04 | 41 | Chemist | 1867-11-07 | 1934-07-04 | 24345 days |
| **4** | Rachel Carson | 1907-05-27 | 1964-04-14 | 56 | Biologist | 1907-05-27 | 1964-04-14 | 20777 days |
| **5** | John Snow | 1813-03-15 | 1858-06-16 | 66 | Physician | 1813-03-15 | 1858-06-16 | 16529 days |
| **6** | Alan Turing | 1912-06-23 | 1954-06-07 | 37 | Computer Scientist | 1912-06-23 | 1954-06-07 | 15324 days |
| **7** | Johann Gauss | 1777-04-30 | 1855-02-23 | 77 | Mathematician | 1777-04-30 | 1855-02-23 | 28422 days |

In [45]:

```
# 利用astype函数去转换年月日
data_scientists['age_days_dt']=data_scientists['age_days_dt'].astype('timedelta64[Y]
data_scientists
```

Out[45]:

| | Name | Born | Died | Age | Occupation | Born_dt | Died_dt | age_days_dt |
|---|---|---|---|---|---|---|---|---|
| **0** | Rosaline Franklin | 1920-07-25 | 1958-04-16 | 61 | Chemist | 1920-07-25 | 1958-04-16 | 37.0 |
| **1** | William Gosset | 1876-06-13 | 1937-10-16 | 90 | Statistician | 1876-06-13 | 1937-10-16 | 61.0 |
| **2** | Florence Nightingale | 1820-05-12 | 1910-08-13 | 45 | Nurse | 1820-05-12 | 1910-08-13 | 90.0 |
| **3** | Marie Curie | 1867-11-07 | 1934-07-04 | 41 | Chemist | 1867-11-07 | 1934-07-04 | 66.0 |
| **4** | Rachel Carson | 1907-05-27 | 1964-04-14 | 56 | Biologist | 1907-05-27 | 1964-04-14 | 56.0 |
| **5** | John Snow | 1813-03-15 | 1858-06-16 | 66 | Physician | 1813-03-15 | 1858-06-16 | 45.0 |
| **6** | Alan Turing | 1912-06-23 | 1954-06-07 | 37 | Computer Scientist | 1912-06-23 | 1954-06-07 | 41.0 |
| **7** | Johann Gauss | 1777-04-30 | 1855-02-23 | 77 | Mathematician | 1777-04-30 | 1855-02-23 | 77.0 |

## 2.8 文件的导入与导出

In [46]:

```
names=data_scientists['Name']
```

In [47]:

```
# 输出文件
names.to_pickle("E:\jupyter notebook storage\Practice in Pandas/ouput file/names.pic
```

注意：\和转译字符会报错，建议/代替。