Practice on Income Dataset

In [2]:
```
#Define functions for the following data points

filepath="DataFiles/incomedata.csv"
import pandas as pd
def readcsvdata(filepath):
    h=pd.read_csv(filepath)
    return h
readcsvdata(filepath)
```

Out[2]:

	GEOID	State	2005	2006	2007	2008	2009	2010	2011	2012	2013
0	04000US01	Alabama	37150	37952	42212	44476	39980	40933	42590	43464	41381
1	04000US02	Alaska	55891	56418	62993	63989	61604	57848	57431	63648	61137
2	04000US04	Arizona	45245	46657	629	46914	45739	46896	48621	47044	50602
3	04000US05	Arkansas	36658	37057	40795	39586	36538	38587	41302	39018	39919
4	04000US06	California	51755	55319	55734	57014	56134	54283	53367	57020	57528

In [25]:
```
#Average Income of all states from 2005 to 2013

incomedf=readcsvdata(filepath)
def rowaccess(df):

    data=df.values
    for i in (data):
        #print(i)
        li=i[2:]

        print(sum(li)/len(li))#for each state average




rowaccess(incomedf)

```

```
41126.444444444445
60106.555555555555
42038.555555555555
38828.88888888889
55350.444444444445
```

In [30]:
```
#Average Income of all states from 2005 to 2013

incomedf=readcsvdata(filepath)
def rowaccessdata(df):
    output=0
    data=df.values
    for j in (data):
        #print(i)
        l=j[2:]
        output=output+(sum(l)/len(l))
    print(output/len(data))
rowaccessdata(incomedf)
```

```
47490.177777777775
```

In [38]:
```
# State with highest average income in the last three years
incomedf=readcsvdata(filepath)
def highestaverageamongstates(df):
    li=[]
    data=df.values
    for dt in data:
        threevalues=dt[-3:]
        average=sum(threevalues)/len(threevalues)
        li.append(average)
    #print(li)
    print(max(li),"maximun ")

highestaverageamongstates(incomedf)
```

```
[42478.333333333336, 60738.666666666664, 48755.666666666664, 40079.66666666666
4, 55971.666666666664]
60738.666666666664 maximun
```

In [68]:
```
# State with highest average income in the last three years
incomedf=readcsvdata(filepath)
def highestaverageamongstates1(df):
    dic={}
    data=df.values
    for dt in data:
        threevalues=dt[-3:]
        average=sum(threevalues)//len(threevalues)
        dic[average]=dt[1]
    print(max(dic.items()))
highestaverageamongstates1(incomedf)
```

```
(60738, 'Alaska')
```

In [73]:
```
#State with lowest average income from 2007 to 2010(inclusive)

incomedf=readcsvdata(filepath)
def lowestaverageamongstates1(df):
    dic={}
    data=df.values
    for dt in data:
        threevalues=dt[4:8]
        print(threevalues)
        average=sum(threevalues)//len(threevalues)
        print(average)
        dic[average]=dt[1]
    print(min(dic.items()))
lowestaverageamongstates1(incomedf)
```

```
[42212 44476 39980 40933]
41900
[62993 63989 61604 57848]
61608
[629 46914 45739 46896]
35044
[40795 39586 36538 38587]
38876
[55734 57014 56134 54283]
55791
(35044, 'Arizona')
```

In [83]:

```
#Print the list of all states in the same line with average income less than

incomedf=readcsvdata(filepath)
def incomelessthancalifornia(df):
    d={}
    data=df.values
    for i in data:
        #print(i)
        li=i[2:]
        avg=sum(li)//len(li)
        d[avg]=i[1]
    print(d)
    sort_d=sorted(d.items())
    print(sort_d)
    for states in sort_d:
        if states[1]=="California":
            break
        else:
            print(states[1])



incomelessthancalifornia(incomedf)
```

```
{41126: 'Alabama', 60106: 'Alaska', 42038: 'Arizona', 38828: 'Arkansas', 55350:
'California'}
[(38828, 'Arkansas'), (41126, 'Alabama'), (42038, 'Arizona'), (55350, 'Californ
ia'), (60106, 'Alaska')]
Arkansas
Alabama
Arizona
```

In [92]:
```
#Print the names of states based on descending order of income in the year 2
incomedf=readcsvdata(filepath)
def descendingorder(df):
    d={}
    data=df.values
    for i in data:
        #print(i)
        li=i[6]
        #print(li)
        d[li]=i[1]
    print(d.items())
    sort_d=sorted(d.items(),reverse=True)
    print(sort_d)
    for state in sort_d:
        print(state[1])

descendingorder(incomedf)
```

```
dict_items([(39980, 'Alabama'), (61604, 'Alaska'), (45739, 'Arizona'), (36538,
'Arkansas'), (56134, 'California')])
[(61604, 'Alaska'), (56134, 'California'), (45739, 'Arizona'), (39980, 'Alabam
a'), (36538, 'Arkansas')]
Alaska
California
Arizona
Alabama
Arkansas
```

In [5]:
```
#State with the lowest recorded income from 2005 to 2013

incomedf=readcsvdata(filepath)
d={}
def lowestrecordedincome(df):
    data=df.values
    for i in data:
        f=i[2:]
        min_f=min(f)
        d[min_f]=i[1]
    s=min(d.items())
    print(s[1])

lowestrecordedincome(incomedf)
```

```
Arizona
```

In [ ]:
```

```

In [4]:

```
incomedf=readcsvdata(filepath)
def rowindexrange(df,key):
    li=[]
    data=df.values
    for i in data:
        if key==i[0] or key==i[1]:
            for index in range(2,len(i)-1):
                k= i[index]
                li.append(k)
            print(sum(li))


    rowindexrange(incomedf,key)

```

```
4
Arkansas
309541
Arizona
327745
Alaska
479822
Alabama
328757
```

In [24]:

```
incomedf=readcsvdata(filepath)
def rowindexrange(df,key):
    li=[]
    s=0
    li1=[]
    data=df.values
    for i in data:
        if key==i[0] or key==i[1]:
            for index in range(2,len(i)-1):
                k= i[index]
                li.append(k)
            variable=sum(li)//len(li)
            #print(variable)
            li1.append(variable)
            #print(li1)
            for k in li1:
                s=s+k
    print(s)



a=int(input())
f=list(map(str,input().split()))
for i in range(a):
    rowindexrange(incomedf,f[i])

```

```
4
Alabama Alaska Arizona Arkansas
41094
59977
40968
38692
```

In [30]:

```
incomedf.index = incomedf['State']
incomedf
```

Out[30]:

State	GEOID	State	2005	2006	2007	2008	2009	2010	2011	2012	2013
Alabama	04000US01	Alabama	37150	37952	42212	44476	39980	40933	42590	43464	41381
Alaska	04000US02	Alaska	55891	56418	62993	63989	61604	57848	57431	63648	61137
Arizona	04000US04	Arizona	45245	46657	629	46914	45739	46896	48621	47044	50602
Arkansas	04000US05	Arkansas	36658	37057	40795	39586	36538	38587	41302	39018	39919
California	04000US06	California	51755	55319	55734	57014	56134	54283	53367	57020	57528

In [40]:
```
#Combined Average Income of all states from 2005 to 2013
incomedf.mean(axis =1)
```

Out[40]:
```
State
Alabama       41126.444444
Alaska        60106.555556
Arizona       42038.555556
Arkansas      38828.888889
California    55350.444444
dtype: float64
```

In [36]:
```
#State with highest average income in the last three years
incomedf.iloc[:,:3].mean(axis =1).idxmax()
```

Out[36]:
```
'Alaska'
```

In [42]:
```
#State with lowest average income from 2007 to 2010(inclusive)
incomedf.iloc[:,4:8].mean(axis=1).idxmin()
```

Out[42]:
```
'Arizona'
```

In [57]:
```
#Print the list of all states in the same line with average income less than
cali=incomedf.iloc[4:,:].mean(axis=1)
print(cali)
cali1=cali[0]
print(cali1)

list(incomedf[incomedf.iloc[:,2:].mean(axis=1)<cali1].index)

```

```
State
California    55350.444444
dtype: float64
55350.444444444445
```

Out[57]:
```
['Alabama', 'Arizona', 'Arkansas']
```

In [58]:
```
1 #Print the names of states based on descending order of income in the year 2
2
3 incomedf.sort_values('lifeExp', inplace=True, ascending=False)
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-58-7014ab05210a> in <module>
      1 #Print the names of states based on descending order of income in the y
ear 2009
      2
----> 3 incomedf.sort_values('lifeExp', inplace=True, ascending=False)

~\Anaconda3\lib\site-packages\pandas\core\frame.py in sort_values(self, by, axi
s, ascending, inplace, kind, na_position)
   4717
   4718             by = by[0]
-> 4719             k = self._get_label_or_level_values(by, axis=axis)
   4720
   4721             if isinstance(ascending, (tuple, list)):

~\Anaconda3\lib\site-packages\pandas\core\generic.py in _get_label_or_level_val
ues(self, key, axis)
   1704             values = self.axes[axis].get_level_values(key)._values
   1705         else:
-> 1706             raise KeyError(key)
   1707
   1708         # Check for duplicates

KeyError: 'lifeExp'
```

In [62]:
```
1
```

In [ ]:
```
1
```