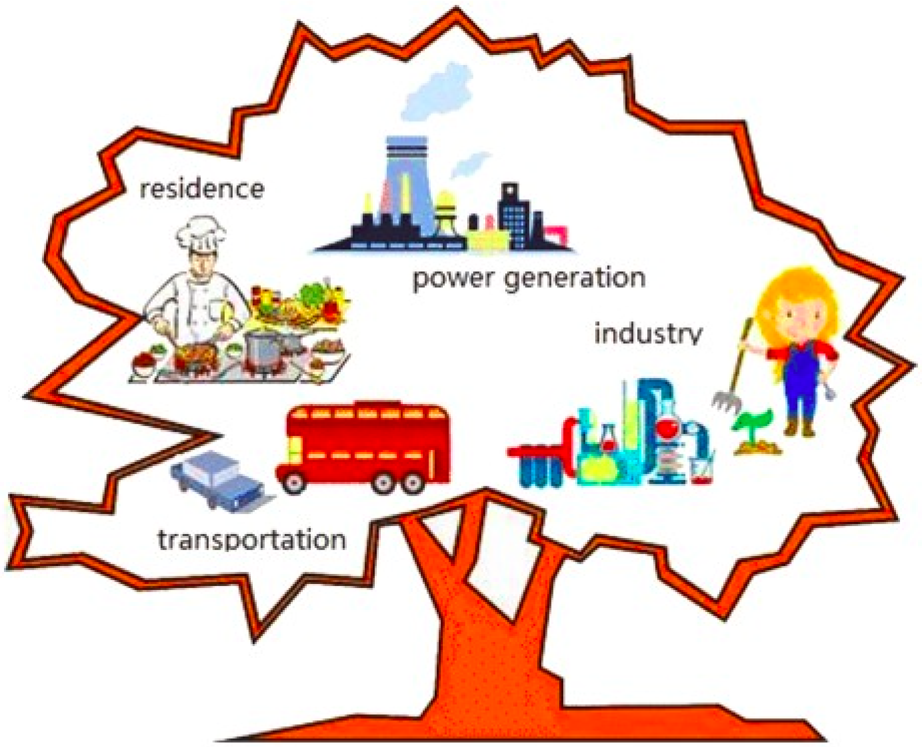
In [2]:
```
print("Name : ")
print("Find the Correlation between the Natural Gas Consumption and Total P
print("Find the Correlation between the Coal Consumption and Coal Price  fr
```

```
Name :
Find the Correlation between the Natural Gas Consumption and Total Popula
tion from 2010-2014.
Find the Correlation between the Coal Consumption and Coal Price  from 20
10-2014.
```

Activty 1- Find the Correlation between the Natural Gas Consumption and Total Population from 2010-2014.

In [4]:
```
#image
#predefine code for image
from IPython.display import Image
Image(filename='natural_gas.png')
#predefine code end
```

Out[4]:



In [6]:
```python
#import libraries and csv
import pandas as pd
import matplotlib.pyplot as plt

pd.set_option('display.max_columns', None)

df = pd.read_csv('Energy Census and Economic Data US 2010-2014.csv')
df
```

Out[6]:

	StateCodes	State	Region	Division	Coast	Great Lakes	TotalC2010	TotalC2011	TotalC2012
0	AL	Alabama	3.0	6.0	1.0	0.0	1931522	1905207	1879716
1	AK	Alaska	4.0	9.0	1.0	0.0	653221	653637	649341
2	AZ	Arizona	4.0	8.0	0.0	0.0	1383531	1424944	1395839
3	AR	Arkansas	3.0	7.0	0.0	0.0	1120632	1122544	1067642
4	CA	California	4.0	9.0	1.0	0.0	7760629	7777115	7564063
5	CO	Colorado	4.0	8.0	0.0	0.0	1513547	1470445	1440781
6	CT	Connecticut	1.0	1.0	1.0	0.0	764970	739130	725019
7	DE	Delaware	3.0	5.0	1.0	0.0	250212	272568	273728
8	FL	Florida	3.0	5.0	1.0	0.0	4282673	4141711	4029903
9	GA	Georgia	3.0	5.0	1.0	0.0	3100144	2982837	2767491
10	HI	Hawaii	4.0	9.0	1.0	0.0	278046	287113	280171
11	ID	Idaho	4.0	8.0	0.0	0.0	516120	516978	510869
12	IL	Illinois	2.0	3.0	0.0	1.0	3955091	3937616	3820547
13	IN	Indiana	2.0	3.0	0.0	1.0	2863396	2847188	2770158
14	IA	Iowa	2.0	4.0	0.0	0.0	1499729	1498973	1440053
15	KS	Kansas	2.0	4.0	0.0	0.0	1117631	1104843	1075435
16	KY	Kentucky	3.0	6.0	0.0	0.0	1978527	1903208	1868483
17	LA	Louisiana	3.0	7.0	1.0	0.0	4385758	4388867	4255161
18	ME	Maine	1.0	1.0	1.0	0.0	415065	413893	399670
19	MD	Maryland	3.0	5.0	1.0	0.0	1464503	1410012	1368600
20	MA	Massachusetts	1.0	1.0	1.0	0.0	1416119	1397164	1363282
21	MI	Michigan	2.0	3.0	0.0	1.0	2753536	2785212	2687926
22	MN	Minnesota	2.0	4.0	0.0	1.0	1857095	1850749	1816866
23	MS	Mississippi	3.0	6.0	1.0	0.0	1177620	1155456	1143099
24	MO	Missouri	2.0	4.0	0.0	0.0	1910500	1856590	1781978
25	MT	Montana	4.0	8.0	0.0	0.0	400855	402355	395724
26	NE	Nebraska	2.0	4.0	0.0	0.0	860741	862675	852984

	StateCodes	State	Region	Division	Coast	Great Lakes	TotalC2010	TotalC2011	TotalC2012
27	NV	Nevada	4.0	8.0	0.0	0.0	645604	632655	639190
28	NH	New Hampshire	1.0	1.0	1.0	0.0	294473	292979	284490
29	NJ	New Jersey	1.0	2.0	1.0	0.0	2395713	2411816	2241207
30	NM	New Mexico	4.0	8.0	0.0	0.0	649962	668675	666540
31	NY	New York	1.0	2.0	1.0	1.0	3723729	3611091	3503309
32	NC	North Carolina	3.0	5.0	1.0	0.0	2685333	2558792	2481060
33	ND	North Dakota	2.0	4.0	0.0	0.0	476072	528508	552326
34	OH	Ohio	2.0	3.0	0.0	1.0	3824933	3792585	3655849
35	OK	Oklahoma	3.0	7.0	0.0	0.0	1579910	1585212	1561913
36	OR	Oregon	4.0	9.0	1.0	0.0	975067	1002476	975044
37	PA	Pennsylvania	1.0	2.0	0.0	1.0	3752280	3725014	3623997
38	RI	Rhode Island	1.0	1.0	1.0	0.0	195314	185731	183879
39	SC	South Carolina	3.0	5.0	1.0	0.0	1643912	1601881	1558766
40	SD	South Dakota	2.0	4.0	0.0	0.0	378514	378470	375950
41	TN	Tennessee	3.0	6.0	0.0	0.0	2247273	2195401	2080953
42	TX	Texas	3.0	7.0	1.0	0.0	11687521	11906249	11931169
43	UT	Utah	4.0	8.0	0.0	0.0	756012	794058	790154
44	VT	Vermont	1.0	1.0	0.0	0.0	153697	150475	130412
45	VA	Virginia	3.0	5.0	1.0	0.0	2483360	2380922	2343908
46	WA	Washington	4.0	9.0	1.0	0.0	2031428	2059630	2037127
47	WV	West Virginia	3.0	5.0	0.0	0.0	738821	726341	720985
48	WI	Wisconsin	2.0	3.0	0.0	1.0	1791199	1778018	1721543
49	WY	Wyoming	4.0	8.0	0.0	0.0	540122	556548	550182
50	DC	District of Columbia	3.0	5.0	0.0	0.0	190529	183806	172963
51	US	United States	NaN	NaN	NaN	NaN	97446021	96827465	94411432

In [7]:
```
#Get the column number of POPESTIMATE2010
POPESTIMATE2010 = df.columns.get_loc("POPESTIMATE2010")
POPESTIMATE2010
```

Out[7]:
```
163
```

In [8]:
```
#Get the column number of POPESTIMATE2014
POPESTIMATE2014 = df.columns.get_loc("POPESTIMATE2014")
POPESTIMATE2014
```

Out[8]:
```
167
```

In [9]:
```
#Create new dataframe only for Population colums and state column
population_dataframe = df[df.columns[POPESTIMATE2010:POPESTIMATE2014+1] ]
population_dataframe['State'] = df['State']
population_dataframe
```

```
<ipython-input-9-2679d715b8d4>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-do
cs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (http
s://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returni
ng-a-view-versus-a-copy)
  population_dataframe['State'] = df['State']
```

Out[9]:

	POPESTIMATE2010	POPESTIMATE2011	POPESTIMATE2012	POPESTIMATE2013	POPESTIMATE20
0	4785822	4801695	4817484	4833996	48493
1	713856	722572	731081	737259	7367
2	2922297	2938430	2949300	2958765	29663
3	6411999	6472867	6556236	6634997	67314
4	37336011	37701901	38062780	38431393	388025
5	5048575	5119661	5191709	5272086	53558
6	3579345	3590537	3594362	3599341	35966
7	899731	907829	916881	925240	9356
8	18852220	19107900	19355257	19600311	198932
9	9714464	9813201	9919000	9994759	100973
10	1363950	1378251	1392766	1408987	14195
11	1570639	1583780	1595590	1612843	16344
12	12840097	12858725	12873763	12890552	128805
13	6490308	6516560	6537632	6570713	65968
14	3050295	3064904	3075935	3092341	31071
15	2858949	2869965	2885966	2895801	29040
16	4349838	4370038	4383465	4399583	44134
17	4545581	4575972	4604744	4629284	46496
18	1327361	1327930	1328592	1328702	13300
19	5788101	5843833	5891819	5938737	59764
20	6564073	6612270	6655829	6708874	67454
21	9876498	9875736	9884781	9898193	99098
22	5310418	5348036	5380615	5422060	54571
23	2970811	2978464	2986137	2992206	29940

	POPESTIMATE2010	POPESTIMATE2011	POPESTIMATE2012	POPESTIMATE2013	POPESTIMATE2
24	5996085	6010544	6025281	6044917	6063
25	990575	997661	1005163	1014864	1023
26	1829865	1842232	1855487	1868969	1881
27	2703493	2718586	2755245	2791494	2839
28	1316517	1318109	1321297	1322616	1326
29	8803580	8842614	8876000	8911502	8938
30	2064950	2078407	2084594	2086895	2085
31	19400867	19521745	19607140	19695680	19746
32	9559488	9651502	9748181	9848917	9943
33	674345	685242	701705	723857	739
34	11540070	11544757	11550901	11572005	11594
35	3759481	3786527	3817059	3853118	3878
36	3837083	3867644	3898684	3928068	3970
37	12711077	12743995	12770043	12781296	12787
38	1053078	1052020	1052637	1053354	1055
39	4636290	4673054	4722621	4771929	4832
40	816192	824171	834504	845510	853
41	6356628	6398389	6455177	6497269	6549
42	25245717	25657477	26094422	26505637	26956
43	2774346	2815324	2855194	2902787	2942
44	625792	626450	626138	626855	626
45	8025376	8110188	8193422	8270345	8326
46	6741911	6822112	6896325	6973742	7061
47	1854176	1854982	1856313	1853595	1850
48	5689268	5708785	5724888	5742953	5757
49	564358	567631	576893	583223	584
50	605210	620427	635040	649111	658
51	309347057	311721632	314112078	316497531	318857

In [10]:
```
#from population_dataframe find the row for New Jersey and create a new dat
new_jersey_population= population_dataframe.loc[population_dataframe['State
new_jersey_population
```

Out[10]:

	POPESTIMATE2010	POPESTIMATE2011	POPESTIMATE2012	POPESTIMATE2013	POPESTIMATE2(
29	8803580	8842614	8876000	8911502	8938

In [11]:
```
#Get the column number of NatGasC2010
NatGasC2010 = df.columns.get_loc("NatGasC2010")
NatGasC2010
```

Out[11]:
```
107
```

In [12]:
```
#Get the column number of NatGasC2014
NatGasC2014 = df.columns.get_loc("NatGasC2014")
NatGasC2014
```

Out[12]:
```
111
```

In [13]:
```
#Create new dataframe only for Natural Gas colums
natural_gas_dataframe = df[df.columns[NatGasC2010:NatGasC2014 + 1] ]
natural_gas_dataframe['State'] = df['State']
natural_gas_dataframe
```

```
<ipython-input-13-d269965c58f1>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-do
cs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (http
s://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returni
ng-a-view-versus-a-copy)
  natural_gas_dataframe['State'] = df['State']
```

Out[13]:

	NatGasC2010	NatGasC2011	NatGasC2012	NatGasC2013	NatGasC2014	State
0	544405	609288	677380	625869	651532	Alabama
1	334978	339819	347228	332963	329585	Alaska
2	336208	293134	339043	340375	315448	Arizona
3	274774	288906	300572	286352	274767	Arkansas
4	2325411	2196252	2456371	2483208	2417476	California
5	510877	481614	461056	487445	499668	Colorado
6	203814	236016	236260	241509	240614	Connecticut
7	56087	81710	104421	100106	106854	Delaware
8	1180462	1235953	1348390	1245287	1246670	Florida
9	541691	532310	625008	635287	666771	Georgia
10	2732	2744	2813	2805	2805	Hawaii
11	85075	83947	90339	106356	94319	Idaho
12	974412	997722	950711	1073734	1116417	Illinois
13	580752	638237	657719	682842	727758	Indiana
14	312941	309672	299315	331358	342431	Iowa
15	280413	285318	268086	288840	289697	Kansas
16	239062	229056	232702	236423	261233	Kentucky
17	1483201	1536147	1586439	1499625	1563857	Louisiana
18	80988	75061	70451	66014	62355	Maine
19	217744	199223	216678	207422	217440	Maryland
20	447429	464018	430921	434893	433246	Massachusetts
21	758696	787337	804084	828483	878699	Michigan
22	427198	424978	430286	474891	489901	Minnesota
23	444875	437873	499946	427323	440003	Mississippi

	NatGasC2010	NatGasC2011	NatGasC2012	NatGasC2013	NatGasC2014	State
24	282141	275308	258945	280845	301360	Missouri
25	72888	79470	75234	81900	79260	Montana
26	169620	173666	161774	178752	179544	Nebraska
27	267808	255970	281432	281972	259438	Nevada
28	62612	72778	74336	55649	58841	New Hampshire
29	671474	677923	670970	712948	798602	New Jersey
30	246201	251838	249815	252867	256085	New Mexico
31	1224485	1247757	1260950	1315282	1386580	New York
32	308710	311189	367948	445898	460876	North Carolina
33	70046	77752	77469	88285	90629	North Dakota
34	810958	849076	869934	954374	1044986	Ohio
35	697351	676910	712411	682277	665799	Oklahoma
36	242914	203607	220578	244025	225576	Oregon
37	909263	1000509	1079497	1176660	1257090	Pennsylvania
38	95710	102465	98435	88274	91259	Rhode Island
39	225961	235465	250482	236710	235860	South Carolina
40	72926	73973	71501	83871	83464	South Dakota
41	263365	267929	281007	285308	311966	Tennessee
42	3689583	3800578	3964121	4143328	4219128	Texas
43	229078	230672	232630	258909	252691	Utah
44	8502	8679	8290	9746	10848	Vermont
45	385931	383527	424050	435247	437322	Virginia
46	294871	272261	271947	327840	319784	Washington
47	121780	124935	140133	153875	161661	West Virginia
48	376627	399247	410305	453608	477923	Wisconsin
49	154759	161776	158546	155960	141763	Wyoming
50	33717	33389	29389	34315	35316	District of Columbia
51	24633507	25014983	26138351	26858134	27513198	United States

In [14]:
```
#from natural_gas_dataframe find the row for New Jersey and create a new da
new_jersey_gas= natural_gas_dataframe.loc[natural_gas_dataframe['State'] ==
new_jersey_gas
```

Out[14]:
```
    NatGasC2010  NatGasC2011  NatGasC2012  NatGasC2013  NatGasC2014       State
29       671474       677923       670970       712948       798602  New Jersey
```

In [15]:
```
#Multiply each natural gas year value with 10 for making it normalize
new_jersey_gas['NatGasC2010'] = new_jersey_gas['NatGasC2010']*10
new_jersey_gas['NatGasC2011'] = new_jersey_gas['NatGasC2011']*10
new_jersey_gas['NatGasC2012'] = new_jersey_gas['NatGasC2012']*10
new_jersey_gas['NatGasC2013'] = new_jersey_gas['NatGasC2013']*10
new_jersey_gas['NatGasC2014'] = new_jersey_gas['NatGasC2014']*10
new_jersey_gas
```

```
<ipython-input-15-864c7ee3aeb3>:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-do
cs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (http
s://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returni
ng-a-view-versus-a-copy)
  new_jersey_gas['NatGasC2010'] = new_jersey_gas['NatGasC2010']*10
<ipython-input-15-864c7ee3aeb3>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-do
cs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (http
s://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returni
ng-a-view-versus-a-copy)
  new_jersey_gas['NatGasC2011'] = new_jersey_gas['NatGasC2011']*10
<ipython-input-15-864c7ee3aeb3>:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-do
cs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (http
s://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returni
ng-a-view-versus-a-copy)
  new_jersey_gas['NatGasC2012'] = new_jersey_gas['NatGasC2012']*10
<ipython-input-15-864c7ee3aeb3>:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-do
cs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (http
s://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returni
ng-a-view-versus-a-copy)
  new_jersey_gas['NatGasC2013'] = new_jersey_gas['NatGasC2013']*10
<ipython-input-15-864c7ee3aeb3>:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-do
cs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (http
s://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returni
ng-a-view-versus-a-copy)
  new_jersey_gas['NatGasC2014'] = new_jersey_gas['NatGasC2014']*10
```

Out[15]:

NatGasC2010	NatGasC2011	NatGasC2012	NatGasC2013	NatGasC2014	State

	NatGasC2010	NatGasC2011	NatGasC2012	NatGasC2013	NatGasC2014	State
29	6714740	6779230	6709700	7129480	7986020	New Jersey

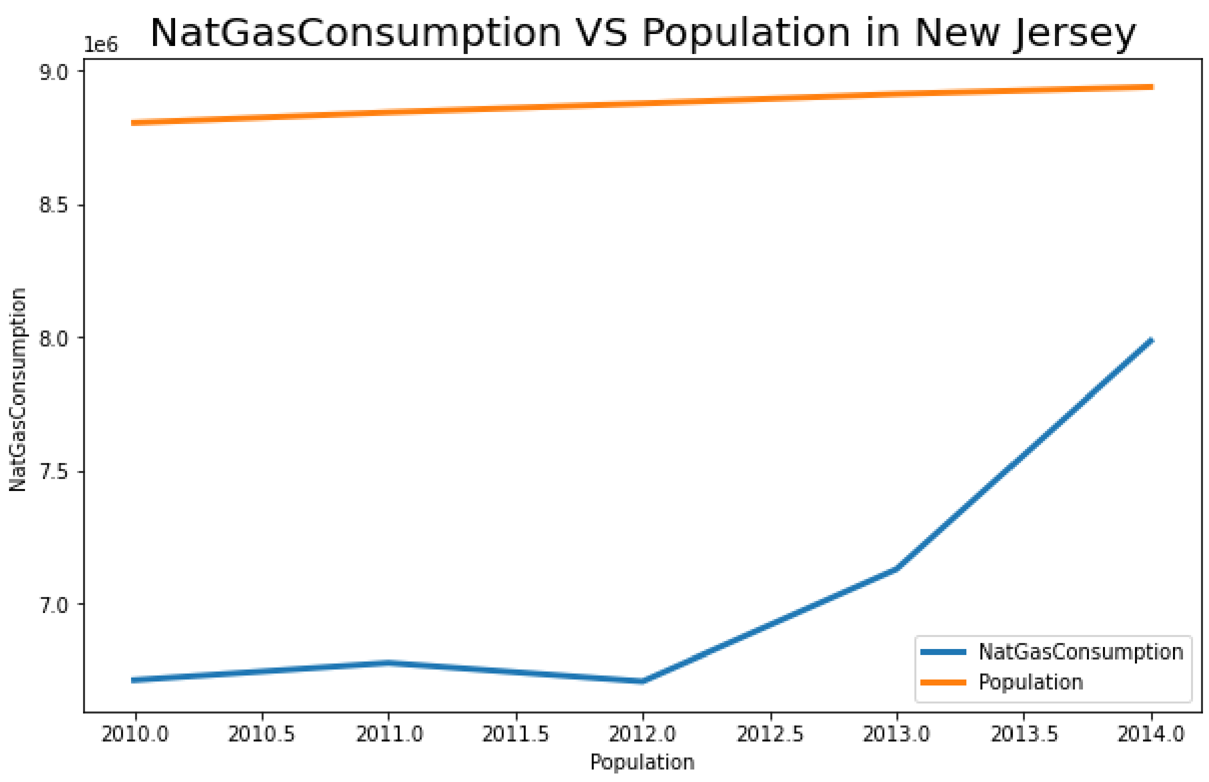
In [16]:

```
#Create a common dataframe for Population and NatGasConsumption
columns_gas = ['Year' , 'Population','NatGasConsumption']
index_gas  = [1,2,3,4,5]
data_gas =[
    [2010,new_jersey_population['POPESTIMATE2010'].values[0],new_jersey_gas
    [2011,new_jersey_population['POPESTIMATE2011'].values[0],new_jersey_gas
    [2012,new_jersey_population['POPESTIMATE2012'].values[0],new_jersey_gas
    [2013,new_jersey_population['POPESTIMATE2013'].values[0],new_jersey_gas
    [2014,new_jersey_population['POPESTIMATE2014'].values[0],new_jersey_gas

]
final_gas = pd.DataFrame(data_gas, index=index_gas, columns=columns_gas)
final_gas
```

Out[16]:

	Year	Population	NatGasConsumption
1	2010	8803580	6714740
2	2011	8842614	6779230
3	2012	8876000	6709700
4	2013	8911502	7129480
5	2014	8938175	7986020

In [17]:
```
#plot a line graph for showing Natural Gas Consumption vs Population
fig = plt.subplots(figsize=(10,6))
label = final_gas['Year']
value = final_gas['NatGasConsumption']


plt.plot(label, value, label = "NatGasConsumption" , linewidth=3.0)

label = final_gas['Year']
value = final_gas['Population']
plt.plot(label, value, label = "Population" , linewidth=3.0)

plt.xlabel('Population')

plt.ylabel('NatGasConsumption')

plt.title('NatGasConsumption VS Population in New Jersey', fontsize=20)

plt.legend()

plt.show()
```



Conslusion : The Natural Gas Consumption and Population growth was constant from 2010 till 2012 in New Jersey, after 2012 there was significant growth in Population which lead to significant growth Natural Gas Consumption

Activity 2 - Find the Correlation between the Coal Consumption and Coal Price from 2010-2014.

In [18]:
```
#image
#predefine code for image
from IPython.display import Image
Image(filename='Coal.Jpg')
#predefine code end
```

Out[18]:

In [1]:
```
#predefine code
#import libraries and csv
import pandas as pd
import matplotlib.pyplot as plt

pd.set_option('display.max_columns', None)

df = pd.read_csv('Energy Census and Economic Data US 2010-2014.csv')
df

#predefine code end
```

Out[1]:

	StateCodes	State	Region	Division	Coast	Great Lakes	TotalC2010	TotalC2011	TotalC2012
0	AL	Alabama	3.0	6.0	1.0	0.0	1931522	1905207	1879716
1	AK	Alaska	4.0	9.0	1.0	0.0	653221	653637	649341
2	AZ	Arizona	4.0	8.0	0.0	0.0	1383531	1424944	1395839
3	AR	Arkansas	3.0	7.0	0.0	0.0	1120632	1122544	1067642
4	CA	California	4.0	9.0	1.0	0.0	7760629	7777115	7564063
5	CO	Colorado	4.0	8.0	0.0	0.0	1513547	1470445	1440781
6	CT	Connecticut	1.0	1.0	1.0	0.0	764970	739130	725019
7	DE	Delaware	3.0	5.0	1.0	0.0	250212	272568	273728
8	FL	Florida	3.0	5.0	1.0	0.0	4282673	4141711	4029903
9	GA	Georgia	3.0	5.0	1.0	0.0	3100144	2982837	2767491
10	HI	Hawaii	4.0	9.0	1.0	0.0	278046	287113	280171
11	ID	Idaho	4.0	8.0	0.0	0.0	516120	516978	510869
12	IL	Illinois	2.0	3.0	0.0	1.0	3955091	3937616	3820547
13	IN	Indiana	2.0	3.0	0.0	1.0	2863396	2847188	2770158
14	IA	Iowa	2.0	4.0	0.0	0.0	1499729	1498973	1440053
15	KS	Kansas	2.0	4.0	0.0	0.0	1117631	1104843	1075435
16	KY	Kentucky	3.0	6.0	0.0	0.0	1978527	1903208	1868483
17	LA	Louisiana	3.0	7.0	1.0	0.0	4385758	4388867	4255161
18	ME	Maine	1.0	1.0	1.0	0.0	415065	413893	399670
19	MD	Maryland	3.0	5.0	1.0	0.0	1464503	1410012	1368600
20	MA	Massachusetts	1.0	1.0	1.0	0.0	1416119	1397164	1363282
21	MI	Michigan	2.0	3.0	0.0	1.0	2753536	2785212	2687926
22	MN	Minnesota	2.0	4.0	0.0	1.0	1857095	1850749	1816866
23	MS	Mississippi	3.0	6.0	1.0	0.0	1177620	1155456	1143099
24	MO	Missouri	2.0	4.0	0.0	0.0	1910500	1856590	1781978
25	MT	Montana	4.0	8.0	0.0	0.0	400855	402355	395724

	StateCodes	State	Region	Division	Coast	Great Lakes	TotalC2010	TotalC2011	TotalC2012
26	NE	Nebraska	2.0	4.0	0.0	0.0	860741	862675	852984
27	NV	Nevada	4.0	8.0	0.0	0.0	645604	632655	639190
28	NH	New Hampshire	1.0	1.0	1.0	0.0	294473	292979	284490
29	NJ	New Jersey	1.0	2.0	1.0	0.0	2395713	2411816	2241207
30	NM	New Mexico	4.0	8.0	0.0	0.0	649962	668675	666540
31	NY	New York	1.0	2.0	1.0	1.0	3723729	3611091	3503309
32	NC	North Carolina	3.0	5.0	1.0	0.0	2685333	2558792	2481060
33	ND	North Dakota	2.0	4.0	0.0	0.0	476072	528508	552326
34	OH	Ohio	2.0	3.0	0.0	1.0	3824933	3792585	3655849
35	OK	Oklahoma	3.0	7.0	0.0	0.0	1579910	1585212	1561913
36	OR	Oregon	4.0	9.0	1.0	0.0	975067	1002476	975044
37	PA	Pennsylvania	1.0	2.0	0.0	1.0	3752280	3725014	3623997
38	RI	Rhode Island	1.0	1.0	1.0	0.0	195314	185731	183879
39	SC	South Carolina	3.0	5.0	1.0	0.0	1643912	1601881	1558766
40	SD	South Dakota	2.0	4.0	0.0	0.0	378514	378470	375950
41	TN	Tennessee	3.0	6.0	0.0	0.0	2247273	2195401	2080953
42	TX	Texas	3.0	7.0	1.0	0.0	11687521	11906249	11931169
43	UT	Utah	4.0	8.0	0.0	0.0	756012	794058	790154
44	VT	Vermont	1.0	1.0	0.0	0.0	153697	150475	130412
45	VA	Virginia	3.0	5.0	1.0	0.0	2483360	2380922	2343908
46	WA	Washington	4.0	9.0	1.0	0.0	2031428	2059630	2037127
47	WV	West Virginia	3.0	5.0	0.0	0.0	738821	726341	720985
48	WI	Wisconsin	2.0	3.0	0.0	1.0	1791199	1778018	1721543
49	WY	Wyoming	4.0	8.0	0.0	0.0	540122	556548	550182
50	DC	District of Columbia	3.0	5.0	0.0	0.0	190529	183806	172963
51	US	United States	NaN	NaN	NaN	NaN	97446021	96827465	94411432

```
In [2]: #Get the column number of CoalC2010
        CoalC2010 = df.columns.get_loc("CoalC2010")
        CoalC2010
```

In [1]:
```
#Get the column number of CoalC2014
CoalC2014 = df.columns.get_loc("CoalC2014")
CoalC2014
```

```
---------------------------------------------------------------------------
--
NameError                                 Traceback (most recent call las
t)
/var/folders/1d/dx5rjvf91qq7f432s9k1r3780000gp/T/ipykernel_80065/47737145
4.py in <module>
      1 #Get the column number of CoalC2014
----> 2 coalC2014 = df.columns.get_loc("CoalC2014")
      3 coalC2014

NameError: name 'df' is not defined
```

In [4]:
```
#Create new dataframe only columns for Coal
Coal_dataframe = df[df.column[CoalC2010:CoalC2014 + 1] ]
Coal_dataframe = ['State'] = df['State']
Coal_dataframe
```

In [5]:
```
#from Coal_dataframe find the row for New Mexico and create a new dataframe
new_mexico_coal= Coal_dataframe.loc[Coal_dataframe['State'] == 'New Mexico'
new_mexico_coal
```

In [6]:
```
#Get the column number of CoalPrice2010
#Get the column number of CoalC2014
CoalC2010 = df.columns.get_loc("CoalC2010")
CoalC2010
```

In [7]:
```
#Get the column number of CoalPrice2014
#Get the column number of CoalC2014
CoalC2014 = df.columns.get_loc("CoalC2014")
CoalC2014
```

In [8]:
```
#Create new dataframe only  columns for Coal_price
Coal_dataframe = df[df.column[CoalC2010:CoalC2014 + 1] ]
Coal_dataframe = ['State'] = df['State']
Coal_dataframe
```

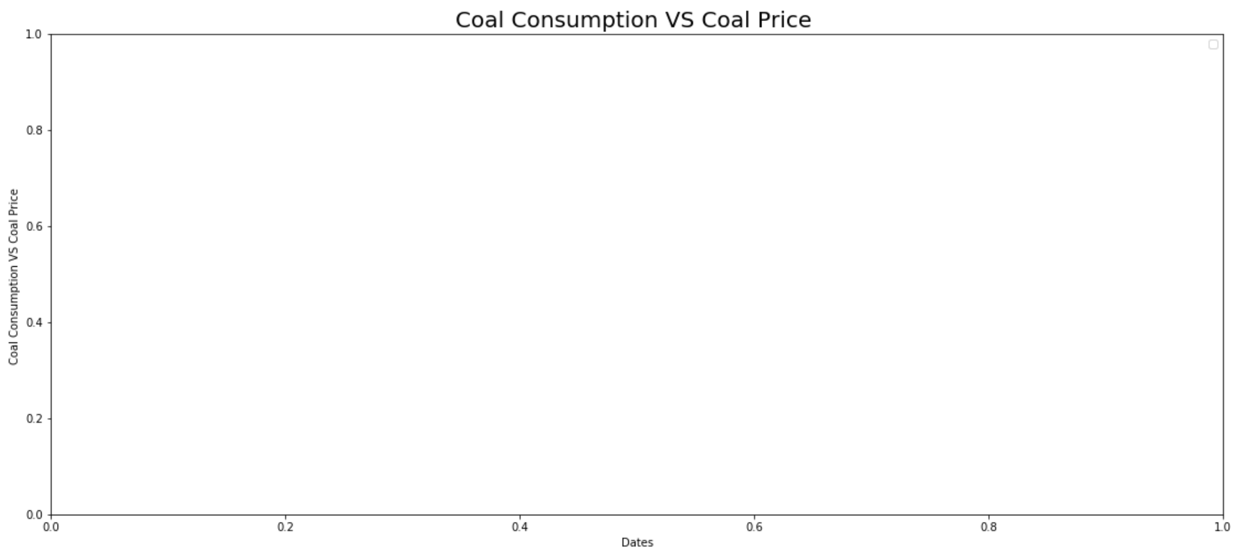
In [9]:
```
#from Coal_price_dataframe find the row for New Mexico and create a new dat
new_mexico_coal= Coal_taframe.loc[Coal_dataframe['State'] == 'New Mexico']
new_mexico_coal
```

In [10]:
```
#Multiply each CoalPrice year wise column values with 100000 for making it
[2010,new_mexico_coal_price['CoalPrice2010'] = new_mexico_coal_price['CoalP
[2011,new_mexico_coal_price['CoalPrice2011'] = new_mexico_coal_price['CoalP
[2012,new_mexico_coal_price['CoalPrice2012'] = new_mexico_coal_price['CoalP
[2013,new_mexico_coal_price['CoalPrice2013'] = new_mexico_coal_price['CoalP
[2014,new_mexico_coal_price['CoalPrice2014'] = new_mexico_coal_price['CoalP
```

In [1]:
```
#Create a common dataframe for coalConsumption and coalPrice
```

In [11]:
```
#Plot a line graph showing the correlation between the Coal Consumption and

#predefine code end
fig = plt.subplots(figsize=(19,8))


#update label and value variable
label =
value =
plt.plot(label, value, label = "coalConsumption" , linewidth=3.0)

#update label and value variable
label =
value =
plt.plot(label, value, label = "coalPrice" , linewidth=3.0)


plt.xlabel('Dates')
plt.ylabel('Coal Consumption VS Coal Price ')
plt.title('Coal Consumption VS Coal Price ', fontsize=20)
plt.legend()
plt.show()

#predefine code end
```

```
No handles with labels found to put in legend.
```



Conslusion :

In [ ]: