```
# Cell 1
import pandas as pd
brainFile = '/content/brainsize.txt'
brainFrame = pd.read_csv(brainFile, sep = '\t')
```

```
# Cell 2
brainFrame.head()
```

	Gender	FSIQ	VIQ	PIQ	Weight	Height	MRI_Count
0	Female	133	132	124	118.0	64.5	816932
1	Male	140	150	124	NaN	72.5	1001121
2	Male	139	123	150	143.0	73.3	1038437
3	Male	133	129	128	172.0	68.8	965353
4	Female	137	132	134	147.0	65.0	951545

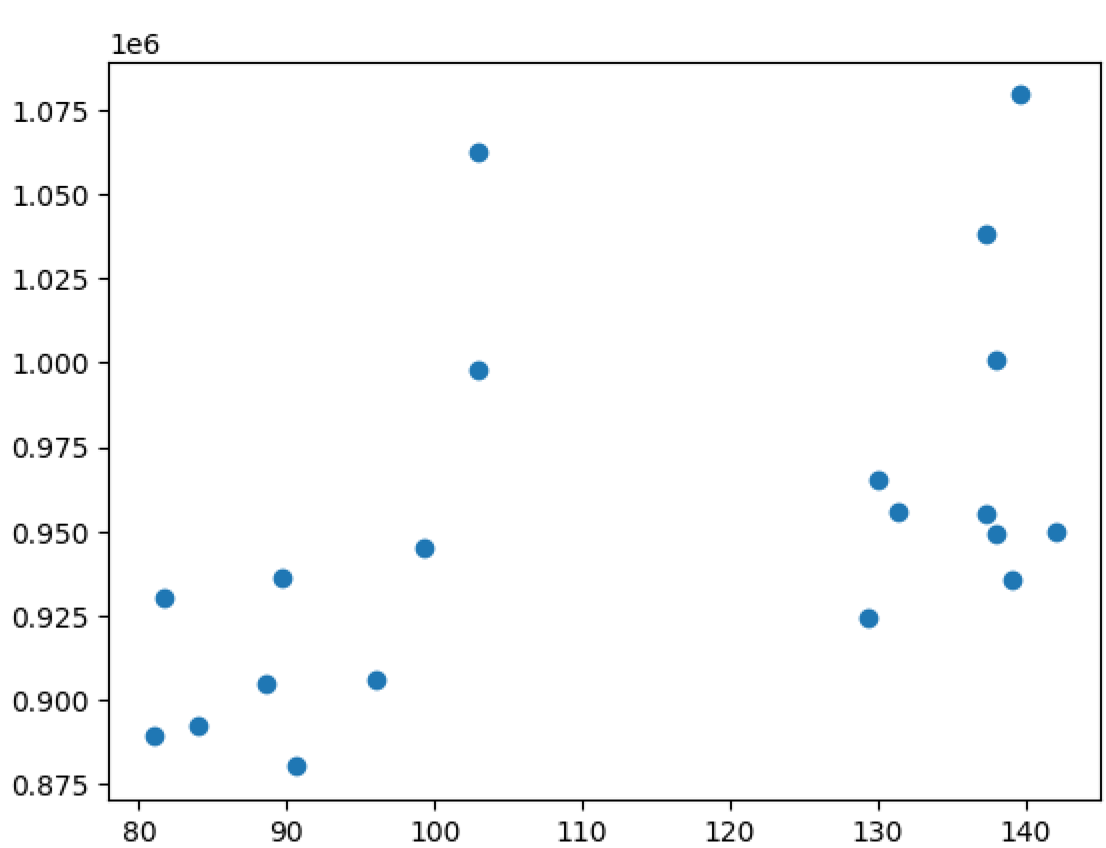
```
# Cell 3
brainFrame.describe()
```

	FSIQ	VIQ	PIQ	Weight	Height	MRI_Count
count	40.000000	40.000000	40.00000	38.000000	39.000000	4.000000e+01
mean	113.450000	112.350000	111.02500	151.052632	68.525641	9.087550e+05
std	24.082071	23.616107	22.47105	23.478509	3.994649	7.228205e+04
min	77.000000	71.000000	72.00000	106.000000	62.000000	7.906190e+05
25%	89.750000	90.000000	88.25000	135.250000	66.000000	8.559185e+05
50%	116.500000	113.000000	115.00000	146.500000	68.000000	9.053990e+05
75%	135.500000	129.750000	128.00000	172.000000	70.500000	9.500780e+05
max	144.000000	150.000000	150.00000	192.000000	77.000000	1.079549e+06

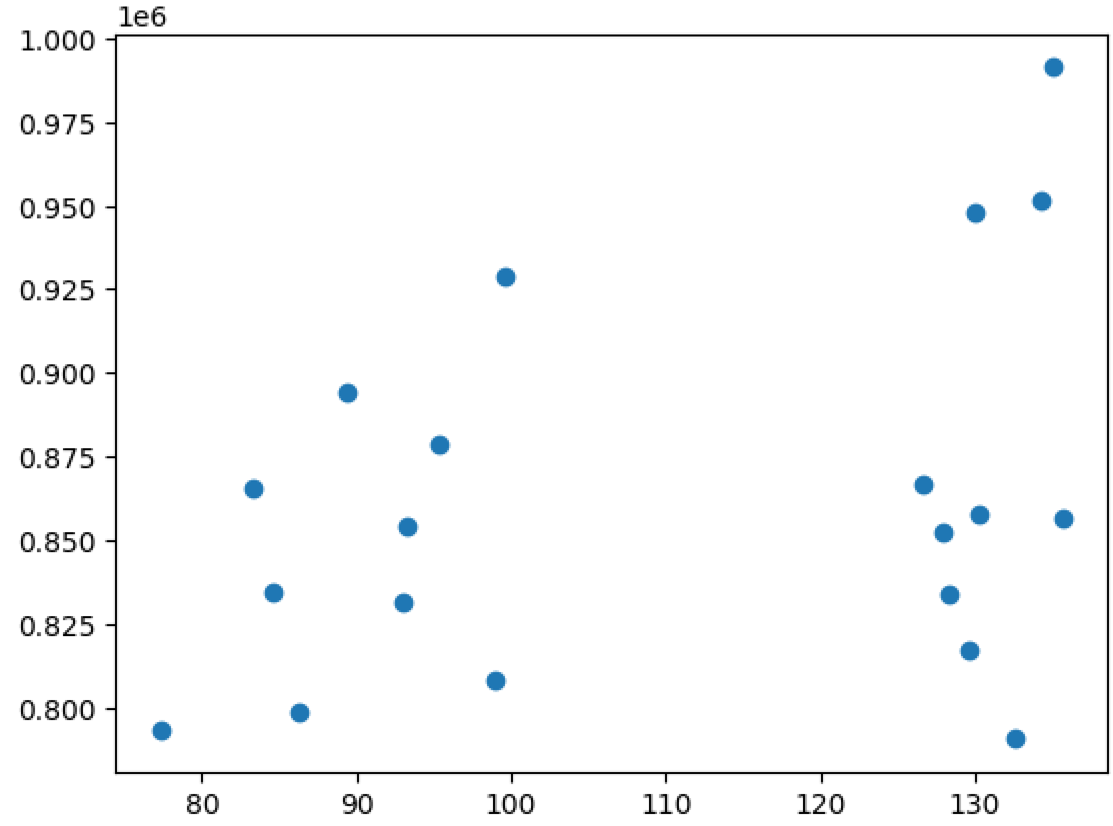
```
# Cell 4
import numpy as np
import matplotlib.pyplot as plt
```

```
# Cell 5
menDf = brainFrame[(brainFrame.Gender == 'Male')]
womenDf = brainFrame[(brainFrame.Gender == 'Female')]
```

```
# Cell 6
menMeanSmarts = menDf[["PIQ", "FSIQ", "VIQ"]].mean(axis=1)
plt.scatter(menMeanSmarts, menDf["MRI_Count"])
plt.show
%matplotlib inline
```



```
# Cell 7
# Graph the women-only filtered dataframe
womenMeanSmarts = womenDf[["PIQ", "FSIQ", "VIQ"]].mean(axis=1)
plt.scatter(womenMeanSmarts, womenDf["MRI_Count"])
plt.show()
%matplotlib inline
```




```
# Cell 8
brainFrame.corr(method='pearson')
```

```
<ipython-input-18-176cb45dad0b>:2: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In
  brainFrame.corr(method='pearson')
```

	FSIQ	VIQ	PIQ	Weight	Height	MRI_Count
FSIQ	1.000000	0.946639	0.934125	-0.051483	-0.086002	0.357641
VIQ	0.946639	1.000000	0.778135	-0.076088	-0.071068	0.337478
PIQ	0.934125	0.778135	1.000000	0.002512	-0.076723	0.386817
Weight	-0.051483	-0.076088	0.002512	1.000000	0.699614	0.513378
Height	-0.086002	-0.071068	-0.076723	0.699614	1.000000	0.601712
MRI_Count	0.357641	0.337478	0.386817	0.513378	0.601712	1.000000

Notice at the left-to-right diagonal in the correlation table generated above. Why is the diagonal filled with 1s? Is that a coincidence? Explain.

- The reason is that it is the same data so it shows a 1.

Still looking at the correlation table above, notice that the values are mirrored; values below the 1diagonal have a mirrored counterpart above the 1 diagonal. Is that a coincidence? Explain.

- The reason is that it is the same value of correlation so it is mirrored because it is the same.

```
# Cell 9
womenDf.corr(method='pearson')
```

```
<ipython-input-19-eb924f62ceff>:2: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In
  womenDf.corr(method='pearson')
```

	FSIQ	VIQ	PIQ	Weight	Height	MRI_Count
FSIQ	1.000000	0.955717	0.939382	0.038192	-0.059011	0.325697
VIQ	0.955717	1.000000	0.802652	-0.021889	-0.146453	0.254933
PIQ	0.939382	0.802652	1.000000	0.113901	-0.001242	0.396157
Weight	0.038192	-0.021889	0.113901	1.000000	0.552357	0.446271
Height	-0.059011	-0.146453	-0.001242	0.552357	1.000000	0.174541
MRI_Count	0.325697	0.254933	0.396157	0.446271	0.174541	1.000000

```
# Cell 10
menDf.corr(method='pearson')
```

```
<ipython-input-20-de98488d4683>:2: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In
  menDf.corr(method='pearson')
```

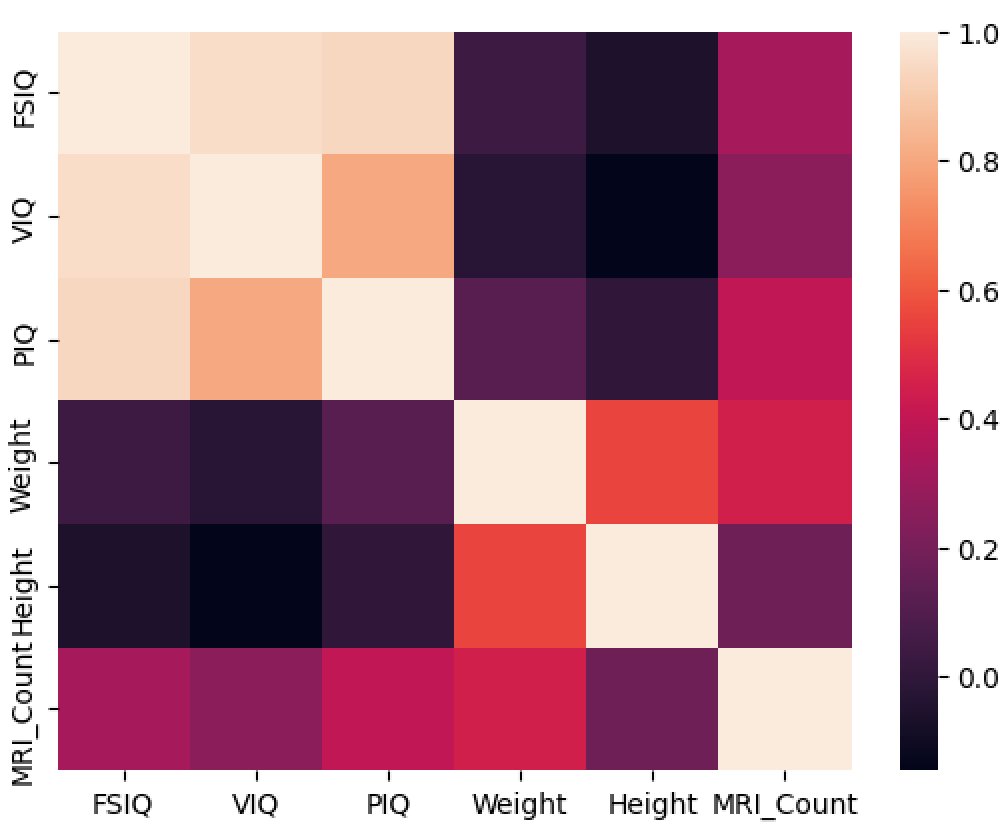
	FSIQ	VIQ	PIQ	Weight	Height	MRI_Count
FSIQ	1.000000	0.944400	0.930694	-0.278140	-0.356110	0.498369
VIQ	0.944400	1.000000	0.766021	-0.350453	-0.355588	0.413105
PIQ	0.930694	0.766021	1.000000	-0.156863	-0.287676	0.568237
Weight	-0.278140	-0.350453	-0.156863	1.000000	0.406542	-0.076875
Height	-0.356110	-0.355588	-0.287676	0.406542	1.000000	0.301543
MRI_Count	0.498369	0.413105	0.568237	-0.076875	0.301543	1.000000

```
# Cell 11
!pip install seaborn
```

```
Requirement already satisfied: seaborn in /usr/local/lib/python3.10/dist-packages (0.13.1)
Requirement already satisfied: numpy!=1.24.0,>=1.20 in /usr/local/lib/python3.10/dist-packages (from seaborn) (1.23.5)
Requirement already satisfied: pandas>=1.2 in /usr/local/lib/python3.10/dist-packages (from seaborn) (1.5.3)
Requirement already satisfied: matplotlib!=3.6.1,>=3.4 in /usr/local/lib/python3.10/dist-packages (from seaborn) (3.7.1
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1,>=3.
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1,>=3.4->s
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1,>=3
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1,>=3
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1,>=3.4
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1,>=3.4->
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1,>=3.
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1,
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.2->seaborn) (202
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7->matplotl
```

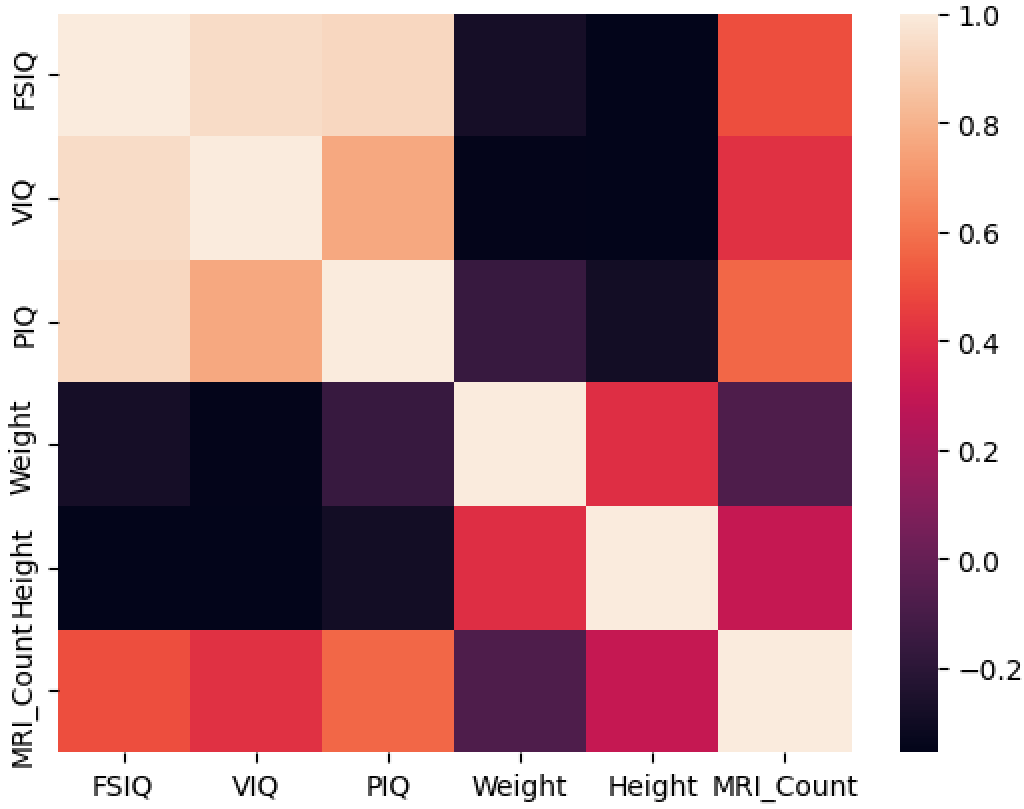
```
# Cell 12
import seaborn as sns

wcorr = womenDf.corr()
sns.heatmap(wcorr)
#plt.savefig('attribute_correlations.png', tight_layout=True)
```

```
<ipython-input-22-ba695c5fc767>:4: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In
  wcorr = womenDf.corr()
<Axes: >
```




```
# Cell 13
wcorr = menDf.corr()
sns.heatmap(wcorr)
#plt.savefig('attribute_correlations.png', tight_layout=True)
```

```
<ipython-input-23-a8cdecfd565e>:2: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In
  wcorr = menDf.corr()
<Axes: >
```



Many variable pairs present correlation close to zero. What does that mean?

- The reason is those variables are not that much correlated so it shows a number closer to zero than to one.

Why separate the genders?

- The reason is to correlate gender to the iq of a person.

What variables have stronger correlation with brain size (MRI_Count)? Is that expected? Explain.

- The variables that have a stronger correlation is FSIQ, VIQ, and PIQ. I think it is to be expected because we believe that a bigger brain capacity will result in higher overall iq.

Supplementary Activity

```
import pandas as pd
PowerConsumption = '/content/Concrete_Data.csv'
ConsumptionFrame = pd.read_csv(PowerConsumption, sep = ',')
```

```
ConsumptionFrame.head()
```

	Cement (component 1)(kg in a m^3 mixture)	Blast Furnace Slag (component 2)(kg in a m^3 mixture)	Fly Ash (component 3)(kg in a m^3 mixture)	Water (component 4)(kg in a m^3 mixture)	Superplasticizer (component 5)(kg in a m^3 mixture)	Coarse Aggregate (component 6)(kg in a m^3 mixture)	Fine Aggregate (component 7)(kg in a m^3 mixture)	Age (day)	Concrete compressive strength(MPa, megapascals)
0	540.0	0.0	0.0	162.0	2.5	1040.0	676.0	28	79.99
1	540.0	0.0	0.0	162.0	2.5	1055.0	676.0	28	61.89
2	332.5	142.5	0.0	228.0	0.0	932.0	594.0	270	40.27
3	332.5	142.5	0.0	228.0	0.0	932.0	594.0	365	41.05
4	198.6	132.4	0.0	192.0	0.0	978.4	825.5	360	44.30

```
ConsumptionFrame.describe()
```

ement ponent in a m^3 ture)	Blast Furnace Slag (component 2)(kg in a m^3 mixture)	Fly Ash (component 3)(kg in a m^3 mixture)	Water (component 4)(kg in a m^3 mixture)	Superplasticizer (component 5)(kg in a m^3 mixture)	Coarse Aggregate (component 6)(kg in a m^3 mixture)	Fine Aggregate (component 7)(kg in a m^3 mixture)	Age (day)	Concrete compressive strength(MPa, megapascals)
00000	1030.000000	1030.000000	1030.000000	1030.000000	1030.000000	1030.000000	1030.000000	1030.000000
67864	73.895825	54.188350	181.567282	6.204660	972.918932	773.580485	45.662136	35.817961
06364	86.279342	63.997004	21.354219	5.973841	77.753954	80.175980	63.169912	16.705742
00000	0.000000	0.000000	121.800000	0.000000	801.000000	594.000000	1.000000	2.330000
75000	0.000000	0.000000	164.900000	0.000000	932.000000	730.950000	7.000000	23.710000
00000	22.000000	0.000000	185.000000	6.400000	968.000000	779.500000	28.000000	34.445000
00000	142.950000	118.300000	192.000000	10.200000	1029.400000	824.000000	56.000000	46.135000
00000	359.400000	200.100000	247.000000	32.200000	1145.000000	992.600000	365.000000	82.600000

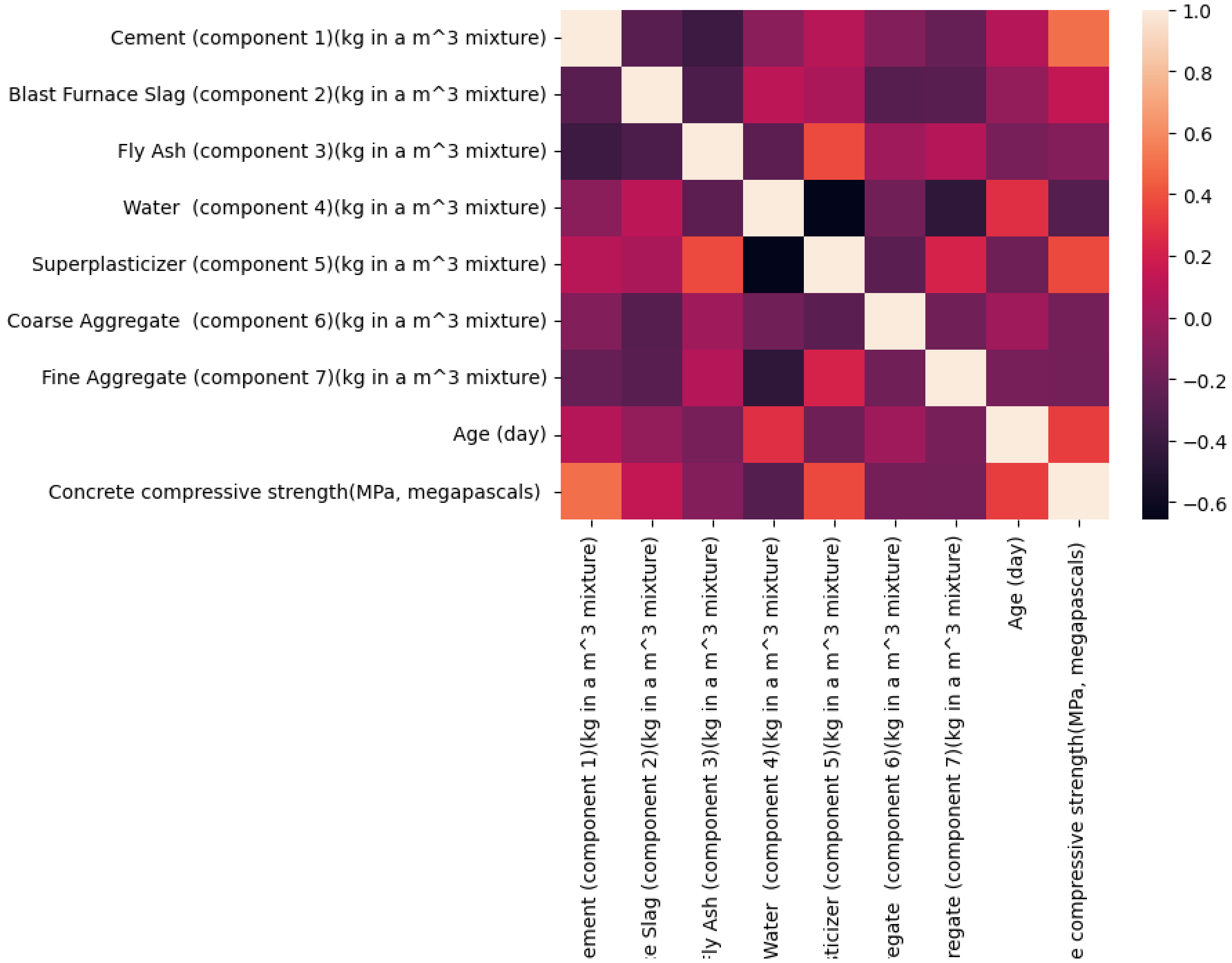
```
import numpy as np
import matplotlib.pyplot as plt
```

```
ConsumptionFrame.corr(method='pearson')
```

	Cement (component 1)(kg in a m^3 mixture)	Blast Furnace Slag (component 2)(kg in a m^3 mixture)	Fly Ash (component 3)(kg in a m^3 mixture)	Water (component 4)(kg in a m^3 mixture)	Superplasticizer (component 5)(kg in a m^3 mixture)	Coarse Aggregate (component 6)(kg in a m^3 mixture)	Fine Aggregate (component 7)(kg in a m^3 mixture)	Age (day)	c str me
Cement (component 1) (kg in a m^3 mixture)	1.000000	-0.275216	-0.397467	-0.081587	0.092386	-0.109349	-0.222718	0.081946	
Blast Furnace Slag (component 2) (kg in a m^3 mixture)	-0.275216	1.000000	-0.323580	0.107252	0.043270	-0.283999	-0.281603	-0.044246	
Fly Ash (component 3) (kg in a m^3 mixture)	-0.397467	-0.323580	1.000000	-0.256984	0.377503	-0.009961	0.079108	-0.154371	
Water (component 4) (kg in a m^3 mixture)	-0.081587	0.107252	-0.256984	1.000000	-0.657533	-0.182294	-0.450661	0.277618	
Superplasticizer (component 5) (kg in a m^3 mixture)	0.092386	0.043270	0.377503	-0.657533	1.000000	-0.265999	0.222691	-0.192700	
Coarse Aggregate (component 6) (kg in a m^3 mixture)	-0.109349	-0.283999	-0.009961	-0.182294	-0.265999	1.000000	-0.178481	-0.003016	
Fine Aggregate (component 7) (kg in a m^3 mixture)	-0.222718	-0.281603	0.079108	-0.450661	0.222691	-0.178481	1.000000	-0.156095	
Age (day)	0.081946	-0.044246	-0.154371	0.277618	-0.192700	-0.003016	-0.156095	1.000000	
Concrete compressive strength(MPa, megapascals)	0.497832	0.134829	-0.105755	-0.289633	0.366079	-0.164935	-0.167241	0.328873	

```
import seaborn as sns

conw = ConsumptionFrame.corr()
sns.heatmap(conw)
#plt.savefig('attribute_correlations.png', tight_layout=True)
```

```
<Axes: >
```



Base on the heatmap that is produce we can tell that Cement and Superplasticizer can result to higher concrete compressive strength and overall age of the concrete.

Conclusion

In conclusion I have learned many things in this module like manipulating data to be easily presentable and how to present the data and correlate it to other things like other attributes. I think that there is still some room for improvement in using datasets and presenting them.