In [1]:
```
import numpy as np
import pandas as pd
```

In [2]:
```
raw_data = pd.read_csv("cereal.csv")
```

In [3]:
```
raw_data.head()
```

Out[3]:

	name	mfr	type	calories	protein	fat	sodium	fiber	carbo	sugars	potass	vitamins	shelf	weight	cups	rating
0	100% Bran	N	C	70	4	1	130	10.0	5.0	6	280	25	3	1.0	0.33	68.402973
1	100% Natural Bran	Q	C	120	3	5	15	2.0	8.0	8	135	0	3	1.0	1.00	33.983679
2	All-Bran	K	C	70	4	1	260	9.0	7.0	5	320	25	3	1.0	0.33	59.425505
3	All-Bran with Extra Fiber	K	C	50	4	0	140	14.0	8.0	0	330	25	3	1.0	0.50	93.704912
4	Almond Delight	R	C	110	2	2	200	1.0	14.0	8	-1	25	3	1.0	0.75	34.384843

In [4]:
```
raw_data.shape
```

Out[4]:
```
(77, 16)
```

In [5]:
```
raw_data.columns
```

Out[5]:
```
Index(['name', 'mfr', 'type', 'calories', 'protein', 'fat', 'sodium', 'fiber',
       'carbo', 'sugars', 'potass', 'vitamins', 'shelf', 'weight', 'cups',
       'rating'],
      dtype='object')
```

In [6]:
```
raw_data.dtypes
```

Out[6]:
```
name         object
mfr          object
type         object
calories      int64
protein       int64
fat           int64
sodium        int64
fiber       float64
carbo       float64
sugars        int64
potass        int64
vitamins      int64
shelf         int64
weight      float64
cups        float64
rating      float64
dtype: object
```

1. Sugar

In [7]:
```
# sugar_per_ounce = suger_per_serving / sugar_per_weight
raw_data["sugar_per_ounce"] = raw_data["sugars"] / raw_data["weight"]
```

In [8]:
```
raw_data
```

Out[8]:

	name	mfr	type	calories	protein	fat	sodium	fiber	carbo	sugars	potass	vitamins	shelf	weight	cups	rating	sugar_per_ounce
0	100% Bran	N	C	70	4	1	130	10.0	5.0	6	280	25	3	1.0	0.33	68.402973	6.0
1	100% Natural Bran	Q	C	120	3	5	15	2.0	8.0	8	135	0	3	1.0	1.00	33.983679	8.0
2	All-Bran	K	C	70	4	1	260	9.0	7.0	5	320	25	3	1.0	0.33	59.425505	5.0
3	All-Bran with Extra Fiber	K	C	50	4	0	140	14.0	8.0	0	330	25	3	1.0	0.50	93.704912	0.0
4	Almond Delight	R	C	110	2	2	200	1.0	14.0	8	-1	25	3	1.0	0.75	34.384843	8.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
72	Triples	G	C	110	2	1	250	0.0	21.0	3	60	25	3	1.0	0.75	39.106174	3.0
73	Trix	G	C	110	1	1	140	0.0	13.0	12	25	25	2	1.0	1.00	27.753301	12.0
74	Wheat Chex	R	C	100	3	1	230	3.0	17.0	3	115	25	1	1.0	0.67	49.787445	3.0
75	Wheaties	G	C	100	3	1	200	3.0	17.0	3	110	25	1	1.0	1.00	51.592193	3.0
76	Wheaties Honey Gold	G	C	110	2	1	200	1.0	16.0	8	60	25	1	1.0	0.75	36.187559	8.0

77 rows × 17 columns

Which product has the least amount of sugar per ounce?

In [9]:
```
raw_data.sort_values(by='sugar_per_ounce').head(10)
```

Out[9]:

	name	mfr	type	calories	protein	fat	sodium	fiber	carbo	sugars	potass	vitamins	shelf	weight	cups	rating	sugar_per_ounce
57	Quaker Oatmeal	Q	H	100	5	2	0	2.7	-1.0	-1	110	0	1	1.00	0.67	50.828392	-1.0
20	Cream of Wheat (Quick)	N	H	100	3	0	80	1.0	21.0	0	-1	0	2	1.00	1.00	64.533816	0.0
63	Shredded Wheat	N	C	80	2	0	0	3.0	16.0	0	95	0	1	0.83	1.00	68.235885	0.0
64	Shredded Wheat 'n'Bran	N	C	90	3	0	0	4.0	19.0	0	140	0	1	1.00	0.67	74.472949	0.0
3	All-Bran with Extra Fiber	K	C	50	4	0	140	14.0	8.0	0	330	25	3	1.00	0.50	93.704912	0.0
54	Puffed Rice	Q	C	50	1	0	0	0.0	13.0	0	15	0	3	0.50	1.00	60.756112	0.0
55	Puffed Wheat	Q	C	50	2	0	0	1.0	10.0	0	50	0	3	0.50	1.00	63.005645	0.0
65	Shredded Wheat spoon size	N	C	90	3	0	0	3.0	20.0	0	120	0	1	1.00	0.67	72.801787	0.0
11	Cheerios	G	C	110	6	2	290	2.0	17.0	1	105	25	1	1.00	1.25	50.764999	1.0
16	Corn Flakes	K	C	100	2	0	290	1.0	21.0	2	35	25	1	1.00	1.00	45.863324	2.0

In [10]:
```
print(raw_data.loc[20, 'name'])
print(raw_data.loc[63, 'name'])
print(raw_data.loc[64, 'name'])
print(raw_data.loc[3, 'name'])
print(raw_data.loc[54, 'name'])
print(raw_data.loc[55, 'name'])
print(raw_data.loc[65, 'name'])
```

```
Cream of Wheat (Quick)
Shredded Wheat
Shredded Wheat 'n'Bran
All-Bran with Extra Fiber
Puffed Rice
Puffed Wheat
Shredded Wheat spoon size
```

What is the average amount of sugar per ounce?

In [11]:
```
raw_data["sugar_per_ounce"].mean()
```

Out[11]:
```
6.555489623158796
```

2. Calories

Calculate calories per gram for each cereal product?

In [12]:
```
ounce_per_serving = raw_data["weight"].astype(float)
```

In [13]:
```
gram_per_serving = 0.035 * ounce_per_serving
```

In [14]:
```
calories_per_serving = raw_data["calories"].astype(float)
#gram_per_serving     = raw_data["weight")].astype(float)

calories_per_gram = calories_per_serving / gram_per_serving
calories_per_gram
```

Out[14]:
```
0     2000.000000
1     3428.571429
2     2000.000000
3     1428.571429
4     3142.857143
         ...
72    3142.857143
73    3142.857143
74    2857.142857
75    2857.142857
76    3142.857143
Length: 77, dtype: float64
```

Identify the product with the highest value of calories per gram?

In [15]:
```
idx = calories_per_gram.idxmax()
raw_data.loc[idx,"name"]
```

Out[15]: 'Muesli Raisins; Dates; & Almonds'

Identify the product with the lowest value of calories per gram?

In [16]:
```
idx=calories_per_gram.idxmin()
raw_data.loc[idx,"name"]
```

Out[16]: 'All-Bran with Extra Fiber'

In [ ]:
```

```