In [1]:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

In [2]:

```
df = pd.read_csv('F:\Classroom\GREATLEARNING DSBA\Statstical method for decision mking
\Abc.csv')
# The file Mcdonald's was downloaded as Abc, for convineince
```

In [3]:

```
# Checking head and tail to get an idea about the dataframe
df.head(3)
```

Out[3]:

	Category	Item	Serving Size	Calories	Calories from Fat	Total Fat	Total Fat (% Daily Value)	Saturated Fat	Saturated Fat (% Daily Value)	Trans Fa
0	Breakfast	Egg McMuffin	4.8 oz (136 g)	300	120	13.0	20	5.0	25	0.(
1	Breakfast	Egg White Delight	4.8 oz (135 g)	250	70	8.0	12	3.0	15	0.(
2	Breakfast	Sausage McMuffin	3.9 oz (111 g)	370	200	23.0	35	8.0	42	0.(

3 rows × 24 columns

In [4]:

```
df.tail(5)
```

Out[4]:

	Category	Item	Serving Size	Calories	Calories from Fat	Total Fat	Total Fat (% Daily Value)	Saturated Fat	Saturated Fat (% Daily Value)	Ti
255	Smoothies & Shakes	McFlurry with Oreo Cookies (Small)	10.1 oz (285 g)	510	150	17.0	26	9.0	44	
256	Smoothies & Shakes	McFlurry with Oreo Cookies (Medium)	13.4 oz (381 g)	690	200	23.0	35	12.0	58	
257	Smoothies & Shakes	McFlurry with Oreo Cookies (Snack)	6.7 oz (190 g)	340	100	11.0	17	6.0	29	
258	Smoothies & Shakes	McFlurry with Reese's Peanut Butter Cups (Medium)	14.2 oz (403 g)	810	290	32.0	50	15.0	76	
259	Smoothies & Shakes	McFlurry with Reese's Peanut Butter Cups (Snack)	7.1 oz (202 g)	410	150	16.0	25	8.0	38	

5 rows × 24 columns

In [5]:

```
# checking info()

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 260 entries, 0 to 259
Data columns (total 24 columns):
 #   Column                          Non-Null Count  Dtype  
---  ------                          --------------  -----  
 0   Category                        260 non-null    object 
 1   Item                            260 non-null    object 
 2   Serving Size                    260 non-null    object 
 3   Calories                        260 non-null    int64  
 4   Calories from Fat               260 non-null    int64  
 5   Total Fat                       260 non-null    float64
 6   Total Fat (% Daily Value)       260 non-null    int64  
 7   Saturated Fat                   260 non-null    float64
 8   Saturated Fat (% Daily Value)   260 non-null    int64  
 9   Trans Fat                       260 non-null    float64
 10  Cholesterol                     260 non-null    int64  
 11  Cholesterol (% Daily Value)     260 non-null    int64  
 12  Sodium                          260 non-null    int64  
 13  Sodium (% Daily Value)          260 non-null    int64  
 14  Carbohydrates                   260 non-null    int64  
 15  Carbohydrates (% Daily Value)   260 non-null    int64  
 16  Dietary Fiber                   260 non-null    int64  
 17  Dietary Fiber (% Daily Value)   260 non-null    int64  
 18  Sugars                          260 non-null    int64  
 19  Protein                         260 non-null    int64  
 20  Vitamin A (% Daily Value)       260 non-null    int64  
 21  Vitamin C (% Daily Value)       260 non-null    int64  
 22  Calcium (% Daily Value)         260 non-null    int64  
 23  Iron (% Daily Value)            260 non-null    int64  
dtypes: float64(3), int64(18), object(3)
memory usage: 48.9+ KB
```

In [6]:

```
# Checking the data spread

df.describe().T
```

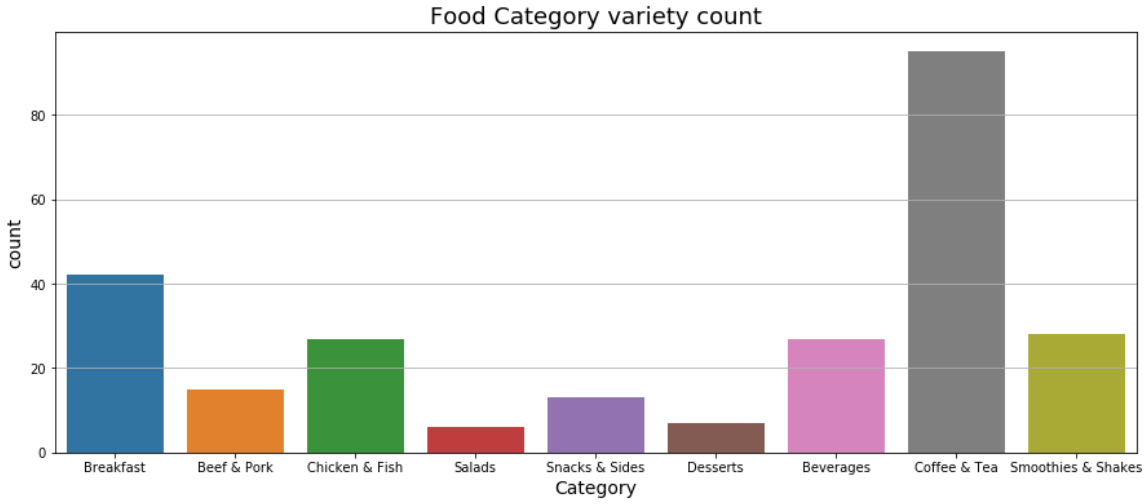
Out[6]:

	count	mean	std	min	25%	50%	75%	max
Calories	260.0	368.269231	240.269886	0.0	210.000	340.0	500.00	1880.0
Calories from Fat	260.0	127.096154	127.875914	0.0	20.000	100.0	200.00	1060.0
Total Fat	260.0	14.165385	14.205998	0.0	2.375	11.0	22.25	118.0
Total Fat (% Daily Value)	260.0	21.815385	21.885199	0.0	3.750	17.0	35.00	182.0
Saturated Fat	260.0	6.007692	5.321873	0.0	1.000	5.0	10.00	20.0
Saturated Fat (% Daily Value)	260.0	29.965385	26.639209	0.0	4.750	24.0	48.00	102.0
Trans Fat	260.0	0.203846	0.429133	0.0	0.000	0.0	0.00	2.5
Cholesterol	260.0	54.942308	87.269257	0.0	5.000	35.0	65.00	575.0
Cholesterol (% Daily Value)	260.0	18.392308	29.091653	0.0	2.000	11.0	21.25	192.0
Sodium	260.0	495.750000	577.026323	0.0	107.500	190.0	865.00	3600.0
Sodium (% Daily Value)	260.0	20.676923	24.034954	0.0	4.750	8.0	36.25	150.0
Carbohydrates	260.0	47.346154	28.252232	0.0	30.000	44.0	60.00	141.0
Carbohydrates (% Daily Value)	260.0	15.780769	9.419544	0.0	10.000	15.0	20.00	47.0
Dietary Fiber	260.0	1.630769	1.567717	0.0	0.000	1.0	3.00	7.0
Dietary Fiber (% Daily Value)	260.0	6.530769	6.307057	0.0	0.000	5.0	10.00	28.0
Sugars	260.0	29.423077	28.679797	0.0	5.750	17.5	48.00	128.0
Protein	260.0	13.338462	11.426146	0.0	4.000	12.0	19.00	87.0
Vitamin A (% Daily Value)	260.0	13.426923	24.366381	0.0	2.000	8.0	15.00	170.0
Vitamin C (% Daily Value)	260.0	8.534615	26.345542	0.0	0.000	0.0	4.00	240.0
Calcium (% Daily Value)	260.0	20.973077	17.019953	0.0	6.000	20.0	30.00	70.0
Iron (% Daily Value)	260.0	7.734615	8.723263	0.0	0.000	4.0	15.00	40.0

1. Plot graphically which food categories have the highest and lowest varieties

In [7]:

```
plt.figure(figsize=(15,6))
plt.grid(True)
plt.title('Food Category variety count', fontsize=18)
plt.xlabel('Category', fontsize = 14)
plt.ylabel('Count',fontsize = 14)
sns.countplot(df.Category)
plt.show()
```



In [8]:

```
df.Category.value_counts()
```

Out[8]:

```
Coffee & Tea          95
Breakfast             42
Smoothies & Shakes    28
Chicken & Fish        27
Beverages             27
Beef & Pork           15
Snacks & Sides        13
Desserts               7
Salads                 6
Name: Category, dtype: int64
```

From the above plot it can be observed that the breakfast category "Coffee and Tea" has the highest variety wheras the category "Salads" has the lowest variety

2 Which all variables have an outlier?

- Outliers can be detected using the 1.5 x IQR range determination

- Or simply by plotting boxplots one could know the outliers

In [9]:

```
df1 = df.describe().T
```

In [10]:

```
df1['IQR'] = df1['75%'] - df1['25%']
df1['Upper limit'] = df1['75%'] + 1.5* df1['IQR']
```

In [11]:

```
df1['Differnece'] = df1['Upper limit']-df1['max']
```

In [12]:

```
df1['Outliers'] = df1['Differnece']<0
df1.head()
```

Out[12]:

	count	mean	std	min	25%	50%	75%	max	IQR	U
Calories	260.0	368.269231	240.269886	0.0	210.000	340.0	500.00	1880.0	290.000	935.
Calories from Fat	260.0	127.096154	127.875914	0.0	20.000	100.0	200.00	1060.0	180.000	470.
Total Fat	260.0	14.165385	14.205998	0.0	2.375	11.0	22.25	118.0	19.875	52.
Total Fat (% Daily Value)	260.0	21.815385	21.885199	0.0	3.750	17.0	35.00	182.0	31.250	81.
Saturated Fat	260.0	6.007692	5.321873	0.0	1.000	5.0	10.00	20.0	9.000	23.

The column Outliers tells if the max value of each column is within the 1.5 x IQR. It shows True/False accordingly

In [13]:

```
print('The following variables contain outliers: -')
df1[df1['Outliers']==True][['Outliers','Differnece','IQR']]
```

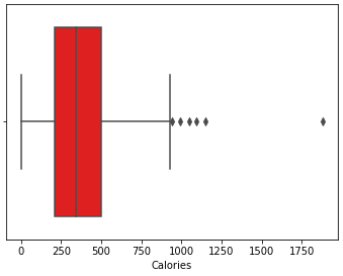
```
The following variables contain outliers: -
```

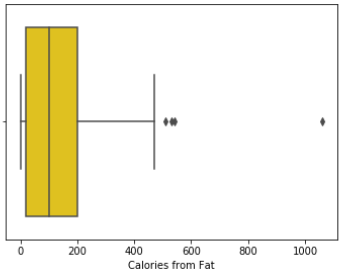
Out[13]:

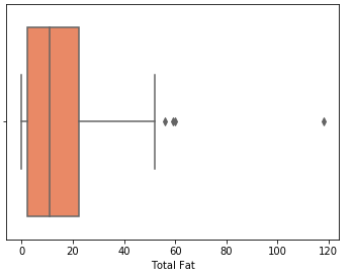
	Outliers	Differnece	IQR
Calories	True	-945.0000	290.000
Calories from Fat	True	-590.0000	180.000
Total Fat	True	-65.9375	19.875
Total Fat (% Daily Value)	True	-100.1250	31.250
Trans Fat	True	-2.5000	0.000
Cholesterol	True	-420.0000	60.000
Cholesterol (% Daily Value)	True	-141.8750	19.250
Sodium	True	-1598.7500	757.500
Sodium (% Daily Value)	True	-66.5000	31.500
Carbohydrates	True	-36.0000	30.000
Carbohydrates (% Daily Value)	True	-12.0000	10.000
Dietary Fiber (% Daily Value)	True	-3.0000	10.000
Sugars	True	-16.6250	42.250
Protein	True	-45.5000	15.000
Vitamin A (% Daily Value)	True	-135.5000	13.000
Vitamin C (% Daily Value)	True	-230.0000	4.000
Calcium (% Daily Value)	True	-4.0000	24.000
Iron (% Daily Value)	True	-2.5000	15.000

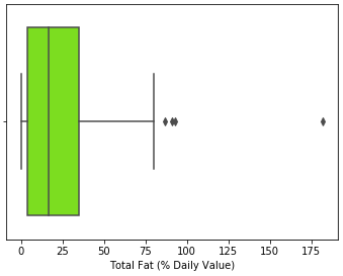
In [14]:

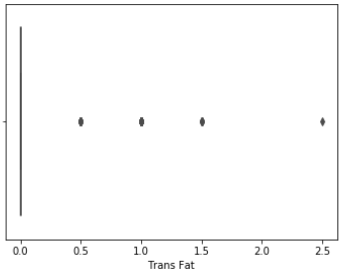
```
# Plotting boxplots for these variables would give a better visual representation


plt.figure(figsize = (20,30))
plt.subplot(6,3,1)
sns.boxplot(df['Calories'], color = 'red')

plt.subplot(6,3,2)
sns.boxplot(df['Calories from Fat'], color = 'gold')

plt.subplot(6,3,3)
sns.boxplot(df['Total Fat'], color = 'coral')

plt.subplot(6,3,4)
sns.boxplot(df['Total Fat (% Daily Value)'], color = 'lawngreen')

plt.subplot(6,3,5)
sns.boxplot(df['Trans Fat'], color = 'cyan')

plt.subplot(6,3,6)
sns.boxplot(df['Cholesterol'], color = 'cyan')

plt.subplot(6,3,7)
sns.boxplot(df['Cholesterol (% Daily Value)'], color = 'chocolate')

plt.subplot(6,3,8)
sns.boxplot(df['Sodium'], color = 'green')

plt.subplot(6,3,9)
sns.boxplot(df['Sodium (% Daily Value)'], color = 'tomato')

plt.subplot(6,3,10)
sns.boxplot(df['Carbohydrates'], color = 'orange')

plt.subplot(6,3,11)
sns.boxplot(df['Carbohydrates (% Daily Value)'], color = 'crimson')

plt.subplot(6,3,12)
sns.boxplot(df['Dietary Fiber (% Daily Value)'],color = 'navy')

plt.subplot(6,3,13)
sns.boxplot(df['Sugars'], color = 'yellow')

plt.subplot(6,3,14)
sns.boxplot(df['Protein'], color = 'blue')

plt.subplot(6,3,15)
sns.boxplot(df['Vitamin A (% Daily Value)'],color ='green')

plt.subplot(6,3,16)
sns.boxplot(df['Vitamin C (% Daily Value)'])

plt.subplot(6,3,17)
sns.boxplot(df['Calcium (% Daily Value)'], color = 'gold')

plt.subplot(6,3,18)
sns.boxplot(df['Iron (% Daily Value)'], color = 'red')

plt.show()
```

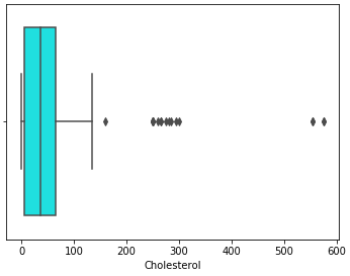

0 250 500 750 1000 1250 1500 1750
Calories

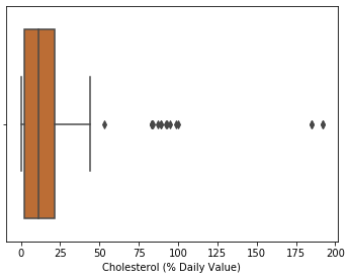

0 200 400 600 800 1000
Calories from Fat

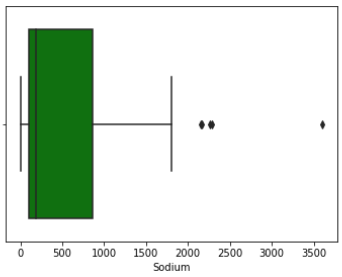

0 20 40 60 80 100 120
Total Fat

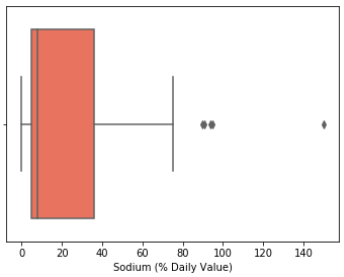

0 25 50 75 100 125 150 175
Total Fat (% Daily Value)

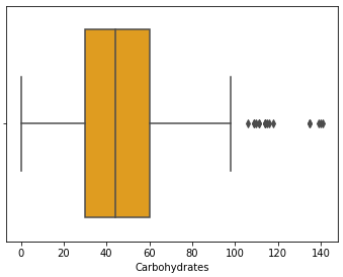

0.0 0.5 1.0 1.5 2.0 2.5
Trans Fat

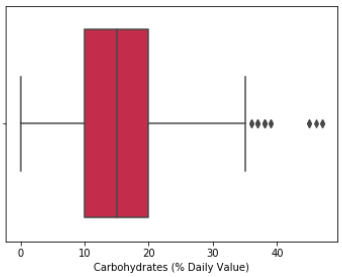

0 100 200 300 400 500 600
Cholesterol

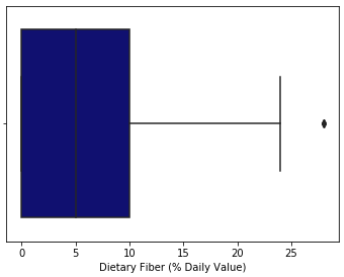

0 25 50 75 100 125 150 175 200
Cholesterol (% Daily Value)

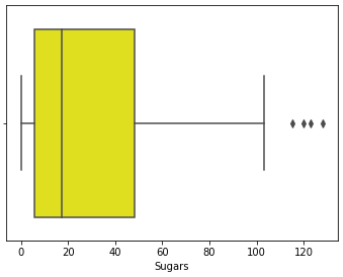

0 500 1000 1500 2000 2500 3000 3500
Sodium

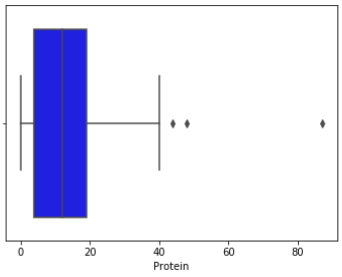

0 20 40 60 80 100 120 140
Sodium (% Daily Value)

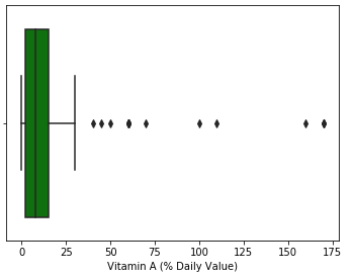

0 20 40 60 80 100 120 140
Carbohydrates

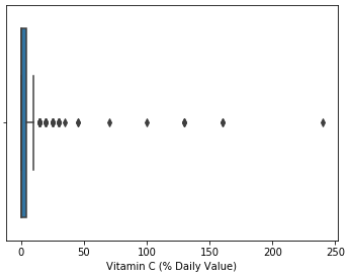

0 10 20 30 40
Carbohydrates (% Daily Value)

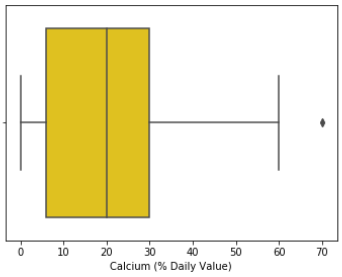

0 5 10 15 20 25
Dietary Fiber (% Daily Value)

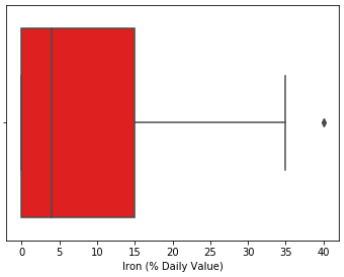

0 20 40 60 80 100 120
Sugars


0 20 40 60 80
Protein


0 25 50 75 100 125 150 175
Vitamin A (% Daily Value)


0 50 100 150 200 250
Vitamin C (% Daily Value)


0 10 20 30 40 50 60 70
Calcium (% Daily Value)


0 5 10 15 20 25 30 35 40
Iron (% Daily Value)

The following variables do not have outliers: -

In [32]:

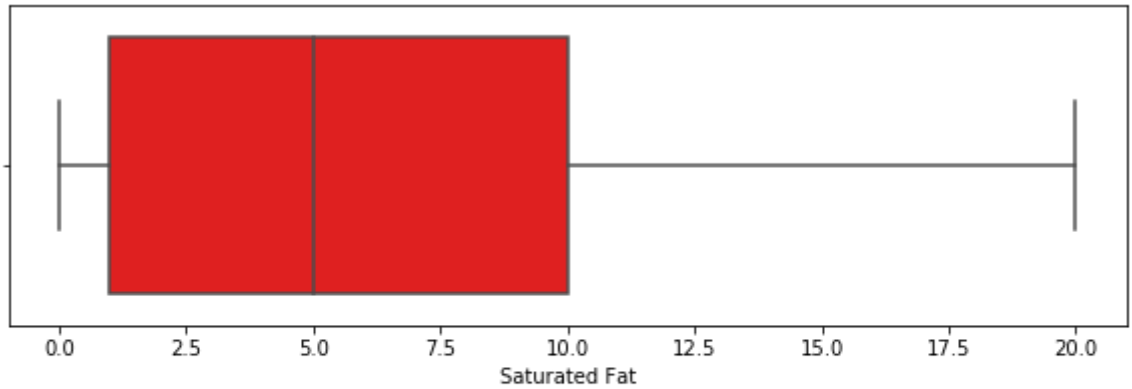
```
df1[df1['Outliers']==False][['Outliers','Differnece','IQR']]
```

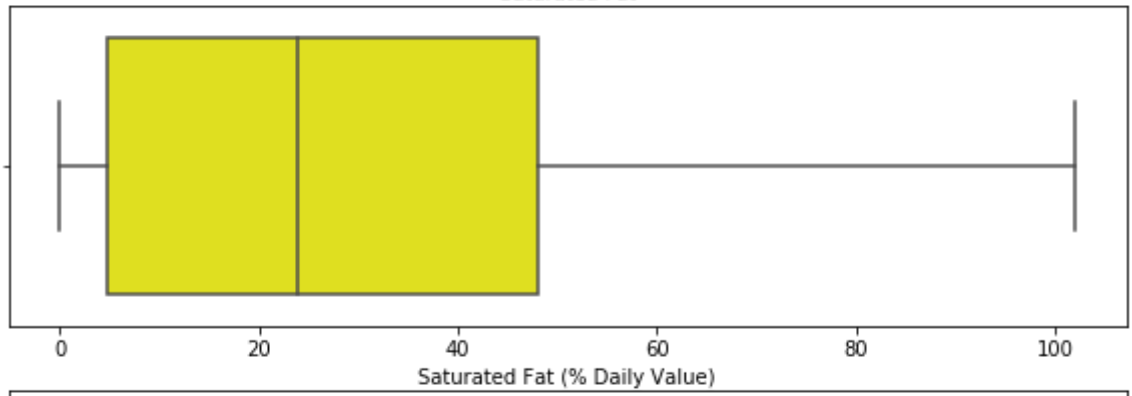
Out[32]:

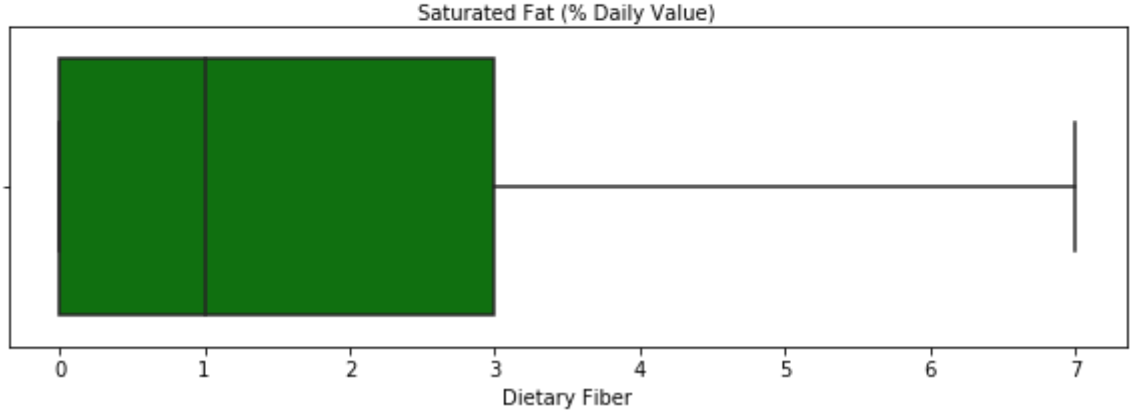
	Outliers	Differnece	IQR
Saturated Fat	False	3.500	9.00
Saturated Fat (% Daily Value)	False	10.875	43.25
Dietary Fiber	False	0.500	3.00

In [35]:

```
plt.figure(figsize = (10,10))

plt.subplot(3,1,1)
sns.boxplot(df['Saturated Fat'], color = 'red')

plt.subplot(3,1,2)
sns.boxplot(df['Saturated Fat (% Daily Value)'], color = 'yellow')

plt.subplot(3,1,3)
sns.boxplot(df['Dietary Fiber'], color = 'green')

plt.show()
```


0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
Saturated Fat


0
20
40
60
80
100
Saturated Fat (% Daily Value)


0
1
2
3
4
5
6
7
Dietary Fiber

3.Which variables have the highest correlation? Plot them and find out the value?

- Heat map can be used to visualize the correaltion

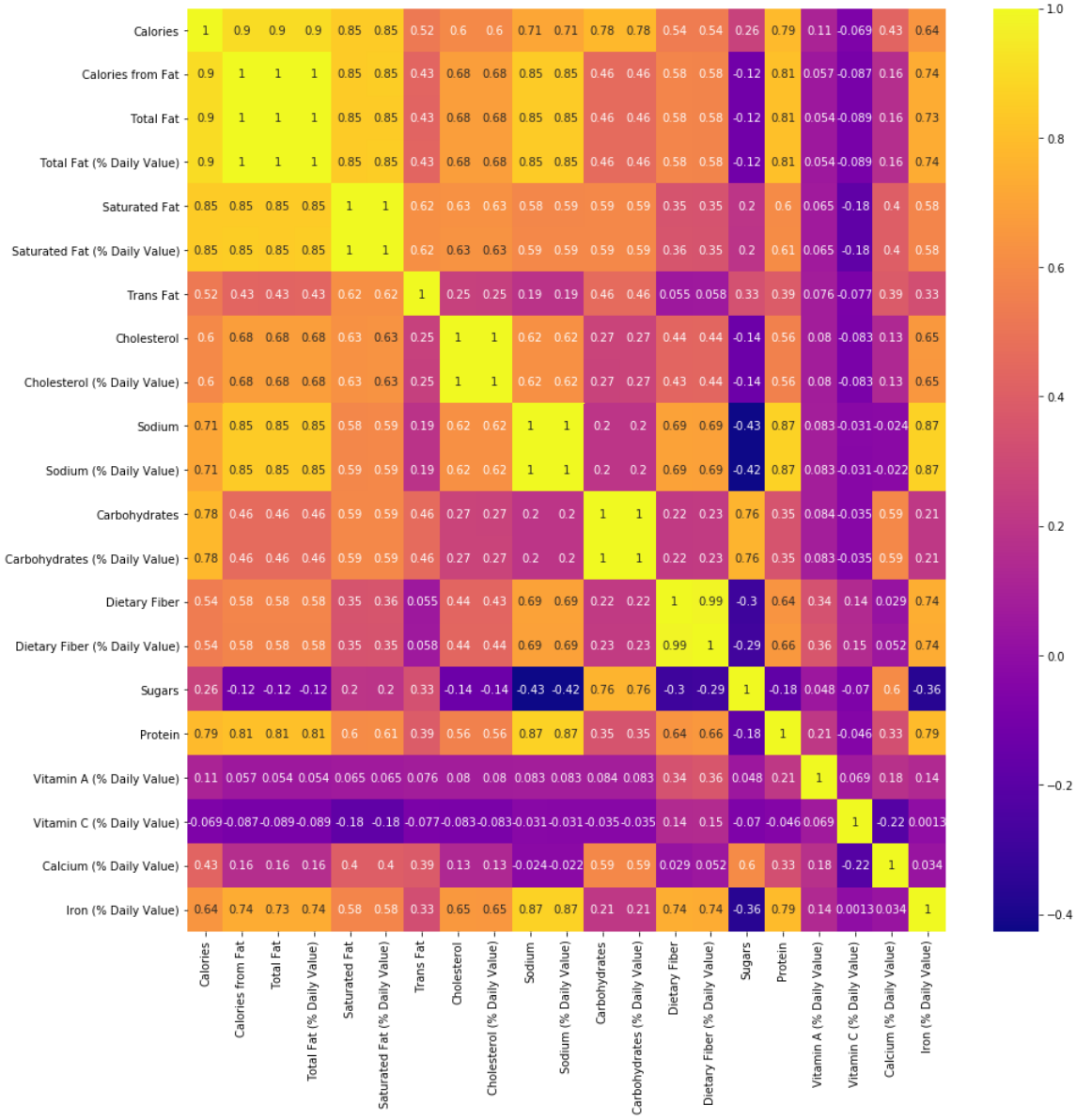
- Further a pairplot can be used to visualize the relationship

In [15]:

```
corr = df.corr()
```

In [16]:

```
plt.figure(figsize=(15,15))
sns.heatmap(corr,annot = True, cmap = 'plasma')
plt.show()
```

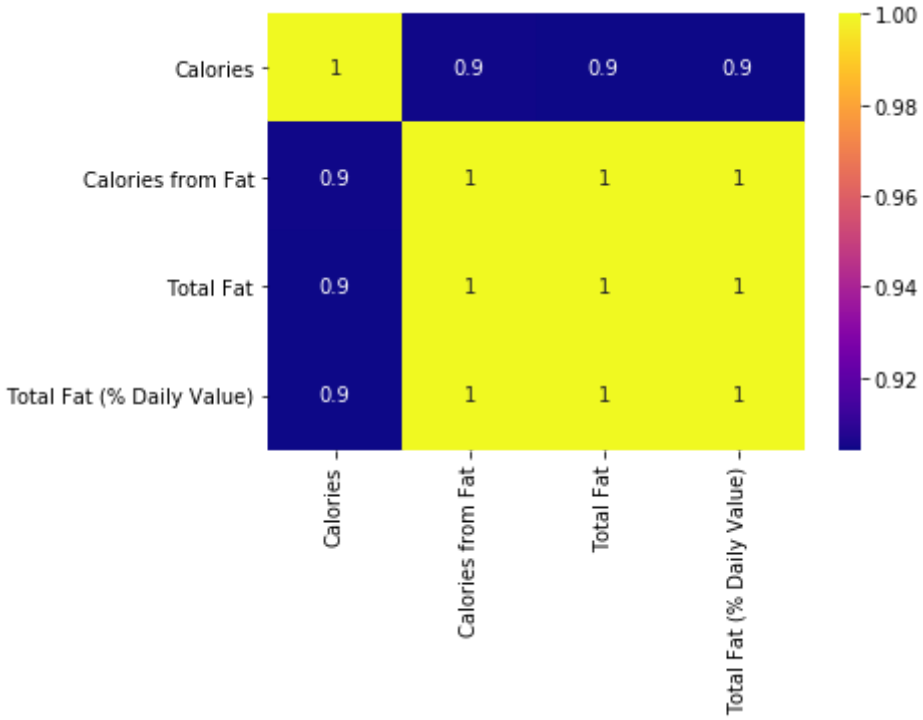


In [17]:

```
cor = df[['Calories','Calories from Fat','Total Fat','Total Fat (% Daily Value)']].corr
()
```

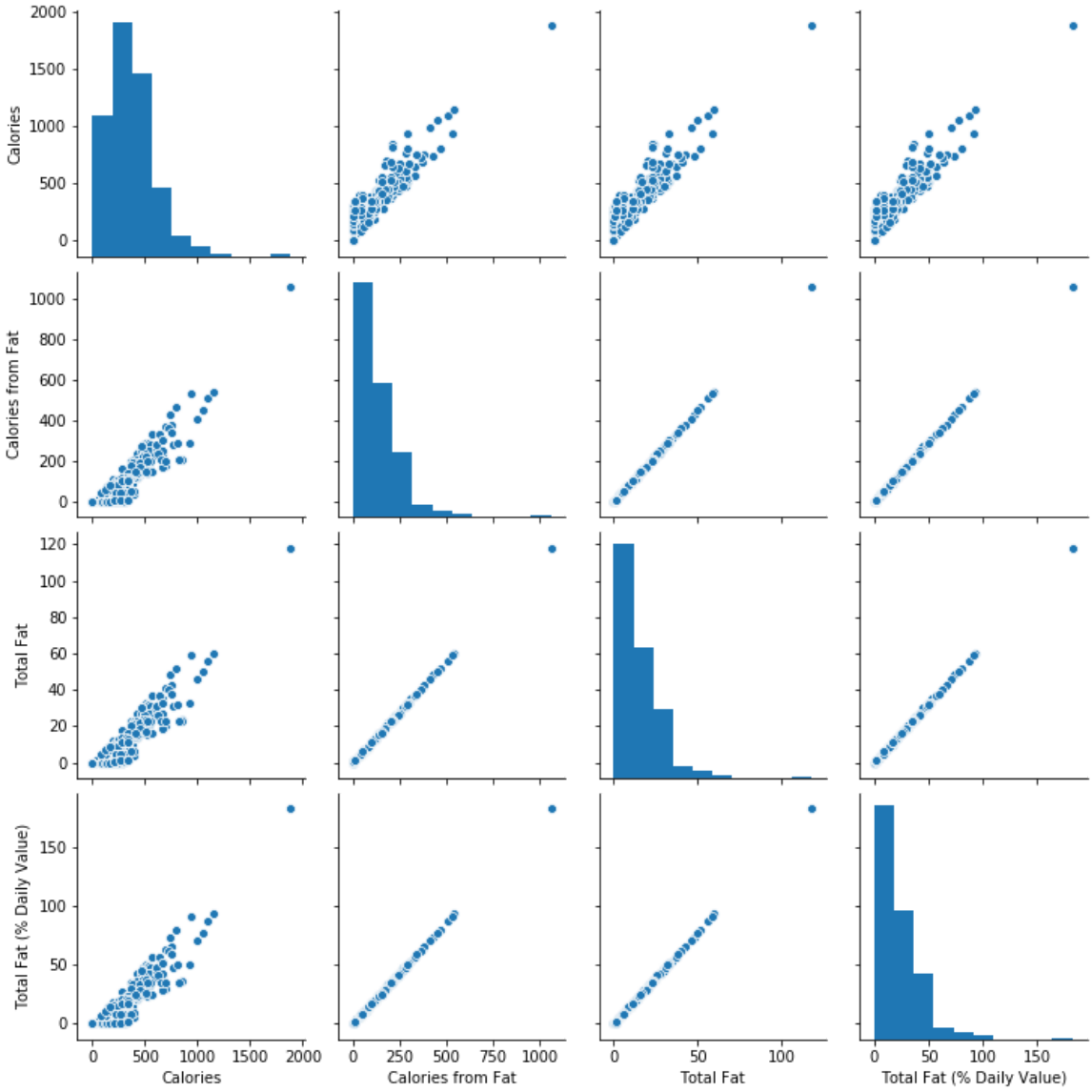
In [18]:

```
sns.heatmap(cor, annot = True, cmap = 'plasma');
```



In [19]:

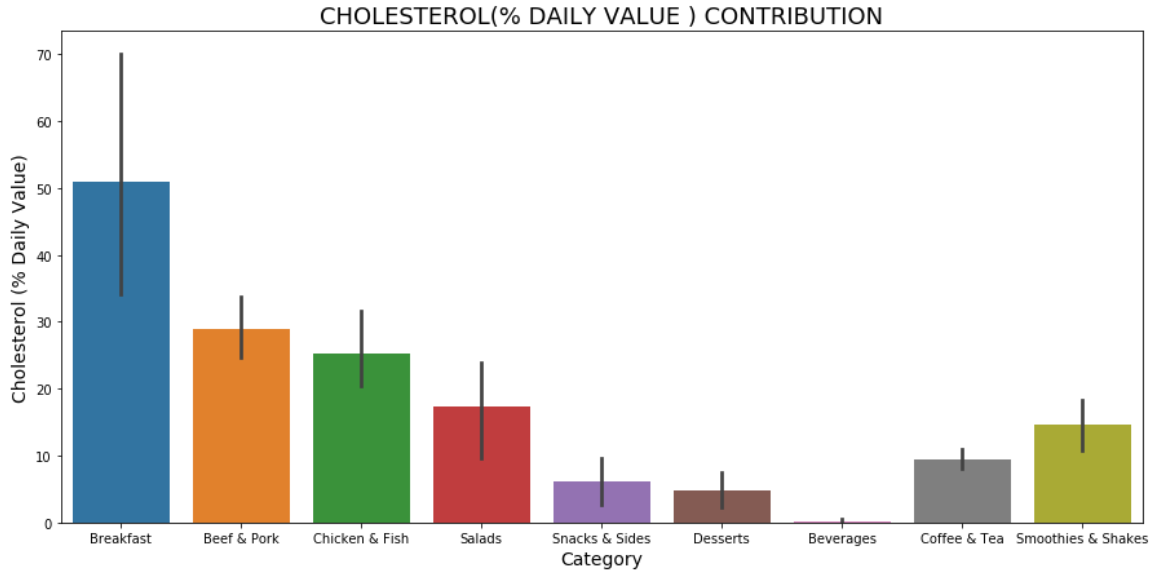
```
sns.pairplot(df[['Calories','Calories from Fat','Total Fat','Total Fat (% Daily Value)'
]]);
```


Calories
Calories from Fat
Total Fat
Total Fat (% Daily Value)
0
500
1000
1500
2000
0
200
400
600
800
1000
0
20
40
60
80
100
120
0
50
100
150
0
250
500
750
1000
0
50
100
150

4.Which category contributes to the maximum % of Cholesterol in a diet (% daily value)?

In [20]:

```
plt.figure(figsize = (15,7))
plt.title('CHOLESTEROL(% DAILY VALUE ) CONTRIBUTION', fontsize = 18)
plt.xlabel('Category', fontsize = 14)
plt.ylabel('Cholesterol in a diet (% daily value)', fontsize = 14)
sns.barplot(df['Category'], df['Cholesterol (% Daily Value)']);
```



In [21]:

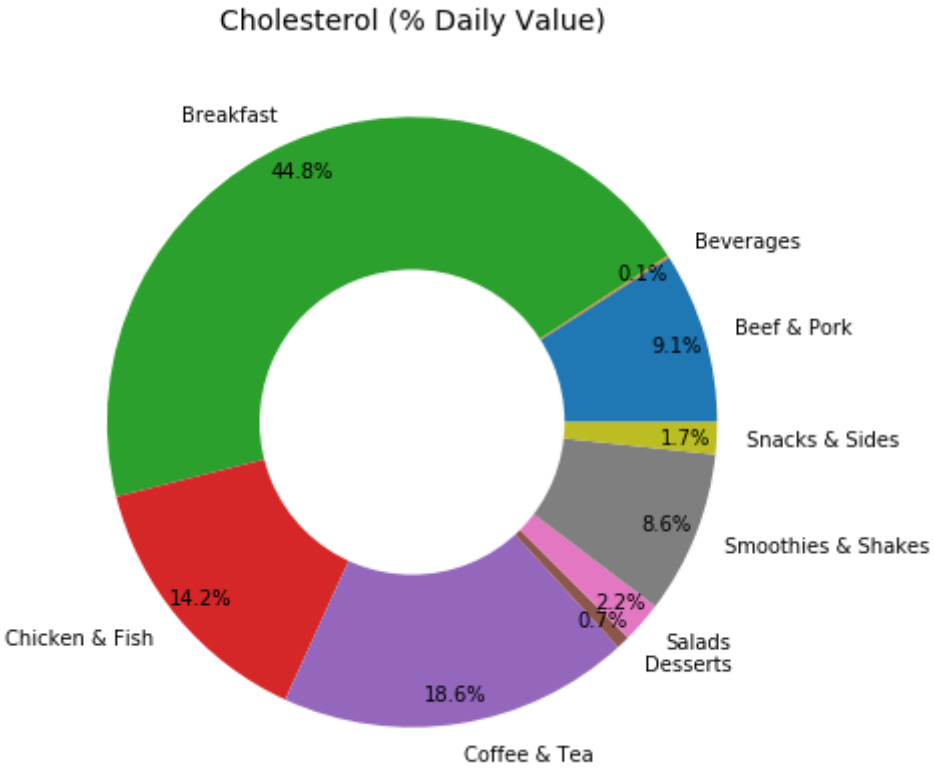
```
df_new = df.groupby('Category').sum()
df_new.reset_index(inplace = True)
df_new = df_new[['Category','Cholesterol (% Daily Value)']]
df_new.head()
```

Out[21]:

	Category	Cholesterol (% Daily Value)
0	Beef & Pork	434
1	Beverages	5
2	Breakfast	2140
3	Chicken & Fish	681
4	Coffee & Tea	891

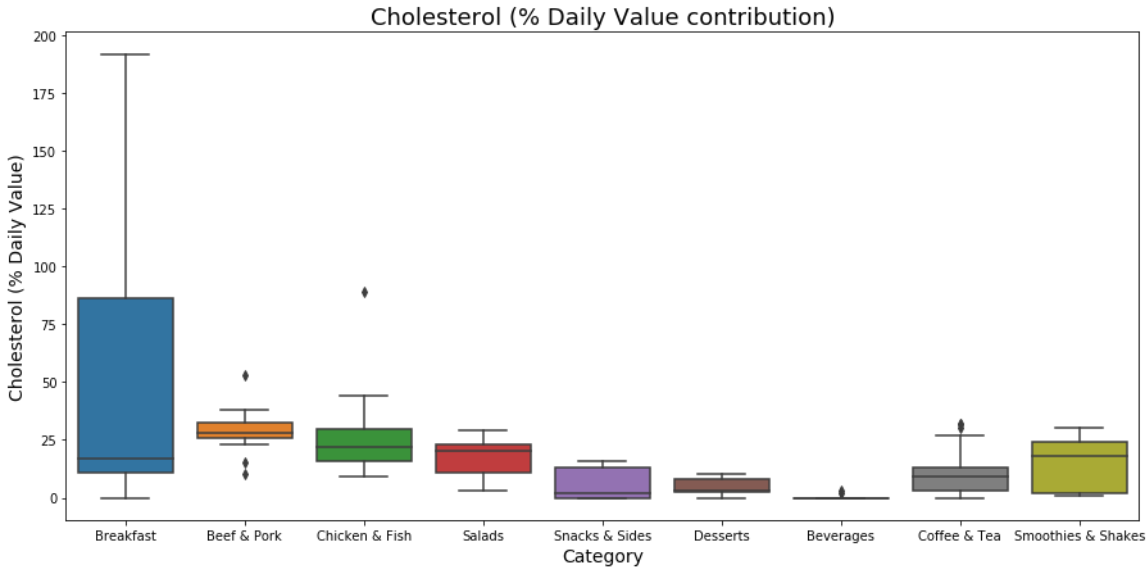
In [22]:

```
plt.figure(figsize = (7,7))
plt.title('Cholesterol (% Daily Value)', fontsize = 14)
plt.pie(df_new['Cholesterol (% Daily Value)'].tolist(), labels = df_new.Category.tolist
(), radius = 1, autopct = '%1.1f%%',pctdistance=.9);
plt.pie([1],colors=['w'],radius=.5)
plt.show()
```



In [23]:

```
plt.figure(figsize = (15,7))
plt.title('Cholesterol (% Daily Value contribution)', fontsize = 18)
plt.xlabel('Category', fontsize = 14)
plt.ylabel('Cholesterol (% Daily Value contribution)', fontsize = 14)
sns.boxplot(df['Category'],df['Cholesterol (% Daily Value)']);
```



From the above charts it can be clearly concluded that breakfast contributes the most to cholesterol

5.Which item contributes maximum to the Sodium intake?

In [24]:

```
df_n = df.groupby('Item').sum()
df_n.reset_index(inplace = True)
df_sodium_max = df_n.sort_values(by = 'Sodium', ascending = False)[['Item','Sodium']].h
ead()
```

In [25]:

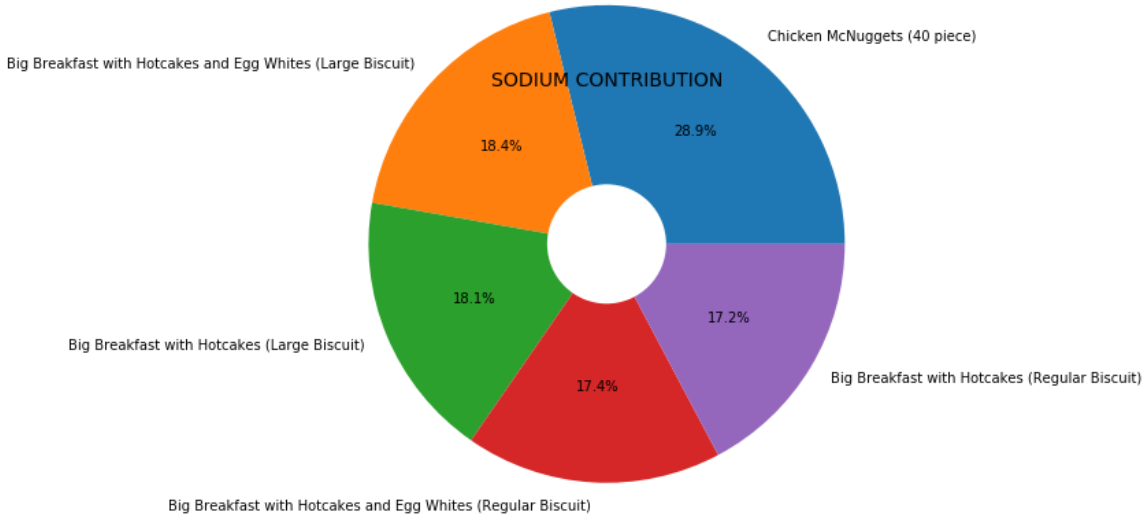
```
df_sodium_max
```

Out[25]:

	Item	Sodium
43	Chicken McNuggets (40 piece)	3600
23	Big Breakfast with Hotcakes and Egg Whites (La...	2290
21	Big Breakfast with Hotcakes (Large Biscuit)	2260
24	Big Breakfast with Hotcakes and Egg Whites (Re...	2170
22	Big Breakfast with Hotcakes (Regular Biscuit)	2150

In [26]:

```
plt.title('SODIUM CONTRIBUTION', fontsize = 14)
plt.pie(list(df_sodium_max['Sodium']), labels = list(df_sodium_max['Item']), radius = 2
, autopct='%1.1f%%');
plt.pie([1],colors=['w'],radius=.5);
```



It can be concluded from the piechart that Chicken McNuggets(40 piece) contributes the most

6.Which 4 food items contain the most amount of Saturated Fat?

In [27]:

```
df_sat_fat = df_n.sort_values(by = 'Saturated Fat', ascending = False)[['Item','Saturat
ed Fat']].head()
```

Items contributing to maximum Saturated Fat

In [28]:

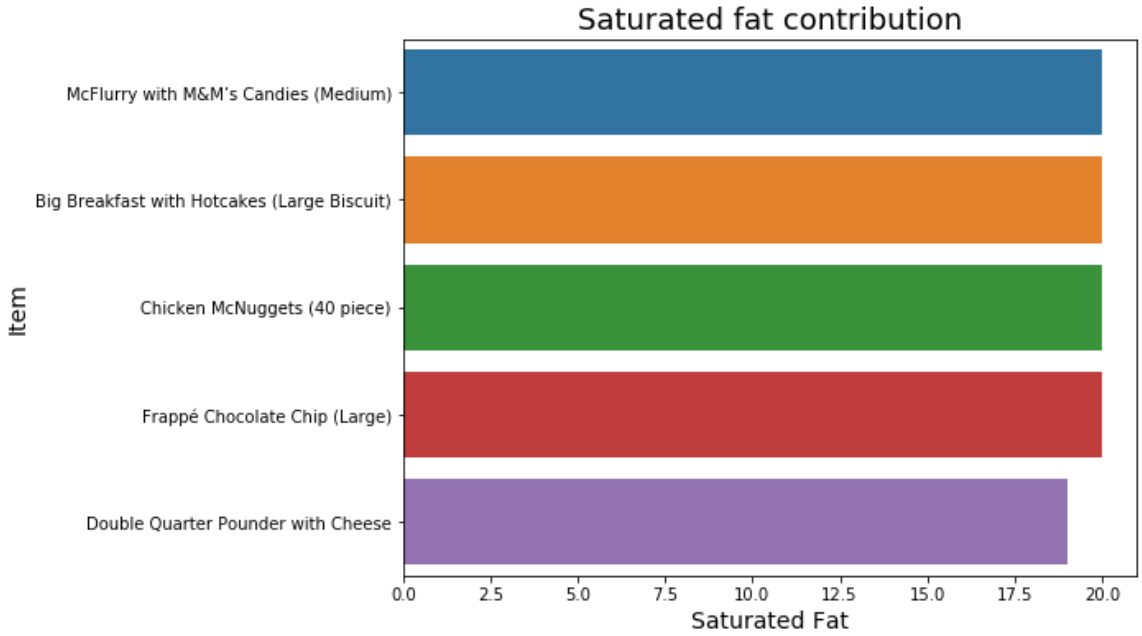
```
df_sat_fat
```

Out[28]:

	Item	Saturated Fat
151	McFlurry with M&M's Candies (Medium)	20.0
21	Big Breakfast with Hotcakes (Large Biscuit)	20.0
43	Chicken McNuggets (40 piece)	20.0
82	Frappé Chocolate Chip (Large)	20.0
70	Double Quarter Pounder with Cheese	19.0

In [43]:

```
plt.figure(figsize = (8,6))
plt.title('Saturated fat contribution',fontsize = 18)
plt.xlabel('Item', fontsize=14)
plt.ylabel('Saturated Fat', fontsize=14)
sns.barplot(df_sat_fat['Saturated Fat'], df_sat_fat['Item']);
```



In [327]:

```
print('items contributing max to saturated fat: \n')
df_sat_fat
```

items contributing max to saturated fat:

Out[327]:

	Item	Saturated Fat
151	McFlurry with M&M's Candies (Medium)	20.0
21	Big Breakfast with Hotcakes (Large Biscuit)	20.0
43	Chicken McNuggets (40 piece)	20.0
82	Frappé Chocolate Chip (Large)	20.0
70	Double Quarter Pounder with Cheese	19.0