Qn 1(a)

In [466]:
```
import pandas as pd
```

In [467]:
```
fao_df = pd.read_csv("FAO.csv", encoding="unicode_escape")
```

In [468]:
```
fao_df.head() # see the dataset
```

Out[468]:

	Area	Item	Element	Y1961	Y1962	Y1963	Y1964	Y1965	Y1966	Y1967	...	Y
0	Afghanistan	Wheat and products	Food	1928.0	1904.0	1666.0	1950.0	2001.0	1808.0	2053.0	...	3
1	Afghanistan	Rice (Milled Equivalent)	Food	183.0	183.0	182.0	220.0	220.0	195.0	231.0	...	
2	Afghanistan	Barley and products	Feed	76.0	76.0	76.0	76.0	76.0	75.0	71.0	...	
3	Afghanistan	Barley and products	Food	237.0	237.0	237.0	238.0	238.0	237.0	225.0	...	
4	Afghanistan	Maize and products	Feed	210.0	210.0	214.0	216.0	216.0	216.0	235.0	...	

5 rows × 56 columns

In [469]:
```
# getting the names of areas, without duplicates
areas = list(fao_df["Area"].unique())
```

In [470]:
```
# getting the names of all columns
columns = list(fao_df.columns)
# do a list slicing to get only the years and nothing else
years = columns[3:]
```

In [471]:
```
# create an empty list (food_data_by_countries) to append all the dictionaries
food_data_by_countries = []
for area in areas:
    location = {}
    for year in years:
        filt = (fao_df["Area"] == area)
        total_production = fao_df.loc[filt][year].sum()
        location[year] = [total_production]
    food_data_by_countries.append(location)

# create an empty dataframe to append all the other dataframes
totalProductionByEachCountryEachYear = pd.DataFrame()

# start appending the dataframes!
for country_dict in food_data_by_countries:
    individual_df = pd.DataFrame(country_dict)
    totalProductionByEachCountryEachYear = totalProductionByEachCountryEachYea
r.append(individual_df)

# naming the rows by countries
totalProductionByEachCountryEachYear.index = areas
print("Qn 1(a):")
print("\n")
totalProductionByEachCountryEachYear




# countries = list(fao_df["Area"].unique())
# country_details = []
# country_grp = fao_df.groupby("Area")
# for country in countries:
#     country_food = country_grp.get_group(country).sum()
#     country_details.append(country_food)

# new_df = pd.DataFrame(country_details)
# new_df.drop(columns=["Area", "Item", "Element"], inplace=True)
# new_df.index = countries
# new_df
```

Qn 1(a):

Out[471]:

	Y1961	Y1962	Y1963	Y1964	Y1965	Y1966	Y1967	Y1968	Y1969	Y
Afghanistan	9481.0	9414.0	9194.0	10170.0	10473.0	10169.0	11289.0	11508.0	11815.0	104
Albania	1706.0	1749.0	1767.0	1889.0	1884.0	1995.0	2046.0	2169.0	2230.0	23
Algeria	7488.0	7235.0	6861.0	7255.0	7509.0	7536.0	7986.0	8839.0	9003.0	93
Angola	4834.0	4775.0	5240.0	5286.0	5527.0	5677.0	5833.0	5685.0	6219.0	64
Antigua and Barbuda	92.0	94.0	105.0	95.0	84.0	73.0	64.0	59.0	68.0	
...	...	...	...	...	...	...	...	...	...	...
Venezuela (Bolivarian Republic of)	9523.0	9369.0	9788.0	10539.0	10641.0	10772.0	11126.0	12014.0	12537.0	135
Viet Nam	23856.0	25220.0	26053.0	26377.0	26961.0	27355.0	27745.0	28698.0	29565.0	308
Yemen	2982.0	3038.0	3147.0	3224.0	3328.0	3358.0	3420.0	3411.0	3386.0	33
Zambia	2976.0	3057.0	3069.0	3121.0	3236.0	3523.0	3688.0	3791.0	3904.0	40
Zimbabwe	3260.0	3503.0	3479.0	3738.0	3940.0	3991.0	4202.0	4443.0	4486.0	47

174 rows × 53 columns

In [472]:
```
# saving the newly created dataframe as a new csv file
totalProductionByEachCountryEachYear.to_csv("total_production_by_country_each_year.csv")
```

Q1(b)

In [473]:
```
# getting a list of the total productions by country
overall_production = []
for area in areas:
    total_production = totalProductionByEachCountryEachYear.loc[area].sum()
    overall_production.append([total_production])
```

In [474]:
```
# creating a dictionary to transform it into a dataframe
overall_production_dict = dict(zip(areas, overall_production))
```

In [475]:
```
# converting the dictionay into a dataframe
overall_production_df = pd.DataFrame(overall_production_dict)
overall_production_df = overall_production_df.transpose()

# give the columns a nice name
overall_production_df = overall_production_df.rename(columns={0: "OPC"})
overall_production_df.index.name = "Countries"
print("Qn1(b):")
print("\n")
overall_production_df


# total = []
# for country in countries:
#     total_by_country = new_df.loc[country].sum()
#     total.append(total_by_country)

# total_dict = dict(zip(countries, total))
# total_df = pd.DataFrame([total_dict])
# total_df = total_df.transpose()
# total_df
```

```
Qn1(b):


```

Out[475]:

Countries	OPC
Afghanistan	689162.0
Albania	237202.0
Algeria	1530613.0
Angola	706016.0
Antigua and Barbuda	4446.0
...	...
Venezuela (Bolivarian Republic of)	1340783.0
Viet Nam	3068282.0
Yemen	459785.0
Zambia	315687.0
Zimbabwe	351093.0

174 rows × 1 columns

Q1(c)

In [476]:
```
# APPYPC = Overall production by a specific country / total number of years
total_number_of_years = len(years)
```

In [477]:
```
average_production_by_year = []
for area in areas:
    average = (overall_production_df.loc[area]["OPC"]/total_number_of_years)
    average_production_by_year.append([average])
```

In [478]:
```
# creating a dictionary to transform it into a dataframe
APPYPC_dict = dict(zip(areas, average_production_by_year))
```

In [479]:
```
# converting the dictionay into a dataframe
APPYPC_df = pd.DataFrame(APPYPC_dict)
APPYPC_df = APPYPC_df.transpose()

# give the columns a nice name
APPYPC_df = APPYPC_df.rename(columns={0: "APPYPC"})
APPYPC_df.index.name = "Countries"
print("Qn1(c):")
print("\n")
APPYPC_df
```

```
Qn1(c):
```

Out[479]:

Countries	APPYPC
Afghanistan	13003.056604
Albania	4475.509434
Algeria	28879.490566
Angola	13321.056604
Antigua and Barbuda	83.886792
...	...
Venezuela (Bolivarian Republic of)	25297.792453
Viet Nam	57892.113208
Yemen	8675.188679
Zambia	5956.358491
Zimbabwe	6624.396226

174 rows × 1 columns

1(d)

In [480]:
```
# GAPPC = Overall production by a specific country / Overall production by all the countries
overall_production_by_all_countries = overall_production_df["OPC"].sum()
```

In [481]:
```
# getting a list of GAPPC by country
global_average_production = []
for area in areas:
    global_average = (overall_production_df.loc[area]["OPC"] / overall_production_by_all_countries)
    global_average_production.append([global_average])
```

In [482]:
```
# creating a dictionary to transform it into a dataframe
GAPPC_dict = dict(zip(areas, global_average_production))
```

In [483]:
```
# converting the dictionay into a dataframe
GAPPC_df = pd.DataFrame(GAPPC_dict)
GAPPC_df = GAPPC_df.transpose()

# give the columns a nice name
GAPPC_df = GAPPC_df.rename(columns={0: "GAPPC"})
GAPPC_df.index.name = "Countries"
print("Qn1(d):")
print("\n")
GAPPC_df
```

```
Qn1(d):
```

Out[483]:

Countries	GAPPC
Afghanistan	0.001820
Albania	0.000626
Algeria	0.004041
Angola	0.001864
Antigua and Barbuda	0.000012
...	...
Venezuela (Bolivarian Republic of)	0.003540
Viet Nam	0.008102
Yemen	0.001214
Zambia	0.000834
Zimbabwe	0.000927

174 rows × 1 columns

For Part (b) to (d), store your result as a DF and display only the top 3 and bottom 3 results from this new DF

In [484]:
```
countries_food_data = overall_production_df
countries_food_data["APPYPC"] = APPYPC_df["APPYPC"]
countries_food_data["GAPPC"] = GAPPC_df["GAPPC"]
```

In [485]:
```
# getting the top 3 and bottom 3 data
top3Data = countries_food_data.head(3)
bottom3Data = countries_food_data.tail(3)
topThreeAndBottomThreeData = pd.concat([top3Data, bottom3Data])
print("Qn 1(b)-(d): The top 3 and bottom 3 results are ")
print("\n")
topThreeAndBottomThreeData
```

```
Qn 1(b)-(d): The top 3 and bottom 3 results are
```

Out[485]:

Countries	OPC	APPYPC	GAPPC
Afghanistan	689162.0	13003.056604	0.001820
Albania	237202.0	4475.509434	0.000626
Algeria	1530613.0	28879.490566	0.004041
Yemen	459785.0	8675.188679	0.001214
Zambia	315687.0	5956.358491	0.000834
Zimbabwe	351093.0	6624.396226	0.000927

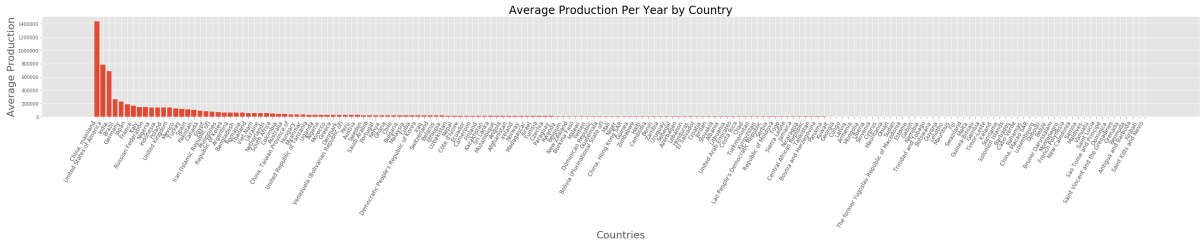
Q2

In [486]:
```
from matplotlib import pyplot as plt
import numpy as np
```

In [487]:
```
fig =  plt.figure(figsize=(45, 4))
ax = fig.add_subplot(111)
APPYPC_df = APPYPC_df.sort_values("APPYPC", ascending=False)
ax.bar(APPYPC_df.index, APPYPC_df["APPYPC"])
ax.set_xticklabels(APPYPC_df.index, rotation=60, horizontalalignment="right",
fontsize="12")

ax.set_title("Average Production Per Year by Country", fontsize=24)
ax.set_ylabel("Average Production", fontsize=22)
ax.set_xlabel("Countries", fontsize=22)

print("Qn2(a):")
print("\n")
plt.show

# From the barchart, we can observe that the countries with the highest and lo
west productions are
# China, mainland and Saint Kitts and Nevis respectively.
```

Qn2(a):

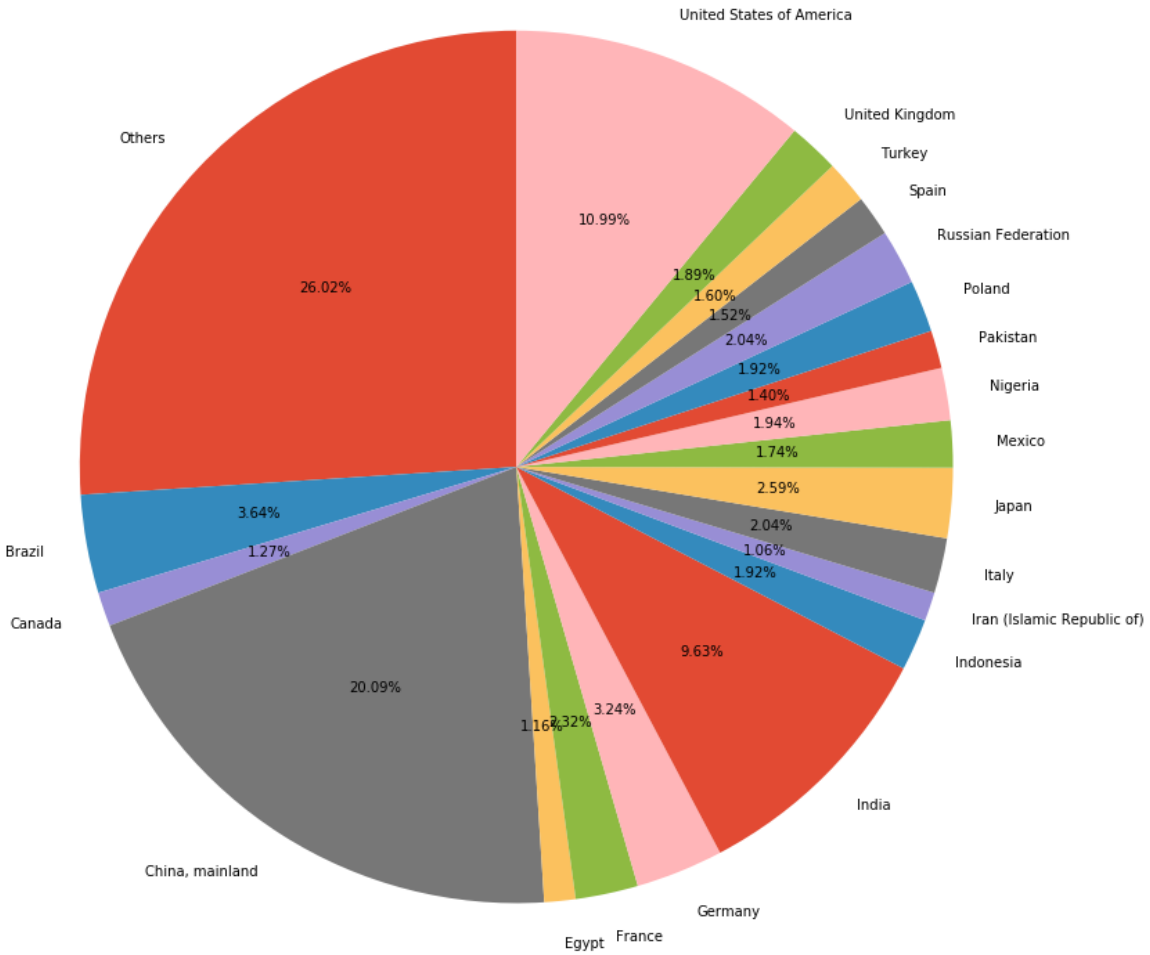
Out[487]: <function matplotlib.pyplot.show(*args, **kw)>



Q3

In [488]:
```
# For this particular problem, I categorize countries whose GAPPC values are l
ess than 1% into "Others"

import itertools
plt.style.use('ggplot')

t = GAPPC_df.sort_values(by=['GAPPC'],ascending=False)
temp = GAPPC_df.to_dict('dict')

newdic={}
for key, group in itertools.groupby(temp['GAPPC'], lambda k: 'Others' if (temp
['GAPPC'][k]<.01) else k):
     newdic[key] = sum([temp['GAPPC'][k] for k in list(group)])

labels = newdic.keys()
sizes = newdic.values()

tot = 0.0
for i in sizes:
    tot += i
newdic['Others'] = 1.02 - tot

fig, ax = plt.subplots(figsize=(12, 10))
ax.pie(sizes, labels=labels, autopct='%1.2f%%', explode=(0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0), startangle=90)
ax.axis('equal')
plt.tight_layout()

print("Qn3:")
print("\n")
plt.show()
```

Qn3:



Q4

In [489]:
```
# set the filter to get countries whose avergae production is more than 5%
filt = (countries_food_data["GAPPC"] > 0.05)
```

In [490]:
```
GAPPC_column = countries_food_data["GAPPC"]
```

In [491]:
```
# get the countries whose average production is more than 5%
countries_averageProduction_more_than_fivePercent = list(GAPPC_column.loc[filt
].index)
```

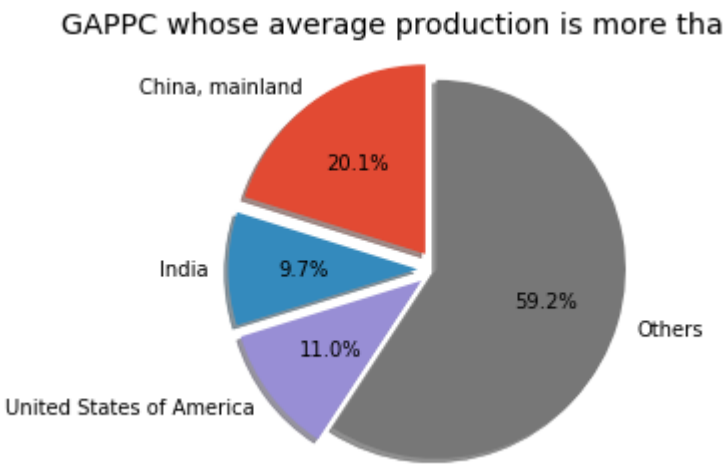
In [492]:
```
# get the respective percentages of these countries
percentages_of_countries_averageProduction_more_than_fivePercent = list(GAPPC_
column.loc[filt])
```

In [493]:
```
# find the percentage of "others"
total_percent_of_countries_more_than_5percent = 0
for i in range(len(percentages_of_countries_averageProduction_more_than_fivePercent)):
    total_percent_of_countries_more_than_5percent += percentages_of_countries_averageProduction_more_than_fivePercent[i]
percent_of_others = 1 - total_percent_of_countries_more_than_5percent
```

In [494]:
```
# create a list of all percentages with all countries
percentages_of_countries_averageProduction_more_than_fivePercent.append(percent_of_others)
```

In [495]:
```
# finally, plotting the pie chart...

labels = "China, mainland", "India", "United States of America", "Others"
sizes = list(map(lambda n: n*100, percentages_of_countries_averageProduction_more_than_fivePercent))
explode = (0.1, 0.1, 0.1, 0)

fig1, ax1 = plt.subplots()
ax1.pie(sizes, explode=explode, labels=labels, autopct="%1.1f%%", shadow=True, startangle=90)
ax1.axis("equal")
plt.title("GAPPC whose average production is more than 5%")
print("Qn4:")
print("\n")
plt.show()
```

Qn4:



Q5(a)

In [496]:
```
# setting the filter to get ONLY honey from fao_df
filt = (fao_df["Item"] == "Honey")

# getting honey_only_df
honey_only_df = fao_df.loc[filt]

# construct a total column which computes the total honey production from 2010
to 2013
honey_only_df["Total"] = honey_only_df["Y2010"] + honey_only_df["Y2011"] + hon
ey_only_df["Y2012"] + honey_only_df["Y2013"]

# if the honey production for a particular country is zero, drop it!
honey_df = honey_only_df.drop(honey_only_df[(honey_only_df["Total"] == 0)].ind
ex)

# construct a new dataframe call honey_df which consists of the area, Y2010 -
 Y2013
honey_df = honey_df[["Area", "Y2010", "Y2011", "Y2012", "Y2013"]]
```

```
C:\Users\S9632298D\Anaconda3\lib\site-packages\ipykernel_launcher.py:8: Setti
ngWithCopyWarning:
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: http://pandas.pydata.org/pandas-docs/st
able/user_guide/indexing.html#returning-a-view-versus-a-copy
```

In [497]:
```
# getting the top three items from honey_df
topThreeItems = honey_df.head(3)
```

In [498]:
```
# getting the bottom three items from honey_df
bottomThreeItems = honey_df.tail(3)
```

In [499]:
```
# merge the two dataframes above into one and display it
topThreeAndBottomThreeDf = topThreeItems.append(bottomThreeItems)
print("Qn5(a):")
print("\n")
topThreeAndBottomThreeDf
```

```
Qn5(a):
```

Out[499]:

	Area	Y2010	Y2011	Y2012	Y2013
13	Afghanistan	3.0	2.0	2	2
106	Albania	3.0	3.0	3	3
229	Algeria	6.0	5.0	6	7
20904	Venezuela (Bolivarian Republic of)	1.0	1.0	1	1
21025	Viet Nam	0.0	0.0	1	0
21138	Yemen	3.0	3.0	3	3

5(b)

In [500]:
```
# total_honey_production_between_2010and2013 =
honey_df["Total"] = honey_df["Y2010"] + honey_df["Y2011"] + honey_df["Y2012"]
+ honey_df["Y2013"]
```

In [501]:
```
# construct a new_df which consists of only the "area" and "total" columns
total_honey_production_between_2010and2013_df = honey_df[["Area", "Total"]]
```

In [502]:
```
# getting and printing the overall honey production (global production)
overall_honey_production_globally = total_honey_production_between_2010and2013
_df["Total"].sum()
print("Qn5(b), part 1:")
print(f'The overall honey production globally is {overall_honey_production_glo
bally}')
# overall_honey_production_globally
```

```
Qn5(b), part 1:
The overall honey production globally is 6399.0
```

In [503]:
```
# appending the total_honey_production_between_2010and2013_df with the honey d
ataframe and saving it as a csv file
# the appended version of the dataframe is already done above, hence, we only
 left with converting in into a csv file
honey_df.to_csv("honey_production_from_year_2010_to_2013.csv", index=False)
```

In [504]:
```
# printing the top three and bottom three daatsetsfrom the honey_df
topThreeHoneyData = honey_df.head(3)
bottomThreeHoneyData = honey_df.tail(3)
topThreeAndBottomThreeDataFromHoneyDF = topThreeHoneyData.append(bottomThreeHo
neyData)
print("Qn5(b):")
print("\n")
topThreeAndBottomThreeDataFromHoneyDF
```

```
Qn5(b):
```

Out[504]:

	Area	Y2010	Y2011	Y2012	Y2013	Total
13	Afghanistan	3.0	2.0	2	2	9.0
106	Albania	3.0	3.0	3	3	12.0
229	Algeria	6.0	5.0	6	7	24.0
20904	Venezuela (Bolivarian Republic of)	1.0	1.0	1	1	4.0
21025	Viet Nam	0.0	0.0	1	0	1.0
21138	Yemen	3.0	3.0	3	3	12.0

5(c)

In [561]:
```
# get the honey production of all countries from 2010 to 2013
honey_from_all_countries_from_2010_to_2013 = honey_only_df[["Area", "Y2010",
"Y2011", "Y2012", "Y2013"]]
```

Out[561]:

	Area	Y2010	Y2011	Y2012	Y2013
13	Afghanistan	3.0	2.0	2	2
106	Albania	3.0	3.0	3	3
229	Algeria	6.0	5.0	6	7
348	Angola	23.0	23.0	23	23
457	Antigua and Barbuda	0.0	0.0	0	0
...	...	...	...	...	...
20904	Venezuela (Bolivarian Republic of)	1.0	1.0	1	1
21025	Viet Nam	0.0	0.0	1	0
21138	Yemen	3.0	3.0	3	3
21255	Zambia	0.0	0.0	0	0
21377	Zimbabwe	0.0	0.0	0	0

169 rows × 5 columns

In [562]:
```
# getting the average honey production for each country
honey_from_all_countries_from_2010_to_2013["average"] = (honey_from_all_countries_from_2010_to_2013["Y2010"] + honey_from_all_countries_from_2010_to_2013["Y2011"] + honey_from_all_countries_from_2010_to_2013["Y2012"] + honey_from_all_countries_from_2010_to_2013["Y2013"]) / 4
```

```
C:\Users\S9632298D\Anaconda3\lib\site-packages\ipykernel_launcher.py: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: http://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
```

Out[562]:

	Area	Y2010	Y2011	Y2012	Y2013	average
13	Afghanistan	3.0	2.0	2	2	2.25
106	Albania	3.0	3.0	3	3	3.00
229	Algeria	6.0	5.0	6	7	6.00
348	Angola	23.0	23.0	23	23	23.00
457	Antigua and Barbuda	0.0	0.0	0	0	0.00
...	...	...	...	...	...	...
20904	Venezuela (Bolivarian Republic of)	1.0	1.0	1	1	1.00
21025	Viet Nam	0.0	0.0	1	0	0.25
21138	Yemen	3.0	3.0	3	3	3.00
21255	Zambia	0.0	0.0	0	0	0.00
21377	Zimbabwe	0.0	0.0	0	0	0.00

169 rows × 6 columns

In [563]:
```
total_average = honey_from_all_countries_from_2010_to_2013["average"].sum()
```

Out[563]: 1599.75

In [564]:
```
five_percent_of_total_average = 0.05 * total_average
```

Out[564]: 79.98750000000001

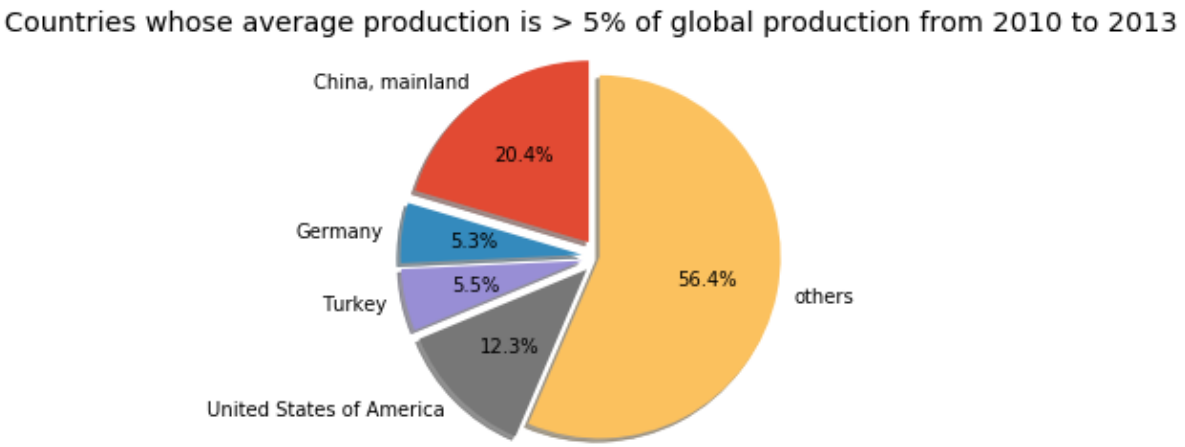
In [568]:
```
filt = (honey_from_all_countries_from_2010_to_2013["average"] > five_percent_of_total_average)
more_than_five_percent = honey_from_all_countries_from_2010_to_2013.loc[filt]
```

Out[568]:

	Area	Y2010	Y2011	Y2012	Y2013	average
4151	China, mainland	302.0	333.0	341	330	326.50
7560	Germany	92.0	84.0	79	86	85.25
19544	Turkey	80.0	93.0	88	91	88.00
20420	United States of America	183.0	191.0	200	214	197.00

In [583]:
```
# plotting the pie chart
labels = 'China, mainland', 'Germany', 'Turkey', 'United States of America',
"others"
sizes = list(more_than_five_percent["average"])
sizes = list(map(lambda x: (x/total_average)*100, sizes))

# get the percenatge for "others"
total = 0
for i in range(len(sizes)):
    total += sizes[i]
left = 100 - total

sizes.append(left)

explode = (0.1, 0.1, 0.1, 0.1, 0)

fig1, ax1 = plt.subplots()
ax1.pie(sizes, explode=explode, labels=labels, autopct='%1.1f%%',
        shadow=True, startangle=90)
ax1.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
plt.title("Countries whose average production is > 5% of global production fro
m 2010 to 2013")
print("Qn5(c):")
print("\n")
plt.show()
```

```
Qn5(c):
```



Q6(a)

In [445]:
```
import re
# getting all the Malaysia datasets from fao_df
filt = fao_df["Area"] == "Malaysia"
malaysia_df = fao_df.loc[filt]
```

In [446]:
```
# filter out the sugar items for Food element
filt_food = (fao_df["Element"] == "Food") & (fao_df["Item"].str.contains("Sugar", flags=re.IGNORECASE))
malaysia_sugar_Food = malaysia_df.loc[filt_food]

# getting only for years between 2010 and 2013
malaysia_sugar_Food_between_2010And2013 = malaysia_sugar_Food[["Area", "Element", "Y2010", "Y2011", "Y2012", "Y2013"]]
malaysia_sugar_Food_between_2010And2013

# getting the total sugar production per year for malaysia (Food)
malaysia_sugar_Food_2010 = malaysia_sugar_Food_between_2010And2013["Y2010"].sum()
malaysia_sugar_Food_2011 = malaysia_sugar_Food_between_2010And2013["Y2011"].sum()
malaysia_sugar_Food_2012 = malaysia_sugar_Food_between_2010And2013["Y2012"].sum()
malaysia_sugar_Food_2013 = malaysia_sugar_Food_between_2010And2013["Y2013"].sum()
```

In [447]:
```
# filter out the sugar items for Feed element
filt_feed = (fao_df["Element"] == "Feed") & (fao_df["Item"].str.contains("Sugar", flags=re.IGNORECASE))
malaysia_sugar_Feed = malaysia_df.loc[filt_feed]

# getting only for years between 2010 and 2013
malaysia_sugar_Feed_between_2010And2013 = malaysia_sugar_Feed[["Area", "Element", "Y2010", "Y2011", "Y2012", "Y2013"]]
malaysia_sugar_Feed_between_2010And2013

# getting the total sugar production per year for malaysia (Feed)
malaysia_sugar_Feed_2010 = malaysia_sugar_Feed_between_2010And2013["Y2010"].sum()
malaysia_sugar_Feed_2011 = malaysia_sugar_Feed_between_2010And2013["Y2011"].sum()
malaysia_sugar_Feed_2012 = malaysia_sugar_Feed_between_2010And2013["Y2012"].sum()
malaysia_sugar_Feed_2013 = malaysia_sugar_Feed_between_2010And2013["Y2013"].sum()
```

In [448]:
```
# creating a dictionary to convert it to dataframe
malaysia_sugar = {"Country": ["Malaysia", "Malaysia"], "Food Type": ["Food", "Feed"], "2010": [malaysia_sugar_Food_2010, malaysia_sugar_Feed_2010], "2011": [malaysia_sugar_Food_2011, malaysia_sugar_Feed_2011], "2012": [malaysia_sugar_Food_2012, malaysia_sugar_Feed_2012], "2013": [malaysia_sugar_Food_2013, malaysia_sugar_Feed_2013]}
```

In [449]:
```
# getting the dataframe for malaysia sugar production
malaysia_sugar_df = pd.DataFrame(malaysia_sugar)
print("Q6(a), Malayisa sugar production:")
print("\n")
malaysia_sugar_df
```

```
Q6(a), Malayisa sugar production:
```

Out[449]:

	Country	Food Type	2010	2011	2012	2013
0	Malaysia	Food	2362.0	2411.0	2541	2587
1	Malaysia	Feed	62.0	48.0	32	28

In [450]:
```
# getting all the France datasets from fao_df
filt = fao_df["Area"] == "France"
france_df = fao_df.loc[filt]
```

In [451]:
```
# filter out the sugar items for Food element
filt_food = (fao_df["Element"] == "Food") & (fao_df["Item"].str.contains("Suga
r", flags=re.IGNORECASE))
france_sugar_Food = france_df.loc[filt_food]

# getting only for years between 2010 and 2013
france_sugar_Food_between_2010And2013 = france_sugar_Food[["Area", "Element",
"Y2010", "Y2011", "Y2012", "Y2013"]]
france_sugar_Food_between_2010And2013

# getting the total sugar production per year for malaysia (Food)
france_sugar_Food_2010 = france_sugar_Food_between_2010And2013["Y2010"].sum()
france_sugar_Food_2011 = france_sugar_Food_between_2010And2013["Y2011"].sum()
france_sugar_Food_2012 = france_sugar_Food_between_2010And2013["Y2012"].sum()
france_sugar_Food_2013 = france_sugar_Food_between_2010And2013["Y2013"].sum()
```

In [452]:
```
# filter out the sugar items for Feed element
filt_feed = (fao_df["Element"] == "Feed") & (fao_df["Item"].str.contains("Suga
r", flags=re.IGNORECASE))
france_sugar_Feed = france_df.loc[filt_feed]

# getting only for years between 2010 and 2013
france_sugar_Feed_between_2010And2013 = france_sugar_Feed[["Area", "Element",
"Y2010", "Y2011", "Y2012", "Y2013"]]
france_sugar_Feed_between_2010And2013

# getting the total sugar production per year for malaysia (Feed)
france_sugar_Feed_2010 = france_sugar_Feed_between_2010And2013["Y2010"].sum()
france_sugar_Feed_2011 = france_sugar_Feed_between_2010And2013["Y2011"].sum()
france_sugar_Feed_2012 = france_sugar_Feed_between_2010And2013["Y2012"].sum()
france_sugar_Feed_2013 = france_sugar_Feed_between_2010And2013["Y2013"].sum()
```

In [453]:
```
# creating a dictionary to convert it to dataframe
france_sugar = {"Country": ["France", "France"], "Food Type": ["Food", "Feed"
], "2010": [france_sugar_Food_2010, france_sugar_Feed_2010], "2011": [france_s
ugar_Food_2011, france_sugar_Feed_2011], "2012": [france_sugar_Food_2012, fran
ce_sugar_Feed_2012], "2013": [france_sugar_Food_2013, france_sugar_Feed_2013]}
```

In [454]:
```
# getting the dataframe for france sugar production
france_sugar_df = pd.DataFrame(france_sugar)
print("Qn6(a), France sugar production:")
print("\n")
france_sugar_df
```

```
Qn6(a), France sugar production:
```

Out[454]:

	Country	Food Type	2010	2011	2012	2013
0	France	Food	4583.0	4601.0	4668	4882
1	France	Feed	80.0	77.0	77	78

6(b)

In [455]:
```
# mergeing the two dataframes into one and display it
sugar_production_by_MalaysiaAndFrance_between_2010And2013 = malaysia_sugar_df.
append(france_sugar_df)
print("Qn6b:")
print("\n")
sugar_production_by_MalaysiaAndFrance_between_2010And2013.reset_index(drop=Tru
e)
```

```
Qn6b:
```

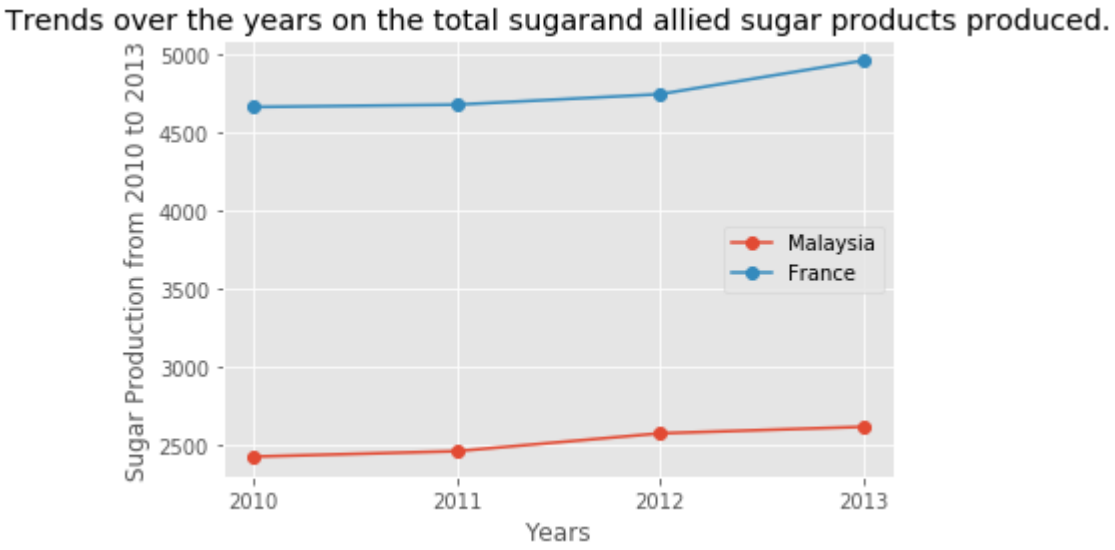
Out[455]:

	Country	Food Type	2010	2011	2012	2013
0	Malaysia	Food	2362.0	2411.0	2541	2587
1	Malaysia	Feed	62.0	48.0	32	28
2	France	Food	4583.0	4601.0	4668	4882
3	France	Feed	80.0	77.0	77	78

6(c)

In [456]:
```
# constant x-axis for both plots - represents years from 2010 to 2013
x_axis = ["2010", "2011", '2012',"2013"]

# total sugar production (sugar + allied sugar products) in the respective yea
rs for Malaysia
malaysia_total_sugar_yaxis = [2424, 2459, 2573, 2615]
# plotting the line plot out
plt.plot(x_axis, malaysia_total_sugar_yaxis, label="Malaysia", marker="o")

# total sugar production (sugar + allied sugar products) in the respective yea
rs for France
france_total_sugar_yaxis = [4663, 4678, 4745, 4960]
# plotting the line plot out
plt.plot(x_axis, france_total_sugar_yaxis, label="France", marker="o")

# labeeling of the axes, and giving the plot a title
plt.xlabel("Years", fontsize=12)
plt.ylabel("Sugar Production from 2010 t0 2013", fontsize=12)
plt.title("Trends over the years on the total sugarand allied sugar products p
roduced.")
plt.legend()
plt.grid(True)
print("Qn6b:")
print("\n")
plt.show()
```

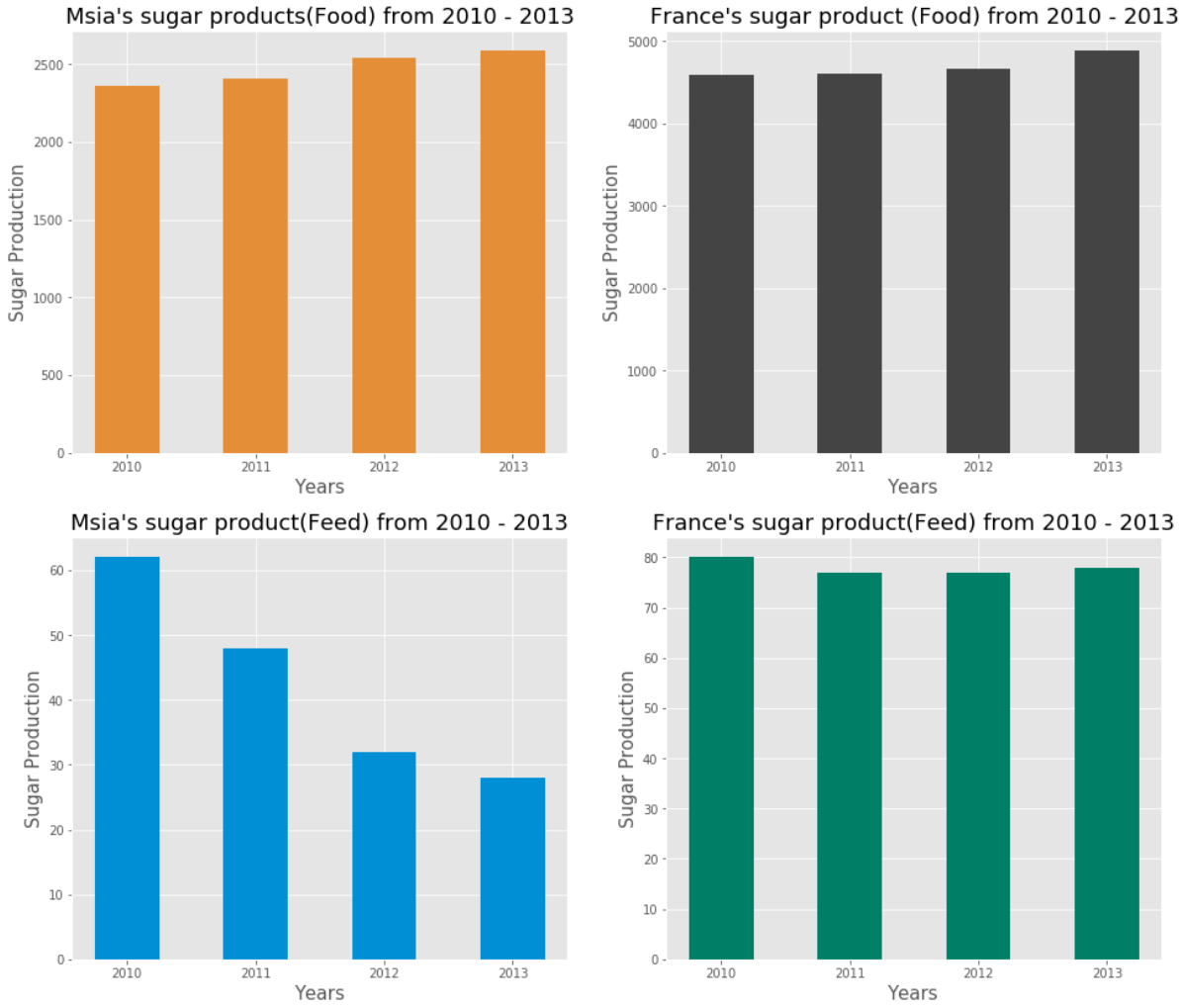
Qn6b:



6(d)

In [457]:
```
import numpy as np
fig = plt.figure(figsize=(16, 14))

pltMalaysiaFood = fig.add_subplot(221)
pltMalaysiaFeed = fig.add_subplot(223)
pltFranceFood = fig.add_subplot(222)
pltFranceFeed = fig.add_subplot(224)

x = ["2010", "2011", "2012", "2013"]
y1 = [2362, 2411, 2541, 2587] # malaysia food
y2 = [62, 48, 32, 28] # malaysia feed
y3 = [4583, 4601, 4668, 4882] # france food
y4 = [80, 77, 77, 78] # france feed

# plotting for Malaysia (food)
pltMalaysiaFood.bar(x, y1, color="#e58e38", width=0.5)
pltMalaysiaFood.set_xlabel("Years", fontsize="15")
pltMalaysiaFood.set_ylabel("Sugar Production", fontsize="15")
pltMalaysiaFood.set_title("Msia's sugar products(Food) from 2010 - 2013", font
size="18")

# plotting for ,Malaysia (feed)
pltMalaysiaFeed.bar(x, y2, color="#008fd5", width=0.5)
pltMalaysiaFeed.set_xlabel("Years", fontsize="15")
pltMalaysiaFeed.set_ylabel("Sugar Production", fontsize="15")
pltMalaysiaFeed.set_title("Msia's sugar product(Feed) from 2010 - 2013", fonts
ize="18")

# plotting for France (food)
pltFranceFood.bar(x, y3, color="#444444", width=0.5)
pltFranceFood.set_xlabel("Years", fontsize="15")
pltFranceFood.set_ylabel("Sugar Production", fontsize="15")
pltFranceFood.set_title("France's sugar product (Food) from 2010 - 2013", font
size="18")

# plotting for France (feed)
pltFranceFeed.bar(x, y4, color="#007f67", width=0.5)
pltFranceFeed.set_xlabel("Years", fontsize="15")
pltFranceFeed.set_ylabel("Sugar Production", fontsize="15")
pltFranceFeed.set_title("France's sugar product(Feed) from 2010 - 2013", fonts
ize="18")

print("Qn6(d):")
print("\n")
plt.show()
```

Qn6(d):



6(f)

In [458]:
```
average_between_two_countries_in_2010 = ((2541 + 32) + (4668 + 77)) / 2
difference_between_two_countries_in_2010 = abs((2541 + 32) - (4668 + 77))
percentage_difference_between_the_two_countries_in_2012 = (difference_between_
two_countries_in_2010 / average_between_two_countries_in_2010) * 100

print(f'Qn6(f): The percentage difference between Malaysia and France in 2010
 is {percentage_difference_between_the_two_countries_in_2012}%')
```

```
Qn6(f): The percentage difference between Malaysia and France in 2010 is 59.3
60481005739274%
```

In [ ]: