In [1]:
```
# import modułów
import pandas as pd
import numpy as np
import warnings
warnings.filterwarnings('ignore')
import matplotlib.pyplot as plt
```

In [2]:
```
# import danych pobranych z Banku Danych Lokalnych GUS
part1 = pd.read_excel('F:\DataScience\Data\Inflacja\data1.xlsx', sheet_name = 'DANE', usecols = ['Rodzaje produktów','Rok','Wartosc'])
part2 = pd.read_excel('F:\DataScience\Data\Inflacja\data2.xlsx', sheet_name = 'DANE', usecols = ['Rodzaje produktów','Rok','Wartosc'])
part3 = pd.read_excel('F:\DataScience\Data\Inflacja\data3.xlsx', sheet_name = 'DANE', usecols = ['Rodzaje produktów','Rok','Wartosc'])
part4 = pd.read_excel('F:\DataScience\Data\Inflacja\data4.xlsx', sheet_name = 'DANE', usecols = ['Rodzaje produktów','Rok','Wartosc'])
part5 = pd.read_excel('F:\DataScience\Data\Inflacja\data5.xlsx', sheet_name = 'DANE', usecols = ['Rodzaje produktów','Rok','Wartosc'])
part6 = pd.read_excel('F:\DataScience\Data\Inflacja\data6.xlsx', sheet_name = 'DANE', usecols = ['Rodzaje produktów','Rok','Wartosc'])
part7 = pd.read_excel('F:\DataScience\Data\Inflacja\data7.xlsx', sheet_name = 'DANE', usecols = ['Rodzaje produktów','Rok','Wartosc'])
part8 = pd.read_excel('F:\DataScience\Data\Inflacja\data8.xlsx', sheet_name = 'DANE', usecols = ['Rodzaje produktów','Rok','Wartosc'])
part9 = pd.read_excel('F:\DataScience\Data\Inflacja\data9.xlsx', sheet_name = 'DANE', usecols = ['Rodzaje produktów','Rok','Wartosc'])
part10 = pd.read_excel('F:\DataScience\Data\Inflacja\data10.xlsx', sheet_name = 'DANE', usecols = ['Rodzaje produktów','Rok','Wartosc'])
part11 = pd.read_excel('F:\DataScience\Data\Inflacja\data11.xlsx', sheet_name = 'DANE', usecols = ['Rodzaje produktów','Rok','Wartosc'])
part12 = pd.read_excel('F:\DataScience\Data\Inflacja\data12.xlsx', sheet_name = 'DANE', usecols = ['Rodzaje produktów','Rok','Wartosc'])
part13 = pd.read_excel('F:\DataScience\Data\Inflacja\data13.xlsx', sheet_name = 'DANE', usecols = ['Rodzaje produktów','Rok','Wartosc'])
part14 = pd.read_excel('F:\DataScience\Data\Inflacja\data14.xlsx', sheet_name = 'DANE', usecols = ['Rodzaje produktów','Rok','Wartosc'])
```

In [3]:
```
#  połączenie danych, ustawienie indeksów oraz unstack kolumny "Rok" z wierszy do kolumn dla lepszej prezentacji danych
part1.set_index(['Rodzaje produktów','Rok'], inplace = True)
data = part1.unstack(level = 'Rok')

parts = [part2, part3, part4, part5, part6, part7, part8, part9, part10, part11, part12, part13, part14]
for part in parts:
    part.set_index(['Rodzaje produktów','Rok'], inplace = True)
    part = part.unstack(level = 'Rok')
    data = data.append(part)
data.head()
```

Out[3]:

	Wartosc										
Rok	**2010**	**2011**	**2012**	**2013**	**2014**	**2015**	**2016**	**2017**	**2018**	**2019**	**2020**
Rodzaje produktów											
chleb mieszany zwykły (pszenno-żytni) - za 1kg	3.24	3.58	3.75	3.89	3.92	3.98	3.96	4.01	4.13	4.41	4.68
kasza gryczana prażona cała - za 1kg	4.52	6.68	4.53	4.85	5.0	5.04	5.36	5.0	4.33	4.09	4.69
kasza jęczmienna "Mazurska" - za 1kg	1.32	1.76	1.88	2.06	2.13	2.1	2.26	2.08	1.83	1.89	1.92
mąka pszenna "Poznańska", workowana - za 1kg	1.01	1.36	1.32	1.3	1.1	1.07	0.97	1.04	1.11	1.18	1.17
cegła budowlana pełna palona kl.15 - za 1szt	1.23	1.21	1.2	1.18	1.17	1.15	1.14	1.14	1.19	1.29	1.37

In [4]:
```
# Usunięcie multiindeksu "Wartosc"
data = data.transpose()
data.reset_index(inplace = True)
data = data.drop(columns =['level_0'])
data.set_index('Rok',inplace = True)
data = data.transpose()
data.head()
```

Out[4]:

Rok	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020
Rodzaje produktów											
chleb mieszany zwykły (pszenno-żytni) - za 1kg	3.24	3.58	3.75	3.89	3.92	3.98	3.96	4.01	4.13	4.41	4.68
kasza gryczana prażona cała - za 1kg	4.52	6.68	4.53	4.85	5.0	5.04	5.36	5.0	4.33	4.09	4.69
kasza jęczmienna "Mazurska" - za 1kg	1.32	1.76	1.88	2.06	2.13	2.1	2.26	2.08	1.83	1.89	1.92
mąka pszenna "Poznańska", workowana - za 1kg	1.01	1.36	1.32	1.3	1.1	1.07	0.97	1.04	1.11	1.18	1.17
cegła budowlana pełna palona kl.15 - za 1szt	1.23	1.21	1.2	1.18	1.17	1.15	1.14	1.14	1.19	1.29	1.37

In [5]:
```
# Zmiana nazwy kolumn
data.reset_index(inplace = True)
data.rename(columns = {'Rodzaje produktów':'ProduktLubUsluga'}, inplace = True)
data.set_index('ProduktLubUsluga', inplace = True)
data.head()
```

Out[5]:

Rok	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020
ProduktLubUsluga											
chleb mieszany zwykły (pszenno-żytni) - za 1kg	3.24	3.58	3.75	3.89	3.92	3.98	3.96	4.01	4.13	4.41	4.68
kasza gryczana prażona cała - za 1kg	4.52	6.68	4.53	4.85	5.0	5.04	5.36	5.0	4.33	4.09	4.69
kasza jęczmienna "Mazurska" - za 1kg	1.32	1.76	1.88	2.06	2.13	2.1	2.26	2.08	1.83	1.89	1.92
mąka pszenna "Poznańska", workowana - za 1kg	1.01	1.36	1.32	1.3	1.1	1.07	0.97	1.04	1.11	1.18	1.17
cegła budowlana pełna palona kl.15 - za 1szt	1.23	1.21	1.2	1.18	1.17	1.15	1.14	1.14	1.19	1.29	1.37

In [6]:
```
# Usunięcie duplikatów i sprawdzenie rozmiaru tablicy
data.reset_index(inplace = True)
data.drop_duplicates(subset = ['ProduktLubUsluga'], inplace = True)
data.set_index('ProduktLubUsluga', inplace = True)
data.shape
```

Out[6]: (193, 11)

In [7]:
```
# Usunięcie produktów i usług z brakującymi wartosciami w wierszach
data = data.replace('-', np.NaN)
data.dropna(how = 'any', inplace = True)
data.shape
```

Out[7]: (117, 11)

In [8]:
```
# Eksport danych do MS Excel w celu korekty nazw produktów i usług oraz usunięcie pozycji nietypowych
data.reset_index(inplace = True)
excelWriter = pd.ExcelWriter('F:\DataScience\Data\Inflacja\data_export.xlsx')
data.to_excel(excelWriter, index = False)
excelWriter.save()
```

In [9]:
```
# Ponowny import poprawionych danych wraz z kolumną zawierierającą szacowane miesięczne spożycie(koszyk inflacyjny)
data = pd.read_excel('F:\DataScience\Data\Inflacja\data_import.xlsx')
data.set_index('ProduktLubUsluga', inplace = True)
data.head(10)
```

Out[9]:

ProduktLubUsluga	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	miesięcznie	rocznie
baleron gotowany - za 1kg	17.58	17.89	19.28	19.80	20.19	19.55	19.61	21.08	21.88	23.72	26.40	0.5	6.0
bateria zlewozmywakowa	134.07	138.05	144.79	151.25	151.01	150.52	152.73	155.21	159.54	164.25	165.55	0.0	1.0
benzyna silnikowa bezołowiowa 95 - za 1L	4.59	5.14	5.73	5.51	5.30	4.65	4.37	4.63	4.95	5.02	4.48	60.0	720.0
bilet do kina	15.07	15.62	16.36	17.06	17.65	18.23	18.73	19.48	19.80	20.02	19.34	2.0	24.0
bilet do teatru	34.53	37.10	40.21	41.61	43.63	45.14	46.62	48.21	49.41	51.01	53.97	1.0	12.0
bilet normalny na przejazd autobusem miejskim	2.23	2.34	2.58	2.72	2.73	2.71	2.71	2.69	2.74	2.77	2.87	20.0	240.0
bilet normalny na przejazd tramwajem	2.53	2.63	2.95	3.28	3.30	3.32	3.32	3.33	3.36	3.40	3.56	20.0	240.0
boczek surowy - za 1kg	12.06	12.54	14.40	14.93	14.51	13.69	14.30	16.47	16.30	17.44	18.99	0.5	6.0
boczek wędzony - za 1kg	16.58	17.13	19.25	20.09	20.15	19.62	19.91	22.28	23.31	24.79	27.91	0.5	6.0
botki męskie skórzane na podeszwie nieskórzanej - za 1parę	188.46	195.95	211.60	223.58	227.91	227.47	225.02	218.93	223.76	228.65	232.23	0.0	2.0

In [10]:
```
# wygenerowanie kolumn z kosztami danych produktów w danym roku

for number in range(0,11):
    data['kwota_'+str(2010+number)] = data[2010+number]*data['rocznie']
data.head(10)
```

Out[10]:

ProduktLubUsluga	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	...	kwota_2011	kwota_2012	kwota_2013	kwota_2014	kwota_2015	kwota_2016	kwota_2017	kwota_2018	kwota_2019	kwota
baleron gotowany - za 1kg	17.58	17.89	19.28	19.80	20.19	19.55	19.61	21.08	21.88	23.72	...	107.34	115.68	118.80	121.14	117.30	117.66	126.48	131.28	142.32	
bateria zlewozmywakowa	134.07	138.05	144.79	151.25	151.01	150.52	152.73	155.21	159.54	164.25	...	138.05	144.79	151.25	151.01	150.52	152.73	155.21	159.54	164.25	
benzyna silnikowa bezołowiowa 95 - za 1L	4.59	5.14	5.73	5.51	5.30	4.65	4.37	4.63	4.95	5.02	...	3700.80	4125.60	3967.20	3816.00	3348.00	3146.40	3333.60	3564.00	3614.40	3
bilet do kina	15.07	15.62	16.36	17.06	17.65	18.23	18.73	19.48	19.80	20.02	...	374.88	392.64	409.44	423.60	437.52	449.52	467.52	475.20	480.48	
bilet do teatru	34.53	37.10	40.21	41.61	43.63	45.14	46.62	48.21	49.41	51.01	...	445.20	482.52	499.32	523.56	541.68	559.44	578.52	592.92	612.12	
bilet normalny na przejazd autobusem miejskim	2.23	2.34	2.58	2.72	2.73	2.71	2.71	2.69	2.74	2.77	...	561.60	619.20	652.80	655.20	650.40	650.40	645.60	657.60	664.80	
bilet normalny na przejazd tramwajem	2.53	2.63	2.95	3.28	3.30	3.32	3.32	3.33	3.36	3.40	...	631.20	708.00	787.20	792.00	796.80	796.80	799.20	806.40	816.00	
boczek surowy - za 1kg	12.06	12.54	14.40	14.93	14.51	13.69	14.30	16.47	16.30	17.44	...	75.24	86.40	89.58	87.06	82.14	85.80	98.82	97.80	104.64	
boczek wędzony - za 1kg	16.58	17.13	19.25	20.09	20.15	19.62	19.91	22.28	23.31	24.79	...	102.78	115.50	120.54	120.90	117.72	119.46	133.68	139.86	148.74	
botki męskie skórzane na podeszwie nieskórzanej - za 1parę	188.46	195.95	211.60	223.58	227.91	227.47	225.02	218.93	223.76	228.65	...	391.90	423.20	447.16	455.82	454.94	450.04	437.86	447.52	457.30	

10 rows × 24 columns

In [11]:
```
# Usunięcie wierszy z produktami/usługami nieobcnymi w koszyku inflacyjnym
data = data.where(data['kwota_2020'] != 0).dropna()
data.head(10)
```

Out[11]:

ProduktLubUsluga	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	...	kwota_2011	kwota_2012	kwota_2013	kwota_2014	kwota_2015	kwota_2016	kwota_2017	kwota_2018	kwota_2019	kwota
baleron gotowany - za 1kg	17.58	17.89	19.28	19.80	20.19	19.55	19.61	21.08	21.88	23.72	...	107.34	115.68	118.80	121.14	117.30	117.66	126.48	131.28	142.32	
bateria zlewozmywakowa	134.07	138.05	144.79	151.25	151.01	150.52	152.73	155.21	159.54	164.25	...	138.05	144.79	151.25	151.01	150.52	152.73	155.21	159.54	164.25	
benzyna silnikowa bezołowiowa 95 - za 1L	4.59	5.14	5.73	5.51	5.30	4.65	4.37	4.63	4.95	5.02	...	3700.80	4125.60	3967.20	3816.00	3348.00	3146.40	3333.60	3564.00	3614.40	3
bilet do kina	15.07	15.62	16.36	17.06	17.65	18.23	18.73	19.48	19.80	20.02	...	374.88	392.64	409.44	423.60	437.52	449.52	467.52	475.20	480.48	
bilet do teatru	34.53	37.10	40.21	41.61	43.63	45.14	46.62	48.21	49.41	51.01	...	445.20	482.52	499.32	523.56	541.68	559.44	578.52	592.92	612.12	
bilet normalny na przejazd autobusem miejskim	2.23	2.34	2.58	2.72	2.73	2.71	2.71	2.69	2.74	2.77	...	561.60	619.20	652.80	655.20	650.40	650.40	645.60	657.60	664.80	
bilet normalny na przejazd tramwajem	2.53	2.63	2.95	3.28	3.30	3.32	3.32	3.33	3.36	3.40	...	631.20	708.00	787.20	792.00	796.80	796.80	799.20	806.40	816.00	
boczek surowy - za 1kg	12.06	12.54	14.40	14.93	14.51	13.69	14.30	16.47	16.30	17.44	...	75.24	86.40	89.58	87.06	82.14	85.80	98.82	97.80	104.64	
boczek wędzony - za 1kg	16.58	17.13	19.25	20.09	20.15	19.62	19.91	22.28	23.31	24.79	...	102.78	115.50	120.54	120.90	117.72	119.46	133.68	139.86	148.74	
botki męskie skórzane na podeszwie nieskórzanej - za 1parę	188.46	195.95	211.60	223.58	227.91	227.47	225.02	218.93	223.76	228.65	...	391.90	423.20	447.16	455.82	454.94	450.04	437.86	447.52	457.30	

10 rows × 24 columns

In [12]:
```
# Wycięcie niezbędnych kolumn do dalszych obliczeń i transpoza

lista = []
for number in range(0,11):
    position = 'kwota_'+ str(2010+number)
    lista.append(position)
lista
koszyk = data[lista]
koszyk = koszyk.transpose()
koszyk
```

Out[12]:

ProduktLubUsluga	baleron gotowany - za 1kg	bateria zlewozmywakowa	benzyna silnikowa bezołowiowa 95 - za 1L	bilet do kina	bilet do teatru	bilet normalny na przejazd autobusem miejskim	bilet normalny na przejazd tramwajem	boczek surowy - za 1kg	boczek wędzony - za 1kg	botki męskie skórzane na podeszwie nieskórzanej - za 1parę	...	sól warzona biała workowana - za 1kg	strzyżenie włosów męskich	szynka wieprzowa gotowana - za 1kg	śledź solony, niepatroszony - za 1kg	śmietana o zawartości tłuszczu 18% - za 1l	truskawki mrożone - za 0,5kg	wino białe gronowe, wytrawne - za 0,75l
kwota_2010	105.48	134.07	3304.8	361.68	414.36	535.2	607.2	72.36	99.48	376.92	...	0.744	174.84	262.44	19.32	32.82	91.44	217.44
kwota_2011	107.34	138.05	3700.8	374.88	445.20	561.6	631.2	75.24	102.78	391.90	...	0.912	184.08	266.16	22.68	33.36	97.32	218.88
kwota_2012	115.68	144.79	4125.6	392.64	482.52	619.2	708.0	86.40	115.50	423.20	...	0.960	192.12	281.16	27.60	33.54	99.60	221.28
kwota_2013	118.80	151.25	3967.2	409.44	499.32	652.8	787.2	89.58	120.54	447.16	...	0.984	196.20	289.08	29.01	34.20	100.80	227.04
kwota_2014	121.14	151.01	3816.0	423.60	523.56	655.2	792.0	87.06	120.90	455.82	...	0.960	201.96	292.68	29.19	34.50	99.60	230.40
kwota_2015	117.30	150.52	3348.0	437.52	541.68	650.4	796.8	82.14	117.72	454.94	...	0.960	205.92	291.00	30.03	33.42	96.00	232.80
kwota_2016	117.66	152.73	3146.4	449.52	559.44	650.4	796.8	85.80	119.46	450.04	...	0.936	210.60	294.24	31.74	33.12	93.96	234.48
kwota_2017	126.48	155.21	3333.6	467.52	578.52	645.6	799.2	98.82	133.68	437.86	...	0.960	220.08	306.72	33.66	36.54	93.84	575.76
kwota_2018	131.28	159.54	3564.0	475.20	592.92	657.6	806.4	97.80	139.86	447.52	...	0.984	232.08	320.04	32.58	39.84	102.00	591.84
kwota_2019	142.32	164.25	3614.4	480.48	612.12	664.8	816.0	104.64	148.74	457.30	...	1.176	252.00	341.28	31.17	41.16	111.72	602.64
kwota_2020	158.40	165.55	3225.6	464.16	647.64	688.8	854.4	113.94	167.46	464.46	...	1.224	302.64	380.88	32.49	42.24	116.64	588.00

11 rows × 69 columns

In [13]:
```
# Dodanie kolumny z sumą za każdy rok

lista = []
for number in range(0,11):
    suma = koszyk.loc['kwota_' + str(2010+number)].sum()
    lista.append(round(suma,2))

suma_koszyk = pd.Series(lista)
koszyk.reset_index(inplace = True)
koszyk['suma_koszyk'] = suma_koszyk
koszyk.rename(columns = {'index':'Rok'}, inplace = True)
koszyk.set_index('Rok', inplace = True)
koszyk
```

Out[13]:

ProduktLubUsluga	baleron gotowany - za 1kg	bateria zlewozmywakowa	benzyna silnikowa bezołowiowa 95 - za 1L	bilet do kina	bilet do teatru	bilet normalny na przejazd autobusem miejskim	bilet normalny na przejazd tramwajem	boczek surowy - za 1kg	boczek wędzony - za 1kg	botki męskie skórzane na podeszwie nieskórzanej - za 1parę	...	strzyżenie włosów męskich	szynka wieprzowa gotowana - za 1kg	śledź solony, niepatroszony - za 1kg	śmietana o zawartości tłuszczu 18% - za 1l	truskawki mrożone - za 0,5kg	wino białe gronowe, wytrawne - za 0,75l	wizyta u lekarza specjalisty
Rok																		
kwota_2010	105.48	134.07	3304.8	361.68	414.36	535.2	607.2	72.36	99.48	376.92	...	174.84	262.44	19.32	32.82	91.44	217.44	817.44
kwota_2011	107.34	138.05	3700.8	374.88	445.20	561.6	631.2	75.24	102.78	391.90	...	184.08	266.16	22.68	33.36	97.32	218.88	873.72
kwota_2012	115.68	144.79	4125.6	392.64	482.52	619.2	708.0	86.40	115.50	423.20	...	192.12	281.16	27.60	33.54	99.60	221.28	938.28
kwota_2013	118.80	151.25	3967.2	409.44	499.32	652.8	787.2	89.58	120.54	447.16	...	196.20	289.08	29.01	34.20	100.80	227.04	986.88
kwota_2014	121.14	151.01	3816.0	423.60	523.56	655.2	792.0	87.06	120.90	455.82	...	201.96	292.68	29.19	34.50	99.60	230.40	1038.12
kwota_2015	117.30	150.52	3348.0	437.52	541.68	650.4	796.8	82.14	117.72	454.94	...	205.92	291.00	30.03	33.42	96.00	232.80	1070.28
kwota_2016	117.66	152.73	3146.4	449.52	559.44	650.4	796.8	85.80	119.46	450.04	...	210.60	294.24	31.74	33.12	93.96	234.48	1113.24
kwota_2017	126.48	155.21	3333.6	467.52	578.52	645.6	799.2	98.82	133.68	437.86	...	220.08	306.72	33.66	36.54	93.84	575.76	1179.36
kwota_2018	131.28	159.54	3564.0	475.20	592.92	657.6	806.4	97.80	139.86	447.52	...	232.08	320.04	32.58	39.84	102.00	591.84	1247.64
kwota_2019	142.32	164.25	3614.4	480.48	612.12	664.8	816.0	104.64	148.74	457.30	...	252.00	341.28	31.17	41.16	111.72	602.64	1349.40
kwota_2020	158.40	165.55	3225.6	464.16	647.64	688.8	854.4	113.94	167.46	464.46	...	302.64	380.88	32.49	42.24	116.64	588.00	1516.44

11 rows × 70 columns

In [14]:
```
# wyodrębnienie kolumny z sumą i obliczenie rok-rocznej inflacji oraz sumarycznej od roku 2010

koszyk.reset_index(inplace = True)
inflacja = koszyk[['Rok','suma_koszyk']]
inflacja.set_index('Rok', inplace = True)
lista = inflacja['suma_koszyk'].tolist()

#obliczenie iflacji rok-rocznej
infl_r_r = [0]
for number in range(0, len(lista)-1):
    value = round(((lista[number+1]/lista[number])*100)-100,2)
    infl_r_r.append(value)

# obliczenie inflacji sumarycznej
infl_sum = [0]
for number in range(0,len(lista)-1):
    value = round(((lista[number+1]/lista[0])*100)-100,2)
    infl_sum.append(value)

# utworzenie nowych kolumn
inflacja.reset_index(inplace = True)
inflacja['inflacja_r_r'] = infl_r_r
inflacja['inflacja_od_2010'] = infl_sum
inflacja.set_index('Rok', inplace = True)
```

In [15]:
```
# zmiana nazw w indeksie 'Rok': np.  kwota_2014 ---> 2014
inflacja.reset_index(inplace = True)
inflacja['Rok'] = inflacja['Rok'].str.replace('kwota_','')
inflacja.set_index('Rok', inplace = True)
inflacja
```

Out[15]:

ProduktLubUsluga	suma_koszyk	inflacja_r_r	inflacja_od_2010
Rok			
2010	22261.22	0.00	0.00
2011	23425.78	5.23	5.23
2012	24984.83	6.66	12.23
2013	25402.51	1.67	14.11
2014	25691.48	1.14	15.41
2015	25267.37	-1.65	13.50
2016	25101.45	-0.66	12.76
2017	25952.81	3.39	16.58
2018	26347.30	1.52	18.36
2019	27036.77	2.62	21.45
2020	27573.04	1.98	23.86

In [16]:
```
# Prezentacja danych na wykresie
plt.figure( figsize = (15,9))
plt.title("Inflacja w Polsce dla wybranego koszyka dóbr i usług")
plt.ylabel("Inflacja %")
plt.xlabel("Rok")
plt.plot(inflacja['inflacja_r_r'], marker = 'o', linestyle = ':', color = 'r', label = 'Inflacja r/r')
plt.plot(inflacja['inflacja_od_2010'],linestyle = 'dashed', color = 'g', marker = 'o', label = 'Inflacja sumaryczna')
plt.grid(linestyle = '--')
plt.legend(loc = 'upper left', fontsize = 15)
plt.xticks(rotation = 45)
plt.show()
```