```python
# Week 12 - Assessed exercises

# This week we learnt some advanced data manipulation methods, about APIs and
# about webscraping. In this last set of assessed exercises you must complete the
# Brightspace quiz 'W12 - Assessed exercises' and submit a .py file with the
# code you used to answer the questions in your quiz. Each question is work
# 0.5 marks and it is either correct (full marks) or incorrect (0 marks).

# This template file contains code that will help you answer the questions in
# the quiz.

# Q1 and Q2 are based on the advanced data manipulation section. You will need to
# use the titantic dataset which is part of the seaborn package and can be loaded
# using the following commands
import seaborn as sb
import re
from bs4 import BeautifulSoup
import requests
import matplotlib.pyplot as plt
import datetime
import wbdata as wbd
from pandas import Series, DataFrame
import pandas as pd
import numpy as np
titanic = sb.load_dataset('titanic')
titanic
# The questions involve using the groupby function and applying functions to that
# grouped object. For questins involving the interquartile range, use the function
# from lecture_12_code.py


# Q1
fare_group = titanic.fare.groupby([titanic.sex, titanic.survived])
q1 = fare_group.mean()
print('The average fare of male passengers who did not survive is: {}'.format(
    round(q1[2], 0)))
# Q2
survied_group = titanic.survived.groupby([titanic.sex, titanic.pclass])
q2 = survied_group.mean()
print('The percentage of female first class passengers (pclass=1) survived:
{:.2f}'.format(
    q2[0]))
# Q3 to Q5 relate to the World Bank API. You will be asked to search of indicator
# and country codes in Q3 and Q4. In Q5 you will need to extract data from the
# the World Bank for a particular indicator, country and year

# Q3
age_group = titanic.groupby(['class', 'sex', 'survived'])
quant_3 = age_group.age.quantile(q=0.75)[9]
quant_1 = age_group.age.quantile(q=0.25)[9]

print(' the interquartile range for the age of female third class passengers
(class=3) who survived (survived=1) is : {}'.format(
    round(quant_3-quant_1, 0)))

# Q4
# the indicator code for "Taxes on exports (% of tax revenue)" is  GC.TAX.EXPT.ZS.

# Q5
indicator1 = {
```

```
    'GC.TAX.YPKG.RV.ZS': 'the taxes on income, profits and capital gains, as a % of revenue, for the Egypt in 2007'}
data_date1 = (datetime.datetime(2007, 1, 1), datetime.datetime(2007, 12, 31))
data1 = wbd.get_dataframe(indicator1, 'EGY', data_date1)
data1
# The taxes on income, profits and capital gains, as a % of revenue, for the Egypt in 2007 is about 28%.

# Q6 to Q8 relate to webscraping and uses the Spotify weekly charts. You will need
# to import BeautifulSoup and the requests package
# The below code loads the data from the Spotify weekly charts for the week
# 2017-06-30 to 2017-07-07, and uses BeautifulSoup to parse the html.
spotify = requests.get(
    'https://spotifycharts.com/regional/global/weekly/2017-06-30--2017-07-07')
soup = BeautifulSoup(spotify.text, "html.parser")
# The following commands extract the information related to the tracks and removes
# the html tags
track = soup.find_all('td', class_="chart-table-track")
tracks = [x.text.strip() for x in track]
# Q6 asks you to search through tracks to find the number of times a particular
# arist appears in this weekly chart
count1 = 0
for i in tracks:
    if 'Justin Bieber' in i:
        count1 += 1
count1
# Justin Bieber appeared 5 times

# The following commands extract the information related to the number of plays,
# removes the html tags and commas, and converts the value to an integers
play = soup.find_all('td', class_="chart-table-streams")
plays = [int(x.text.strip().replace(',', '')) for x in play]

# Q7 asks you to perform some statistical analysis on these numbers
count2 = pd.Series(plays).mean()
print('the mean number of plays for the 200 songs in the 2017-06-30 to 2017-07-07 is {}'.format(round(count2,0)))


# Q8 asks you to load the charts for a different week and determine how many of
# the songs from the original week 2017-06-30 to 2017-07-07 are still in the
# charts at this later week. To load in the data for the new week, change the
# date range in the url to the date range specified in your question.
spotify2 = requests.get(
    'https://spotifycharts.com/regional/global/weekly/2017-08-25--2017-09-01')
soup2 = BeautifulSoup(spotify2.text, "html.parser")
# The following commands extract the information related to the tracks and removes
# the html tags
track2 = soup2.find_all('td', class_="chart-table-track")
tracks2 = [x.text.strip() for x in track2]

count3 = 0
for i in tracks:
    if i in tracks2:
        count3 += 1
count3
# 125 songs were still in the chart.

```