In this coursework you will be writing Python functions as well as some JavaScript code to perform various computations.

To create these functions you will use Spyder (or some other program editing software if you prefer) to edit the template files given. The template files contain the names of the functions you need to define. Do not change the name of the files or the names of functions in the file, otherwise it will cause difficulty for marking your code and you will lose marks. However, you can add more functions if needed. On top of correct program output, you are also marked for coding style. Make sure you have ticked the Enable code style linting option on Spyder for conformance to the PEP 8 convention. Your answer file should be submitted via Minerva in a single zip file.

(a) – Reading data from CSV file and displaying it in HTML. [7 Marks]

(b) – Process data from CSV and displaying structured information. [14 Marks]

(c) – Retrieve data from the web [9 Marks]

(d) – JavaScript. [10 Marks] (Not required or COMP0011 (IFY) students)

The total mark for this coursework is 40 marks for XJCO1011 students. Out of the total mark, 6 marks are allocated for coding style conformance to PEP 8. The weighting of this mark in relation your overall module grade will depend on the particular module you are taking. Full details are on Minerva.

Questions

(a) Reading data from CSV file and displaying it in HTML [7 Marks]

Write a program that read a CSV file and display the data in a table in an HTML file. CSV files are files in Comma Separated Value format. Please refer to the module reading material Writing HTML using Python for an example of creating a HTML file and reading data from a CSV file.

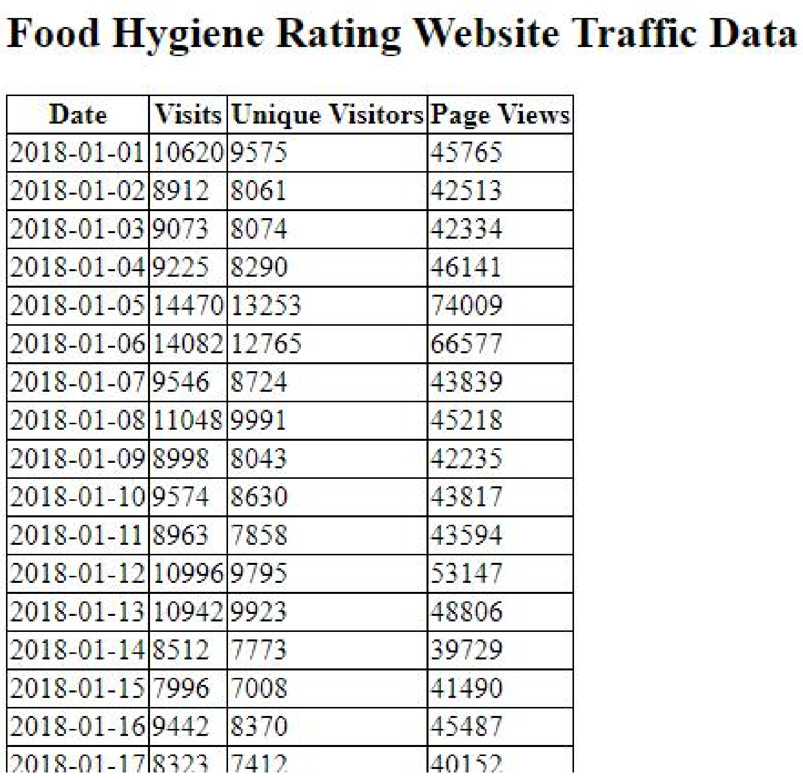
You are provided with the following files to help you with this task:

- Starting template file for your Python program:

  display_data_html.py [Download]
- A useful example program that creates an HTML file that displays data in the form of a table:

  create_html_table.py [Download]
- CSV files containing food hygiene website traffic data, on which to test your program (same as for Part A, but your program should work for any file in a similar format):
  - web-traffic-2018.csv
  - web-traffic-2017.csv

The output for a basic solution when the program is run should look something like this:



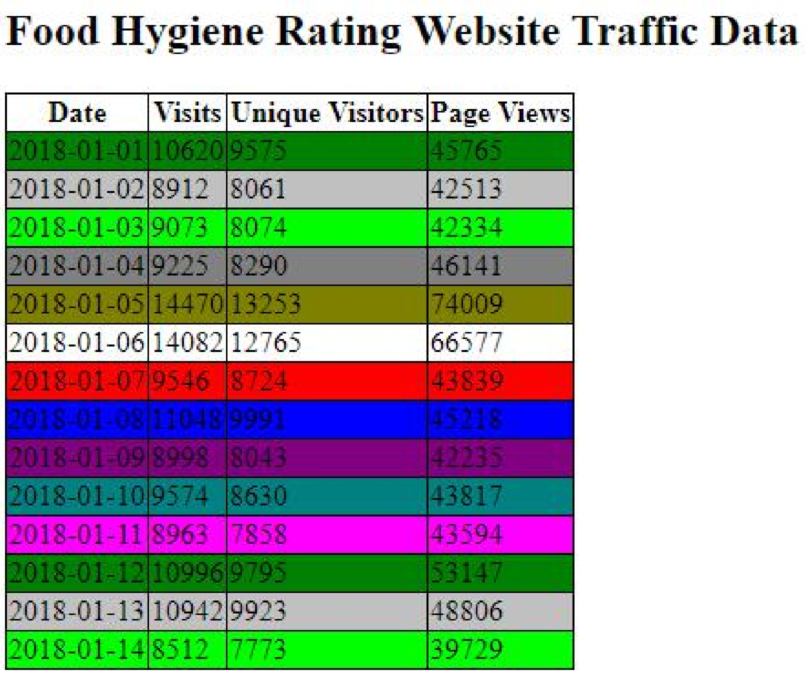
Food Hygiene Rating Website Traffic Data

Date	Visits	Unique Visitors	Page Views
2018-01-01	10620	9575	45765
2018-01-02	8912	8061	42513
2018-01-03	9073	8074	42334
2018-01-04	9225	8290	46141
2018-01-05	14470	13253	74009
2018-01-06	14082	12765	66577
2018-01-07	9546	8724	43839
2018-01-08	11048	9991	45218
2018-01-09	8998	8043	42235
2018-01-10	9574	8630	43817
2018-01-11	8963	7858	43594
2018-01-12	10996	9795	53147
2018-01-13	10942	9923	48806
2018-01-14	8512	7773	39729
2018-01-15	7996	7008	41490
2018-01-16	9442	8370	45487
2018-01-17	8323	7412	40152

You can pass this coursework with a basic solution but you can gain higher marks if you add further decorations, more sophisticated formatting to your output.

For example, you could get your output to look something like the figure below with alternating background colour for each row. In this example, the following list of colours is used.

HTML_COLORS = ['Green', 'Silver', 'Lime', 'Gray', 'Olive', 'White', 'Red', 'Blue', 'Purple', 'Teal', 'Fuchsia', 'Aqua']

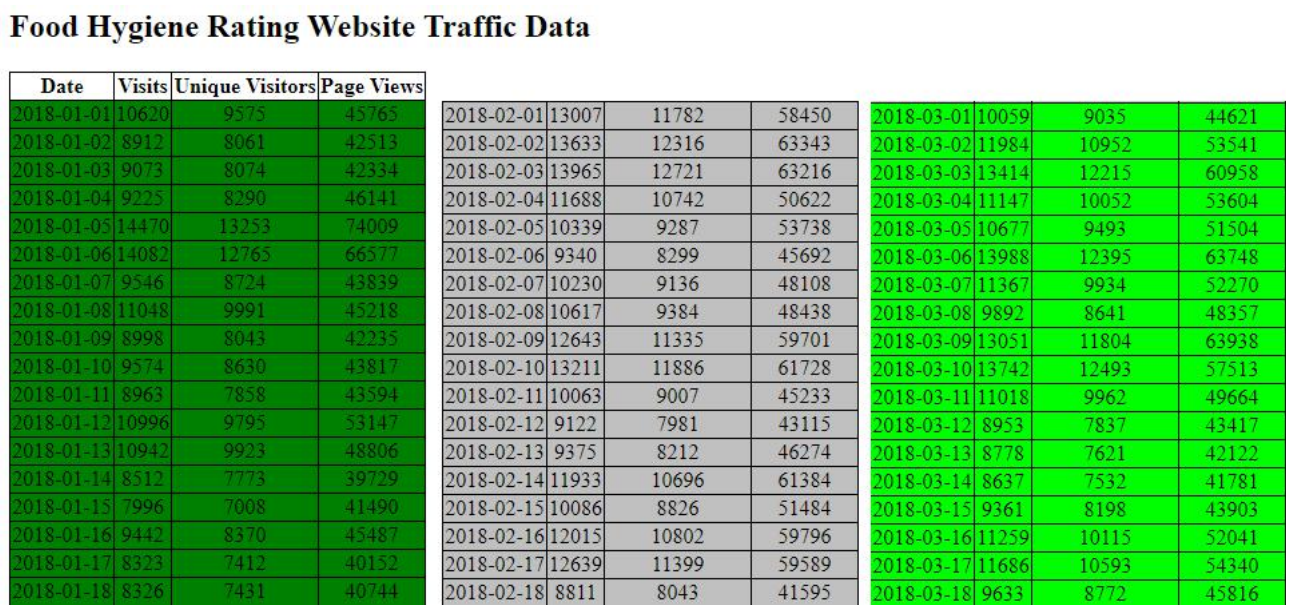


Food Hygiene Rating Website Traffic Data

Date	Visits	Unique Visitors	Page Views
2018-01-01	10620	9575	45765
2018-01-02	8912	8061	42513
2018-01-03	9073	8074	42334
2018-01-04	9225	8290	46141
2018-01-05	14470	13253	74009
2018-01-06	14082	12765	66577
2018-01-07	9546	8724	43839
2018-01-08	11048	9991	45218
2018-01-09	8998	8043	42235
2018-01-10	9574	8630	43817
2018-01-11	8963	7858	43594
2018-01-12	10996	9795	53147
2018-01-13	10942	9923	48806
2018-01-14	8512	7773	39729

However, you can also produce a more advanced version to look something like the figure below with different background colour for different month of the year and centre justified text. In this example, the following list of colours is used.

HTML_COLORS = ['Green', 'Silver', 'Lime', 'Gray', 'Olive', 'White', 'Red', 'Blue', 'Purple', 'Teal', 'Fuchsia', 'Aqua']



Food Hygiene Rating Website Traffic Data

Date	Visits	Unique Visitors	Page Views
2018-01-01	10620	9575	45765
2018-01-02	8912	8061	42513
2018-01-03	9073	8074	42334
2018-01-04	9225	8290	46141
2018-01-05	14470	13253	74009
2018-01-06	14082	12765	66577
2018-01-07	9546	8724	43839
2018-01-08	11048	9991	45218
2018-01-09	8998	8043	42235
2018-01-10	9574	8630	43817
2018-01-11	8963	7858	43594
2018-01-12	10996	9795	53147
2018-01-13	10942	9923	48806
2018-01-14	8512	7773	39729
2018-01-15	7996	7008	41490
2018-01-16	9442	8370	45487
2018-01-17	8323	7412	40152
2018-01-18	8326	7431	40744
2018-02-01	13007	11782	58450
2018-02-02	13633	12316	63343
2018-02-03	13965	12721	63216
2018-02-04	11688	10742	50622
2018-02-05	10339	9287	53738
2018-02-06	9340	8299	45692
2018-02-07	10230	9136	48108
2018-02-08	10617	9384	48438
2018-02-09	12643	11335	59701
2018-02-10	13211	11886	61728
2018-02-11	10063	9007	45233
2018-02-12	9122	7981	43115
2018-02-13	9375	8212	46274
2018-02-14	11933	10696	61384
2018-02-15	10086	8826	51484
2018-02-16	12015	10802	59796
2018-02-17	12639	11399	59589
2018-02-18	8811	8043	41595
2018-03-01	10059	9035	44621
2018-03-02	11984	10952	53541
2018-03-03	13414	12215	60958
2018-03-04	11147	10052	53604
2018-03-05	10677	9493	51504
2018-03-06	13988	12395	63748
2018-03-07	11367	9934	52270
2018-03-08	9892	8641	48357
2018-03-09	13051	11804	63938
2018-03-10	13742	12493	57513
2018-03-11	11018	9962	49664
2018-03-12	8953	7837	43417
2018-03-13	8778	7621	42122
2018-03-14	8637	7532	41781
2018-03-15	9361	8198	43903
2018-03-16	11259	10115	52041
2018-03-17	11686	10593	54340
2018-03-18	9633	8772	45816

(b) Reading data from CSV and display structured information [14 marks]

Similar to part(a) in this coursework, write a program to read the data from a CSV file and displaying data in a HTML table; but this time you will need to process the data before displaying the summarised information.

You are provided with the following files to help you with this task:

- Starting template file for your Python program:

  summarised_data_html.py [View] [Download]

- A useful example program that creates an HTML file that displays summarised data in the form of a table:

  process_summary_html.py [View] [Download]

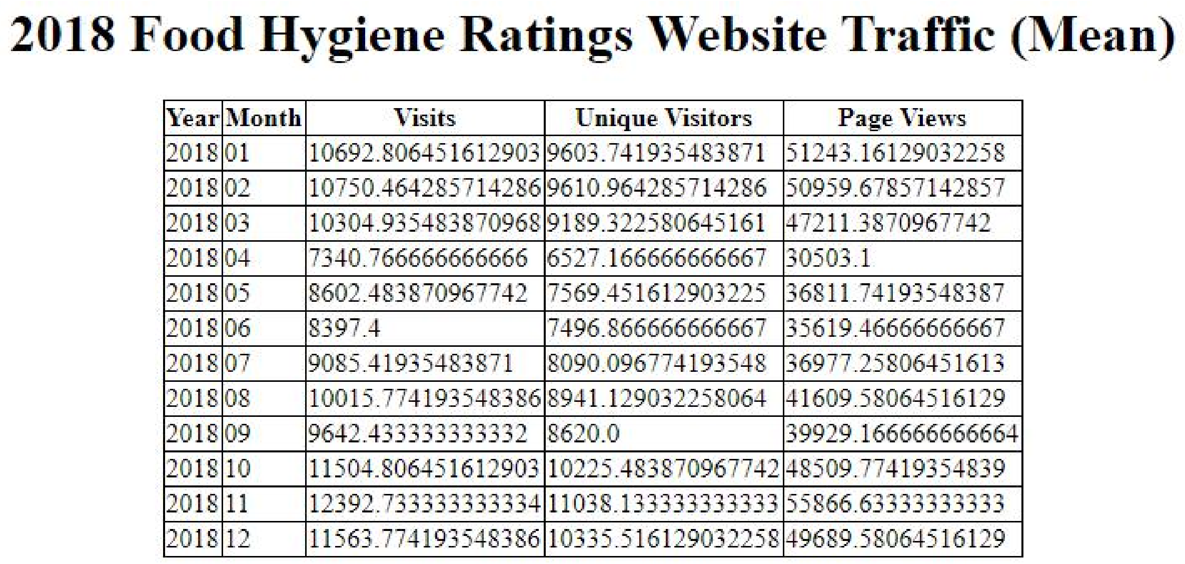
- CSV files containing food hygiene website traffic data, on which to test your program (same as for Part A, but your program should work for any file in a similar format):
  - web-traffic-2018.csv [View] [Download]
  - web-traffic-2017.csv [View] [Download]

You need to modify and extend summarised_data_html(title, csvfile, htmlfile) so that the function creates a HTML file displaying the average value for each month in the given CSV files. For example, the table below is the data for the month of January 2018 in the file web-traffic-2018.csv.

Date	Visits	UniqueVisitors	Pageviews
01/01/2018	10620	9575	45765
02/01/2018	8912	8061	42513
03/01/2018	9073	8074	42334
04/01/2018	9225	8290	46141
05/01/2018	14470	13253	74009
06/01/2018	14082	12765	66577
07/01/2018	9546	8724	43839
08/01/2018	11048	9991	45218
09/01/2018	8998	8043	42235
10/01/2018	9574	8630	43817
11/01/2018	8963	7858	43594
12/01/2018	10996	9795	53147
13/01/2018	10942	9923	48806
14/01/2018	8512	7773	39729
15/01/2018	7996	7008	41490
16/01/2018	9442	8370	45487
17/01/2018	8323	7412	40152
18/01/2018	8326	7431	40744
19/01/2018	10733	9530	51035
20/01/2018	11493	10359	52138
21/01/2018	10333	9446	49331
22/01/2018	10734	9640	52800
23/01/2018	9804	8813	49617
24/01/2018	9476	8363	49916
25/01/2018	9724	8651	48253
26/01/2018	12024	10816	57542
27/01/2018	12703	11467	60628
28/01/2018	11859	10671	54237
29/01/2018	18045	16283	96096
30/01/2018	14406	12880	70320
31/01/2018	11095	9821	51028

The sum value for the field Visits is 331477, UniqueVisitors is 297716, and Pageviews is 1588538. Therefore, the mean (average) value for the field Visits is (331477/31) = 10692.80645, UniqueVisitors is (297716/31) = 9603.741935, and Pageviews is (1588538/31) = 51243.16129.

The output for a basic solution when the program is run should look something like this:

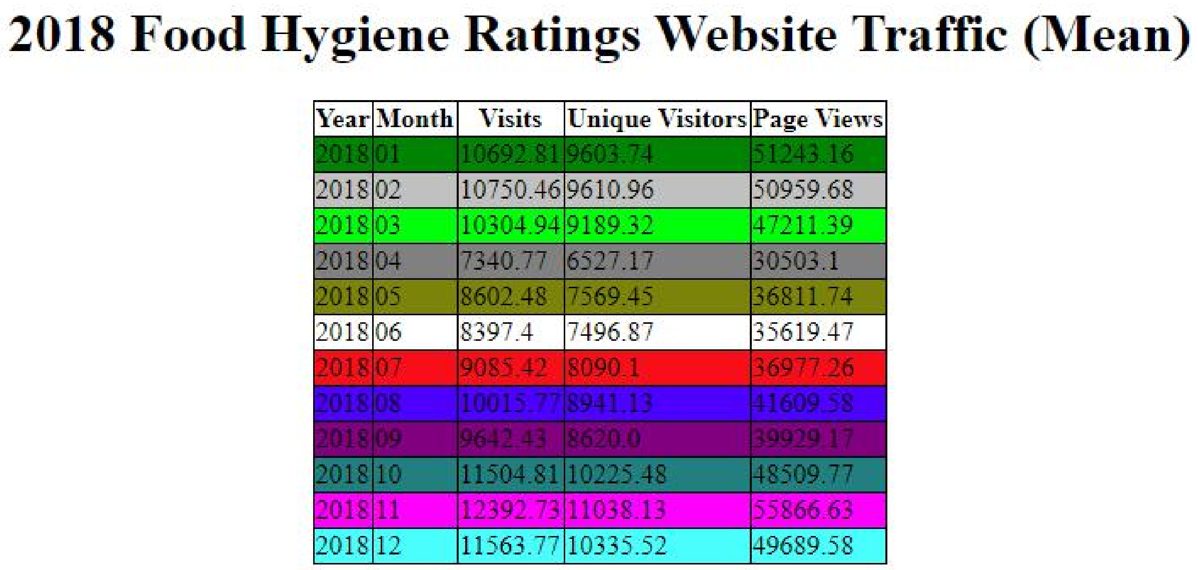


2018 Food Hygiene Ratings Website Traffic (Mean)

Year	Month	Visits	Unique Visitors	Page Views
2018	01	10692.806451612903	9603.741935483871	51243.16129032258
2018	02	10750.464285714286	9610.964285714286	50959.67857142857
2018	03	10304.935483870968	9189.322580645161	47211.3870967742
2018	04	7340.766666666666	6527.166666666667	30503.1
2018	05	8602.483870967742	7569.451612903225	36811.74193548387
2018	06	8397.4	7496.866666666667	35619.46666666667
2018	07	9085.41935483871	8090.096774193548	36977.25806451613
2018	08	10015.774193548386	8941.129032258064	41609.58064516129
2018	09	9642.433333333332	8620.0	39929.166666666664
2018	10	11504.806451612903	10225.483870967742	48509.77419354839
2018	11	12392.733333333334	11038.133333333333	55866.63333333333
2018	12	11563.774193548386	10335.516129032258	49689.58064516129

You can pass this coursework with a basic solution but you can gain higher marks if you add further decorations, more sophisticated formatting to your output.

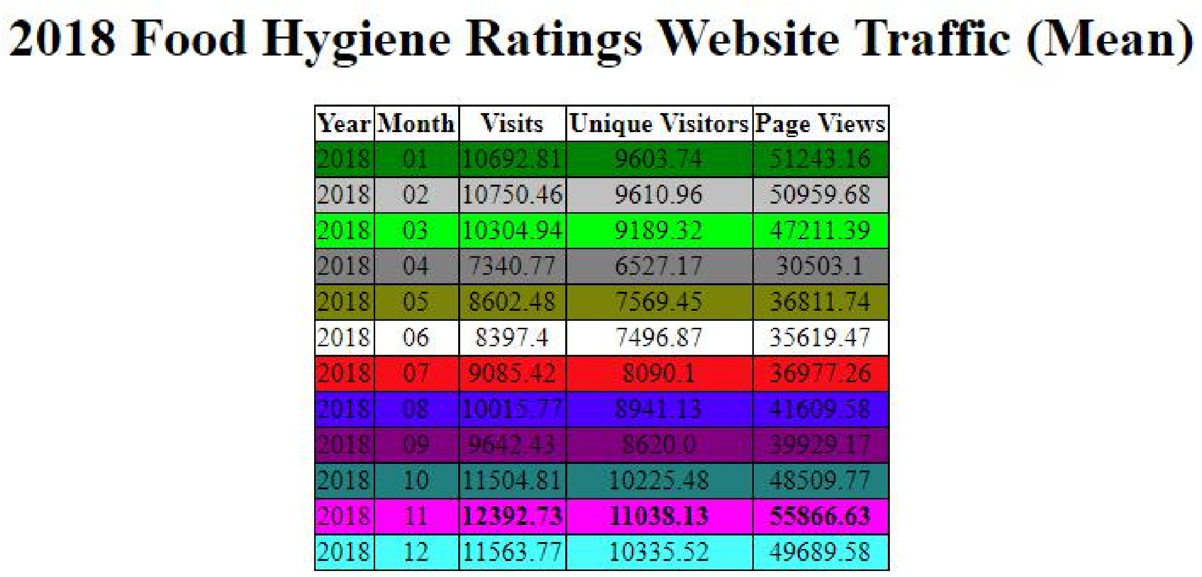
For example, you could get your output to look something like the figure below with alternating background colour for each row.



2018 Food Hygiene Ratings Website Traffic (Mean)

Year	Month	Visits	Unique Visitors	Page Views
2018	01	10692.81	9603.74	51243.16
2018	02	10750.46	9610.96	50959.68
2018	03	10304.94	9189.32	47211.39
2018	04	7340.77	6527.17	30503.1
2018	05	8602.48	7569.45	36811.74
2018	06	8397.4	7496.87	35619.47
2018	07	9085.42	8090.1	36977.26
2018	08	10015.77	8941.13	41609.58
2018	09	9642.43	8620.0	39929.17
2018	10	11504.81	10225.48	48509.77
2018	11	12392.73	11038.13	55866.63
2018	12	11563.77	10335.52	49689.58

You can also produce a more advanced version to look something like the figure below with highlighted cell for fields with the highest mean value in bold (Nov 2018 for Visists, Nov 2018 for UniqueVisitors, and Nov 2018 for Pageviews). Note that the field with highest mean may not be the same month.



2018 Food Hygiene Ratings Website Traffic (Mean)

Year	Month	Visits	Unique Visitors	Page Views
2018	01	10692.81	9603.74	51243.16
2018	02	10750.46	9610.96	50959.68
2018	03	10304.94	9189.32	47211.39
2018	04	7340.77	6527.17	30503.1
2018	05	8602.48	7569.45	36811.74
2018	06	8397.4	7496.87	35619.47
2018	07	9085.42	8090.1	36977.26
2018	08	10015.77	8941.13	41609.58
2018	09	9642.43	8620.0	39929.17
2018	10	11504.81	10225.48	48509.77
2018	11	**12392.73**	**11038.13**	**55866.63**
2018	12	11563.77	10335.52	49689.58

(c) Retrieve data from the Web [9 Marks]

For this coursework, you will retrieve data in CSV format from the web. Data stored on web pages can be accessed in Python using the urllib module.

The New Zealand Government publishes online datasets on population, business, labour market, society, economy, and environment. These datasets are licensed under the Creative Common Attribution 4.0 International licence. For this coursework, you are going to use the marine economy data from 2007 to 2018 from the URL:

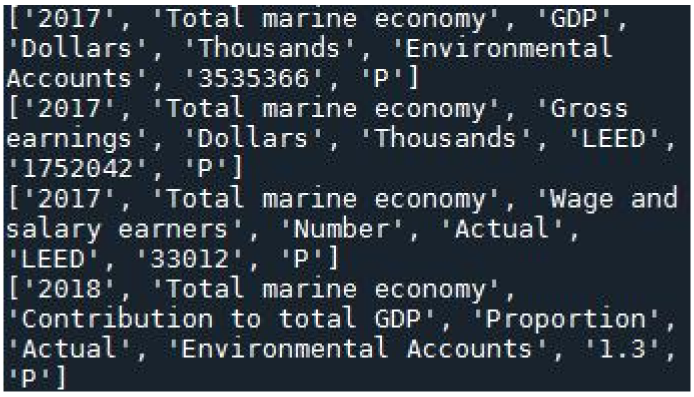
https://www.stats.govt.nz/assets/Uploads/Environmental-economic-accounts/Environmental-economic-accounts-2020-tables/Download-data/marine-economy-2007-18.csv

In this coursework, you need to write the following functions:

1. display_detail_data(data) [5 Marks]

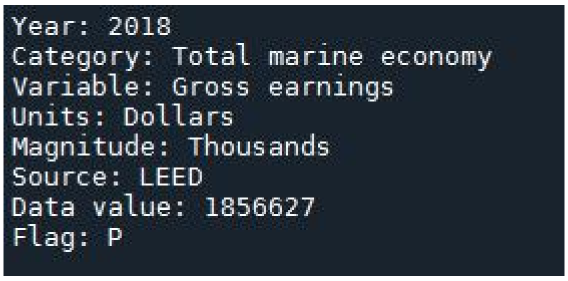
This function will take the marine economy data as its argument and display the details. A sample function to retrieve data from the web called get_csv_data_from_url(url) is provided in the template file query_web.py. You can use this function to retrieve the data from the url and supply it to your function display_detail_data(data). Note that the first row of the data returned by get_csv_data_from_url(url) is the header of the columns.

For basic solution, you can simply loop over the data and display it as shown in figure below.



```
['2017', 'Total marine economy', 'GDP',
'Dollars', 'Thousands', 'Environmental
Accounts', '3535366', 'P']
['2017', 'Total marine economy', 'Gross
earnings', 'Dollars', 'Thousands', 'LEED',
'1752042', 'P']
['2017', 'Total marine economy', 'Wage and
salary earners', 'Number', 'Actual',
'LEED', '33012', 'P']
['2018', 'Total marine economy',
'Contribution to total GDP', 'Proportion',
'Actual', 'Environmental Accounts', '1.3',
'P']
```

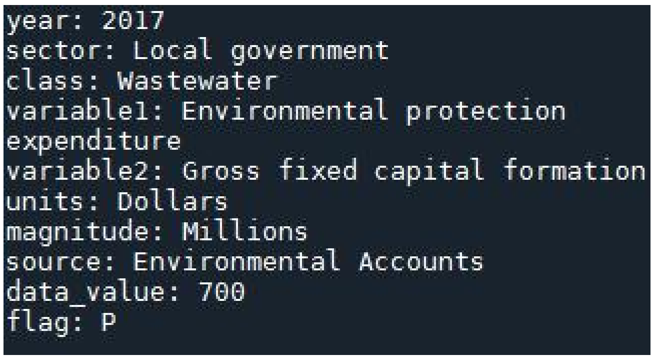
However, you can gain higher marks if you can format to your output as shown below by hardcoding the headers for each column.



```
Year: 2018
Category: Total marine economy
Variable: Gross earnings
Units: Dollars
Magnitude: Thousands
Source: LEED
Data value: 1856627
Flag: P
```

You can also produce a more advanced version with column headers extracted from the first row of the data. In this case, other online datasets such as environment protection expenditure from 2009 to 2018 at

https://www.stats.govt.nz/assets/Uploads/Environmental-economic-accounts/Environmental-economic-accounts-2020-tables/Download-data/environmental-protection-expenditure-account-2009-18.csv

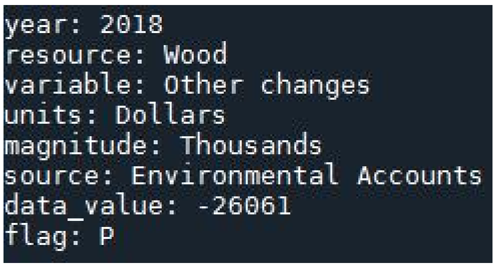


```
year: 2017
sector: Local government
class: Wastewater
variable1: Environmental protection
expenditure
variable2: Gross fixed capital formation
units: Dollars
magnitude: Millions
source: Environmental Accounts
data_value: 700
flag: P
```

and renewable energy stock account from 2007 to 2018 at

https://www.stats.govt.nz/assets/Uploads/Environmental-economic-accounts/Environmental-economic-accounts-2020-tables/Download-data/renewable-energy-stock-account-2007-18.csv

can also be displayed correctly.



```
year: 2018
resource: Wood
variable: Other changes
units: Dollars
magnitude: Thousands
source: Environmental Accounts
data_value: -26061
flag: P
```

2. display_GDP_data_year(data, year) [4 Marks]

This function will take 2 arguments: the retrieved marine economy data, and the year. The function will return the data_value for the variable 'GDP'. You can use the same get_csv_data_from_url(url) function to retrieve the data from the url and pass it to your function display_GDP_data_year(data).

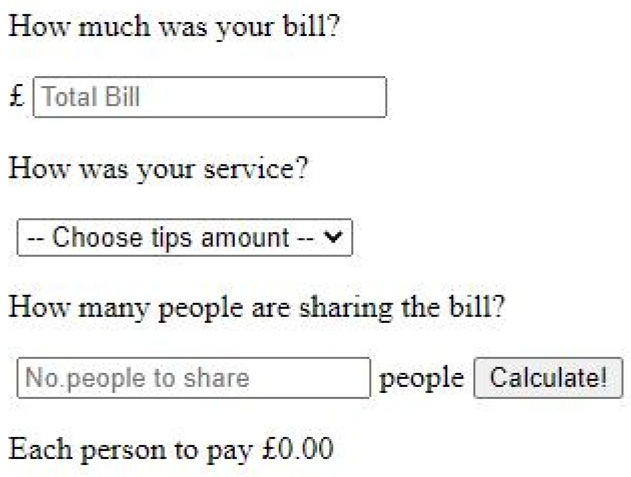
The marine economy data from 2007 to 2018 is assessable at the URL:

https://www.stats.govt.nz/assets/Uploads/Environmental-economic-accounts/Environmental-economic-accounts-2020-tables/Download-data/marine-economy-2007-18.csv

Example	
year	**output**
2008	564630
2012	839196
2018	1133460

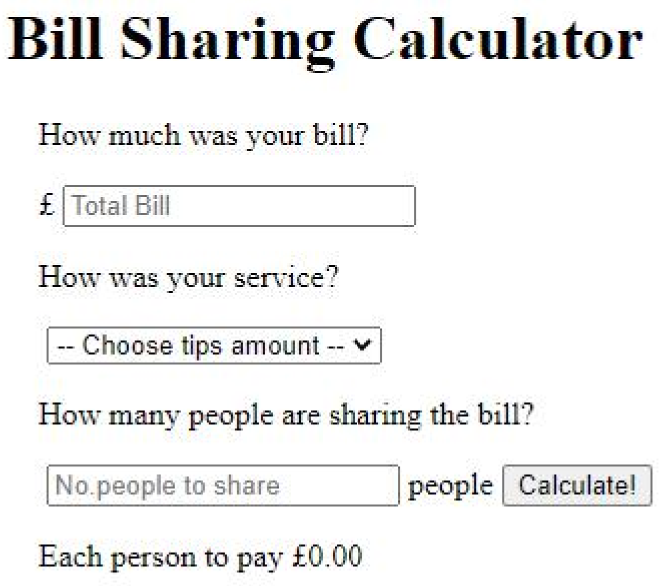
(d) JavaScript [10 Marks]

For this coursework you will add code to HTML and modify a JavaScript function for a bill sharing. You start by working on the given template file bill_sharing.html. The template file contains HTML and Javascript code to display input boxes, selection box, and text to display calculated amount to be paid by each person as in figure below.



1. Add heading [1 Mark]

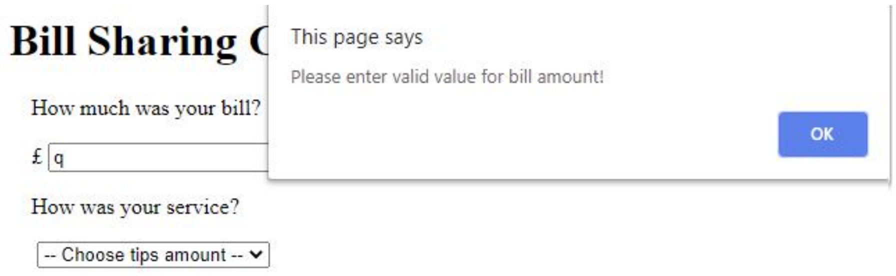
Add code to the bill_sharing.html to display a heading “Bill Sharing Calculator” similar to figure below.



2. Add validations to function [6 Marks]

Modify the **billSharing()** function to include validations for input for bill amount, tips amount, and the number of people to share the bill. If invalid data is entered, the following alert() messages should appear.

Pressing the Calculate button with invalid bill amount such as “” (empty) or non-numeric value such ‘q’ gives the following message. You can use JavaScript isNaN() function to check whether a value is not a number.


Bill Sharing C

This page says

Please enter valid value for bill amount!

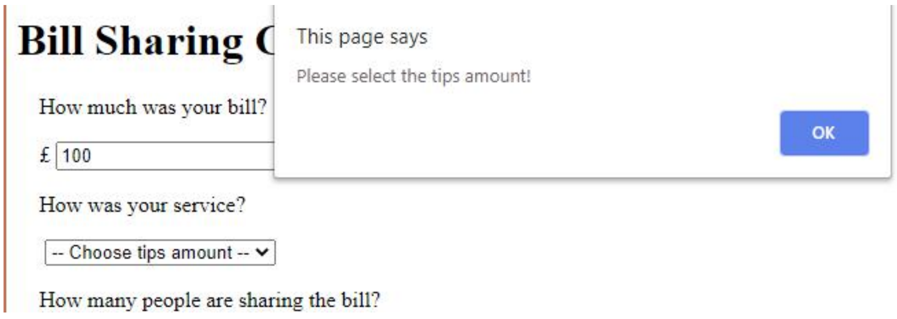
How much was your bill?

£ q

How was your service?

-- Choose tips amount --

Pressing the Calculate button without selecting the tips amount gives the following message.


Bill Sharing C

This page says

Please select the tips amount!

How much was your bill?

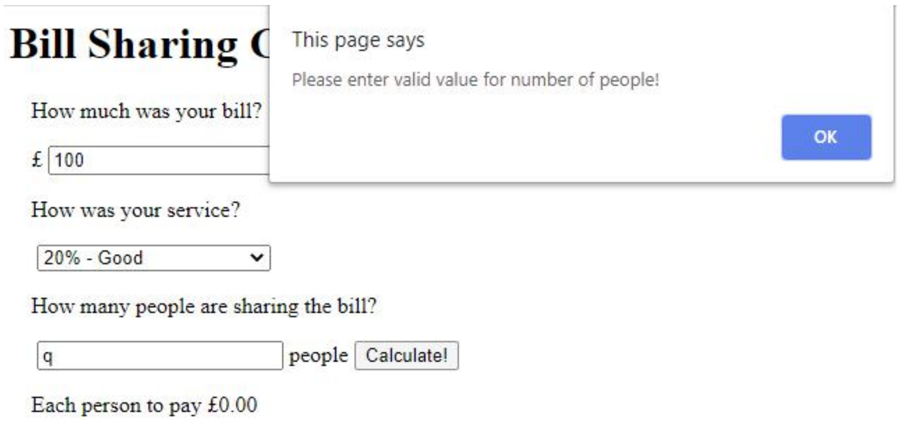
£ 100

How was your service?

-- Choose tips amount --

How many people are sharing the bill?

Pressing the Calculate button with invalid number of people to share the bill such as “” (empty) or non-numeric value such ‘q’ gives the following message. You can use JavaScript isNaN() function to check whether a value is not a number.


Bill Sharing C

This page says

Please enter valid value for number of people!

How much was your bill?

£ 100

How was your service?

20% - Good

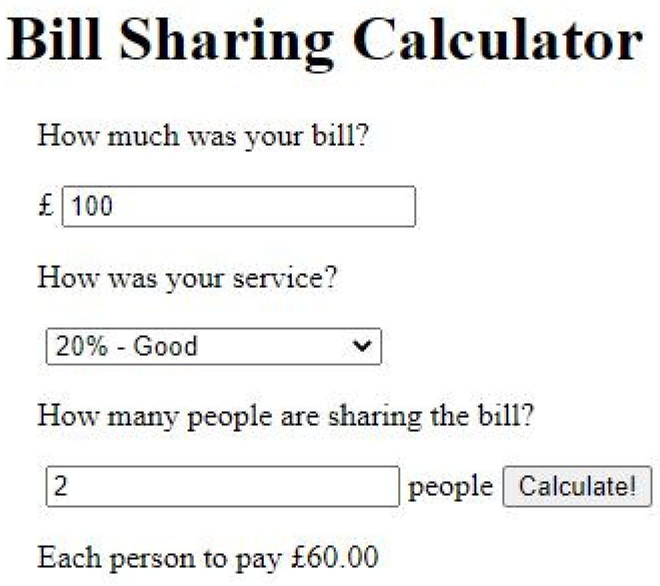
How many people are sharing the bill?

q people

Each person to pay £0.00

3. Add bill sharing calculation codes [3 Marks]

Further modify the **billSharing()** function to calculate the amount to be share between the people involved. For example, if the user enter 100 for the bill amount, select 20% - Good, and 2 people to share, the page will display "Each person to pay £60.00" similar to the figure shown. You can use toFixed() method tor round a number to specified number of decimals. For example, vNum.toFixed(2) will convert the vNum to two decimal points.


Bill Sharing Calculator

How much was your bill?

£ 100

How was your service?

20% - Good

How many people are sharing the bill?

2 people

Each person to pay £60.00

Submission Instructions

You should submit via the submission widget on the Assessment page of the module's Minerva pages.

Your submission should be in the form of a zip file cw3.zip containing all program files for all questions.

The deadline for submissions is **23:59 pm on Friday 25 December.** The standard university penalty of 5% of available marks per day will apply to late work.