Assignment 2 Software Technologies 2810ICT – ReadMe.txt Guide:

Task 1 – created_food.py:

1. Command to Run the Program:

   Python3 createdb_food.py

2. Notes:

   This program reads the raw data from the excel spreadsheets (inspections.xlsx and violations.xlsx) and creates a database called inspections.db. The tables of the database (if they exist) are dropped when running the program. The reason I decided to drop the tables is that the raw data may have changed / been updated since the program was last run.

Task 2 – sql_food.py:

1. Command to Run the Program:

   Python3 sql_food.py

2. Notes:

   This program assumes the database is up to date (i.e that created_food.py has been run previously). If the database does not exist or is not up to date – please run createdb_food.py before running this program.

   This program searches the database and finds the distinctive businesses that have at least 1 Violation found within the Inspections, and outputs the names of the businesses to console ordered alphabetically.

   After printing the businesses with at least 1 Violation to the console, this program searches the database and finds the name, address, zip code, city and the number of Violations for each business with at least 1 Violation – before printing the name and the number of Violation for each of these businesses to the console window, ordered by the number of Violations (descending). **Important Note**: Due to the number of businesses / violations within the raw data – only a portion of the data may be presented in the console window, and the previous output (business names with at least 1 Violation ordered alphabetically) may not be visible in the console window due to its limited size. You may choose to only output one of the results of this program (the business names ordered alphabetically, or the businesses and their number of Violations ordered by the number of Violations via commenting out the line within the relevant for loop).

Task 3 – excel_food.py:

1. Command to Run the Program:

   Python3 excel_food.py

2. Notes:

   This program assumes the database is up to date (i.e that created_food.py has been run previously). If the database does not exist or is not up to date – please run createdb_food.py before running this program.

   This Program searches the database to find the number of each type of Violation (determined by the violation_code), and outputs the violation_code, violation_description and the number of Violations for each to a new excel spreadsheet named 'ViolationTypes.xlsx'. The previous version is overwritten in the event the excel spreadsheet already exists.

<u>Task 4 – numpy_food.py:</u>

1. Command to Run the Program:

   ```
   Python3 numpy_food.py
   ```

2. Notes:

   This program assumes the database is up to date (i.e that created_food.py has been run previously). If the database does not exist or is not up to date – please run createdb_food.py before running this program.

   This program searches the database and returns 2 plots (one after the other). The first plot to be output consists of the number of Violations per month for the postcode with the highest number of Violations, the number of Violations per month for the postcode with the lowest number of Violations and the average number of Violations per month for all of California**. Important Note** 1: As the instructions of the assignment indicated to distinguish the Violations by the zip code of the excel spreadsheet – the program treats zip codes such as “90005” and “90005-3209” as 2 different zip codes. The program therefore determines that the zip code with the least number of Violations is 90005-2586, which was found to have only 1 Violation from 1 inspection. As a result – the monthly violations for the zip code with the least number of Violations is 0 for all months with the exception of the single month where the above Violation was found (you can zoom in on the plot to reduce the size of the axis to see the single Violation during the month 11/2017).

   The second plot for this program compares the total number of violations for each month for all McDonalds against the total number of violations for each month for all Burger Kings. **Important Note 2:** As the program displays 2 plots, and I did not want to present both plots as subplots on a single plot (as the information is then too cluttered / hard to read), the second plot is therefore displayed after the window of the first plot is closed (as the program is stalled when presenting a plot, and does not resume until the plot is closed).

   Important Note 3: Due to the size of the raw data, the plots may take some time before they are shown – so you may need to be patient when waiting for the plots to appear on screen. Please bear this is mind particularly after closing the first plot – as the second plot may also take a while before being presented.