ICT112 Assignment

University of the Sunshine Coast

2019 ATMC Semester 1

Generated Web Site for Aussie Road Fatalities

Your task in this assignment is to write a Python program that generates an informative web site about Australian Road Fatalities over the last 30 years, showing summary statistics about the crash types, speed limits, vehicle types, times of day, and the ages and genders of the people who died. This will allow viewers to get insights into the data, and to identify trends that are changing over time.

Figure 1. https://commons.wikimedia.org/wiki/File:Ljubljana_car_crash_2013.jpg

Your program will read some input data from a text file (e.g., a spreadsheet in comma-separated-values, or CSV, format) and automatically generate several web pages with links going between the pages. The input data is updated each month, so the idea is that your program will be run on the updated input data each week or each month to automatically generate an updated web site that shows the latest trends and statistics.

Each web page should show a different aspect of road fatalities, and how that aspect has changed over time. Some of your pages must have photos and graphs. Here are some suggestions for different pages you might generate:

- Fatalities per year, and how this rate is changing over time; bar graph
- Fatalities per Australian state;
- Fatalities at different times of the day (each hour);
- The impact of different types of crash (single vehicle, multiple vehicle, pedestrian, etc.);
- The contribution of crashes involving buses and trucks, compared to other kinds of crashes; there's no point where buses have a yes
- The impact of different speed limit zones on road fatalities;
- The impact of different road-user roles (driver; passenger; motor cyclist; cyclist, pedestrian, etc.) on road fatalities;
- The impact of gender on road fatalities - note that the data shows the gender of the deceased person, not necessarily of the driver;
- The impact of age (of the deceased) on road fatalities.

Four more important goals:

A. For most of these pages, it would be even more interesting and useful if you could display the *change* in these statistics over time, such as comparing 1989 against 2017, or the first 10 years against the most recent 10 years.
B. Also, for many of these pages, it would be more meaningful if you could display the statistics *relative* to the total number of vehicles or vehicle-kilometres in that category. For example, display the fatalities for buses and trucks relative to how many buses and trucks are on the roads, rather than just the absolute numbers of fatalities - this makes it possible to compare *fatality rates* between those different types of vehicles. To generate some of these relative percentages, you will need to obtain additional data from other sources - for example, the ABS has lots of data about vehicle usage in Australia (http://www.abs.gov.au/ausstats/abs@.nsf/mf/9208.0). Make sure you clearly document all your data sources!
C. There must be one top-level page, **index.html**, that is the entry point for the web site. This must give an overview of the whole website, and should have links to all the other pages.
D. All the HTML files and graphs in your whole web site must be generated by your Python program, so that your whole web site can be automatically updated when the input data values are updated, just by rerunning your Python program.

Learning Objectives

In this assignment you will learn how to:

1. *use top-down design to divide a larger data processing task into parts;*
2. *use Python to read and write text files (CSV data files);*
3. *use functions to raise the abstraction level of your program;*
4. *analyse data to produce graphs and summary statistics;*
5. *use string and list methods to generate HTML.*

Marking Criteria

This assignment is due at the end of Week 12 (Friday 7 June 11:59pm). You must submit your files in a single .zip file, via Blackboard. Your .zip file should include your Jupyter Notebook (containing your Python program), your input data files, all your generated output web pages; and any image files, graphs and CSS files that are used in your web pages.

To create a .zip file on a Windows PC, right-click on the folder containing all your Task 2 files, then select **'Send to / Compressed (zipped) folder**'. On a Mac, right-click on your folder and select the '**Compress *foldername***' option.

It will be marked out of 30. The marking criteria is as follows:

- **Jupyter Notebook Structure [6 marks].**
  This should use markup cells and headings to divide your program into

parts:
    - **Title**, plus your name and student number;
    - **Introduction,** which explains the structure of your web site, and the features / statistics / graphics that you have implemented;
    - **Web-site overview**, to explain the structure of your web site;
    - **Python code**, preferably broken into subsections for each part of your program;
- **Web Design and Functionality [6 marks].** To achieve maximum marks, the generated web site should contain several (5-10) web pages, with an entry page (index.html) that links to all the other pages, and the other pages linking back to the entry page; good use of images; the layout and organisation of the pages should be elegant and readable, with appropriate choices of colours; good use of HTML tables for layout; the generated pages should conform to HTML standards; a simple CSS file should be used to specify the look of the web pages (this CSS file does not have to be generated by your program).
- **Data Analysis and Visualisation [6 marks].** Your program should analyse the data and generate (and include in the web pages):
    - one or more **informative graphs** that visualise various aspects of the data;
    - **summary statistics** for various group of data (e.g. an average, or maximum, or minimum, or sum);
    - some **percentages that are relative** to the available population (e.g. display the fatalities for each Australian State relative to the population of that state, rather than just the absolute numbers of fatalities - this makes it possible to compare accident rates between states). **NOTE: To generate some of these relative percentages, you will need to obtain additional data from other sources, such as the population of each state, the number of drivers of each gender or each age group, etc. Make sure you clearly document your data sources;**
- **Python Functions [6 marks].** Most your Python code should be inside functions; function parameters should be used to make your program concise and flexible; each function should include appropriate function docs; correct and meaningful naming conventions should be followed for function names and variable names.
- **Program Design and Algorithms [6 marks].** To achieve maximum marks, your program should be elegantly designed: be concise; use constants for important values/strings that are likely to change; use appropriate data structures (e.g. strings, lists, tuples, dictionaries, or objects) to store the data; make good use of Python libraries and methods (e.g. for strings and lists); and be reasonably efficient during execution (e.g. reading the input files just once, rather than many times). A key criterion is that your program should not contain blocks of code that are duplicated or similar (such blocks of code should be simplified into a single block of code by good use of loops or functions).

Note: the generated web pages should NOT contain any Javascript, as the goal of this assignment is to generate simple static web pages using Python.