THE UNIVERSITY of ADELAIDE

ASSESSMENT 1 PART A PRACTICAL 2

Due Date: Sunday 11:59 pm, end of Week 2.

Weighting: 6% (this is the second of five practical assessments totalling 30%).

Introduction: Files and functions

Purpose

The purpose of this assessment is for you to demonstrate your ability to apply what you have learnt each week in the creation of a program to solve a problem.

Your tasks

Making decisions with health information

In this assessment you are required to build a Jupyter Notebook that includes Python code to present health data and analysis. You can either update your notebook from week 1 Assessment 1 Part A or start a new notebook.

You are encouraged to discuss general strategies and functions that can help you solve this problem with your peers in the Assessment 1: discussion board. You can also refer to examples from the weekly activities that solve general problems such as writing loops or making decisions with conditions. However, you must not share actual code examples that solve all or part of this assessment and you must not submit any code or markdown that you did not write yourself.

Getting Started

Create a new notebook or rename your week 1- Assessment 1- Part A notebook to: Week 2 - Assessment 1- Part A.

Section 1 – More blood Glucose Data

Create simulated blood glucose readings for 500 persons without diabetes (instead of the 5 you created in Week 1). The normal values are below 126 mg/dL and over 60 mg/dL. Write Python code that uses the random library to generate and print 500 values between these ranges.

Make sure this section has a heading and description of this data in a markdown cell.

Section 2 – Unit and data point chooser

In Australia, blood glucose levels are reported in mmol/L instead of mg/dL. The conversion is:

mmol/L = (mg/dL) / 18

Write Python code that first prompts the user to specify the preferred unit: (mmol/L or mg/dL) and then generates 500 data points in the specified units.

Make sure this section has a heading and description of this data in a markdown cell.

Section 3 – Giving more choices

Extend your code from Section 2 to also ask the user to input the number of data points that they want to generate.

Your Python code should then generate the number of data points the user requested in the units that the user requested.

Directions

Submit your assessment via online submission, as a Jupyter Notebook, providing your responses to questions in this Assessment brief. This assessment will be written in Python.

Save your notebook and upload it through the assignment link. Your tutor will assess your work based on the rubric. Make sure you have a look at the rubric so you understand what your tutor will be looking for.

Requirements

You must submit your Jupyter notebook file (containing your Python code) through the Canvas submission link.

Rubric

Assessment Criteria	**Level of performance**					
Jupyter Notebook structure	**N/A**	**Distinction (5/5)** All problems include a header related to what data is being presented and a clear description of the data.	**Credit (3.7/5)** All problems include a header related to what data is being presented and a minimal description of the data. (for example, a list instead of text describing the data).	**Pass (3.2/5)** All problems include a header and a description of the data but relation to data unclear.	**Fail (0/5)** No headers or description of the process.	**/5**

Python code	**High Distinction (15/15)** Meets Distinction level plus: output clearly indicates what the data values printed represent: label and units.	**Distinction (12.6/15)** Code is functional (correct results), uses meaningful variable names, comments describe the purpose of the code.	**Credit (11.1/15)** Code is functional (correct results).	**Pass (9.6/15)** Code is almost functional but not fully correct due to differences between assessment specification and produced code. Python code use must still be correct to pass.	**Fail (0/15)** Code is incorrect.	**/15**
TOTAL						**/20**