THE UNIVERSITY of ADELAIDE

ASSESSMENT 1 PART A PRACTICAL 1

Due Date: Sunday 11:59 pm, end of Week 1.

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

Introduction: Creating a Jupyter Notebook of health information

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

In this assessment you are required to build a Jupyter Notebook that includes Python code to present health data and analysis

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 removing characters or making strings upper case. 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 Jupyter Notebook named Week 1- Assessment 1- Part A. In a markdown cell, create a title for your notebook that reflects the information contained in the notebook (Example: Australian Weather Analysis) and give a brief description of what the analysis is about. Your notebook must contain the following sections.

Section 1 – Blood Glucose Data

The first section of your notebook should simulate the behaviour of the glucose in the blood in 5 persons without diabetes, the normal values are below 126 mg/dL and over 60 mg/dL. Write section heading and description of this data in a markdown cell. Then in a code cell, write Python code that uses the random library to print 5 values between these ranges.

Section 2 – BMI calculator

The next section of your notebook calculates the Body Mass Index (BMI) in kg/m^2. The BMI is calculated as body mass (colloquially called weight) divided by height squared or:

$BMI = m/h^2$

(where *m* is the person's weight in kg and *h* is their height in metres)

Write a description of this in a markdown cell in your Jupyter Notebook. Don't forget to add a header for this section.

Add a code cell and write code that prompts the user for a weight and a height and then prints their BMI with 2 digits of precision after the decimal. To get 2 digits of precision you will need to use the Python round() function. Use the built in Python help() function to look up how to use round().

Section 3 – DNA

The final section in your notebook determines the number of times each of the four nucleobases: adenine, guanine, cytosine, and thymine appears in a DNA sequence. The DNA sequence is represented as the letters of the nucleobases in the order they appear. For example: GAATAACA

Write a notebook section with a header, description and Python code that prompts the user for a DNA sequence and prints the number of adenine, guanine, cytosine and thymine nucleobases that appear in the sequence.

For example, the sequence GAATAACA should print:

Adenine: 5

Guanine: 1

Cytosine: 1

Thymine: 1

Once the code to count the different bases is working, extend your code to help the poor typist by accepting upper- or lower-case letters and removing any extra comma characters ‘,’. For example, if the user types: g,aTaC that would be converted in the program to GATAC. Print the converted DNA before printing the nucleobase counts.

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**