# DPIT121 Assignment 3: Visualised Analysis of Weekly Food and Nutrient Intake

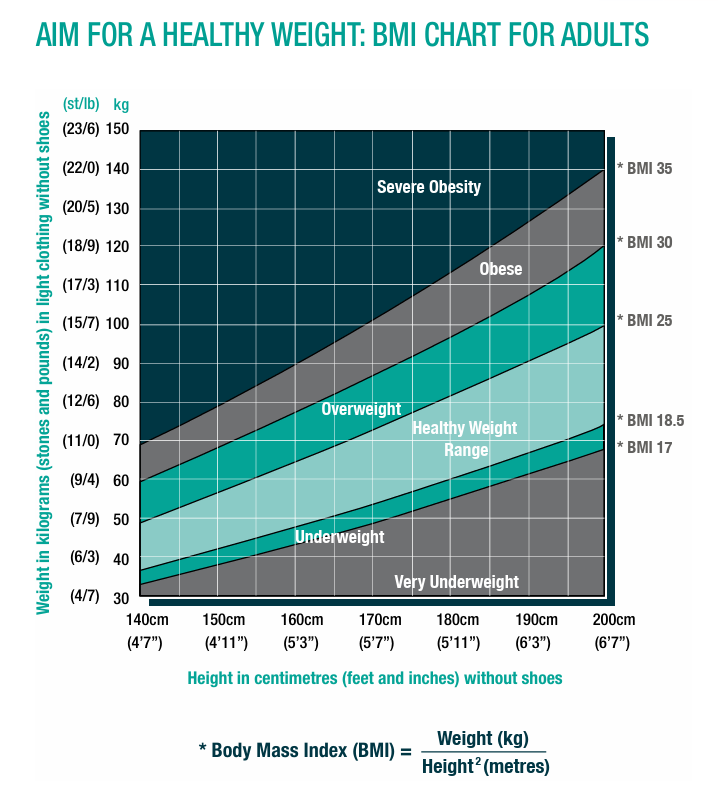
***Due Date: Week 13***

***Marks: 10***

# Objective

The objective of Assignment 3 is to apply the object-oriented design and programming knowledge acquired from Week 1 to Week 12 lectures. In this assignment, students will design a Python program, utilising UML class diagrams, to assist the Australian dairy industry in providing visualised analysis of people's weekly food, nutrient, and/or energy intakes based on their genders and ages. Additionally, the program will compare individual people's records with average and recommended values (refer to Fig. 1 and Fig. 2).

**Figure 1 - General BMI information needed for BMI calculation.**



**Figure 2 – Average recommended daily serving size information for five food categories for an adult: Vegetables, Fruits, Grains, Meats and Dairy**

|  |  |  |
| --- | --- | --- |
| **Food Category** | **Recommended Daily Serves** | **Single Serve size** |
| **Vegetables** | **6** | **20g** |
| **Fruits** | **2** | **20g** |
| **Grains** | **6** | **40g bread equivalent** |
| **Meats** | **3** | **30g** |
| **Dairy** | **2.5** | **20ml yogurt or 10g cheese** |

The proposed Python program should facilitate object serialization processes and allow for visualisation (**using matplotlib**) of the analysis results using different styles (**refer to Fig. 3**) and enable the export of visual analysis results for printing, reuse, and documentation purposes. Students are required to apply at **least one Python class design pattern** introduced in the lecture in this assignment to gain a better understanding of object-oriented programming code reuse principles in practical scenarios.

# Project Description

To provide a comprehensive visual analysis of weekly food, nutrient, and/or energy intakes, students are asked to design a Python program that fulfils the following requirements:

## Design

* Create a **UML class diagram** to represent the object-oriented design of the Python program, incorporating at least **one class design pattern (Decorator, Observer, Singleton or Adapter)** for code reuse.
* Develop **suitable classes** to capture information related to food, nutrients, energy intake, genders, and ages, utilising appropriate class relationships and OOD such as inheritance and polymorphism.
* Implement object serialisation processes to enable the persistence of object data.
  + Each type of chart should be saved as an individual **JPG file**.
  + User information such as **age, gender, weight, height and daily serving intake of vegetables, fruits, grains, meats and dairy** should be saved as a **text file (named user\_info.txt)** upon serialization.

**Sample user\_info.txt content**

Age: 45 years  
Gender: male  
Weight: 87.0 kg  
Height: 178.0 cm  
Daily Serving Intake:  
Vegetables: 2.0 servings  
Fruits: 3.0 servings  
Grains: 4.0 servings  
Meats: 2.0 servings  
Dairy: 1.0 servings

**Figure 3 – Examples of Visualisation Styles in Python**

|  |  |
| --- | --- |
| **BMI** | **Bar Chart** |
| **Pie Chart** | **Bubble Chart** |
| **Line Chart** | |

**Example/Sample Program Output (Bolded portion represent user input for illustration)**

Enter your age (in years): **45**

Enter your gender (male or female): **male**

Enter your weight (in kg): **87**

Enter your height (in cm): **178**

Enter your daily intake of Vegetables (in servings): **2**

Enter your daily intake of Fruits (in servings): **3**

Enter your daily intake of Grains (in servings): **4**

Enter your daily intake of Meats (in servings): **2**

Enter your daily intake of Dairy (in servings): **1**

Your BMI is: **27.46**

Would you like to save the BMI chart? (yes/no): **yes**

Visualize BMI saved as visualize\_bmi\_chart.jpg

Would you like to save the charts? (yes/no): **yes**

Visualize BMI saved as visualize\_bmi\_chart.jpg

Visualize Bar Chart saved as visualize\_bar\_chart\_chart.jpg

Visualize Pie Charts saved as visualize\_pie\_charts\_chart.jpg

Visualize Bubble Chart saved as visualize\_bubble\_chart\_chart.jpg

Visualize Line Chart saved as visualize\_line\_chart\_chart.jpg

Do you also want to export your data? (yes/no): **yes**

Data has been exported to user\_info.txt

## Program Implementation

* **Collect data** on people's weekly food, nutrient, and/or energy intakes (i.e., vegetables, fruits, grain, meats, and dairy products), along with their genders and ages (see Fig. 1).
* Perform a **visual analysis** of the collected data, comparing individual records with *average recommended daily serving size information for five food categories for an adult* (see Fig. 2).
* Use different visualisation styles (**at least three different styles**), such as bar chart, line chart, pie chart, bubble chart etc., to present the analysis results as informative and visually appealing. The user should be able to **export these charts as .jpg files**.
* Enable the object serialisation to save the **user’s data** in a **text file** **(named user\_info.txt).**
* Use **matplotlib** to allow users to **save visual analysis results** for different purposes such as printing, reuse, and documentation.

## Testing and Analysis

* Design and implement test cases using the Python **unittest** module to verify the correctness of the implemented functionality (**at least one test per class**).

Hint for unitest: python -m unittest test\_assignment3.py

* Analyse the test results and evaluate the effectiveness of the applied class design patterns in achieving code reuse and modularity. You should indicate which **design pattern/s** is applied and how it is reused. You should use **coverage** to evaluate the effectiveness of **code coverage.**

Hint for code coverage: coverage run -m unittest discover  
 coverage report

# Submission with Marking Criteria

Students must submit the following components to Moodle as a **zipped file**.

* **Report**: Submit a PDF file (named **assignment3\_visualisation.pdf**) describing the program's functionality, designs, and the class design pattern/s used. Include a UML class diagram representing the object-oriented design of the Python program, highlighting the applied **class design patterns**. (2 marks)
* **Python Program:** Submit the complete Python program (named **assignment3.py**) implementing the visualised analysis system, adhering to the visualisation requirements (**at least three different styles)**. (5 marks)
* **Test Cases and Analysis:** Submit test cases developed using the Python **unittest** module, along with an analysis of the test results **(including screenshots)** and **insights into the effectiveness of the applied class design patterns using code coverage**. Python unittest module should be named **test\_assignment3.py** (2 marks)
* **Demonstration Video:** Because the submission deadline is in Week 13, the **demonstration will NOT be required in the lab**. Alternatively, you are required to **submit a short video record (up to 5 mins)** to demonstrate your solution. (1 mark)  
    
  **Demonstration Video Submission Guidelines**: You can upload the demonstration video to YouTube (private mode) and include the **video link in the REPORT. Please check if your video link works correctly.**

# Submission Guidelines

* You will submit **3 files in total** on Moodle
* Submit your files " **assignment3\_visualisation.pdf** " "**assignment3.py**" and “**test\_assignment3.py**” to Moodle Assignment 3 section.
* Note: Email submissions will **NOT** be accepted.
* **Late Submission Penalty**
  + Up to 1 day late: 10% deduction of the original mark.
  + Up to 2 days late: 20% deduction of the original mark.
  + Up to 3 days late: 30% deduction of the original mark.
  + **After 3 days without academic consideration： 0**

# Appendix

<https://www.eatforhealth.gov.au/guidelines/guidelines>