Macromedia University of Applied Sciences

Course Title: Advanced coding skills
Name of Examiner: Prof. Ahmet Yildiz, Prof. William B Morrison

To be completed by students:

302560	B-UBr DT ATI 6e 24S
Student ID number	Matriculation
Gautam	Jayant
Last name	First name

The student work will be submitted as:
(Please fill in the letter X in the appropriate box)

- [X] Individual work
- [ ] Group work

Does only apply to group work: (Complete only if it is a group work)

If you are submitting a group work, please list the first and last names of all group members. The names must be entered electronically by the respective group members themselves. By entering the name, it is confirmed that the student agrees to submit the paper in the present form. Furthermore, by entering their names it is declared by the individual groups members to have created the project paper (in case of a group work: the part which the respective student has contributed to the paper and has marked accordingly within the paper) on their own, without the help of others. In the process, the student has not used any aids other than those cited in the listing of sources and literature. All passages taken either verbatim or in adapted form from publications are indicated as such. The work has not been submitted to another examination office in the same or a similar form. With the submission, the group members agree that all evaluations and notes of the examiners are deposited in the uploaded work. The group member who uploaded the work must make the correction notes available to the other group members.

1)	5) .....
2)	6) .....
3)	7) .....
4)	8) .....

Assessment of group work:
(Please fill in the letter X in the appropriate box)

- [ ] I apply for an individual evaluation (i.e. each member of the group will receive an individual mark)
- [ ] I apply for a group evaluation (i.e. each member of the group receives an identical grade)

Berlin, 22.01.2025	Jayant Gautam
Place/Date	Complete First Name and Last Name

Evaluation (according to grading scale), result of initial inspection: total points: _____

Date: **Name, first name First Examiner (to be filled in digitally)**

To be completed by the examiner: (Text area for the second examiner)

ADVANCED CODING SKILLS

-by Jayant Gautam

Fitness Tracker Project Documentation

Project Overview

The Fitness Tracker application was developed to provide a simple yet effective solution for tracking fitness activities, calculating BMI, and viewing stored data. The project includes object-oriented design principles, data persistence, and a graphical user interface (GUI) to enhance usability.

Steps to Develop the Project.

1. **Project Initialization**

The project was structured into multiple Python modules to ensure modularity and maintainability:

- fitness_tracker.py: Core application logic.

- data_persistence.py: Handles saving and loading data.

- bmi_calculator.py: Provides BMI calculation and categorization.

- user_interface.py: Originally provided a terminal-based interface.

Each module was organized into the tracker package.

2. **Object-Oriented Development**

Classes Implemented:

- FitnessTracker: Orchestrates the main application logic.

- DataPersistence: Manages data storage and retrieval using JSON files.

- BMICalculator: Provides static methods for BMI calculation and determining BMI categories.

- UserInterface: Handles user interaction (initially terminal-based).

Inheritance: Although inheritance was not directly required in this project, the design pattern supports scalability. For example, a new AdvancedBMICalculator class could inherit from BMICalculator to extend functionality.

3. **Adding BMI Calculation**

The BMICalculator class was created to calculate BMI using the formula:

BMI = weight / height^2

It also includes a method to categorize BMI into underweight, normal weight, overweight, or obese.

4. **Data Persistence**

The DataPersistence class was implemented to save fitness data to a JSON file (fitness_data.json).

It uses Python's json module for serialization and deserialization, ensuring data is retained across sessions.

5. **Transition to a GUI**

A GUI was created using Python's Tkinter library to replace the terminal-based interface.

Features of the GUI:

- A main menu with buttons for adding data, viewing data, calculating BMI, and exiting.

- Separate windows for each functionality, ensuring a clean and user-friendly design.

- Error handling and validation for user inputs using message boxes.

Structure:

The FitnessAppGUI class encapsulates the GUI logic and interacts with other modules like DataPersistence and BMICalculator.

6. **Testing the Application**

The application was tested using simulated inputs to verify:

- Fitness data is saved and displayed correctly.

- BMI calculations are accurate and match the correct categories.

- The GUI flows smoothly between functionalities.

7. **Packaging the Project**

The updated project, including the GUI, was packaged into a ZIP file for portability.

The following structure was maintained:

- main.py: Entry point for the application.

- fitness_app_gui.py: Main script for running the GUI.

- tracker package: Contains all core modules.
- tests folder: Placeholder for testing scripts.
- docs folder: Placeholder for additional documentation.

Final Features of the Project

- Object-Oriented Design: Modular and scalable.
- Packages and Modules: Organized in a clear structure for maintainability.
- Data Persistence: Uses JSON to store and retrieve fitness data.
- Graphical User Interface: Built with Tkinter for better usability.

Future Improvements

- Advanced Analytics: Add graphs and progress tracking for user activities.
- Cloud Integration: Enable data synchronization across devices.
- Mobile Application: Port the application to mobile platforms for broader accessibility.