# COMP1811 – Python Project Report

| **Your Name:** |  | **Student ID** |  |
| --- | --- | --- | --- |
| **Partner’s name:** |  | **Student ID** |  |

# Brief statement of features you have completed

*THIS SECTION SHOULD BE THE SAME FOR ALL GROUP MEMBERS*

| 1.1 Circle the parts of the coursework you have **fully completed and are fully working**. Please be accurate. | **Features F1a:** i ii  F1b: i ii  **F2a:** i ii  F2b: i ii  **F3a:** i ii  iii F3b: i ii |  |
| --- | --- | --- |
| 1.2 Circle the parts of the coursework you have **partly completed or are partly working.** | **Features F1a:** i ii  F1b: i ii  **F2a:** i ii  F2b: i ii  **F3a:** i ii  iii F3b: i ii |  |
| Briefly explain your answer if you circled any parts in 1.2 | | |

# Concise List of Bugs and Weaknesses

*A concise list of bugs and/or weaknesses in your work (if you don't think there are any, then say so). Bugs that are declared in this list will lose you fewer marks than ones that you don't declare! (****100-200 words****, but word count depends heavily on the number of bugs and weaknesses identified.)*

*THIS SECTION SHOULD BE COMPLETED INDIVIDUALLY FOR F1 AND F2 AND AS A GROUP FOR F3.*

## Bugs

*List each bug plus a brief description. A bug is code that causes an error or produces unexpected results.*

## Weaknesses

*List each weakness plus a brief description. A weakness is code that only works under limited scenarios and at some point produces erroneous or unexpected results or code/output that can be improved.*

# Description of the features implemented

*Describe your implementation design and the choices made (e.g. choice of data structures, custom data types, code logic, choice of functions, etc) and indicate how the features developed were integrated. (****200-400 words***

*THIS SECTION SHOULD BE COMPLETED INDIVIDUALLY FOR F1 AND F2 AND AS A GROUP FOR F3.*

## My Feature (1 or 2 – Delete as appropriate)

*Design description for implemented feature…*

## My Feature (1 or 2 – Delete as appropriate)

*Design description for implemented feature…*

## Feature 3

*Design description for implemented feature…*

# Classes and OOP Features

*List the classes you developed and provide an exposition on the choice of classes, class design, and OOP features implemented. List all the classes used in your program and include the attributes and behaviours for each. You may use a class diagram to illustrate these classes – do not include the class code here. Your narrative for section 4.2 should describe the design decisions you made, and the OOP techniques used (abstraction, encapsulation, inheritance/polymorphism).* ***Note****: stating definitions here will not get you marks, you must clearly outline how you implemented the techniques in your code and WHY. (****400-600 words****)*

*THIS SECTION SHOULD BE COMPLETED INDIVIDUALLY FOR F1 AND F2 AND AS A GROUP FOR F3.*

## List of Classes Used with a Brief Explanation

## F1

## F2

## F3

## Brief Explanation of Class Design and OOP Features Used

## F1

## F2

## F3

# Code for the Classes Created

*Add the* ***code for each of the classes you have implemented yourself*** *here. If you have contributed to parts of classes, please highlight those parts in a different colour and label them with your name. Copy and paste relevant code - actual code please, no screenshots! Make it easy for the tutor to read. Add an explanation if necessary – though your in-code comments should be clear enough. You will lose marks if screenshots are provided instead of code.* ***DO NOT provide a listing of the entire code. You will be marked down if a full code listing is provided, or you include the code as a screenshot****.*

*THIS SECTION SHOULD BE COMPLETED INDIVIDUALLY FOR F1 AND F2 AND AS A GROUP FOR F3.*

## Class …

## Class …

## Class …

…

# Testing

*Describe the process you took to test your code and to make sure the program functions as required.* ***Make sure you include a test plan and demonstrate thorough testing of your own code as well as the integrated code****.*

*THIS SECTION SHOULD BE COMPLETED INDIVIDUALLY FOR F1 AND F2 AND AS A GROUP FOR F3.*

## F1

*Test plan for F1…*

## F2

*Test plan for F3…*

## F3

*Test plan for F3…*

# Annotated Screenshots Demonstrating Implementation

*Provide screenshots that demonstrate the features implemented running – i.e. showing the output produced by all of the subfeatures. Annotate each screenshot and if necessary, provide a brief description for* ***each*** *(****up to 100 words****) to explain the code in action.*

*THIS SECTION SHOULD BE COMPLETED INDIVIDUALLY FOR F1 AND F2 AND AS A GROUP FOR F3.*

## Feature F1

## F1a.i- screenshots …

## F1b.ii- screenshots …

## F1b.i- screenshots …

## F1b.ii- screenshots …

## Feature F2

## F2a.i- screenshots …

## F2a.ii- screenshots …

## F2b.i- screenshots …

## F2b.ii- screenshots …

## Feature F3

## F3a.i- screenshots …

## F3a.ii- screenshots …

## F3a.iii- screenshots …

## F3b.i- screenshots …

## F3b.ii- screenshots …

# OpenAI Comparison

*Provide any code generated using OpenAI along with a listing of the code you initially wrote from scratch in a table showing the generated and your code side-by-side for each feature. Examine and explain the generated code’s design, describing its quality and efficiency compared to the initial code you wrote. The narrative must also describe how you used the generated code to improve your own code or describe how the generated code may be improved.*

# SELF-ASSESSMENT

*Please assess yourself objectively for each section shown below and then enter the total mark you expect to get. Marks for each assessment criterion are indicated between parentheses.*

Code development (70)

1. Features Implemented [40] (group work and integration will be assessed here)

Partner A or Partner B features (up to 20)

Sub-features have not been implemented – 0

Attempted, not complete or very buggy – 1 to 5

Implemented and functioning without errors but not integrated – 6 to 10

Implemented and fully integrated but buggy – 11 to 16

Implemented, fully integrated and functioning without errors – 17 to 20

Group Features (up to 20)

Sub-features has not been implemented – 0

Attempted, not complete or very buggy – 1 to 3

Implemented and functioning without errors but not integrated – 4 to 8

Implemented and fully integrated but buggy – 9 to 14

Implemented, fully integrated and functioning without errors – 15 to 20

**For this criterion I think I got: out of 40**

1. Use of OOP techniques [20]

Abstraction (up to 7)

No classes have been created – 0

Classes have been created superficially and not instantiated or used – 1

Classes have been created but only some have been instantiated and used – 2 or 3

Useful classes and objects have been created and used correctly – 4 or 5

The use of classes and objects exceeds the specification – 6 or 7

Encapsulation (up to 7)

No encapsulation has been used – 0

Class variables and methods have been encapsulated superficially – 1 to 3

Class variables and methods have been encapsulated correctly – 4 to 6

The use of encapsulation exceeds the specification – 6 to 8

Inheritance or polymorphism (up to 6)

No inheritance or polymorphism has been used – 0

Inheritance or polymorphism has been used superficially – – 1 or 2

Inheritance or polymorphism has been used correctly – 3 or 4

The use of inheritance or polymorphism exceeds the specification – 5 or 6

**For this criterion I think I got: out of 20**

1. Quality of Code [10]

Code Duplication (up to 4)

Code contains too many unnecessary code repetition – 0

Regular occurrences of duplicate code – 1

Occasional duplicate code – 2

Very little duplicate code – 3

No duplicate code – 4

PEP8 Conventions and naming of variables, methods and classes (up to 3)

PEP8 and naming convention has not been used – 0

PEP8 and naming convention has been used occasionally – 1

PEP8 and naming convention has been used regularly – 2

PEP8 convention used professionally and all items have been named correctly – 3

In-code Comments (up to 3)

No in-code comments – 0

Code contains occasional in-code comments – 1

Code contains useful and regular in-code comments – 2

Thoroughly commented, good use of docstrings, and header comments describing.py files – 3

**For this criterion I think I got: out of 10**

2. Documentation (20)

Design (up to 10) clear exposition about the design and decisions for OOP use

The documentation cannot be understood on first reading or is mostly incomplete – 0

The documentation is readable, but a section(s) are missing – 1 to 3

The documentation is complete – 4 to 6

The documentation is complete and of a high standard – 7 to 10

Testing (10)

Testing has not been demonstrated in the documentation – 0

A test plan has been included but is incomplete – 1 or 2

A test plan has been included with some appropriate test cases – 3 to 6

A full test plan has been included with thorough test cases and evidence of carrying it out – 7 to 10

**For this criterion I think I got: out of 20**

3. Screencast - Acceptance Test (10)

Recorded demonstration of code and accompanying explanatory commentary (up to 10)

Not submitted or no work demonstrated or lacking commentary – 0

Work demonstrated not to expected standard, unclear commentary,   
 superficial team contribution – 1 to 3

Work demonstrated to expected standard, sufficient commentary and team contribution – 4 to 7

Work demonstrated exceeded the standard expected – 8 to 10

**For this criterion I think I got: out of 10**

**I think my overall mark would be: out of 100**

# Appendix A: Code Listing

*Provide a complete listing of all the \*.py files in your PyCharm project. Make sure your code is well commented and applies professional Python convention (refer to* [*PEP 8*](https://www.python.org/dev/peps/pep-0008/) *for details). The code listed here must match that uploaded to Moodle. Please copy and paste the actual code – no screenshots please! You will lose marks if screenshots are provided instead of code. Clearly label the parts each partner created with their name and SID.*