



FIT9136 Algorithms and Programming Foundations in Python

Assignment 2

Aug 2022

Table of Contents

[1. Key Information](#)

[2. Instruction](#)

[2.1. Book Class](#)

[2.2. Book Operation Class](#)

[2.3. User Class](#)

[2.4. User Operation Class](#)

[2.5. Reader Class](#)

[2.6. ReaderOperations Class](#)

[2.7. Main File](#)

[2.8. User Manual](#)

[3. Group Communication](#)

[4. Git Management](#)

[5. Do and Do NOT](#)

[5.1. Important NOTES](#)

[6. Submission Requirements](#)

[7. Academic Integrity](#)

[8. Marking Guide](#)

[9. Getting help](#)

[9.1. English language skills](#)

[9.2. Study skills](#)

[9.3. Things are tough right now](#)

[9.4. Things in the unit don't make sense](#)

[9.5. I don't know what I need](#)

1. Key Information

Purpose	<p>This assignment will develop your skills in designing, constructing, testing, and documenting a Python program according to specific programming standards. This assessment is related to the following learning outcome (LO):</p> <ul style="list-style-type: none"> • LO2 - Restructure a computational program into manageable units of modules and classes using the object-oriented methodology • LO3 - Demonstrate Input/Output strategies in a Python application and apply appropriate testing and exception handling techniques
Your task	<p>This assignment is a group task where you will write Python code for a simple application whereby you will be developing an e-book reader and analyser application as per the specifications.</p>
Value	<p>30% of your total marks for the unit.</p>
Due Date	<p>Friday, 23 September 2022, 4:30 PM (AEST)</p>
Submission	<ul style="list-style-type: none"> • Via Moodle Assignment Submission. • FIT GitLab check-ins will be used to assess the history of development • MOSS will be used for similarity checking of all submissions.
Assessment Criteria	<p>This assessment includes a compulsory interview with your tutor following the submission date. At the interview you will be asked to explain your code/design/testing, modify your code, and discuss your design decisions and alternatives. Marks will not be awarded for any section of code/design/functionality that you cannot explain satisfactorily. Failure to attend the interview will result in your assessment not being marked. You will be provided with the timing of the interviews at a later date.</p> <p>The following aspects will be assessed:</p> <ol style="list-style-type: none"> 1. Program functionality in accordance to the requirements 2. Code Architecture and Adherence to Python coding standards 3. The comprehensiveness of documented code
Late Penalties	<ul style="list-style-type: none"> • 10% deduction per calendar day or part thereof for up to one week • Submissions more than 7 calendar days after the due date will receive a mark of zero (0) and no assessment feedback will be provided.
Support Resources	<p>See Moodle Assessment page and Section 7 in this document.</p>
Feedback	<p>Feedback will be provided via two formats:</p> <ul style="list-style-type: none"> • general cohort performance • specific student feedback ten working days post submission

2. Instruction

In this assignment, you are going to develop an e-book reading and analysing application called Findle. This application allows the user to perform a series of operations with various e-books (you have been provided with the 100 e-books in the **data** folder).

The basic operations of the application are listed below:

- Register in the application (Users have to register before using this application)
- Login to the application
- Show information for a specified book (i.e., Number of Chapters, Number of Words, Number of Lines)
- Show the title of all available books
- Show the contents of a specified book
- Read a specified book
- Show the available books by author
- Show the number of books by year
- Bookmark a specific page in a specified book
- Delete bookmarks
- Show all bookmarks
- Add a book to favourites
- Delete a book from favourites
- Show all favourite books
- Logout of the application.

In order to develop this application you need to develop 6 classes where each class is defined in one individual python file (i.e., *.py):

- Book Class (i.e., book.py)
- Book Operation Class (i.e., book_operation.py)
- User Class (i.e., user.py)
- User Operation Class (i.e., user_operation.py)
- Reader Class (i.e., reader.py)
- Reader Operation class (i.e., reader_operation.py)

And one main python file (i.e., main.py) where the execution of the program is implemented.

2.1. Book Class

Contains all the information about a book.

Required Attributes

- *title*
- *author*
- *release_date*
- *last_update_date*
- *language*
- *producer*
- *book_path* (A relative path such as `"/data/books_data/11-0.txt"`)

Required Methods

2.1.1. `__init__()`

Constructs a book object.

Positional Arguments*

- *title* (i.e., unique identifier)
- *author*
- *release_date*
- *last_update_date*
- *language*
- *producer*
- *book_path*

Returns

- N/A

2.1.2. `__str__()`

Return all the book's attributes as a formatted string.

Positional Arguments

- N/A

Returns

- String returned in the format of:
`"title;;;author;;;release_date;;;last_update_date;;;language;;;producer;;;book_path"`

**All positional arguments of the constructor must have a default value.*

2.2. Book Operation Class

Contains all the operations related to a book.

Required Class Variables

- *book_folder_path* (A relative folder path, i.e., `"/data/books_data/"`)
- *book_info_path* (A relative path, i.e., `"/data/result_data/books.txt"`)
- *book_title_list* (List of all books titles such as `"[title1, title2, title3, ...]"`)
- *book_info_dict* (Key-value pairs of a book title and book object such as `"{title1: obj1, title2:obj2, title3:obj3.....}"`)

Required Methods

2.2.1. extract_book_info()

Extracts book attributes from each book contained within a folder (i.e., *book_folder_path*) and writes these as a formatted string (See 2.1.2. *__str__()*) in a given file (i.e., *book_info_path*).

Positional Arguments

- N/A

Returns

- A boolean result should be returned to indicate the success of the method, i.e., True if completed, False if an error has occurred.

Notes:

- If this method is called multiple times, you need to make sure the result book file does not contain duplicates.
- Every time this method is called, the *book_title_list* and *book_info_dict* should also be updated if any changes are found.

2.2.2. load_book_info()

Loads all the book's information from a given file (i.e., *book_info_path*) into the *book_info_dict* dictionary and the *book_title_list* list.

Positional Arguments

- N/A

Returns

- A boolean result should be returned to indicate the success of the method, i.e., True if completed, False if an error has occurred.

Notes:

- Some book titles may have more than one line of text, which needs to

be merged into one row.

- Duplicate titles are not allowed in *book_info_list* list
- This method should be only executed at the beginning of the program.

2.2.3. **get_counts()**

Return the number of chapters, words, and lines for the specified *book_title*

Positional Arguments	
	<ul style="list-style-type: none">• <i>book_title</i>
Returns	<ul style="list-style-type: none">• A tuple consisting of the number of chapters, words, and lines for the given book (all int type).

Notes:

- Before counting words, it is necessary to remove all punctuation.
- When counting words and lines, only consider the content between the book start sign:
“*** START OF THE PROJECT GUTENBERG EBOOK ALICE’S ADVENTURES IN WONDERLAND ***”
and the book end sign:
“*** END OF THE PROJECT GUTENBERG EBOOK ALICE’S ADVENTURES IN WONDERLAND ***”.
Do not count the lines in the book that contain the start and end signs.
- Empty lines should be counted within the start and end signs.
- Hyphenated words such as “child-life” or “after-time” can be considered as two words after removing the punctuation “-”.
- If the book does not show any chapter information, record the number of chapters as 0.

2.2.4. **display_titles()**

Display titles of the books listed in *book_title_list* list.

Positional Argument	
	<ul style="list-style-type: none">• <i>page_number</i> (a positive integer value starting from one)
Returns	<ul style="list-style-type: none">• Display one page of the book titles from the <i>book_title_list</i> list, where each title starts with the number that shows the title number in the list. As an example:

	<table border="1" data-bbox="368 208 1310 656"> <tr> <td data-bbox="368 208 579 656"></td><td data-bbox="579 208 1310 656"> <pre> -----List of Book Titles----- 11. Alice's Adventures in Wonderland 12. Through the Looking-Glass, And What Alice Found There 13. Peter Pan Peter Pan and Wendy 14. The Time Machine 15. The Legend of Sleepy Hollow 16. 17. 18. 19. 20. Current Page: 2 Total Pages: 10 -----end----- </pre> </td></tr> </table> <p>Notes:</p> <ul style="list-style-type: none"> Each page shows a maximum of 10 titles. In the above example, we have 95 titles in the <i>book_title_list</i> list, and the <i>page_number</i> value is 2. So the calculated total page is 10 (i.e., 95/10), and the second 10 items in the <i>book_title_list</i> list will be printed. Invalid <i>page_number</i> (e.g., -1, 0, or number > total page) should be handled correctly. <table border="1" data-bbox="368 994 1310 1814"> <tr> <td colspan="2" data-bbox="368 994 1310 1111"> 2.2.5. show_book_content() Display the contents of a book for the specified <i>book_title</i> </td></tr> <tr> <td data-bbox="368 1111 579 1216">Positional Argument</td><td data-bbox="579 1111 1310 1216"> <ul style="list-style-type: none"> <i>book_title</i> </td></tr> <tr> <td data-bbox="368 1216 579 1814">Returns</td><td data-bbox="579 1216 1310 1814"> <ul style="list-style-type: none"> The book title with the list of contents display in the following format: <pre> Title:Alice's Adventures in Wonderland Contents CHAPTER I. Down the Rabbit-Hole CHAPTER II. The Pool of Tears CHAPTER III. A Caucus-Race and a Long Tale CHAPTER IV. The Rabbit Sends in a Little Bill CHAPTER V. Advice from a Caterpillar CHAPTER VI. Pig and Pepper CHAPTER VII. A Mad Tea-Party CHAPTER VIII. The Queen's Croquet-Ground CHAPTER IX. The Mock Turtle's Story CHAPTER X. The Lobster Quadrille CHAPTER XI. Who Stole the Tarts? CHAPTER XII. Alice's Evidence </pre> </td></tr> </table> <p>Notes:</p> <ul style="list-style-type: none"> Print out "No Contents", If the book does not have contents (chapters). 		<pre> -----List of Book Titles----- 11. Alice's Adventures in Wonderland 12. Through the Looking-Glass, And What Alice Found There 13. Peter Pan Peter Pan and Wendy 14. The Time Machine 15. The Legend of Sleepy Hollow 16. 17. 18. 19. 20. Current Page: 2 Total Pages: 10 -----end----- </pre>	2.2.5. show_book_content() Display the contents of a book for the specified <i>book_title</i>		Positional Argument	<ul style="list-style-type: none"> <i>book_title</i> 	Returns	<ul style="list-style-type: none"> The book title with the list of contents display in the following format: <pre> Title:Alice's Adventures in Wonderland Contents CHAPTER I. Down the Rabbit-Hole CHAPTER II. The Pool of Tears CHAPTER III. A Caucus-Race and a Long Tale CHAPTER IV. The Rabbit Sends in a Little Bill CHAPTER V. Advice from a Caterpillar CHAPTER VI. Pig and Pepper CHAPTER VII. A Mad Tea-Party CHAPTER VIII. The Queen's Croquet-Ground CHAPTER IX. The Mock Turtle's Story CHAPTER X. The Lobster Quadrille CHAPTER XI. Who Stole the Tarts? CHAPTER XII. Alice's Evidence </pre>
	<pre> -----List of Book Titles----- 11. Alice's Adventures in Wonderland 12. Through the Looking-Glass, And What Alice Found There 13. Peter Pan Peter Pan and Wendy 14. The Time Machine 15. The Legend of Sleepy Hollow 16. 17. 18. 19. 20. Current Page: 2 Total Pages: 10 -----end----- </pre>								
2.2.5. show_book_content() Display the contents of a book for the specified <i>book_title</i>									
Positional Argument	<ul style="list-style-type: none"> <i>book_title</i> 								
Returns	<ul style="list-style-type: none"> The book title with the list of contents display in the following format: <pre> Title:Alice's Adventures in Wonderland Contents CHAPTER I. Down the Rabbit-Hole CHAPTER II. The Pool of Tears CHAPTER III. A Caucus-Race and a Long Tale CHAPTER IV. The Rabbit Sends in a Little Bill CHAPTER V. Advice from a Caterpillar CHAPTER VI. Pig and Pepper CHAPTER VII. A Mad Tea-Party CHAPTER VIII. The Queen's Croquet-Ground CHAPTER IX. The Mock Turtle's Story CHAPTER X. The Lobster Quadrille CHAPTER XI. Who Stole the Tarts? CHAPTER XII. Alice's Evidence </pre> 								

2.2.6. show_book_text() Display the book text for the specified page of a book	
Positional Argument	<ul style="list-style-type: none"> • <i>book_title</i> • <i>page_number</i>(a positive integer value start from one)
Returns	<ul style="list-style-type: none"> • Display one page of the book text where each page includes the title of the book, chapter number, book text, and page number. An example is provided below: <pre>-----Alice's Adventures in Wonderland - Chapter I----- There was nothing so _very_ remarkable in that; nor did Alice think it so _very_ much out of the way to hear the Rabbit say to itself, "Oh dear! Oh dear! I shall be late!" (when she thought it over afterwards, it occurred to her that she ought to have wondered at this, but at the time it all seemed quite natural); but when the Rabbit actually _took_ a watch out of its waistcoat-pocket_, and looked at it, and then hurried on, Alice started to her feet, for it flashed across her mind that she had never before seen a rabbit with either a waistcoat-pocket, or a watch to take out of it, and burning with curiosity, she ran across the field after it, and fortunately was just in time to see it pop down a large rabbit-hole under the hedge. In another moment down went Alice after it, never once considering how in the world she was to get out again. -----end page 2-----</pre>
Notes: <ul style="list-style-type: none"> • Each page shows 15 lines of the book text. • If the book has a chapter, The first line of the book text is the chapter number, and the last line of the book text is the book end sign, e.g.: <pre>**** END OF THE PROJECT GUTENBERG EBOOK ALICE'S ADVENTURES IN WONDERLAND ****</pre> Do not show the lines of the book end sign. • If the book doesn't have a chapter (i.e., contents) the book text is all the lines between the book start sign, e.g.: <pre>**** START OF THE PROJECT GUTENBERG EBOOK ALICE'S ADVENTURES IN WONDERLAND ****</pre> and the book end sign, e.g.: <pre>**** END OF THE PROJECT GUTENBERG EBOOK ALICE'S ADVENTURES IN WONDERLAND ****</pre> Do not show the lines of the book start sign and end sign. • In the above example we display page 2 of "Alice's Adventures in Wonderland". 	
2.2.7. get_book_by_author() Display all the book titles belonging to the specified author	
Positional Arguments	<ul style="list-style-type: none"> • <i>author_name</i>

Returns	<ul style="list-style-type: none"> Print the book title followed by the full author name as shown in the image below. <pre> -----Books by frank----- 1. Book Title 1 _ frank guvanier 2. Book Title 2 _ L. Frank Baum 3. Book Title 3 _ Frank Gabriel -----end----- </pre>				
<p>Notes:</p> <ul style="list-style-type: none"> If the value of author_name is in a book's author string the book is returned. For instance, author_name="frank" returns an author such as "L. Frank Baum" because it contains "frank". The comparison should be case insensitive. <div data-bbox="368 913 1310 1167"> <p>2.2.8. get_book_release_year()</p> <p>Display the total number of books for the following release year</p> <ul style="list-style-type: none"> before 1990 (year < 1990), between 1990 and 2000 (1990 <= year <= 2000), after 2000 (2000 < year) </div> <table border="1" data-bbox="368 1167 1310 1574"> <tr> <td data-bbox="368 1167 579 1279"> Positional Arguments </td><td data-bbox="579 1167 1310 1279"> <ul style="list-style-type: none"> N/A </td></tr> <tr> <td data-bbox="368 1279 579 1574"> Returns </td><td data-bbox="579 1279 1310 1574"> <ul style="list-style-type: none"> Print out the total number of books for each specified release year. An example: <pre> -----Total Number of Books----- Number of books released before 1990: 10 Number of books released between 1990 and 2000: 5 Number of books released after 2000: 43 </pre> </td></tr> </table>		Positional Arguments	<ul style="list-style-type: none"> N/A 	Returns	<ul style="list-style-type: none"> Print out the total number of books for each specified release year. An example: <pre> -----Total Number of Books----- Number of books released before 1990: 10 Number of books released between 1990 and 2000: 5 Number of books released after 2000: 43 </pre>
Positional Arguments	<ul style="list-style-type: none"> N/A 				
Returns	<ul style="list-style-type: none"> Print out the total number of books for each specified release year. An example: <pre> -----Total Number of Books----- Number of books released before 1990: 10 Number of books released between 1990 and 2000: 5 Number of books released after 2000: 43 </pre>				
<p>Note:</p> <ul style="list-style-type: none"> If no book is within the range, print out 0. Show your output in a clear format. 					

2.3. User Class

Contains all the information about a user.

Required Attributes

- *user_id* (i.e., unique identifier)
- *user_name*
- *user_password*
- *user_role*

Required Methods

2.3.1. `__init__()`

Constructs a user object.

Positional Arguments

- *user_id* (i.e., unique identifier)
- *user_name*
- *user_password*
- *user_role*

Returns

- N/A

Notes:

- All positional arguments of the constructor must have a default value.
- *user_id* is a unique 10 digit number that would be generated randomly when a user object is constructed.

2.3.2. `__str__()`

Return all the attributes of a user as a formatted string.

Positional Arguments

- N/A

Returns

- String returned in the format of:
"user_id;;;user_name;;;user_password;;;user_role"

2.4. User Operation Class

Contains all the operations related to a user.

Required Class Variables

- *user_info_path* (A relative path i.e. `"/data/result_data/users.txt"`)
- *user_info_list* (List of all users such as `"[user1, user2, user3, ...]"`)

Required Methods

2.4.1. load_user_info()

Loads all the registered users information from a given file (i.e., *user_info_path* class variable) into the *user_info_list* list

Positional Arguments

- *N/A*

Returns

- A boolean result should be returned to indicate the success of the method i.e. True if completed, False if an error has occurred.

Notes:

- Every line in the file will be stored as one element in the list in the form of the user object.
- This method should be executed only at the beginning of the program.

2.4.2. user_registration()

Create a user object and save it in *user_info_list* list.

Positional Arguments

- *user_name*
- *user_password*
- *user_role*

Returns

- A boolean result should be returned to indicate the success of the method i.e. True if completed, False if an error has occurred.

Notes:

- Duplicate *user_id* and *user_name* can not be accepted and needs to be handled correctly.

2.4.3. user_login()

Authenticate a user login attempt.

	Positional Argument	<ul style="list-style-type: none"> • <i>user_name</i> • <i>user_password</i>
	Returns	<ul style="list-style-type: none"> • A boolean result should be returned to indicate the success of the method i.e. True if user registered , False with an appropriate error message if the user is not registered or an error has occurred.
<u>Notes:</u> <ul style="list-style-type: none"> • Do not contain any file reading/writing operations in this method. 		
2.4.4. write_user_info() Write all the users 's information provided in <i>user_info_list</i> class variable in the provided file located in <i>user_info_path</i> class variable		
	Positional Arguments	<ul style="list-style-type: none"> • <i>N/A</i>
	Returns	<ul style="list-style-type: none"> • Boolean result should be returned to indicate the success of the method i.e. True if user registered , False with an appropriate error message if an error has occurred.
<u>Notes:</u> <ul style="list-style-type: none"> • The format for each user object written in the file should follow the format of the <code>__str__</code> method. • This method should be executed only at the end of the program. 		

2.5. Reader Class

Contains all the information about a reader. This class inherits from the user class.

Required Attributes	<ul style="list-style-type: none"> • <i>user_id</i> (i.e., unique identifier) • <i>user_name</i> • <i>user_password</i> • <i>user_role</i> • <i>favourite_book_list</i> (List of reader's favorite book such as [book_title1, book_title2,]) • <i>bookmark_list</i>(list of tuple where each tuple refers to bookmarked book title and page # such as [(book_title1, page#1),(book_title2, page#2),])
----------------------------	--

Required Methods	<table border="1"> <tr> <td colspan="2"> 2.5.1.__init__() Constructs a reader object. </td></tr> <tr> <td>Positional Arguments</td><td> <ul style="list-style-type: none"> • <i>user_id</i> (i.e., unique identifier) • <i>user_name</i> • <i>user_password</i> • <i>User_role</i> • <i>favourite_book_list</i> • <i>bookmark_list</i> </td></tr> <tr> <td>Returns</td><td> <ul style="list-style-type: none"> • N/A </td></tr> </table> <table border="1"> <tr> <td colspan="2"> 2.5.2.__str__() Return all the attributes of a reader as a formatted string. </td></tr> <tr> <td>Positional Arguments</td><td> <ul style="list-style-type: none"> • N/A </td></tr> <tr> <td>Returns</td><td> <ul style="list-style-type: none"> • String returned in the format of: "UserID;;;Username;;;Password;;;Role;;;[book_title1, book_title2,];;[(book_title1, page#1),(book_title2, page#2),]" </td></tr> </table> <p>Notes:</p> <ul style="list-style-type: none"> • All positional arguments of the constructor must have a default value. • The default value for Role attribute is "reader". 	2.5.1.__init__() Constructs a reader object.		Positional Arguments	<ul style="list-style-type: none"> • <i>user_id</i> (i.e., unique identifier) • <i>user_name</i> • <i>user_password</i> • <i>User_role</i> • <i>favourite_book_list</i> • <i>bookmark_list</i> 	Returns	<ul style="list-style-type: none"> • N/A 	2.5.2.__str__() Return all the attributes of a reader as a formatted string.		Positional Arguments	<ul style="list-style-type: none"> • N/A 	Returns	<ul style="list-style-type: none"> • String returned in the format of: "UserID;;;Username;;;Password;;;Role;;;[book_title1, book_title2,];;[(book_title1, page#1),(book_title2, page#2),]"
2.5.1.__init__() Constructs a reader object.													
Positional Arguments	<ul style="list-style-type: none"> • <i>user_id</i> (i.e., unique identifier) • <i>user_name</i> • <i>user_password</i> • <i>User_role</i> • <i>favourite_book_list</i> • <i>bookmark_list</i> 												
Returns	<ul style="list-style-type: none"> • N/A 												
2.5.2.__str__() Return all the attributes of a reader as a formatted string.													
Positional Arguments	<ul style="list-style-type: none"> • N/A 												
Returns	<ul style="list-style-type: none"> • String returned in the format of: "UserID;;;Username;;;Password;;;Role;;;[book_title1, book_title2,];;[(book_title1, page#1),(book_title2, page#2),]" 												

2.6. ReaderOperations Class

Contains all the operations related to a reader.

Required Class Variables

- *N/A*

Required Methods

2.6.1. add_bookmark()

Add specified reader's bookmarked (i.e., title and page#) to the *bookmark_list* list

Positional Arguments

- *book_title*
- *page_no*
- *reader object*

Returns

- One tuple (***book_title***, ***page***) will be added to the reader's instance variable ***bookmark_list***.

2.6.2. delete_bookmark()

Delete the specified reader's bookmark based on the index from *bookmark_list* list

Positional Arguments

- *num*(the value starts from 1 not 0)
- *reader object*

Returns

- The corresponding bookmark at position ***num*** will be removed from the reader's instance variable ***bookmark_list***.

2.6.3. save_favorite_book ()

Add the specified reader's favourite book (i.e., title) in *favourite_book_list* list

Positional Argument

- *book_title*
- *reader object*

Returns

- The *book_title* is saved into the reader's instance variable *favourite_book_list*.

2.6.4. delete_favorite_book()

Remove specified reader's favourite book from <i>favourite_book_list</i> list	
Positional Arguments	<ul style="list-style-type: none"> • <i>num/title (have a int/str value)</i> • <i>reader object</i>
Returns	<ul style="list-style-type: none"> • If the type is int, delete one item from the reader's instance variable <i>favourite_book_list</i> based on the index. • If the type is str, search the match item in the <i>favourite_book_list</i> to perform deletion.
<p>Note:</p> <ul style="list-style-type: none"> • If the index number is invalid or the str value is not found, print out messages to users. 	
2.6.5. show_all_favorite_book () Display all the Reader's favourite books	
Positional Argument	<ul style="list-style-type: none"> • <i>reader object</i>
Returns	<ul style="list-style-type: none"> • Print out all favourite books with a row number shown at the beginning of each item for the specified reader.
2.6.6. show_all_bookmarks() Display all the Reader's bookmarks	
Positional Argument	<ul style="list-style-type: none"> • <i>reader object</i>
Returns	<ul style="list-style-type: none"> • Print out all bookmarks with a row number shown at the beginning of each item for the specified reader.

2.7. Main File

In this component, your team will construct the menu and control logic for the application. The design and implementation is up to your team but must include the menu items outlined in section 2 using the classes and methods outlined.

A tip here is when you try to accept user input, try to use a command that includes multiple arguments to simplify the get-user-input process. For example, if a reader hopes to delete a bookmark, you can have a user input "3 4". These two arguments can be translated to 3-> delete bookmark command and 4-> the book row number that the user expects to delete. The user input format is "command arg1 arg2 arg3 ...". The validations can be performed directly for all the arguments of commands. As a consequence, handling the multi-level user input does not require to handle several nested loops.

You must ensure that your menu and control logic handles exceptions appropriately.

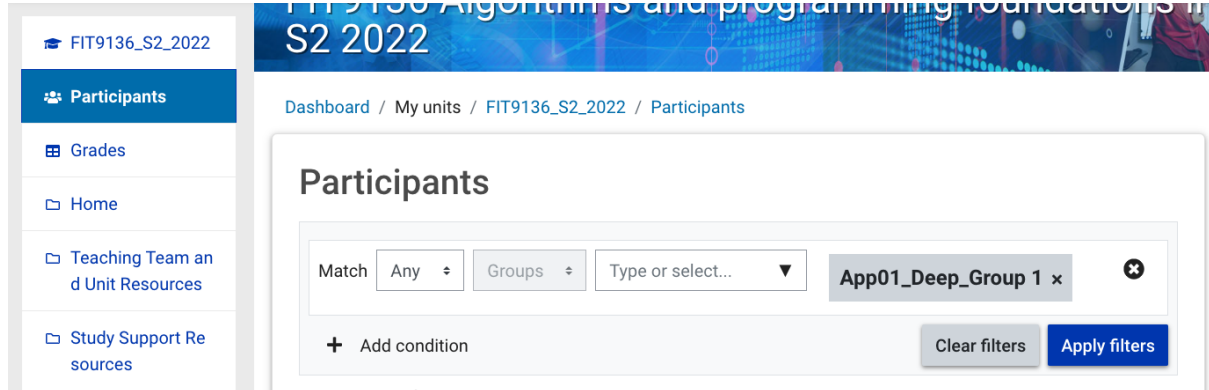
You can break down your code to several functions if you wish but you still need to call the extra defined functions in the main function. In the if `__name__=="__main__"` part, only call `main()` function. Your tutor will only run your `main.py` file.

2.8. User Manual

It is required to provide user instruction saved into a file named **README.pdf** which describes how to use your application. Please do not show too much content in this document. No more than 5 pages.

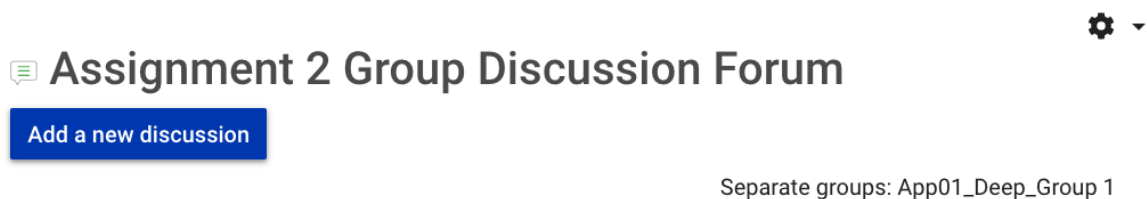
3. Group Communication

This is a group assignment. You are going to work in a group of 2-3. You can find your group number, group members, and allocated tutor through Moodle> Participant:



The screenshot shows the Moodle interface for the course FIT9136_S2_2022. On the left is a sidebar with navigation links: Participants (selected), Grades, Home, Teaching Team and Unit Resources, and Study Support Resources. The main content area has a breadcrumb trail: Dashboard / My units / FIT9136_S2_2022 / Participants. Below this is a 'Participants' section with a filter bar. The filter bar includes a 'Match' dropdown set to 'Any', a 'Groups' dropdown, and a text input field with a dropdown arrow. A filter tag 'App01_Deep_Group 1' is active. Below the filter bar is a '+ Add condition' link and 'Clear filters' and 'Apply filters' buttons.

You can communicate with your group members and your allocated tutors using Assignment 2 Group Discussion Forum on Moodle> Assessments> Assignment 2 Group Discussion Forum:



The screenshot shows the 'Assignment 2 Group Discussion Forum' page. At the top right is a gear icon. Below the title is a blue button labeled 'Add a new discussion'. At the bottom right, it says 'Separate groups: App01_Deep_Group 1'.

4. Git Management

ENSURE your group number and members names are shown on each .py files as a part of header comment at the top of file you submit.

GIT STORAGE

Your work **MUST** be saved in your group local working directory (repo) in the Assignment 2 folder and **regularly pushed to the FIT GitLab server** to build a clear history of development of your application. Any submission with less than twelve pushes to the FIT GitLab server will incur a grade penalty. Please note twelve pushes is a minimum, in practice we would expect significantly more. **This number of pushes must be evenly distributed amongst group members.** All commits must include a meaningful commit message which clearly describes what the particular commit is about.

Groups must regularly check that their pushes have been successful by logging in to the web interface of the FIT GitLab server; you must not simply assume they are working. Before submission, via Moodle, you must log in to the [web interface of the GitLab server](#) and ensure your submission files are present on the GitLab server.

GIT automatically maintains a history of all files pushed to the server, you do not need to, and **MUST** not, add a version name to your various versions, please ensure you use the same name for all versions of a particular file.

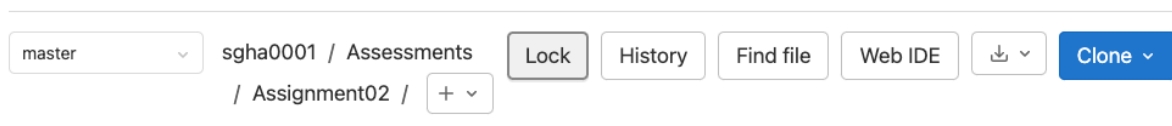
If you have problems in pushing to the remote group repo, you should move your current local group repo out of the way (to a new folder) and then reclone your group repo.

If multiple students work on a .py file at the same time, merging these changes can be quite difficult. For this reason, you are required to take a simple approach to working on the .py file - **lock the remote repo when making changes.**

Whenever a particular student wishes to work on the model, they should go to the Git Server web interface and check if the assignment 2 folder has been locked by another member of the group.

If it has, you must not carry out any work on the assignment task.

If it has not been locked, you can proceed to lock the folder by selecting "Lock":



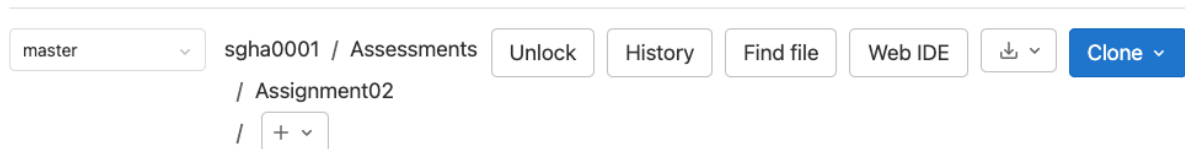
Ensure you are in the correct folder when this lock is applied.

You will know the items are locked as each will have a lock icon attached to it:



If you hover over the padlock icon, you will be able to see who currently has the folder locked.

When you have completed your work, and pushed it to Git, you should return to the Git web interface and unlock the folder:



It is our expectation that all members of the group will contribute to complete the program, just one member must not complete it. In assessing your group's work, we will examine the commit log to ensure all members of the group have participated.

5. Do and Do NOT

Do	Do NOT
<ul style="list-style-type: none">• Maintain appropriate citing and referncing¹,• Get support early from this unit and other services within the university,• Apply for special consideration or for extensions² early if needed.	<ul style="list-style-type: none">• Leave your assignment in draft mode (assignments in draft mode will not be marked),• Submit late (10% daily penalty applies)³,• Attempt to submit after 7 days of the due date (they will not be accepted), unless you have special consideration.

5.1. Important NOTES

- DO NOT use absolute paths and follow the exact structure for the path as provided in the examples in each section. All the path issues that cause your program crash or exception will lead to no mark for functionality.
- You must implement all required methods but you may add additional methods if you require.
- Please make sure your file reading/writing operations include encoding type **utf8**. Changing the application running environment could cause issues if you do not have the encoding type. Any character encoding issues/exceptions will cause serious mark deduction.
- If one method is not working and it hinders the program from continuing running to show other functionalities, the following functionalities will get no mark. For example, if your system can register but cannot login, the functionality that needs to be marked after login will get no mark and you will only get marks for the register part. Therefore, it is important to finish the methods one by one and make sure they can work properly.
- If any exception/errors happen when running your program, you will lose 50% of allocated function logic marks. For example, if the main menu function returns any error, then the maximum mark you can get is 5% instead of 10% in the function logic.
- Add correct validation and output messages to make your code more user-friendly to users.
- The assignment must be done using the **Pycharm, Python Version 3.9**.

¹<https://www.monash.edu/library/help/citing-and-referencing/citing-and-referencing-tutorial>

² <https://www.monash.edu/exams/changes/special-consideration>

³ e.g.: The original mark was 70/100, submitting 2 days late results in 50/100 (20 mark deduction). This includes weekends and public holidays.

- The Python code for this assignment must be implemented according to the [PEP 8-Style Guide for Python Code](#).
- The allowed libraries are **random**, **math**, **os**, **string**. You will not receive marks for the components if you use any other libraries apart from the mentioned library.
- Commenting on your code is an essential part of the assessment criteria. In addition to inline and function commenting on your code, you should include comments at the beginning of your program file which specify your name, Student ID, the creation date, and the last modified date of the program, as well as a high-level description of the program.
- This assignment cannot be completed in a few days and requires students to apply what we learn each week as we move closer to the submission date. Please remember to show your progress weekly to your tutor.
- You must keep up to date with the Moodle Ed Assignment 1A forum where further clarifications may be posted (this forum is to be treated as your client).
- Please be careful to ensure you do not publicly post anything which includes your reasoning, logic or any part of your work to this forum, doing so violates Monash plagiarism/ collusion rules and has significant academic penalties. Use private posts or email your allocated tutor to raise questions that may reveal part of your reasoning or solution.

6. Submission Requirements

The assignment must be submitted by **Friday, 23 September 2022, 4:30 PM (AEDT)**.

The following files are to be submitted and must exist in your Group FITGITLab server repo:

- A series of **.py** files (i.e., book.py, book_operation.py, reader.py, reader_operation.py, user.py, user_operation.py, and main.py).
 - A template, A2_student_template.zip, is available on the Moodle Assessments page. You have to use this template.
- A **userManual_group#.pdf** file.
- A PDF document of your Group Diary named as Group##_Diary.pdf (replace ## with your group number eg. Group01_Diary.pdf for Group01). A template is available on the Moodle Assessments page to provide a suggested structure for your group diary.

The above files must be compress to a .zip file named **ass2_group#.zip** and submitted via Moodle. **The files only need to be submitted by one member of the group after the group has agreed that the submission is complete and ready to be graded.**

The **.py** files must also have been pushed to your group FIT GitLab server repo with an appropriate history as you developed your solutions. Please ensure your committed comments are meaningful. **Do NOT** need to push the history of userManual_group#.pdf file to the group FIT GitLab server. Only push the final version of PDF file. **DO NOT** push .zip file.

- No submissions will be accepted via email,
- Please note we **cannot mark any work on the GitLab Server**, you need to ensure that you submit correctly via Moodle since it is only in this process that you complete the required student declaration without which work cannot be assessed.
- It is your responsibility to **ENSURE** that the submitted files are the correct files. We strongly recommend after uploading a submission, and prior to actually submitting in Moodle, that you download the submission and double-check its contents.
- Please **carefully** read the documentation under the “**Special Consideration**” and “**Assignment Task Submission**” on the Moodle Assessments page which covers things such as extensions, correct submission, and resubmission.
- **Please note, if you need to resubmit, you cannot depend on your tutors' availability, for this reason, please be VERY CAREFUL with your submission. It is strongly recommended that you submit several hours before due to avoid such issues.**
- **Marks will be deducted for any of these requirements that are not strictly complied with.**

7. Academic Integrity

Students are expected to be familiar with the [University Academic Integrity Policy](#) and are particularly reminded of the following:

Section 1.9:

Students are responsible for their own good academic practice and must:

- undertake their studies and research responsibly and with honesty and integrity;
- credit the work of others and seek permission to use that work where required;
- not plagiarise, cheat or falsify their work;
- ensure that their work is not falsified;
- not resubmit any assessment they have previously submitted, without the permission of the chief examiner; appropriately acknowledge the work of others;
- take reasonable steps to ensure that other students are unable to copy or misuse their work; and
- be aware of and comply with University regulations, policies and procedures relating to academic integrity.

and Section 2.9:

Unauthorised distribution of course-related materials: Students are not permitted to share, sell or pass on to another person or entity external to Monash:

2.9.1 any course material produced by Monash University (such as lecture slides, lecture recordings, class handouts, assessment requirements, examination questions; excluding Handbook entries) as this is a breach of the Copyright Compliance Policy and such conduct may be a copyright law infringement subject to legal action; or

2.9.2 any course-related material produced by students themselves or other students (such as class notes, past assignments), nor to receive such material, without the permission of the chief examiner. The penalties for breaches of academic misconduct include

- a zero mark for the assessment task
- a zero mark for the unit
- suspension from the course
- exclusion from the University.

Where a penalty or disciplinary action is applied, the outcome is recorded and kept for seven years, or for 15 years if the penalty was excluded.

8. Marking Guide

Your work will be marked as per the following:

- Application Functionality - 35 Marks
 - Your tutor will run your main.py to check all the basic operations listed the **section 2. Instruction**.
 - If your programme crash/ any operation won't be implemented correctly, then 0 mark for this criteria.
- Classes Implementation - 25 Marks
 - Book Class - 3 Marks
 - Class Requirements (i.e., attributes/methods) - 1 Marks
 - Code Architecture and Style - 1 Marks
 - Documentation and Commenting - 1 Marks
 - BookOperation Class - 6 Marks
 - Class Requirements (i.e., class variables/methods)- 4 Marks
 - Code Architecture and Style - 1 Marks
 - Documentation and Commenting - 1 Marks
 - User Class - 3 Marks
 - Class Requirements (i.e., attributes/methods) - 1 Marks
 - Code Architecture and Style - 1 Marks
 - Documentation and Commenting - 1 Marks
 - UserOperation Class - 5 Marks
 - Class Requirements (i.e., class variables/methods) - 3 Marks
 - Code Architecture and Style - 1 Marks
 - Documentation and Commenting - 1 Marks
 - Reader Class - 3 Marks
 - Class Requirements (i.e., attributes/methods) - 1 Marks
 - Code Architecture and Style - 1 Marks
 - Documentation and Commenting - 1 Marks
 - ReaderOperation Class - 5 Marks
 - Class Requirements (i.e., class variables/methods) - 3 Marks
 - Code Architecture and Style - 1 Marks
 - Documentation and Commenting - 1 Marks
- Main File Design - 15 Marks
 - Program Logic - 11 Marks
 - Code Architecture and Style - 2 Marks
 - Documentation and Commenting - 2 Marks
- User Manual - 5 Marks
- Interview - 10 Marks
- Peer Evaluation - 10 Marks
 - Contribution and Participation in your group:
 - Communication
 - Project Management

- Quality of contribution
 - Quantity of contribution
 - Support for the group's working environment
- as assessed by self-evaluation and group members (peer) evaluation.
- Penalty - Up to 20 marks
 - Missing submission requirements

9. Getting help

9.1. English language skills

if you don't feel confident with your English.

- Talk to English Connect: <https://www.monash.edu/english-connect>

9.2. Study skills

If you feel like you just don't have enough time to do everything you need to, maybe you just need a new approach.

- Talk to a learning skills advisor: <https://www.monash.edu/library/skills/contacts>

9.3. Things are tough right now

Everyone needs to talk to someone at some point in their life, no judgement here.

- Talk to a counsellor: <https://www.monash.edu/health/counselling/appointments>
(friendly, approachable, confidential, free)

9.4. Things in the unit don't make sense

Even if you're not quite sure what to ask about, if you're not sure you won't be alone, it's always better to ask.

- Ask in Ed: <https://edstem.org/au/courses/8843/discussion/>
- Attend a consultation:
<https://lms.monash.edu/course/view.php?id=141449§ion=21>

9.5. I don't know what I need

Everyone at Monash University is here to help you. If things are tough now they won't magically get better by themselves. Even if you don't exactly know, come and talk with us and we'll figure it out. We can either help you ourselves or at least point you in the right direction.