The Brief

Your task is to develop solutions to a series of coding exercises that will test your knowledge of the fundamental programming techniques introduced through the course of the module. These exercises complement the content delivered in class each week and allow you to put into practice what you have learned. The exercises can be found in *Programming Skills Portfolio* folder of your CodeLab I GitHub repository (located inside the *Assessment* folder*)*.

You should compile an organised repository of code during the module. When completing the assessment exercises It is recommended that you create a new project for each exercise and save these to the *Programming Skills Portfolio* folder in your repository. Each exercise should be appropriately named (e.g. *01-CodingIsCool*) so they are easy to find. You should commit changes in your repository often (e.g. after completing each exercise), use descriptive messages for these commits, and ensure you are regularly pushing your code back to GitHub. Throughout the module there are staged deadlines where specific exercises must be completed by:

Deadlines

Deadlines for the programming skills portfolio are staged throughout the module. The required exercises and dates for each deadline are listed below:

Required Exercises	Deadline Date
1. Coding is Cool 2. Simple Sums 3. Biography 4. Primitive Quiz 5. Days of the Month 6. Brute Force Attack 7. Some Counting 8. Simple Search 9. Hello 10. Is it even?	November 1, 2024

All deadlines are 11:59am, only code submitted before the deadline will be marked. Mark penalties may be applied to late submissions without prior approval of an extension. Please ensure that you prepare and submit your work in good time to allow for any issues that may arise.

Assessment feedback will be returned within 15 working days from the deadline date for submission.

Deliverables

The deliverables for this assignment are as follows:

- Your Python code in response to the programming skills portfolio test exercise(s). Code must be pushed to your GitHub repository before the corresponding deadlines listed above.
- Certificate of completion for the SoloLearn Introduction to Python course.

Submission

For each submission you should ensure your code for the required exercises has been submitted to your GitHub repository and links to these exercises submitted to via the submission portal on Minerva. Please adhere to the following method:

- Check your exercises are functioning as expected.
- Commit and push your code to your GitHub repository.
- Copy the links to each exercise required by the submission in your repository
- Paste the links into the submission portal to confirm your submission

For the second submission you should also upload your certificate of completion for the SoloLearn Introduction to Python course.

Only code pushed to your Github repository before the assessment deadline will be marked. Ensure you give yourself enough time before your final push. Guides on how to push your code to GitHub are provided on Minerva. If you are unsure ask your tutor.

Use of online sources

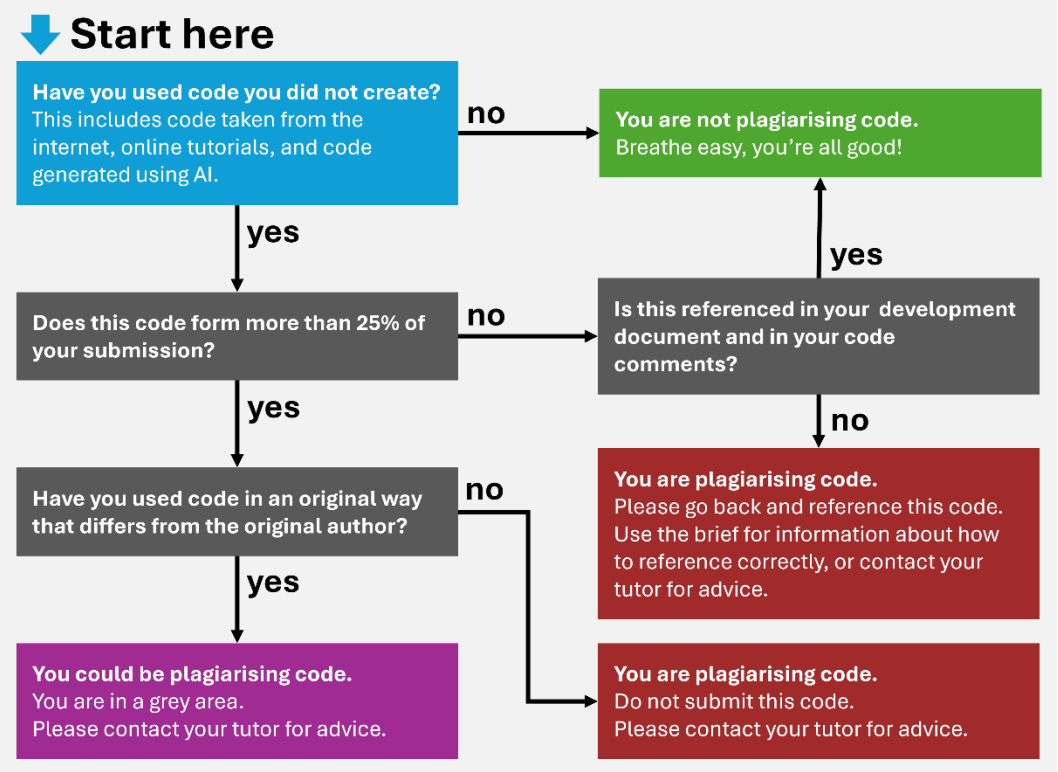
Relying on online sources, especially Generative AI, is **strongly discouraged** for this assessment. The tasks are designed to be manageable without these tools, and using them at this point in your programming journey can hinder your learning and cause misunderstandings.

Also be aware that any use of online resources to develop your code should be kept to a minimum and excessive use may be deemed academic misconduct. Given the small scope of each exercise, any use of online sources will tread this fine line of being deemed academic misconduct.

If you decide to ignore this advice and do take any code from online resources (including Generative AI tools) to support the development of your solutions you should fully acknowledge their use by referencing the source in your code comments.

- Code Comments: A reference in the code comments should be added ahead of the section of code that has been taken or heavily developed using online resources.
- *For standard online sources you should include the title of the webpage and full URL.*
- *For Generative AI tools you should include the name of the tool and the prompt provided to generate the code.*

If you are unsure of how to appropriately reference please consult your tutor. Failure to appropriately acknowledge use of online resources may result in an academic misconduct accusation. The flowchart below is offered as guidance to what is and is not acceptable. Again if you are unsure please consult your tutor.



Marking Criteria

There are 335 marks available for this assessment, split across the following criteria:

Technical Implementation - 235 marks (~70%)

Each exercise has a set number of marks available. You will be marked on the following:

- Solution successfully compiles
- Adherence to the task requirements
- Correct usage of programming techniques
- Efficiency of the code
- Implementation of advanced requirements (where relevant)
- Correct use of coding conventions (e.g. formatting, indentation, commenting)

Repository Presentation - 35 marks (~10%)

Evaluates your GitHub repository and whether you have saved your exercises to the correct location and made appropriate use of commits (e.g. per task with descriptive messages).

Extended Learning - 65 marks (~20%)

Independent study is a crucial element of university study and helps solidify and extend learning. Additional marks are available for the completion of the following online course: https://www.sololearn.com/en/learn/courses/python-introduction

The first submission will only be marked on the technical implementation (110 marks available for the first five exercises). Marks for repository presentation and extended learning will be provided after the final submission.

Criteria	Weighting	Mark Range Description	Mark Range
Technical Implementation	**90%**	Solutions are either missing or contain significant errors and do not compile. Overall limited evidence of the taught techniques are demonstrated.	0 - 19 (Low Fail)
		Poor solutions that may not fully complete the stated task or some solutions are missing. Little to no adherence to coding conventions	20 - 39 (Fail)
		Basic solutions that demonstrate a limited understanding of the programming techniques tested. All required exercises are attempted but may be incomplete or feature errors. Limited adherence to coding conventions	40 - 49 (Third)
		Fair solutions that show an understanding of the techniques introduced. Exercises are functional, although may not use the most efficient method. Coding conventions have mostly been adhered to.	50 - 59 (2:2)

		Good solutions that demonstrate a sound understanding of programming techniques introduced. Exercises are functional and may be delivered beyond their base requirements. Coding conventions have been followed with only minor slips in detail.	60 - 69 (2:1)
		Very good code that provides efficient solutions to the coding challenges. Strong understanding of programming techniques is evident. Exercises are often extended beyond their base requirements. Code is structured and commented to a very good standard.	70 - 79 (First)
		Excellent solutions that extend the coding challenges to demonstrate techniques outside the scope of the class. Code is structured and commented to a high standard.	80 - 89 (High First)
		Beyond expectations for this level of study.	90 - 100 (Outstanding)
Repository Presentation	**10%**	Limited repository organisation that does not adhere to the method specified.	0 - 19 (Low Fail)

		Poor repository organisation with unclear exercise naming or exercises saved in incorrect locations making them difficult to find. Commit messages lack description and code is likely pushed in a single commit.	20 - 39 (Fail)
		Basic repository organisation. Some exercises may be difficult to find. Commit messages are basic and would benefit from further clarity. Code is pushed on an irregular basis.	40 - 49 (Third)
		Fair repository organisation, though there may be some minor slips in presentation. Commit messages are good, though there is room for refinement. Code is pushed on a semi-regular basis	50 - 59 (2:2)
		Good repository organisation, though some exercises may need better naming conventions. Commit messages are good, though there is room for refinement. Code is pushed regularly.	60 - 69 (2:1)

		Very good repository organisation, exercises are easy to find and clearly labelled. Commit messages are clear with code frequently pushed.	70 - 79 (First)
		Excellent highly organised repository, with clear commit messages. Code is pushed very frequently.	80 - 89 (High First)
		Beyond expectations for this level of study.	90 - 100 (Outstanding)
Extended Learning	**20%**	Completion certificate for the specified course has not been provided.	0 (Fail)
		Completion certificate for the specified course has been provided.	100 (Pass)

Intended Learning Outcomes

ILO	**Assessed**

An understanding of the key features of programming including input, output, maths, sequence, iteration and repetition.	✓
The application of coding conventions to ease the review, maintenance, troubleshooting and debugging of developed software.	✓
The successful implementation of a prototype system that is driven by procedural programming techniques.	
An ability to specify applications, discuss their technical implementation and reflect critically on the results.	