1. Assessment Cover Sheet 2025/26

Module code and title	6G4Z0044: Introduction to Programming
Assessment set by	Ayah Helal
Assessment ID	2CWK50
Assessment weighting	50%
Assessment title	Digital Artefact
Type	Individual (100%)
Hand-in deadline	See Moodle
Hand-in format and mechanism	Submission is online, via Moodle

Learning outcomes being assessed

LO1: Apply the main structuring features of the chosen high level programming language(s) to solve a variety of problems

LO2: Design well-structured solutions to problems of appropriate complexity by applying standard techniques.

LO3: Verify the correctness of developed software using standard debugging and testing techniques, and remedy identified defects.

Note: it is your responsibility to make sure that your work is complete and available for marking by the deadline. Make sure that you have followed the submission instructions carefully, and your work is submitted in the correct format, using the correct hand-in mechanism (e.g., Moodle upload). If submitting via Moodle, you are advised to check your work after upload, to make sure it has uploaded properly. If submitting via OneDrive, ensure that your tutors have access to the work. **Do not alter your work after the deadline**. You should make at least one full backup copy of your work.

Penalties for late submission

The timeliness of submissions is strictly monitored and enforced.

All coursework has a late submission window of 7 calendar days, but any work submitted within the late window will be capped at 40%, unless you have an agreed extension. Work submitted after the 7-day late window will be capped at zero unless you have an agreed extension. See 'Assessment Mitigation' below for further information on extensions.

Please note that individual tutors are unable to grant any extensions to assessments.

Assessment Mitigation

If there is a valid reason why you are unable to submit your assessment by the deadline you may apply for Assessment Mitigation. There are two types of mitigation you can apply for via the module area on Moodle (in the 'Assessments' block on the right-hand side of the page):

- **Non-evidenced extension**: does **not** require you to submit evidence. It allows you to add a **short** extension to a deadline. This is not available for event-based assessments such as in-class tests, presentations, interviews, etc. You can apply for this extension during the assessment weeks, and the request must be made **before** the submission deadline. For this assessment, the non-evidenced extension is 2 days.

- **Evidenced extension**: requires you to provide independent evidence of a situation which has impacted you. Allows you to apply for a longer extension and is available for event-based assessment such as in-class test, presentations, interviews, etc. For event-based assessments, the normal outcome is that the assessment will be deferred to the summer reassessment period.

Further information about Assessment Mitigation is available on the dedicated Assessments page.

Personal Learning Plans (PLP)

If you have a Personal Learning Plan (PLP) which states you can negotiate an extended deadline, submit an evidenced extension request on the Moodle site for the module.

Plagiarism

Plagiarism is the unacknowledged representation of another person's work, or use of their ideas, as one's own. Manchester Metropolitan University takes care to detect plagiarism, employs plagiarism detection software, and imposes severe penalties, as outlined in the Student Code of Conduct and Academic integrity and misconduct. Poor referencing or submitting the wrong assignment may still be treated as plagiarism. If in doubt, seek advice from your tutor.

As part of a plagiarism check, you may be asked to attend a meeting with the Module Leader, or another member of the module delivery team, where you will be asked to explain your work (e.g. explain the code in a programming assignment). If you are called to one of these meetings, it is very important that you attend.

Use of generative AI

Permitted – with changes

⚠ The use of generative AI is permitted in this assessment, but please make sure you follow these specific instructions:

"You should remain the developer of the code. This means you should be able to explain all the code you submit. You can use generative AI to generate code snippets or suggest improvements, but the final code should be your work."

For any other uses of generative AI, you should also follow the instructions in the 'Are you allowed to use AI in assessments?' section of the AI Literacy Rise Study Pack or speak to your tutor. All submitted work must be your own original content.

If you are unable to upload your work to Moodle

If you have problems submitting your work through Moodle, you can raise a ticket with the Assessment Management Team using the Assist Portal. This must be done **before the published deadline**, else your work will be logged as a late submission. Alternatively, you can save your work into a single zip folder then upload the zip folder to your university OneDrive and submit a Word document to Moodle which includes a link to the folder. **It is your responsibility to make sure you share the OneDrive folder with the Module Leader, or it will not be possible to mark your work.**

Assessment Regulations

For further information see the Undergraduate Assessment Regulations on the Assessments and Results information pages

Formative feedback:	*Verbal feedback on your progress will be available during timetabled lab sessions.*
Summative feedback:	*This is through Moodle.*

Department of Computing and Mathematics

2. Introduction

In this assignment, you will create a digital artefact in the form of a novel piece of software. This software should function as a single tool and will be written in the Java programming language. You will develop the software iteratively throughout the semester, with guidance and support provided during the weekly lab sessions.

The piece of software that you develop will exhibit the following programming concepts:

- Console output
- User input
- Selection (If, If-else, Switch)
- Loops
- Methods
- Arrays
- Testing
- Objects

We will provide specific teaching on each of these topics, with lab exercises that will contribute to your final digital artefact. The lab exercises will not be assessed weekly, but the knowledge you learn of each technique will allow you to build up a body of code that supports your learning and development. In addition, most weeks we will provide a self-directed exercise that directly contributes to the completion of your digital artefact.

The digital artefact you produce will be based on the labs you complete. As you know, we have already been guiding you through the lab sessions to help you build this artefact. Starting this week, you will begin transforming that digital artefact into the program or idea you plan to implement.

In addition to the basic features provided throughout the lab sessions, we will also give additional marks for features beyond the scope of the brief that require additional learning and efforts outside of the sessions. We do not expect all students to complete additional features and the mark scheme (see Section 3) does not require this for a passing grade. Additional features should be in the spirit of the labs and should make use of the core java API, without external reference to additional code libraries. Specifically: the implementation of databases, the incorporation of externally produced files (e.g., datasets, model files, etc.), the use of Web-based APIs and the development of interfaces using external code libraries (e.g., web apps, or mobile apps) are all out-of-scope. Please discuss additional features with the lab tutors prior to implementing them.

The assignment should be completed individually, and submitted code will be analysed using Java-specific plagiarism detection software. You are permitted to discuss ideas in class, but you should not share code, or share your specific method of solving a problem. In addition all code must be well commented, with

bespoke comments explaining the functionality of your code. These comments will contribute to the **style** component of the final mark.

3. The Submission

All work is to be submitted via Moodle to the submission box entitled **2CWK50 - 2 Computer Based 50%**. Details on the submission date, personal extensions and feedback release are available via Moodle.

For your submission, please ensure that you have all the required elements. You will need to submit all the code for your digital artifact, a readme file, and self-assessment doc. You should ensure that all elements are submitted together as a single zip file. You may use any IDE to complete the work, but you should submit an appropriate folder containing all files used.

You must convert your folders into a zip file (this reduces the file size and creates a single submissible file). You should submit only the zip file.

Please ensure that you have read the mark scheme below (Section 4) to understand how your code will be marked and what elements are required.

4. Mark Scheme

Your code will be marked based on two factors: (1) the code **features** that you have implemented and (2) the code **style** that you have used. We will provide teaching throughout the lectures and labs on good code style.

The code features are worth 75% of your final mark and your code style is worth 25%. To attain each mark band, you must have completed the criterion in that band as well as all criteria below it. Placement within the band will be dependent on the quality of the implemented features. The marker will use some discretion where features from higher bands are correctly implemented, but features from lower bands are unimplemented, incomplete or incorrect.

Mark Band	Code Features (75%)	Code Style (25%)
86-100	1-3 additional features	Self-directed code management
75-85	Inheritance + Advanced OO	Appropriate package structure
60-74	Object-orientation	Extensive comments
50-59	Arrays, Testing	Consistent indentation
40-49	Loops, Methods	Variables named appropriately
20-39	Selection	Partial code submitted
0-19	Basic user input	Little or no code submitted

For the top band on each column you will be required to demonstrate self-directed learning and incorporate this into your assignment. To attain a mark of 86-100 in the Code Features column, you may implement features such as encapsulation, abstraction, and/or interfaces. To attain a mark of 86-100 in Code Style, you should provide evidence in your submitted files of code management features. This may

be the incorporation and appropriate use of version control software (SVN, Git, etc.) or package management software (Gradle, Maven, Ant, etc.), but may also be some other code management feature. To attain the 20 % mark for the bonus features, you would need to implement features that were not explored heavily in the course.

5. Feedback

You will receive **formative feedback** on your assignments in the weekly lab sessions. There will be on-hand support in these lab sessions to guide you through the code development process. The lab exercises that will form part of the digital artefact will be clearly specified each week. We will provide dedicated lab sessions for assignment support at the end of the semester.

At the time of submission, you will be asked to submit a mark band proposal based on the grid in section 3. The marker will adjust this mark proposal (if necessary) and give you feedback on the reasoning for this. The marker will also provide reasoning for the specific mark given within the mark bands for each category.

6. Support for the Assignment

The weekly lab exercises will give you all you need to complete a working version of the assignment. We expect every student to attempt to complete all the digital artefact lab exercises. Not all the lab exercises we give you will contribute directly towards the completion of the digital artefact, but completing all the lab exercises will allow you to learn more about the programming techniques that we are using in class and ultimately ensure that you have a correct understanding of the technique being used.

It is not unusual for students who have never written programming code to feel disheartened, or left behind. If this is you, please do two things: (1) speak to a lab tutor and ask specific questions on the things you do not understand (2) spend as much time coding as you can. Consider repeating earlier exercises from a point in the course that you felt comfortable, or seeking additional coding exercises to give you more practice. Coding is a skill and you will only improve with practice.