University of Northampton

Assessment Brief

Module title:	Problem Solving and Programming	**Module code:**	CSY1020
Assessment code/title:	PJ1	**Assessment weighting/word-limit:**	50%
Submission date:	22/01/2026 at 15:00pm	**Feedback date:**	19/02/2026
Module Leader:	Chris Rafferty	**Resit date:**	26/03/2026

Assessment Task:

Title: Assessment 2

Overview:

For this second assignment you are required to design, implement, and document a Python application that brings together the concepts taught in Weeks 8–13, including object-oriented programming, inheritance, encapsulation, abstraction, and GUI programming with Tkinter.

The assessment is divided into three components:

- Python Code – You will submit a complete program as a zip file containing two files:
    1. bookstore_core.py – all business logic, classes, and simple test cases.
    2. bookstore_gui.py – a Tkinter-based graphical interface that imports and uses the core module.
- Your code must follow Python conventions, use docstrings throughout, include input validation, and demonstrate appropriate use of inheritance and class design.

- Report – You will produce a word report in Word or PDF format. The report should include:
    - Checklist of implemented functionality– A checklist of features implemented from the task document.
    - Design – a class diagram or structured class table, plus a rationale for your design and how the GUI interacts with the core.
    - Implementation – selected code snippets (not the full program) with explanation of how key features work.
    - Reflection – a discussion of challenges faced, any changes made during development, known limitations, and possible extensions.
    - Video link – the URL to your recorded demonstration.

- Video Demonstration – You must provide a short screen recording with narration that:
    - Runs your program from start to finish.
    - Demonstrates the required features in both the core and GUI.
    - Includes a brief spoken explanation of how the system works.

Important: Failure to include a video demonstration will result in an automatic fail grade for this assessment. If the marker suspects that your work is not your own, you may be required to attend a face-to-face viva to demonstrate your code in person.

Requirements:
You must submit your work to all three submission points on NILE:

- Zip File Submission – Upload a single .zip file containing your full program:
  - bookstore_core.py – all classes, business logic, and simple test cases.
  - bookstore_gui.py – Tkinter interface that imports and uses the core module.

- Report Submission (Turnitin) – Upload your report (Word or PDF).

- Code Submission (Turnitin) – Paste the complete code for both bookstore_core.py and bookstore_gui.py into a Word document and submit to the separate Turnitin submission point. This must be plain text (not screenshots).

Failure to submit to all three submission points will result in an automatic fail grade for this assessment.
- You are responsible for checking that your submissions are complete and correct.
- Lack of preparation or consideration will impact your grade and will not be accepted as valid grounds for extensions or deferrals.

Learning Outcomes aligned to this Assessment:
On successful completion of this assessment, you will be able to:

a) Appreciate the principles and practice of analysis and design in the construction of robust, maintainable programs, which satisfy their specifications.
b) Design, write, compile, test and execute straightforward programs using a high level language; appreciate the principles of programming.
c) Appreciate the need for a professional approach to design and the importance of good documentation to the finished programs.
d) Use an appropriate programming language to construct robust, maintainable programs, which satisfy their specifications.
e) Design, write, compile, test and execute programs taking into consideration principles of programming.

f) Recognise problems and develop a strategy for problem solving.

Learning Outcomes are available on the Module Specification for the module, and on the NILE site

Academic Practice (referencing style, literature usage, AI Usage):

Students are expected to apply the professional standards and good practice outline in the module.

Assessment Guidance:

Reading List: You will find a link to your online reading list on NILE in the "About this module" section.

Use of Generative AI (Artificial Intelligence) within this Assessment:
Some uses of Generative AI may be deemed unethical in your Assessment. Further guidance on the conditions for allowable use of Generative AI will be given by the module team.

Please access the following position guidance from University of Northampton on the use of Generative AI within assessments.

AI Categories:

Category 2: GenAI can be used in an assistive role
You may make use of GenAI in your assessment in an assistive role, but you must acknowledge this appropriately.

Academic Practice support
The Skills Hub is a central repository where you will find a range of support for your study and assessments: https://skillshub.northampton.ac.uk/

Feedback:

Feedback should be received within 4 weeks
An announcement will be sent out via NILE to inform you of when feedback is available.
Instruction on Anonymity for students **Further guidance is available online**

CSY1020 AS1: Undergraduate Marking Rubric

	Excellent Work of high quality	**Good** Work of worthy quality	**Satisfactory** Work of satisfactory quality	**Pass** Work achieves requirements	**Fail** Work does not achieve requirements
Code Features (50%)	All functionality implemented flawlessly. Core and GUI integrated cleanly, validation thorough, invoices correct, and all required features fully operational.	Most functionality (bare pass to very good pass) implemented correctly, with only small logic or usability issues. GUI works reliably with minor flaws.	Functionality mainly from bare pass to good pass complete, but some features buggy or missing (e.g. invoice search, shipping toggle).	Only the bare pass requirements implemented; system runs but is incomplete.	Code does not run, ignores requirements, or irrelevant to task.
Code Quality (10%)	Clear separation between core and GUI. Code is Pythonic, well-structured, consistent naming, meaningful docstrings, minimal duplication. Strong evidence of validation and testing.	Mostly well-structured; separation generally clear. Naming and comments good, with some inconsistency. Basic testing shown.	Adequate structure but sometimes messy or inconsistent. Limited commenting/docstrings. Weak validation or test coverage.	Poorly structured, little separation of concerns, minimal documentation.	Very poor quality code, disorganized, no clear structure or documentation.
Report (35%)	Comprehensive, well-structured, professional. Includes overview, accurate design diagram or class table, rationale for design, clear code snippets, and thoughtful reflection.	Covers all required sections with good clarity. Some areas less detailed (e.g. weaker design rationale or reflection) but overall coherent.	Adequate report covering most sections, though vague or superficial. Limited testing or weak design detail.	Very limited report, poorly structured, superficial coverage.	No report submitted or report irrelevant.
Video Demonstration (5%)	Clear, professional recording showing full system running with confident spoken explanation.	Recording shows system running with some explanation, though rushed or uneven.	Recording included but unclear, incomplete, or superficial.	Very poor-quality recording, minimal execution evidence or little explanation.	No video submitted – automatic fail.