

Assessment Brief

Module title:	Problem Solving and Programming	Module code:	CSY1020
Assessment code/title:	AS1	Assessment weighting/word-limit:	50%
Submission date:	13/11/2025 at 15:00pm	Feedback date:	11/12/2025
Module Leader:	Chris Rafferty	Resit date:	19/03/2026

Assessment Task:

Title: Assignment 1

Overview:

For this first assignment you are required to design, implement, and document a set of Python programs that demonstrate your understanding of the material covered in Weeks 1 – 6.

The assessment is divided into three components:

- Python Code (50%) – You will complete four programming tasks, each saved as a separate .py file. Your code should be correct, well-structured, and fully commented with docstrings.
- Report (45%) – You will produce a single Word document containing the following for each task:
 1. Requirements – A short summary of what the program must do.
 2. Design – Pseudocode or a flowchart.
 3. Implementation – Paste your code and write an explanation of how it works.
 4. Testing – At least two screenshots of your program running, with explanations of the results.
 5. Reflection – A reflection on challenges faced and how you overcame them.
 6. The URL to the video demonstration
- Video Demonstration (5%) – You must record a short video (screen recording with audio) demonstrating your programs running. The video should:
 - Show each program being executed once.
 - Include a brief spoken explanation of what each program does.

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:

1. Zip File Submission – Upload a single .zip file containing all four Python programs (Task1.py, Task2.py, Task3.py, Task4.py).
2. Word Document Report (Turnitin Submission) – Upload your report containing requirements, pseudocode/flowcharts, pasted code with explanations, testing evidence, and reflections for each task.
3. Word Document Code Submission (Turnitin Submission) – Paste the code for Tasks 1–4 as plain text (not screenshots) into the Turnitin submission point provided.

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.
- 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
Task 1 (10%)	Program runs flawlessly with correct conversion, validates input, repeats loop, and includes meaningful comments.	Program works well but may contain minor logical issues or formatting inconsistencies, mostly well documented.	Program partially works with errors in loop or validation; documentation inconsistent and often unclear.	Program runs with significant errors, output often wrong, very limited documentation or readability present.	Code does not run or is irrelevant.
Task 2 (10%)	Fully correct analysis of sentence, accurate counts, clear outputs, and consistently documented with strong readability.	Word count and longest word mostly correct, occasional logic errors, outputs clear, generally well documented.	Word count or longest word logic flawed, average length miscalculated, code lacks consistent readability.	Program partially executes with major flaws in analysis or results; documentation weak or incomplete.	Code fails completely.
Task 3 (15%)	Stores student data in dictionary, calculates averages, outputs to file, with well-structured and commented code.	Handles student marks well with only minor errors in averages or file writing, mostly clear structure.	Dictionary implemented but averages or top mark calculations wrong; file writing incomplete or inconsistent.	Dictionary created but calculations largely incorrect; file output missing or barely functional.	No usable solution.
Task 4 (15%)	Program uses at least three well-designed functions, demonstrates decomposition, correct totals, and highly readable coding style.	Functions mostly used effectively, some errors or repetition present, code understandable and reasonably well commented.	Functions used but not effectively; program may still run, but modularity is weak or missing.	Few functions used; program logic flawed, only parts of requirements addressed; comments missing.	No working program.
Report – Content (30%)	Comprehensive: clear requirements, logical pseudocode/flowcharts, detailed explanations of code, well-chosen test cases, insightful reflection.	Covers most elements well, pseudocode/flowcharts mostly logical, explanations adequate, testing and reflection included but less detailed.	Basic attempt at elements, some missing or vague (e.g., weak testing, minimal reflection).	Very incomplete content, vague requirements, missing design/testing, reflection superficial.	Report missing or with no meaningful evidence of content.
Report – Presentation & Academic Quality (15%)	Professionally written, well-structured, fluent style, excellent formatting, correct referencing, free of language errors.	Well written and structured, some minor issues with formatting, referencing, or flow.	Adequately presented, some errors in writing or structure, formatting inconsistent.	Weakly presented, frequent errors in writing, structure hard to follow.	Poorly presented, riddled with errors, no structure or professionalism.
Video (5%)	High-quality recording shows each program running clearly with confident spoken explanation of functionality.	Video shows programs running, explanation present but brief or uneven.	Video included but superficial, unclear, or incomplete in showing tasks.	Very poor-quality video, minimal execution evidence or lacking explanation.	No video submitted