ChatGPT

in Computing Education

A Policy White Paper

Editors:
Matthew Shardlow, Annabel Latham

Contributing Authors:

Matthew Shardlow, Annabel Latham, Daniel Gooch, Kevin Waugh, Mike Richards, Mark Slaymaker, Alistair Willis, John Halloran, David Edmundson-Bird, Leanne Fitton, Anthony Kleerekoper, Ashley Williams

Foreword

Foreword by Professor Rob Aspin, Chair of the Council of Professors and Heads of Computing (CPHC), Head of Computing and Associate Dean, University of Central Lancashire.

In this paper, Latham and Shardlow present a detailed and objective review of Large Language Models (LLM), such as ChatGPT. This work draws on both current research and their own workshop activity, funded by the Council of Professors and Heads of Computing (CPHC) 2023 Special Projects Fund. They seek to identify and normalise the place of LLMs in Higher Education teaching, specifically for computing, recognising the disruptive impact such technologies are likely to have and exploring how these can be positively integrated into teaching and assessment. The paper builds on previous work, through the use of a sponsored workshop, to assess the potential threat LLMs present in Computing Education, before exploring policy considerations that can be taken forward to inform how the positive benefits of the technology can be incorporated in, HE institutions.

LLMs offer significant potential to revolutionise the ability to efficiently develop code and support associated processes such as documentation, review and information gathering that inform and reflect on the development of complex systems. For industry, where the majority of computing students will spend most of their working career, these techniques and tools are likely to become adopted and expected. These tools will be integrated into the professional software development process and will students offer a more strategic perspective on the crafting of solutions.

However, in academia where students are forming their knowledge, skills and understanding and developing expertise in their ‘craft’, such tools could be considered inappropriate and raise concerns about unfair means, plagiarism and academic misconduct. In both academia and industry there are considerable concerns over the robustness and reliability of both code and text generated by LLMs which call into question the fidelity of their output against the task they have been set. The ability to critically evaluate the output of LLMs against the requirements set and context of the problem will be essential moving forward and our obligation is to ensure our students can use such technologies safely, critically, and responsibly. This is not massively dissimilar to concerns raised when search engines became more prevalent. At that time, there were concerns that the accessibility of readily available access to ‘open’ knowledge would impinge on academic integrity for students and adversely affect the reliability and robustness of the sources used to inform technical development and critical thinking. Time has shown responsible use and the education around criticality and appropriateness of sources has turned previous disruptive technologies into well integrated practices. The research here, navigated and presented in an objective study, shows potential for the similar incorporation of LLMs to support future opportunity and evolution of the computing domain.

Executive Summary

On Monday 17 July 2023, 65 academics from 33 universities across the UK joined forces for a workshop to explore the affect of generative AI tools (such as ChatGPT) on Computing Higher Education, and to co-develop guidelines for university assessment policy. The **'ChatGPT in Computing Education: A workshop to Co-Develop Guidelines for Assessment Policy'** workshop was funded by a Council of Professors and Heads of Computing (CPHC) 2023 Special Project Fund grant, and hosted by Dr Annabel Latham and Dr Matthew Shardlow at Manchester Metropolitan University, UK.

The day started with a talk on How Large Language Models (LLMs) work, to give context for the discussions. In the first workshop task, six different modes of assessment common in computing-related higher education courses were evaluated in terms of the threat level and opportunities for redesign in light of LLMs. A group feedback session explored findings and thoughts about the six traditional assessment types and ideas for future assessments. The group noted that assessment types are already evolving away from the traditional knowledge-based assessments (factual recall, closed exams) towards skills-based assessment (coursework, practical activities). Whilst knowledge-based assessment may be threatened by LLM-based plagiarism, skills-based assessments require the learner to demonstrate a practical ability. If this is assessed through a written piece, it may be vulnerable to academic misconduct, however there are many alternative ways of assessing skills such as practical exercises and Vivas. The group examined a number of forms of written assessments (knowledge recall, critical analysis, long essay, and experiential) as well two code-based assessment formats (code production and code analysis). The first part of this report gives a summary of the findings for each of these types of assessment.

The final workshop activity was a World Cafe activity, whereby each table group was assigned a policy topic, and a host who remained with the table to lead discussions. Each group of attendees spent 10 minutes at each table discussing and shaping policy guidelines for universities. These groups discussed a wide range of policy topics including how to incorporate LLMs into HE practices, mitigation of academic misconduct, delivery strategies and appropriate timescales for adoption. These discussions, along with relevant policy points are summarised in the second section of this report.

This report is intended for policy makers in Computing HE settings and beyond. Our findings demonstrate the need for informed decisions to be made within our university settings. LLMs are here and their power is increasing. As educators, we must stay at the forefront of this curve, incorporating this technology into our teaching practices to the benefit of our students' education and future employment prospects.

Table of Contents

Terminology

AI ..... Artificial Intelligence

LLM ..... Large Language Model

GPT ..... Generative Pre-trained Transformer

HE ..... Higher Education

1. Introduction

1.1. Assessing the Outputs of Large Language Models

Large Language Models (LLMs) can provide near-instantaneous responses to queries, written in fluent prose that gives a seductive veneer of authority. The human response to their abilities is a demonstration of 'The ELIZA Effect' – the projection of human traits, including empathy and authority into text generated by computational systems (Turkle, 2007).

LLMs appear convincing and knowledgeable even though their underlying models contain no knowledge about the meaning of words and phrases – only their statistical relationships to one another in different contexts. Without *at-least* some knowledge of the workings of LLMs and an understanding of their inherent limitations, users can be tempted into unquestioningly accepting their outputs, treating these programs as infallible oracles rather than 'stochastic parrots' (Bender *et al.*, 2021) churning out unaccountable, unverified and potentially dangerous text.

1.2. Academic Literacy Skills

A key set of skills taught at university level are those allowing students to perform independent research in order to collect, categorise, assess and verify information. Whilst introductory learning, concentrating on the knowledge and comprehension stages of Bloom's model can often be satisfied from within supplied materials; higher levels of study require students to supplement provided learning with additional materials acquired through independent study.

The PROMPT criteria developed by the Open University (Open University, 2020) is an approach to evaluating information recommended to the University's higher-level students and has been adopted by other institutions. PROMPT encourages learners to consider the sources of information included in their work and to evaluate these sources in a critical manner.

Approaches such as PROMPT are compatible with the use of LLMs in assessment. Designing assessment to award credit for choosing reliable, objective and authoritative searches encourages learners to develop academic literacy skills such as questioning the source of information. Learners should not expect to receive much, if any credit, if their solutions simply use material from a LLM; rather they should draw on a range of sources and reference them accordingly as well as demonstrating engagement with these resources rather than relying on the formidable summarisation skills of this technology.

Table 1, The PROMPT Criteria (Open University, 2020).

Provenance	Is the origin of the information clear? Are the authors or publisher identifiable? Do they have credibility and authority?
Relevance	Is the information relevant to the question being asked?
Objectivity	Is the article objective or does it come with a particular agenda? Is it written in a dispassionate manner, or is it inappropriately emotive or manipulative?
Method	How was the information acquired? Was the methodology appropriate?
Presentation	Is the information clearly presented without obvious omissions and distortions?
Timeliness	Can the material be dated? Does the age of the material have an impact on its usefulness?

Any consideration that LLMs are acceptable in assessment requires that students have some awareness of the technologies used. There is no need for any deep understanding of the workings of LLMs, only the broad concept that they take a probabilistic approach to producing semantically and syntactically correct text – but have no inherent understanding of their outputs. Such information should serve to dissuade the unfortunately widespread view that LLMs are truly intelligent and can be trusted to produce meaningful responses. Students armed with the knowledge that LLMs are – by their very nature – unreliable and prone to falsification are more likely to critically appraise their outputs than those learners who trust them implicitly.

1.3. Hallucinations

Whilst the issue of 'hallucinations' has attracted a great deal of interest as a potential method of detecting improper use of LLMs, it should be remembered that falsification of references by ChatGPT is in good part a reflection of the sandbox environment within which early versions of the model functioned; the program having no access to academic libraries and other sources of factual information beyond those included in its training sets.

The 'opening up' of LLMs through the development and deployment of Application Programming Interfaces (APIs) that allow for the exchange of information between LLMs, and other online services will effectively eliminate obvious fake references and this relatively straightforward method of identifying cheating. Instead, hallucinations will take the form of distorted outputs reflecting inherent biases and omissions in the underlying training that will be much harder to detect.

We propose that all students should receive appropriate training into LLMs giving them a broad understanding of their operation, training, abilities and shortcomings in order that they can critically assess their outputs.

1.4. Is This Really New?

In the early 2000s, educators were concerned about the potential for Wikipedia to supply learners with misleading or erroneous material. The open-source nature of the online encyclopaedia led to a perception that the encyclopaedia's content would be of lower quality than conventional reference sources where material was checked and corrected by professional reviewing and editing teams. Furthermore, since Wikipedia pages could be edited by any user, content could be changed to reflect fringe points of view, omit inconvenient information, or be vandalised by malicious actions (D'Agostino, 2022).

Rather than imposing an artificial ban on Wikipedia – and by doing so removing access to a huge volume of high-quality information; most institutions have developed policies that allow for limited use of Wikipedia as an information source. One approach is to allow students to draw on Wikipedia, so long as it is correctly referenced, as well as requiring evidence from other, authoritative sources. This method allows students to continue using their existing skills at finding online content but requires them to develop new academic skills in proving the validity of the source and finding supporting evidence. This approach is supported by studies (Meseguer-Artola *et al.*, 2020; Sielicka-Barylka, 2021) showing that well-informed use of Wikipedia has a positive impact on university students due to the service having generally up-to-date material which is considered reliable and useful.

Other approaches to incorporating Wikipedia in education combine the usefulness and familiarity of the online encyclopaedia with requirements for academic credibility (University of Edinburgh, 2020) as well as exposing learners to the creation and curation processes underpinning Wikipedia by having them create their own content for the encyclopaedia as part of their studies (Soler-Adillon, Pavlovic and Freixa, 2018).

A corollary can be drawn with the use of calculators in mathematics assessment. Following their widespread availability in the mid-1970s, calculators have been incorporated into the mathematical curriculum, albeit frequently provoking controversy about their impact on basic mathematical skills. Adoption of calculators has often been followed by later restrictions on where and how they can be used.

Critics of calculators in mathematics education frequently cite learners becoming reliant on the device without needing to understand the mathematical operations themselves, and in doing so, failing to demonstrate a step-by-step working of problems. Research does not support these arguments (Cockcroft, 1982; McIntosh, 1990; Sparrow, Kershaw and Jones, 1994; Ruthven, 2009) and more recent restrictions on the use of calculators in the UK SAT assessment has been criticised by mathematical educators:

'In fact, students who use calculators regularly in lessons score as high or higher in tests, taken without calculators, compared to those who do not. On the whole, the use of calculators as an integral mathematical tool has been shown to be beneficial, particularly in the development of mathematical problem solving. It is a pity that current policy is retrogressive in this respect.' (Sellgren, 2014)

Whilst a ban on the use of LLMs in assessment might seem a reasonable approach, experience with previous disruptive technologies suggests that engaging assessment models can be developed that will utilise the powers of LLMs to create more engaging, richer learning. It may be necessary to put short-term bans on the use of LLMs for certain modules or topics until new assessment models and supporting learning materials are developed, but such is the pace of development that bans cannot be sustainable. We propose that educators perform a systematic review of the literature covering experiences of educators facing previous technological changes to learn lessons that could inform future assessment models incorporating LLMs.

1.5. All This Will Pass

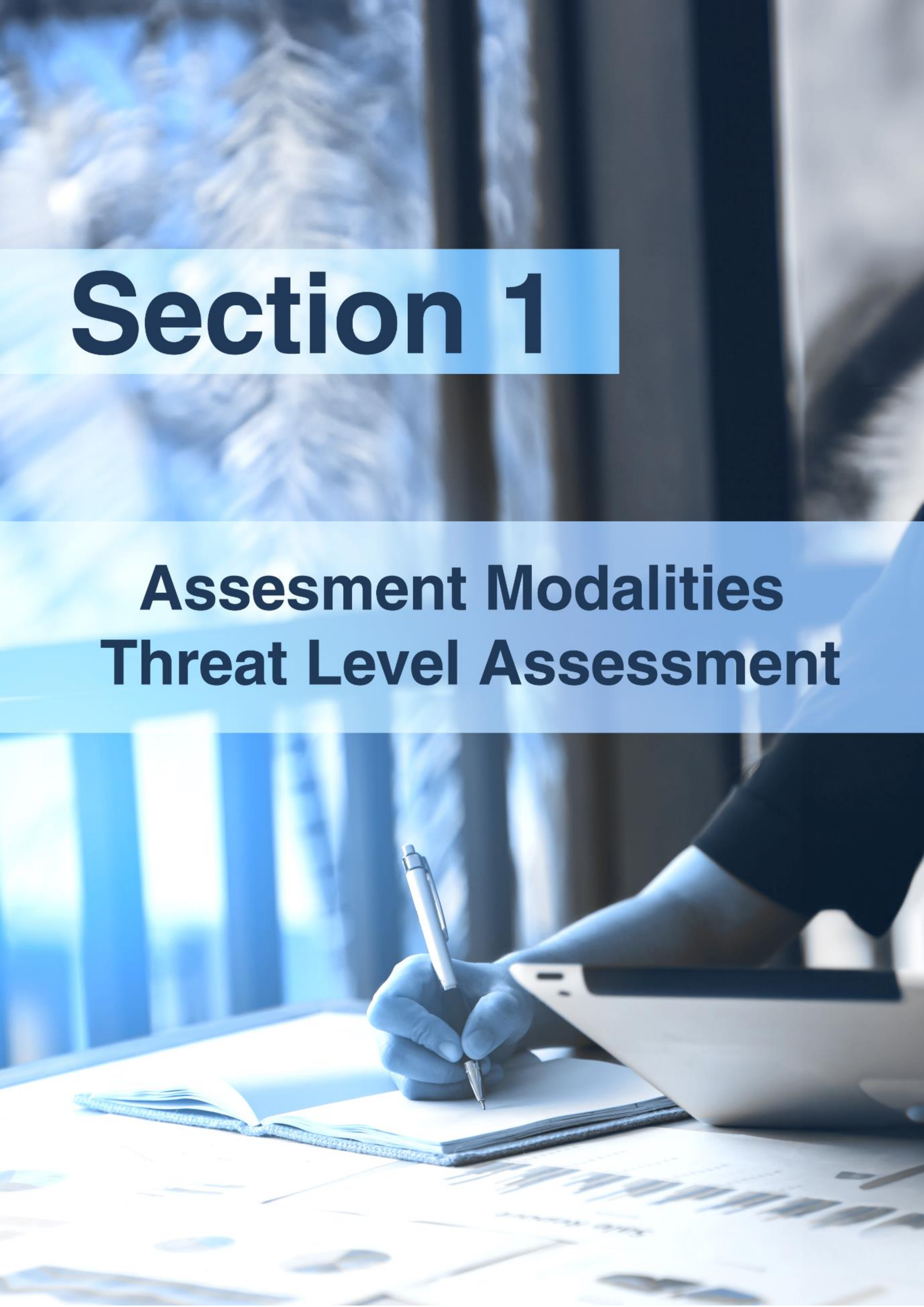
Many of the concerns raised by educators about the improper use of LLMs in assessment reflect the current state-of-the-art where these tools exist as stand-alone services. Inevitably, LLMs and other forms of generative AI will be incorporated into other forms of software.

Already, Microsoft has begun to use the GPT-4 model to provide enhanced search results in Bing (Mehdi, 2023) and will soon incorporate LLM functionality into the Microsoft Office suite of tools (Spataro, 2023). Other major software providers used in education; including Google (Google, 2023) and Adobe (Adobe Inc., 2023) have either introduced, or will shortly do so, AI tools as part of a standard workflow.

Increasingly, AI will become just another tool – perhaps even an 'invisible' function – used to create a final piece of work. Arguably, this has already happened with the use of spell, style and grammar checkers not only being routine, but actively encouraged by educators to improve the quality of student submissions. Each of these tools uses AI technology to perform its task, but their adoption has not raised great concerns amongst educators.

We suggest educators should develop assessment strategies concentrating on sound research methodologies emphasising the quality of sources and the requirement to find multiple lines of evidence rather than becoming distracted by the short-term disruption caused by LLMs.

Despite their many flaws, LLM detection tools to provide opportunities for ensuring academic integrity. Recent studies investigating the prevalence of detection - without interpreting what a detection means - found that there was a significant increase in the number of scripts flagged as containing AI-generated material (Gooch et al., 2024). The paper goes on to present an analysis of the demographic profile of flagged scripts, finding that male students, students with lower prior educational attainment, and younger students are more likely to be flagged. It remains an open question as to how effective LLM detection tools will be in supporting academic.



Section 1

Assesment Modalities
Threat Level Assessment

2. Selection of Assessment Methodologies

To better understand the impact of LLMs and particularly ChatGPT on our education practices, we have selected six forms of assessment that are typically given in a Computing Higher Education setting. These assessment types are not exclusive to the domain of the computational sciences and the findings that we give in the following sections may well be of value to readers from other domains who provide similar modes of assessment. We are also not seeking to limit the types of assessment that could possibly be given within a computing setting. Our selected modalities are an inexhaustive list, designed to cover a majority of computing assessments, prioritising those that are particularly vulnerable. For example, we do not consider video or oral-based assignments, as it is obvious that these are low-risk settings for academic misconduct through LLMs (indeed, face-to-face direct oral assessment has been used since antiquity as a robust means of assessing an individual's knowledge). We have explicitly not considered future possible means of assessing students that may arise from the widespread use of LLMs in education. Whilst we will make recommendations for the integration of Language-based Generative AI into assessment, we cannot now comment or draw conclusion on what is not yet widespread or accepted as a valid means of assessment.

When considering assessment styles, there are a number of factors that are important to take into account. Decisions around assessment modalities should be learner centric (Bloxham and Boyd, 2007), prioritising accessibility (Baker et al., 2020). The instructor must consider whether to deliver assessment material at an individual or group level (Lejk, Wyvill and Farrow, 1997). Assessment timing is key and both continuous and final assessments may be used in tandem to improve learner outcomes (Cox, Imrie and Miller, 2014). The skills to be assessed by an assessment type must be taken into consideration. As well as subject specific skills, a tutor may include assessment of general skills (Clanchy and Ballard, 1995) and (particularly for computing education) additional assessment of problem-solving tasks (Lu et al., 2022., Pérez et al., 2017). The delivery method of assessment must also be considered. Remote assessment is increasingly prevalent since the global pandemic of 2020, although left unchecked this may lead to further academic misconduct if mitigation is not given (Guangul et al., 2020), such as providing each learner with a bespoke, or watermarked problem set.

2.1. Assessment Modalities

We selected three categories of assessment type, which are further subdivided to give 6 separate assessment modalities. The three categories and their subcategories are as follows. Firstly, recall/synthesis tasks. These are further subdivided into (1) Knowledge Recall, (2) Critical Analysis and (3) Long Essay format. This form of question requires students to draw on knowledge delivered in didactic settings and produce some written response reinterpreting and delivering that knowledge at various levels of criticality and word count. Secondly, we survey code based tasks, which are subdivided into (4) Code Analysis and (5) Code Production. These elements are particular to Computing education where we require students to fluently learn one or many programming languages and to fluently produce and interpret programming code. Finally, (6) Experiential assessments rely on a student's own personal experience through appropriate reflective practice of a task, event or circumstance. We have briefly overviewed each assessment modality below and provided some comparisons. Each assessment modality is analysed in detail throughout the rest of this report, considering the threat level with respect to LLM and appropriate mitigation strategies.

2.1.1. Knowledge Recall (KR)

Knowledge recall questions seek factual answers on specific topics. Typically this would be knowledge that has been delivered as part of a prior taught session. These may be delivered as part of a take home exam, or within a piece of coursework

2.1.2. Critical Analysis (CrA)

Critical Analysis questions ask students to compare contrasting theories or viewpoints. They require the student to understand multiple perspectives and draw out differences and similarities between elements of learning. The intended responses to these questions are typically short-form and require conciseness.

2.1.3. Long Essay (LE)

Long-form essay questions provide the student with a topic and expect them to write an extended discourse drawing on factual recall, critical analysis as well as soft skills such as discourse planning, argumentation and often institution or instructor specific structuring at the discourse level (introduction, definitions, argumentation, conclusion) and independent argument level (point, evidence, explanation). The intended responses to these questions are typically long-form and require verboseness.

2.1.4. Code Analysis (CoA)

Code analysis tasks start with a piece of code and require a student to understand the workings of the code structures that are given to them. Typical questions may range from debugging a broken piece of code to suggesting strategies to refactor or commenting on code efficiency. These questions typically form part of coursework tasks or exam questions.

2.1.5. Code Production (CP)

At the heart of computer science education, we must teach our students to write their own code. Typically, this is assessed by asking the students to produce code and evaluating the quality of the results. This may range from answers to known tasks (fizzbuzz, sorting, etc.) to working within bespoke code frameworks.

2.1.6. Experiential (Ex)

Experiential tasks draw on a student's lived experience of a learning event. Rather than communicating the knowledge that has been gained, a learner must instead reflect on their personal development as a result of the act of knowledge gaining. This assessment form may be delivered within a very focussed environment, e.g. reflection on a specific classroom learning event, or may be structured as a much more open ended reflection on a event spanning a wide time period (e.g., a course of learning or placement year). Reflections are typically written in the first person.

2.2. Discussion

Regarding our three recall and synthesis modalities (Knowledge recall, Critical Analysis and Long Essay), we may observe that each of these requires a similar approach from the learner. The learner must engage in a subject, seeking and understanding new concepts from varied sources. Two key differences in the modalities are (1) the length of the required answer and (2) the skills being evaluated by the instructor. Firstly, there is a clear length difference in the required responses to the types of questions that would be posed. For knowledge recall the answer may be a single word or concept name, a bullet pointed list or a brief sentence. For Critical Analysis the student may write several sentences or paragraphs, briefly defining their terms and drawing comparisons. For long essays, A student is required to write at length. Forming critical arguments and drawing on multiple sources to convey a point. Secondly, the level of skill evaluated is also different across the modalities. Knowledge recall is exclusively testing low-level basic knowledge of a field. Critical Analysis requires a deeper understanding of each concept and appropriate lines of argumentation to distinguish them. Long essays require a student to not only regurgitate the opinions of others, but form convincing rhetoric for their own opinions on a topic.

We have provided two separate analyses for Programming code based tasks. We felt that it was important to separate out the two tasks of producing code and analysing code. These two tasks are taught together in programming labs and progress in either is dependent on the other. Whereas a student cannot learn to program without knowing how to read and interpret code and they may find that their analysis skills outstrip their production skills. Although these two skills are typically taught together and exhibit some co-dependency, they are often assessed separately. A piece of coursework may ask a student to solve a task using a given language or framework according to the course, but never ask them to interpret and extend the function of a piece of code. We have therefore included both of these modalities separately to help academics delivering higher education computing assessments to consider the impact of LLMs from both angles.

Our three categories (and six sub-categories thereof) allow us to analyse the threats to various forms of assessment in a holistic manner. Conclusions which are drawn throughout the rest of this report are intended to stimulate redevelopment of a wide set of assessments. We also hope that the findings of this report will allow academics to better understand how LLMs affect all assessment types differently. By identifying the assessment type, the specific threats and appropriate mitigation strategies, we can navigate a safe course through the LLM storm.

3. Knowledge Recall

Bloom's taxonomy is a widely adopted method of structuring learning outcomes across curricula using a series of hierarchical models developed in (Bloom and Krathwohl, 1956). Of the three, the *Cognitive* (Knowledge-Based) is perhaps the most appropriate for designing assessment Table 2. Bloom divided his *Cognitive* model into six levels; ranging from foundational 'knowledge' skills that establish basic comprehension of a topic through to advanced 'evaluation' skills. The names and order of the levels were revised by (Anderson and Krathwohl, 2001).

Table 2, Bloom's Cognitive model in original (1956) and revised (2001) form.

	Bloom (1956)	**2001 revision**
Most advanced	Evaluation	Create
	Synthesis	Evaluate
	Analysis	Analyse
	Application	Apply
	Comprehension	Understand
Least advanced	Knowledge	Remember

According to Bloom's model, assessment for introductory learners should emphasise acquiring the least advanced skills (*knowledge* and *comprehension*) in order to build a strong foundation for later assessment further in their academic journey which will develop and test higher-level skills.

Common types of assessment for foundational level modules require learners to demonstrate recollection of basic facts and terminology even if their understanding is somewhat limited. Students are gently introduced to assessment by being asked to write short answers on tightly constrained topics; to define terms introduced in their learning materials and to apply specific techniques or knowledge to often artificially simple scenarios.

This approach to learning is effective for most students and helps support learners who are unfamiliar with topics, lack confidence, struggle with language skills or those who are returning to learning. Students feel a genuine sense of accomplishment in being able to answer short questions that do not require too much depth of knowledge; this helps build confidence and encourages them to continue studying. However, the abbreviated level of knowledge required to answer such questions means that they are extremely vulnerable to mainstream, free-to-use LLMs; which through the inclusion of large factual knowledge sources during training, can generate correct, original solutions to typical introductory assignment questions without needing to have been trained on extremely specialist data sources.

(Chamberlain et al., 2011) argue that well-designed short questions can be used to assess a range of tightly focused competencies, including higher level academic skills; making it possible to identify areas requiring further development through teaching and other interventions. Eliminating short form questions may well be highly detrimental to learners.

3.1. LLM Knowledge Recall Demonstration

(Yeadon *et al.*, 2022) outlined a study of ChatGPT's performance in essay-based assessment for an undergraduate physics curriculum. ChatGPT was used to generate 10 scripts for an exam comprised of five questions requiring short-form (300 word) answers. It was found that the AI responded in a more discursive manner when given richer prompts including limiting word counts or including a known historical figure or event in the question. Not only was ChatGPT capable of quickly generating synthetic solutions, but these solutions received an average mark of 71 ± 2% with grades more tightly grouped around the average than in student populations. The researchers report that students performing in the lower third of their cohort would improve their grade if they relied on ChatGPT for their solutions.

A study at the Open University (Richards et al., 2023) blind marked a mixture of genuine student solutions and synthetic solutions generated by ChatGPT to two introductory computing modules. In this case, every single synthetic solution achieved *at least* a passing grade for an end-of-module assessment with multiple scripts achieving scores of 85% or higher. These results indicate that a student wishing to cheat by using ChatGPT to generate the entirety of their assignment solutions could expect to pass end assessments for introductory modules without any knowledge of the topic.

We propose that there are substantial educational benefits to short-form assessment in establishing basic skills and knowledge; however, given the ability of LLMs to provide competent solutions to such questions, this form of assessment should not count towards classification.

To illustrate the ability of generative AI tools to set and answer basic knowledge recall questions, the following two screenshots show Bard producing questions on interaction design, and ChatGPT 3.5 producing questions on SQL:

Write four questions you would ask a first year University student in an exam about the evaluation aspects of interaction design? Please answer the question you just set.

View other drafts

Sure, here are four questions I would ask a first year university student in an exam about the evaluation aspects of interaction design:

1. **What are the different types of evaluation methods used in interaction design?**
   - **Heuristic evaluation:** A method of evaluating the design of a product or service by comparing it to a set of usability heuristics.
   - **Usability testing:** A method of evaluating the usability of a product or service by testing it with users.
   - **Eye tracking:** A method of evaluating the user's attention to different elements of a design.
   - **Heat maps:** A method of visualizing the user's mouse movements and clicks on a design.
2. **What are the benefits of using evaluation methods in interaction design?**
   - Evaluation methods can help to identify usability problems in a design.
   - Evaluation methods can help to improve the usability of a design.
   - Evaluation methods can help to ensure that a design meets the needs of the users.
3. **What are the challenges of using evaluation methods in interaction design?**
   - Evaluation methods can be time-consuming and expensive.
   - It can be difficult to get users to participate in evaluation studies.
   - It can be difficult to interpret the results of evaluation studies.
4. **What are the ethical considerations of using evaluation methods in interaction design?**
   - It is important to obtain informed consent from users before participating in an evaluation study.
   - It is important to protect the privacy of users' data.
   - It is important to be respectful of users' time and effort.

I hope these questions are helpful!

KE What questions would you ask a first year University student to test their knowledge of SQL? Please answer the question you just set.

Sure, here's a question you could ask a first-year university student to test their knowledge of SQL:

2. **Write a SQL query to find all students who have a higher average grade than the average grade of the Computer Science majors.**
   Assuming you have a table named "Students" with columns "Name," "Major," and "Grade," and you want to compare average grades:

```sql
SELECT s.Name
FROM Students s
WHERE s.Grade > (SELECT AVG(Grade) FROM Students WHERE Major = 'Computer Sc
```

1. **Explain the concept of database normalization. Why is it important?**
   Database normalization is the process of organizing a database to reduce data redundancy and improve data integrity. It involves breaking down tables into smaller, related tables and applying specific rules to ensure data is stored efficiently and without anomalies. Normalization helps maintain data accuracy, reduces storage space, and simplifies data manipulation.
2. **What is a primary key? How does it relate to the concept of uniqueness in a database?**
   A primary key is a unique identifier for a row in a database table. It ensures that each row in the table is uniquely identifiable. It enforces the concept of uniqueness by preventing duplicate values in the primary key column(s). This is crucial for maintaining data integrity and enabling efficient data retrieval.

Such illustrative images highlight how the use of short-answer questions as summative assessment are now problematic, while there are possibilities in terms of supporting personalised formative skills development and self-checking.

3.2. Moving Forward on Knowledge Recall

Returning to Bloom's taxonomy (Table 1); higher levels of study develop the higher-level skills categorised as Analysis, Synthesis and Evaluation. Here, students are expected to develop and demonstrate proficiencies such as comparing and contrasting disparate sources of information; applying existing knowledge to develop new information and performing critical reviews of information to make a judgement. Whilst LLMs *can* answer assessment questions designed to test these skills, their proficiency is greatly reduced compared to earlier levels of assessment.

The Open University study mentioned above examined ChatGPT's performance on one postgraduate computing module. Not only did this assessment obviously require deeper knowledge of topics than the undergraduate modules, but there is an expectation on students to demonstrate greater proficiency of high-level learning skills such as synthesis and application of knowledge. A key aspect of the module's assessment is the expectation that students will apply (and reference) the theoretical knowledge found in the module materials to an organisation of their choosing as part of a module-long case study. This combination of deep subject-specific knowledge, high-level learning skills and intense personalisation of solutions proved challenging for ChatGPT which produced unsatisfactory superficial and generic answers to the assessment. None of the ChatGPT-generated solutions in the experiment received a passing grade, and the marker referred to the answers being formulaic, superficial, and blandly written.

There are assessment modalities - such as in-class quizzes, proctored examinations, oral Vivas or portfolios and learning journeys - where there remains space for the use of knowledge recall skills, particularly if integrated with elements of critical analysis. Some questions which make explicit use of LLMs and that bridge the gap between knowledge recall and critical analysis include:

1. "Can you use this tool" - use a MT generated prompt in a tool which is known to generate an incorrect, incomplete, or poor response which needs the student to review and improve the original answer. This could be done live through a tool, or in response to a MT-generated artefact from a tool.
2. Compare and contrast the response to two different prompt formulations.
3. If the module uses a particular framing of content, use prompt to generate generic text through a gen-AI tool, and ask the student to convert into the module framing.
4. Reflect on the capabilities and limitations of tool X in the context of [module]
5. Integrate use of a gen-AI tool into a learning journal, and document uses.
6. Generate assessment questions using tool X and ask students to critique them.
7. Ask students to write assessment questions which tool X cannot answer.
8. Student writes an answer -> use a gen-AI to critique answer -> reflect on the critique.
9. Simplify the explanation of X given in the module with support from tool Y. Note student vs tool changes.

4. Critical Analysis

Critical Analysis questions ask students to compare contrasting theories or viewpoints. They require the student to understand multiple perspectives and draw out differences and similarities between elements of learning. In this case, we would expect the answers to these questions to be short form, i.e., 200-300 words.

For computer science, critical analysis abilities are, as with other disciplines, essential, but also differentiated / specific to computer science. The sorts of abilities we want to promote include, for example:

- Explaining code clearly and concisely.
- Being aware of options and able to explain decisions and justify the approach taken: this could be choice of algorithms, AI or machine learning technique, data model, or software development approach or pattern.
- Being aware and able to apply relevant social, legal, ethical and professional issues for a given problem space / piece of software.
- Situating work against current state of the art, able to explain value and contribution.
- Collegial criticality: the ability to work with others, helping shape and successfully participate in (for example) complex software development project.
- Understanding what can and can’t be computed, and why.

What follows is a series of critical analysis questions we asked ChatGPT together with its responses, to gauge how far and in what ways LLMs might plausibly emulate these skills.

4.1. Question 1

Compare the waterfall methodology with the agile methodology for software engineering.

Result Produced by LLM (in Italics):

The Waterfall methodology and the Agile methodology are two distinct approaches to software engineering project management, each with its own set of principles, practices, and advantages. Here's a comparison of the two methodologies:

Approach to Development:

- ***Waterfall:*** *Waterfall is a sequential and linear approach. It follows a predefined sequence of phases, such as requirements, design, implementation, testing, deployment, and maintenance. Each phase must be completed before moving on to the next.*
- ***Agile:*** *Agile is an iterative and incremental approach. It breaks the project into small increments or iterations, typically 2-4 weeks long, and focuses on delivering a working product at the end of each iteration. It allows for flexibility and adjustment throughout the project.*

Flexibility:

- ***Waterfall:*** *Waterfall is less flexible when it comes to changing requirements. Once a phase is completed, it's challenging and costly to make significant changes.*
- ***Agile:*** *Agile is highly adaptable. It welcomes changing requirements, even late in the development process. Changes can be incorporated in the next iteration.*

Client Involvement:

- ***Waterfall:*** *In Waterfall, client involvement is primarily at the beginning and the end of the project. Clients usually see the final product after a lengthy development cycle.*
- ***Agile:*** *Agile encourages regular client involvement. Clients are part of the development team and provide feedback throughout the project, ensuring that the product meets their evolving needs.*

Testing:

- ***Waterfall:*** *Testing occurs at the end of the development cycle in Waterfall. This can lead to the discovery of major issues late in the project.*
- ***Agile:*** *Testing is integrated into the development process in Agile, with continuous testing and quality assurance throughout each iteration.*

Risk Management:

- ***Waterfall:*** *Waterfall tends to have higher project risk because issues may not be identified until late in the process.*
- ***Agile:*** *Agile reduces risk through regular inspection, adaptation, and client feedback, which helps in identifying and addressing problems early.*

Documentation:

- ***Waterfall:*** *Waterfall requires extensive documentation at each phase of development, which can be time-consuming.*
- ***Agile:*** *Agile focuses on working software over comprehensive documentation, though it still maintains necessary documentation.*

Project Control:

- ***Waterfall:*** *Waterfall provides more control over project scope and requirements upfront but less flexibility as the project progresses.*
- ***Agile:*** *Agile offers more flexibility but requires strong project management to control scope and ensure the project stays on track.*

Delivery Time:

- ***Waterfall:*** *Waterfall projects tend to have longer delivery times because the entire project is completed before delivery.*
- ***Agile:*** *Agile allows for quicker delivery of incremental features or releases, which can be beneficial in rapidly changing markets.*

In summary, the choice between Waterfall and Agile depends on the specific project, its requirements, and the client's preferences. Waterfall is suitable for projects with well-defined and stable requirements, while Agile is more appropriate when flexibility, *client involvement, and adaptability is essential. Some teams also adopt hybrid approaches that combine elements of both methodologies to suit their project needs.*

4.1.1. Analysis

This is a highly plausible answer neatly built around a series of criteria under which each approach is compared and evaluated. The selection of these criteria (and not others) is not explained, but if produced by a student the approach would be recognised as sophisticated. It could be criticised for not discussing some specific examples of software development to help ground and illustrate the points made, but this would also challenge many students.

4.2. Question 2

Analyse the efficiency and scalability of bubble sort and merge sort. Discuss their trade-offs in terms of time complexity and space complexity.

Result Produced by LLM (in Italics):

Bubble Sort and Merge Sort are two sorting algorithms that differ significantly in terms of efficiency, scalability, and trade-offs in time and space complexity.

Bubble Sort: *Bubble Sort is a simple comparison-based sorting algorithm that repeatedly steps through the list, compares adjacent elements, and swaps them if they are in the wrong order. The process continues until no more swaps are needed.*

Efficiency: *Bubble Sort has a time complexity of O(n^2), where 'n' is the number of elements to be sorted. This makes it highly inefficient for large datasets because the number of comparisons and swaps increases quadratically with the input size.*

Scalability: *Bubble Sort is not scalable for large datasets. Its performance degrades rapidly as the dataset size increases.*

- ***Trade-offs:***
  - ***Time Complexity:*** *The main trade-off with Bubble Sort is its time complexity. It is suitable for small datasets but becomes impractical for larger ones due to its quadratic time complexity.*
  - ***Space Complexity:*** *Bubble Sort has a space complexity of O(1), as it sorts the elements in place without requiring additional memory.*

Merge Sort: *Merge Sort is a divide-and-conquer sorting algorithm that divides the unsorted list into 'n' subsists, each containing one element, and then repeatedly merges subsists to produce new sorted subsists until there is only one subsist remaining.*

- ***Efficiency:*** *Merge Sort has a time complexity of O (n log n), which makes it much more efficient than Bubble Sort, especially for large datasets. It is a stable and reliable sorting algorithm.*

- ***Scalability:*** *Merge Sort is highly scalable. Its performance remains relatively consistent even as the dataset size grows, thanks to its efficient divide-and-conquer approach.*

- ***Trade-offs:***
  - ***Time Complexity:*** *Merge Sort's time complexity makes it a strong choice for larger datasets but might be overkill for very small lists due to the overhead of dividing and merging.*
  - ***Space Complexity:*** *Merge Sort has a space complexity of O(n) due to the need for auxiliary storage (an additional array) during the merging process. This can be a drawback for sorting large datasets with limited memory.*

Comparison:

- *Bubble Sort is easy to understand and implement, making it suitable for educational purposes or small datasets where its simplicity doesn't result in excessive execution time.*
- *Merge Sort, on the other hand, is a much more efficient and scalable sorting algorithm, making it the preferred choice for sorting large datasets in real-world applications.*
- *The primary trade-off between the two algorithms is that Bubble Sort sacrifices time complexity for simplicity and minimal space usage, while Merge Sort achieves better time complexity at the cost of higher space usage.*

In summary, Bubble Sort is inefficient and not scalable for large datasets, while Merge Sort is efficient and scalable but consumes more memory. The choice between these two algorithms depends on the specific requirements of the sorting task, particularly the size of the dataset and available memory.

4.2.1. Analysis

Again, this answer is built around criteria which support and scaffold the comparison, facilitating critical analysis on space and time complexity as well as efficiency and scalability. These are classic criteria for doing such a critical analysis.

However, while the points made are accurate, the critical analysis feels abstract. Neither sort is explained. The analysis could be improved through the use of worked examples, perhaps as diagrams, to show how each sort processes the same input sequence of some plausible size.

Such examples might also help establish why the space and time characteristics apply in the way that they do - in particular, that merge sort is usually implemented recursively so needs more memory (and if iteratively needs copies) - and why merge sort is preferred for large sorts.

Similarly, while Big O analysis is used, the time difference between n and log(n) isn’t explained or illustrated.

Some nuance is missing, for example what implementation of bubble sort is being considered - with or without a rightward lock, for example.

However, this is still a very good answer.

4.3. Question 3

Highlight relevant similarities in code features between the object-oriented programming paradigm and the procedural programming paradigm.

Result Produced by LLM (in Italics):

Object-oriented programming (OOP) and procedural programming are two different programming paradigms, but they do share some common code features. Here are some relevant similarities:

- ***Variables and Data Types:*** *Both paradigms use variables to store data and support common data types such as integers, floating-point numbers, characters, and arrays. These data types are often available in both OOP and procedural languages.*
- ***Control Structures:*** *Both paradigms utilize control structures like loops (e.g., for, while, do-while) and conditional statements (e.g., if, else, switch) to control the flow of a program. These structures are fundamental in both approaches.*
- ***Functions/Methods:*** *In procedural programming, functions or procedures are used to encapsulate a set of operations that can be reused. In OOP, methods are similar to functions in that they encapsulate behavior, and both paradigms rely on modularization to organize code.*
- ***Parameter Passing:*** *Both paradigms allow passing parameters to functions or methods. Parameters can be passed by value or by reference, depending on the programming language.*
- ***Comments and Documentation:*** *Comments and documentation are important in both paradigms to improve code readability and maintainability. Developers in both OOP and procedural programming use comments to explain the purpose of code segments.*
- ***Error Handling:*** *Error handling mechanisms, such as try-catch blocks or error codes, are used in both paradigms to handle exceptions and errors gracefully.*
- ***Libraries and Modules:*** *Both OOP and procedural languages support the use of libraries or modules to organize and reuse code. These libraries often contain functions, classes, or procedures that can be imported and used in the main program.*
- ***Input and Output:*** *Both paradigms provide mechanisms for input (e.g., reading from the keyboard or a file) and output (e.g., printing to the console or writing to a file).*
- ***Scope and Visibility:*** *Both paradigms define rules for variable scope and visibility. In both cases, variables can be local to a function or global, and access control modifiers (public, private, protected) are used to restrict access to certain parts of the code.*
- ***Modularity:*** *Both paradigms emphasize code modularity and the organization of code into manageable, reusable units. In procedural programming, this is achieved through functions, while in OOP, it's achieved through classes and objects.*

It's important to note that while there are similarities in code features, the main difference between the two paradigms lies in their approach to organizing and structuring code. Procedural programming focuses on procedures and functions, while OOP emphasizes the use of objects and classes for modeling real-world entities and their interactions. The choice between these paradigms depends on the problem domain and the goals of the software being developed.

4.3.1. Analysis

Again, this is a plausible answer built around relevant criteria which provide structure for comparison. And again, the issues are similar: the discussion misses some basic explanation, as well as examples. In particular, the explanation of what procedural and OOP programming are seems perfunctory and appears in the wrong place - at the end. The discussion seems broad and high level and perhaps if submitted by student feedback might suggest using fewer criteria, going into greater depth.

4.4. Question 4

Evaluate the role of machine learning algorithms in predicting user behaviour and discuss the ethical considerations and potential biases that may arise from their implementation.

Result Produced by LLM (in Italics):

Machine learning algorithms play a significant role in predicting user behavior across various applications, such as e-commerce, social media, advertising, and recommendation systems. These algorithms analyze user data to make predictions about future actions or preferences. While they offer numerous benefits, their implementation raises ethical considerations and potential biases that need careful attention.

Role of Machine Learning Algorithms in Predicting User Behavior:

- ***Personalization:*** *Machine learning algorithms analyze historical user data to personalize user experiences. For instance, they can recommend products, content, or services tailored to individual preferences, leading to higher user engagement and satisfaction.*
- ***Targeted Advertising:*** *Advertisers use machine learning to predict user interests and display relevant ads, improving ad relevance and increasing click-through rates.*
- ***Fraud Detection:*** *Machine learning models can detect unusual patterns in user behavior to identify potential fraud or security threats, enhancing the security of online platforms.*
- ***Healthcare:*** *Predictive analytics can assist in diagnosing and predicting medical conditions based on patient data, allowing for early intervention and better patient care.*

Ethical Considerations and Potential Biases:

- ***Bias in Data:*** *Machine learning models rely on historical data, and if the data used for training contains biases, the models can perpetuate those biases. For example, biased training data may lead to discriminatory predictions, such as biased recommendations or unfair treatment.*
- ***Privacy Concerns:*** *Predicting user behavior often involves collecting and analyzing large amounts of personal data. Ensuring user privacy and obtaining informed consent are crucial ethical considerations.*

- ***Transparency and Explain ability:*** *Many machine learning algorithms, particularly deep learning models, can be complex and difficult to interpret. This lack of transparency raises concerns about accountability and understanding how decisions are made.*
- ***Algorithmic Fairness:*** *Ensuring fairness in predictions is vital. Algorithms should not discriminate against certain user groups based on attributes such as race, gender, or age. Unintended biases must be identified and mitigated.*
- ***Filter Bubbles and Echo Chambers:*** *Personalization algorithms can create filter bubbles, where users are only exposed to content that reinforces their existing beliefs. This can contribute to the spread of misinformation and limit exposure to diverse perspectives.*
- ***User Manipulation:*** *Predictive algorithms are used to optimize user engagement, which can lead to addictive design patterns that manipulate user behavior, raising ethical concerns about user agency and well-being.*
- ***Security* Risks:** Machine learning models used for security may produce false positives or false negatives, potentially impacting individuals' security and privacy.
- ***Mitigation Strategies:*** *To address these ethical considerations and potential biases:*
- ***Data Quality:*** *Ensure high-quality training data by identifying and mitigating biases in data sources.*
- ***Diversity in Teams:*** *Create diverse development teams to consider various perspectives and minimize bias in algorithm design.*
- ***Bias Detection:*** *Regularly audit and evaluate algorithms for bias using fairness metrics and tools.*
- ***Transparency and Explainability:*** *Develop models that are more interpretable, and provide explanations for model predictions.*
- ***User Control:*** *Give users control over their data and the ability to customize their experiences.*
- ***Ethical Guidelines:*** *Adhere to ethical guidelines and regulations, such as GDPR and HIPAA, to protect user privacy and data.*
- ***Education and Awareness:*** *Educate users about how predictive algorithms work and their implications.*

In summary, while machine learning algorithms are powerful tools for predicting user behavior, their use must be guided by strong ethical principles to mitigate potential biases and ensure fairness, transparency, and respect for user privacy and autonomy.

4.4.1. Analysis

Once again, this response has a by-now-familiar structure: the use of multiple criteria and points related to the question made under each. And again, the discussion seems high level and somewhat abstract, lacking depth, as well as examples.

4.5. Conclusion

While none of these critical analysis questions has been badly handled by ChatGPT, the responses are similar, with similar advantages - and similar problems.

The advantages include:

- Strong structure with comparison and evaluation built around relevant criteria.
- Generally accurate content
- Readability and conciseness
- Broad ranging

The problems / issues include:

- Lack of examples, especially explanations and worked examples of named algorithms (Question 2); examples of the named problems, and the types of machine learning involved (Question 4) or code showing how the same thing could be implemented in procedural or OOP (Question 2).
- Breadth rather than depth
- Lack of referencing
- Lack of deep technical insight which examples and code discussion help establish.

The main issue with the ChatGPT responses in this section is perhaps to do with level of abstraction. Excellent critical analysis will move between levels of abstraction, making general claims of the type provided in these ChatGPT response, and then grounding them in specifics. This can be done in reverse, too: specific examples can give rise to a general claim. There is a sense that these ChatGPT responses are not close enough to the issues they discuss, but trawl for commonality between the vast numbers of sources the technology uses, rather than the specificity that comes from zooming into the particular.

When student work of this type is submitted, educators are likely to want to see grounding examples, and code (where relevant), to underpin and justify the general claims and points ChatGPT is good at generating. As others have pointed out, ChatGPT could work as an extra source to be used against others when researching and learning, but on the evidence given here, cannot substitute for the level of critical skills we want to see in our students.

5. Long Form Essay Questions

Long-form essay questions provide the student with a topic and expect them to write an extended discourse drawing on factual recall, critical analysis as well as soft skills such as discourse planning, argumentation and often institution or instructor specific structuring at the discourse level (introduction, definitions, argumentation, conclusion) and independent argument level (point, evidence, explanation). The intended responses to these questions are typically long-form and require verboseness.

For the purposes of this chapter, we will consider long-form essays to be those that require a word limit of 3,000+ words. This requires a student to form a rhetorical argument on the topic and provide their own interpretation of the facts, i.e., beyond basic knowledge generation.

5.1. Analysis

Sample questions for Long-form essays in the computing domain are as follows:

1. How will current advances in the field of artificial intelligence shape the future of technology?
2. What are the key principles to effective collaboration in a software development environment?
3. What role does blockchain technology play in contemporary applications?
4. How do virtual reality and augmented reality enhance user experiences?
5. How does the Internet of Things (IoT) impact the development of smart cities?
6. How have computer networks evolved from LAN to 5G?
7. What are the challenges and opportunities in developing mobile applications?
8. How does machine learning contribute to healthcare and medicine?

A sample of prompting ChatGPT for a response to the first question is given below:

MA

Write a 3000 word essay on the following topic:

How will current advances in the field of artificial intelligence shape the future of technology?

Title: The Future of Technology: Shaping Tomorrow with Artificial Intelligence

Abstract:
Artificial intelligence (AI) has rapidly evolved in recent years, becoming a transformative force across various industries. This essay explores the current advances in the field of artificial intelligence and their profound implications for shaping the future of technology. We delve into the impact of AI on various sectors, including healthcare, finance, education, and transportation, and discuss ethical and societal considerations that arise with AI's increasing integration. Furthermore, we examine the potential challenges and opportunities that AI presents, with a focus on innovation, job displacement, and AI's role in solving global challenges. By understanding the trajectory of AI's development, we can better prepare for the technological landscape that awaits us.

Introduction:

The field of artificial intelligence (AI) has made significant strides in recent years, transforming how we live, work, and interact with technology. From virtual personal assistants like Siri and Alexa to self-driving cars and advanced medical diagnostics, AI has become a pervasive force in our lives. These current advances in AI have the potential to shape the future of technology in unprecedented ways, impacting nearly every aspect of our society.

In this essay, we will explore the multifaceted influence of AI on the future of technology. We will examine its impact on various sectors, discuss ethical considerations, and analyze both the challenges and opportunities it presents. As we embark on this journey through the AI landscape, it becomes evident that AI is not just a technology but a catalyst for profound societal and technological change.

I. Current Advances in Artificial Intelligence:

Before delving into how current advances in AI will shape the future of technology, it is essential to understand the key components and developments in the field of AI. Over the past few years, several breakthroughs have driven AI's rapid growth, including:

1. **Deep Learning**: Deep neural networks have revolutionized AI by enabling machines to process vast amounts of data and learn complex patterns. This breakthrough has empowered applications such as image and speech recognition, natural language processing, and recommendation systems.
2. **Big Data**: The availability of large datasets has been instrumental in training AI models effectively. Advances in data collection, storage, and processing have propelled AI to new heights, allowing for more accurate predictions and insights.
3. **Compute Power**: The development of powerful GPUs (Graphics Processing Units) and TPUs (Tensor Processing Units) has accelerated AI training and inference processes. This increase in compute power has made it possible to train deep neural networks with billions of parameters.
4. **Transfer Learning**: Transfer learning techniques enable AI models to leverage knowledge learned in one domain to perform better in another. This approach has significantly reduced the need for massive labeled datasets and accelerated model development.
5. **Generative Models**: Generative adversarial networks (GANs) and transformers have enabled AI systems to create content, including images, text, and music, that is often indistinguishable from human-generated content.

B. Opportunities:

1. **Innovation**: AI fosters innovation across industries, enabling the development of new products, services, and business models.
2. **Global Challenges**: AI can be a powerful tool in addressing global challenges like climate change, disease outbreaks, and disaster response.
3. **Enhanced Decision-Making**: AI systems can augment human decision-making by providing data-driven insights and predictions.

V. Conclusion:

The current advances in the field of artificial intelligence have the potential to reshape the future of technology in profound ways. From healthcare to finance, education, and transportation, AI is becoming a driving force across various sectors. While ethical considerations and challenges exist, the opportunities for innovation and positive societal impact are immense.

To harness the full potential of AI and navigate its complexities, it is crucial for society to foster collaboration between technologists, policymakers, and ethicists. Developing responsible AI, ensuring transparency, addressing biases, and providing lifelong learning opportunities will be essential in shaping a future where AI augments human capabilities and improves the overall quality of life. The future of technology is undeniably intertwined with the continued advancement of artificial intelligence, and our actions today will determine whether this future is one of progress and prosperity for all.

The output of the model appears to give a reasonable attempt at answering the question. It generates a new essay title 'The Future of Technology: Shaping Tomorrow with Artificial Intelligence', although this was not specified as a requirement. It then continues to give an introductory paragraph before continuing with a list of areas that are to be explored as follows:

I. Current Advances in Artificial Intelligence

II. Impact on Various Sectors

A. Healthcare

B. Finance:

C. Education

D. Transportation

III. Ethical Considerations

IV. Challenges and Opportunities

A. Challenges

B. Opportunities

V. Conclusion

Each point is completed with further bulleted lists of relevant topics. The essay length is 1,251 words (not all shown) - despite the prompt requesting 3,000. Whilst the subject matter of the essay is passable (it discusses a broad range of relevant topics and relates them back to the original question), the formatting and style of the response leaves a lot to be desired. The response is informative, but not discursive. It does not provide critical evaluation of the subject matter nor an appropriate rhetorical form to the argumentation. There is no referencing, and no attempt is made to link the work to external sources, or to give sensible critique of other's opinions. If this were submitted for examination, it would perform very poorly. A student may wish to take this as a preliminary outline for an essay, however significant work would be required to expand upon the given points, link them to appropriate sources and combine the bullet points to give a free-flowing discourse.

5.2. Key Threats to Assessment Type

- LLMs produce a viable essay prototype that can be submitted as is.
- A student may further adapt the output of an LLM, obfuscating it's use.
- Further prompting may be used to generate specific sections of an essay.
- Students may request essay answers using a given style as promoted in university.

5.3. Mitigation Strategies

- Require students to submit essay plans, as well as the final essay.
- Allow students to use LLMs to plan essays but be clear on evaluating documentation of essay plan as well as essay.
- Enforce a specific structure on the students (e.g., Introduction, Topic, Evidence, Comparison, Conclusion) which the Language Model will not provide.
- Set essay length at a limit beyond the output range of an AI chatbot. Chatbots typically provide a maximum of 1,000 words as a response.
- Focus on hard to measure outcomes. e.g., problem-solving critical thinking Rolling assessment.
- Requiring stylised references and placing a focus on the appropriate use and evaluation of external sources. References must be checked for correctness.

6. Code Analysis

Code analysis tasks start with a piece of code and requires a student to understand the workings of the code structures that are given to them. Typical questions may range from debugging a broken piece of code to suggesting strategies to refactor or commenting on code efficiency. These questions typically form part of coursework tasks or exam questions. They are designed to test understanding of:

- syntax (the characters, words, statements, and expressions that compose a program).
- semantics (the meaning of these; what the program does);
- correctness (does the program do what it is designed to do; does it work for all relevant cases);
- efficiency and conciseness (does the program run according to expected time and space; is there redundancy that creates overheads);
- code structure and integrity is the code free of redundancy and duplication; is it modular; can it be reused; how far will it generalise to relevant instances of the problem it is designed to solve.

The ability to analyse code is often assumed to depend on the ability to write it. Hence, many introductory programming modules and courses begin by teaching programming concepts including (for procedural programming) variables, datatypes, sequence, selection and conditions, iteration; and / or, for object-oriented programming, objects, classes, instances, inheritance, methods, mutators and so on. Students are then given exercises, tasks, or challenges where they need to implement concepts in a given language to create a program for some purpose (could do with some refs).

With the advent of LLMs, including ChatGPT, the danger is that such tasks can be delegated to the LLM, which may return acceptable solutions without the student necessarily understanding either the question or its answer: in that case, the ability to analyse code may not emerge.

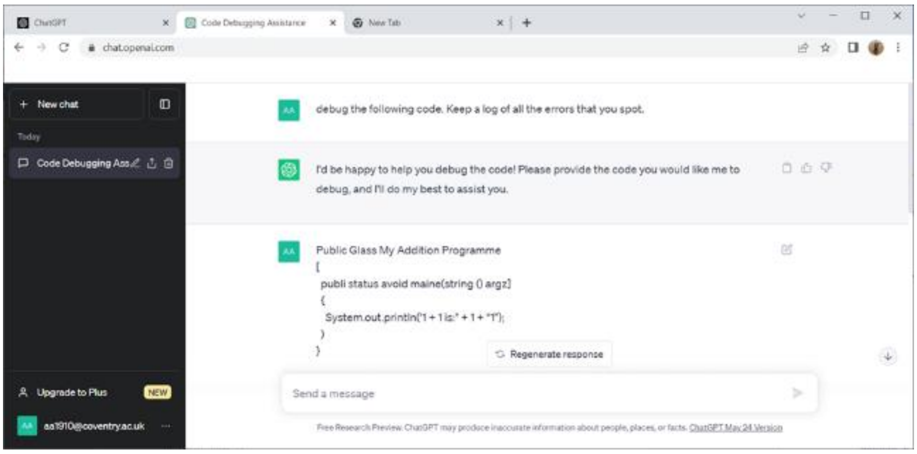
What follows is some samples of code analysis tasks which might be set for students to solve, which have been entered into ChatGPT, with the solutions ChatGPT then generates. We discuss the plausibility and usefulness of these solutions in terms the kinds of programming ability we want our students to develop.

Analysis - Sample Questions

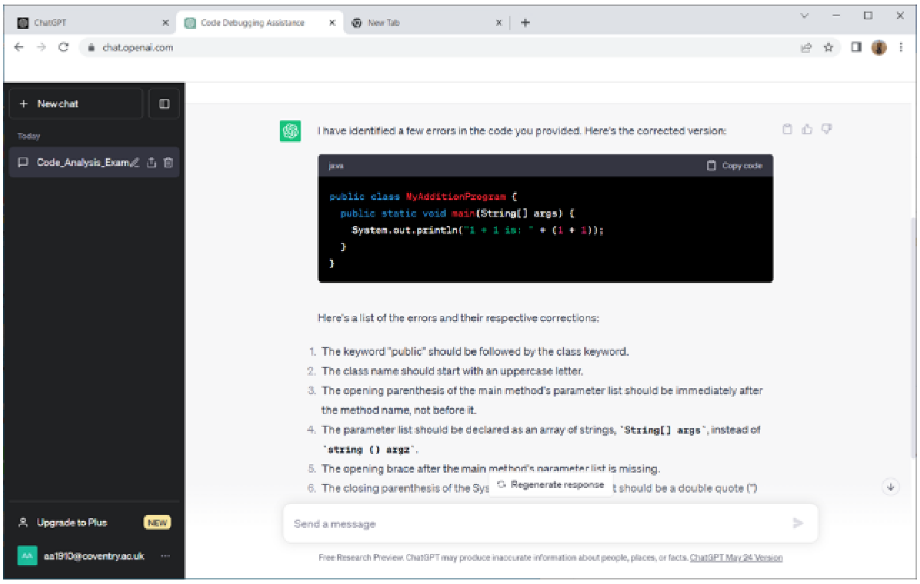
6.1. Debug The Following Code. Keep A log of All Errors that You Spot.

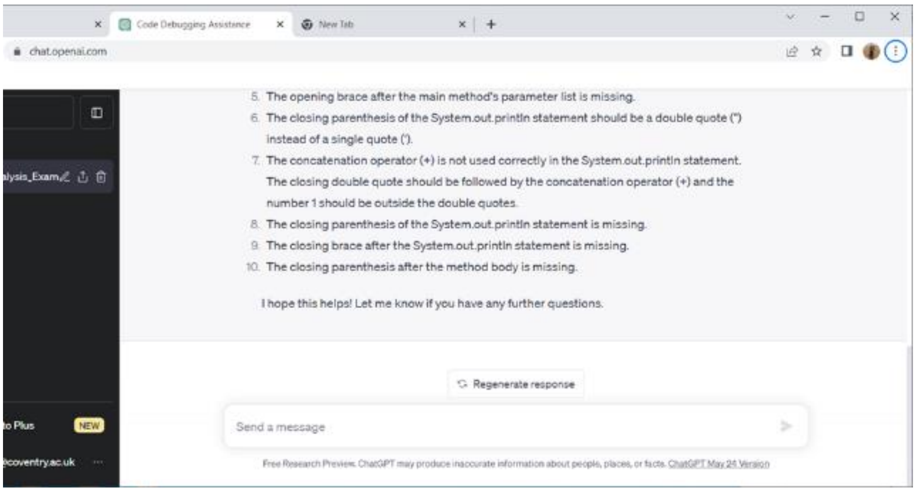
```
Public Glass My Addition Programme
[publi status avoid maine(string () argz]
{
        System.out.println('1 + 1 is:" + 1 + "1");
)
}
```

Solution to (1) produced by ChatGPT Query:



Result:





6.1.1. Analysis

ChatGPT has successfully and comprehensively debugged the code it was given. The program is basically sound, and the bugs concern relatively trivial typing errors or, for example, brace-matching errors. This type of error is familiar to all coders, and the implication of this is that the question is asking the student to do some basic code 'housekeeping' or clean-up, rather than spot more serious errors of syntax or logic. As a code analysis task, it is conceptually very simple.

The solution offers real value to any coder, as trivial bugs like these can sometimes be difficult to spot, consuming time, especially for learner programmers who may be worried that their program is not running for more serious reasons. Using ChatGPT to do automated bug checking of this type seems perfectly acceptable assuming students know how to produce the code being checked. Indeed, automated cleanup frees up any coder to concentrate on more challenging issues of program design. Thus, while coders do in principle need the analysis skills to spot such bugs, they are perhaps of a similar order to typos, which (as with word processors) can and should be fixed automatically.

At the same time, the ChatGPT response does not only correct the code; it also explains what the errors are, so providing some useful instruction and consolidation. The terminal image ChatGPT provides also correctly identifies the language: Java.

This question seems a good one to ask an LLM and to provide real value. The value does depend, however, on the student having written / being able to write the code in the first place.

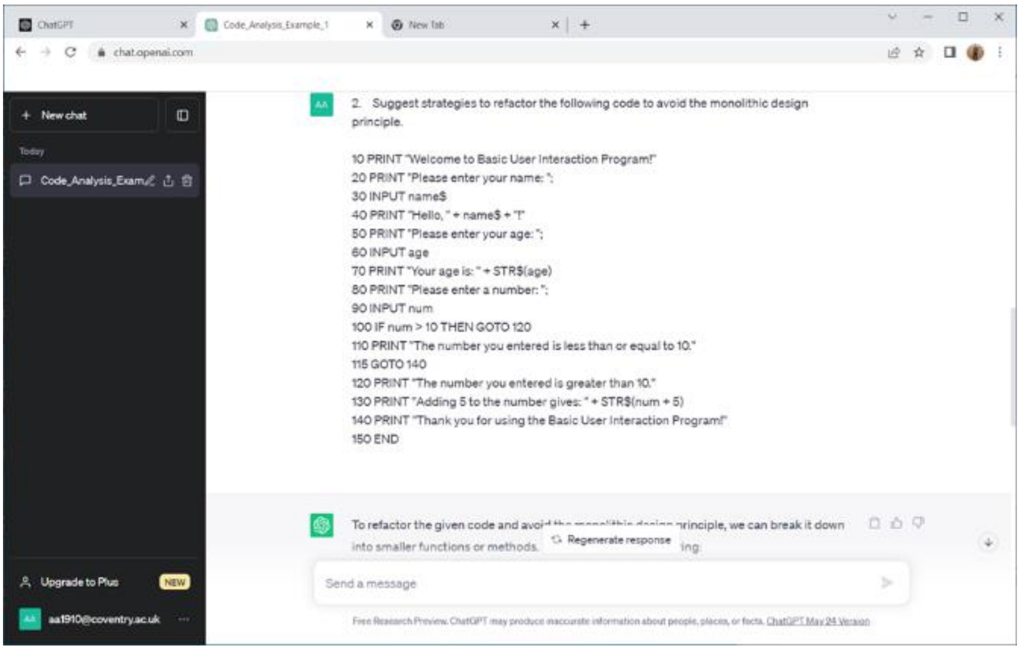
6.2. Suggest Strategies to Refactor the Following Code to Avoid the Monolithic Design Principle.

```
10 PRINT "Welcome to Basic User Interaction Program!"
20 PRINT "Please enter your name: ";
30 INPUT name$
```

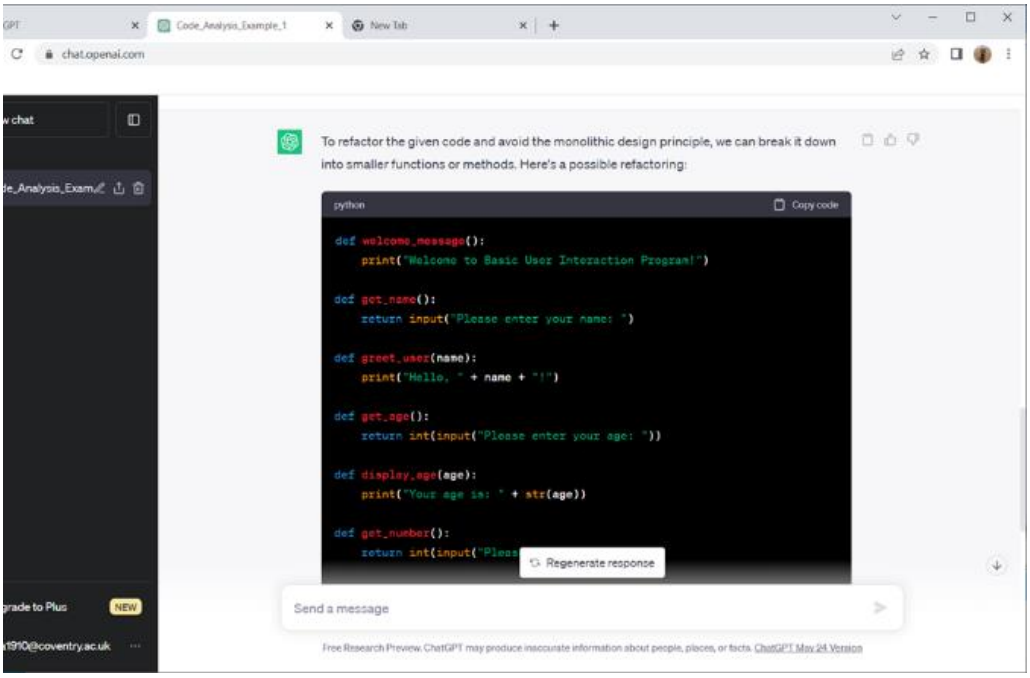
```
40 PRINT "Hello, " + name$ + "!"
50 PRINT "Please enter your age: ";
60 INPUT age
70 PRINT "Your age is: " + STR$(age)
80 PRINT "Please enter a number: ";
90 INPUT num
100 IF num > 10 THEN GOTO 120
110 PRINT "The number you entered is less than or equal to
10."
115 GOTO 140
120 PRINT "The number you entered is greater than 10."
130 PRINT "Adding 5 to the number gives: " + STR$(num + 5)
140 PRINT "Thank you for using the Basic User Interaction
Program!"
150 END
```

Solution to (2) produced by ChatGPT

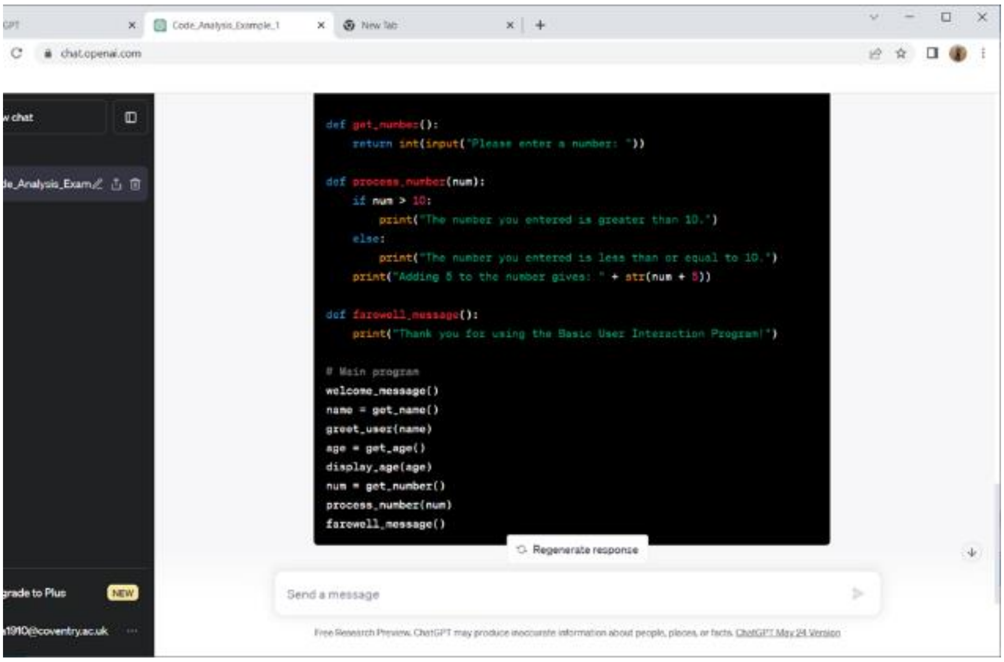
Query:



Result (1):



Result (2):



6.2.1. Analysis

In terms of code analysis, this question is at a higher level of challenge than Question (1) above. The ChatGPT response is much less satisfactory, and if a student were to submit something like this, it would be unlikely to score a very good grade. The main issues are:

1. Too many functions, and related function call overheads
2. Breaking the code into functions has been done arbitrarily

3. The programming language has changed from BASIC to Python
4. ChatGPT does not explain what it has done or why
5. ChatGPT has not answered the question

The program is a very basic chatbot. While it makes some sense to encapsulate it as one or (at most) a few functions, ChatGPT has produced 9. This introduces an overhead, which is the number of (non-encapsulated) function calls, which also deal with return values then passed as arguments in subsequent calls.

The breaking into functions seems arbitrary. For example, code to retrieve and display the user's name has been split into two separate functions get_name() and greet_user(). A single function that both gets and displays the user's name seems more plausible, and reusable. The same goes for get_age() and display_age(). Other functions which simply print lines, welcome_message() and farewell_message() seem unnecessary. However, the ChatGPT refactoring has recognised the decision in the code and neatly encapsulated it as the process_number() function. While the wider use of this seems doubtful (the chatbot is something of a toy program), it makes sense that this processing is separated in case re-use is required.

One of the key issues with the solution is that ChatGPT has produced is that it has arbitrarily switched languages from BASIC to Python. If the question implies thinking about how refactoring should be applied to, or is affected by, a particular language; or is about improving existing code written in a given language for practical purposes, ChatGPT has not considered this, or explained the language switch.

The question is, 'Suggest strategies to refactor the following code to avoid the monolithic design principle'. However, while monolithic design is mentioned, it is not explained. Additionally, strategies are not suggested (although the solution implies a strategy or strategies). There are many strategies for refactoring, including red-green refactoring, refactoring by abstraction and composition (to remove duplication), simplifying (to improve readability), single responsibility (e.g., one function does one thing), moving features, or (for OOP) removing complex conditionals through polymorphism, and so on. One way to handle the question would be select and apply some of these to the code, then contrast and critique the results. But this is entirely missing. Thus, ChatGPT has not answered the question. While a strategy or strategies may have been used, this remains implicit / opaque.

Additionally, a good answer would explain the monolithic design principle. This means 'composed in one piece', which can be bad if code becomes overlong, complex and hard to read. A skill programming student are taught is how to separate functionality to enable readability and reusability, as well as (for example) the limitation and control of variables. If a program consists of one huge function, it can be hard to follow, or to locate and reuse specific functionality. The question implies that the given code is monolithic and needs to be refactored to address this, but students might question how far it really is monolithic and just how much refactoring is merited.

This question has not been well handled by ChatGPT, which has two implications: (1) the uncritical use of ChatGPT to produce content for assessment purposes may not produce the sought for outcome; and (2) the successful use of ChatGPT for this kind of question needs to take place against a background of knowledge and critical skills. ChatGPT could be be a valuable resource in the development of code analysis skills, in a similar way to the use of a range of bibliographic sources (see OU chapter - which one) but (at least on this evidence) cannot replace them.

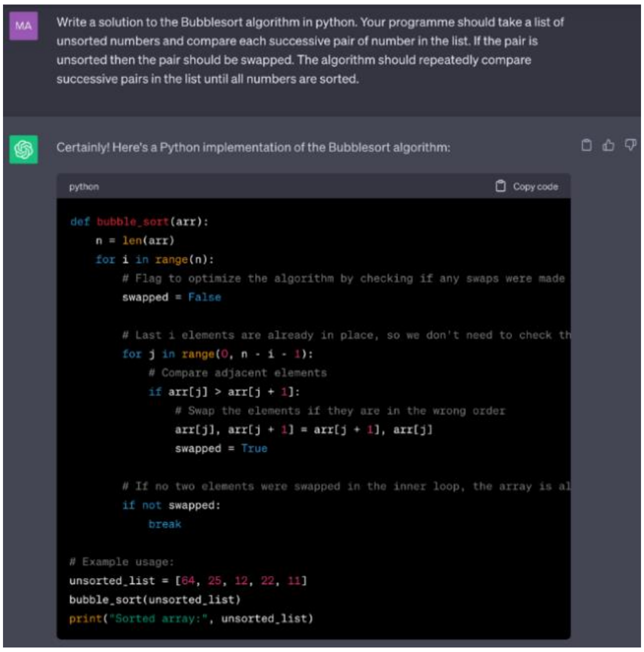
7. Code Production

At the heart of computer science education, we must teach our students to write their own code. This is an essential part of computing education. Ultimately, we must teach students how the machines that they will interact with are programmed. Typically, this is assessed by asking the students to produce their own code in a variety of settings. Exercises may range from small weekly tasks covering specific skills or fundamental programming concepts to large summative synoptic programming assignments requiring cross-sectional skills and the integration of external libraries and APIs. The quality of the resulting code is typically evaluated by an instructor who both runs the programme and reads the code to check for function and form.

To use a LLM to produce code, a student must be able to describe the task in appropriate language. Indeed, a large part of computing education is to teach students how to rephrase and understand a task so as they are able to properly frame their thinking prior to developing the code for a task. A danger of the use of LLMs in code production is that it removes the need for the student to consider how to produce the code. In many cases the student can simply type in (or copy-paste) the instructor-provided task description and receive some competent output without understanding the code. This may backfire on the student if the LLM fails to complete the task as they will have no way of knowing whether the generated code is correct or not.

7.1. Analysis

A sample programming question, and the response from ChatGPT are given in the screenshots below:



In this question, the algorithm (bubble sort) and the programming language (python) are specified. Additionally, a brief description of how the algorithm should work is also provided. The language model correctly provides code in python to complete the bubble sort algorithm. It follows appropriate syntactic conventions; provides a procedure and code to call it and augments the code with relevant comments describing each step of the programme. However, note that the algorithm provided does not match the one requested. The provided algorithm uses an optimisation technique specific to bubble sort, namely iterating n times, where n is the number of elements in the list, and not sorting the last 'i' items at each iteration as these will have been sorted. This is not clear from the original description, which asks for the algorithm to run indefinitely until the list is sorted.

Additionally, the Language Model provided a brief explanation of each step in the code (not shown). This explanation does not refer to the optimisation step given above, and so a student may be unaware of why the algorithm is implemented in this way. Note that this is not some sign of intelligence on the model's behalf, rather the code-based training data it has seen will have contained examples of bubble sort that also leveraged this optimisation.

It is important to note that there is a worthwhile debate around authenticity in programming assessments which has existed before the advent of generative AI. The key question is, should we allow our students access to the tools (Syntax Highlighters, Debuggers, Integrated Development Environments, Frameworks, Version Control, etc.) that they will have when they become developers. One side may argue that teaching in a closed environment limits distractions and allows the student and instructor to focus on learning the core concepts. The other side may argue that students are not typically workplace ready when leaving university as they are not exposed to an authentic development environment. In reality, computing educators have learnt to drip-feed toolchains throughout a programme. It may well be the case that an appropriate argument can be made that best practices for code-based Generative AI should also be taught to improve student's employment prospects.

7.2. Key Threats to Assessment Type

- ChatGPT and other LLMs can produce high-quality error-free code for a task.
- Well-known tasks that are typically used in programming exercises will have formed part of the original training data.
- The code that is produced usually contains well-written comments and an explanation of the functionality of the code.

7.3. Mitigation Strategies

- Require students to learn to code offline (pen and paper, CLI, sandboxed machines)
- Teach students in a bespoke programming language, for which examples on the internet are not readily available.
- focus assessment on problem solving and understanding, rather than code itself.
- Teach best practices for using Generative AI in a coding environment (with industry participation).
- Assess students on larger synoptic codebases, requiring the integration of bespoke APIs.

8. Experiential

Experiential assessments emphasise student development. Typically students work on a real world problem over a period of time, often as a group, to experience the practical application of knowledge and develop employability skills such as group work, leadership and communication. Assessment can involve self-reflections, reflective journals, project reports, presentations, group discussions and peer evaluations. Self-reflections encourage critical thinking, self-awareness and the integration of theory and practice, and often follow a structured model, such as Gibbs Reflective Cycle (1988).

8.1. Analysis

Sample questions for assessing Experiential Learning in the computing domain are given below:

1. Write a reflection on collaborating with peers on your presentation on the ethics of AI.
2. Using a reflective model, write a reflection on your experience working on the group project.
3. Using Gibbs model (1988), write a reflection on collaborating with peers on your analysis of the case study.
4. Reflect on your choice of agile approach for developing your team project.
5. Reflect on your experience of leadership during the group project.
6. Write a reflective coding journal including your thoughts, challenges, and breakthroughs while working on coding assignments, and reflecting on your problem-solving approach, difficulties encountered, and your strategies for overcoming difficulties.

A sample of prompting ChatGPT for a response to the third question is given below:

A

3. Using Gibbs model, write a reflection on collaborating with peers on your analysis of the case study.

Reflection on Collaborating with Peers on Analysis of a Case Study:

Using Gibbs' Reflective Cycle, I will reflect on my experience of collaborating with my peers on the analysis of a case study. This reflection will focus on the key stages of the Gibbs' model: description, feelings, evaluation, analysis, conclusion, and action plan.

Description: Our group was assigned a case study that required a thorough analysis of a complex business scenario. We divided the tasks among ourselves, and each member took responsibility for researching and analyzing specific aspects of the case. We held regular meetings to share our findings, discuss the implications, and develop a cohesive analysis.

Feelings: Initially, I felt a mixture of excitement and nervousness about collaborating with my peers. I was eager to learn from their diverse perspectives and contribute my own insights, but I also felt some pressure to ensure that my contributions were valuable. As we progressed, I experienced a sense of camaraderie and enjoyment in working together towards a common goal.

Evaluation: Overall, our collaboration was successful, as we effectively combined our individual strengths and knowledge to develop a comprehensive analysis of the case study. Our regular meetings and open discussions allowed us to refine our analysis, identify gaps in our understanding, and address any conflicting viewpoints. However, there were moments when we could have improved our time management and communication to avoid last-minute rushes and ensure a smoother workflow.

Analysis: Reflecting on our collaboration, effective communication and active listening were fundamental to our success. We created a supportive environment where everyone felt comfortable expressing their ideas and challenging each other's perspectives constructively. This open exchange of thoughts allowed us to consider different viewpoints and develop a more nuanced analysis. However, we could have benefited from more structured communication channels and clearer expectations regarding deadlines and deliverables.

Conclusion: Collaborating with my peers on the analysis of the case study was a rewarding experience. Through our collective efforts, we were able to develop a well-rounded and insightful analysis that considered various aspects of the business scenario. The collaboration allowed me to broaden my understanding of the subject matter and appreciate the value of teamwork in achieving comprehensive results.

Action plan: To improve future collaborations, I will take several actions. Firstly, I will work on enhancing my time management skills, setting realistic deadlines, and encouraging the team to adhere to them to avoid last-minute rushes. Secondly, I will proactively seek clarification on expectations and encourage open communication channels to ensure everyone is on the same page throughout the project. Lastly, I will continue to actively listen to my peers, appreciate their contributions, and provide constructive feedback to foster an environment of continuous learning and growth.

Overall, the experience of collaborating with my peers on the analysis of the case study was highly beneficial. It allowed me to develop not only a deeper understanding of the subject matter but also valuable collaboration and communication skills. By implementing the lessons learned from this reflection, I aim to enhance future collaborations and contribute effectively in a team setting.

The output of the model gives a well-structured 'personal reflection' using the reflective model specified and using first person language as expected for this style of writing. At 498 words, the response would meet a common 500-word limit for a section of a report. In terms of structure, the output is good, however it is not authentic and there is no evidence of deep reflection. The content is vague, the general 'learnings' are missing details, and the content of the sections is not clearly linked. For example:

- 'Regular meetings' could include detail of how regular, format, how they were chaired or managed, whether action plans were developed.
- 'There were moments when we could have improved our time management and communication' should have specific examples of issues and their consequences.
- 'We created a supportive environment' needs detail of how this was done;
- The action plan needs examples of how these actions might be achieved.

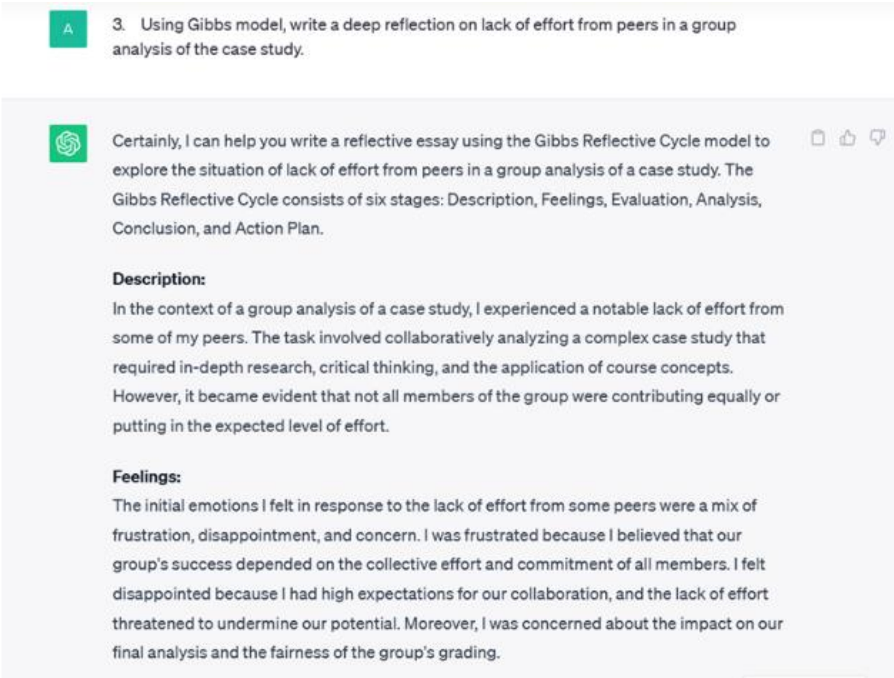
On regenerating the response for the question, the new output wording is more direct and typical of a computer science student, but the themes and lack of detail remain the same.

On re-prompting by asking for a 1000-word reflection, ChatGPT produced a response of 935 words that expanded the example shown above. The new response had added more aspects to reflect on and expanded the action plan as shown below. Despite this, the same issues with lack of detail remain.

Action Plan:
As I reflect on this experience and look ahead to future collaborative projects, there are several key takeaways and action points that I intend to implement:

1. **Enhance Time Management:** To avoid unnecessary stress and ensure smoother project execution, I plan to set more structured timelines and proactively manage tasks. Regular progress checks and interim deadlines will be vital in this regard.
2. **Embrace Diverse Perspectives:** I will continue to prioritize open communication and actively seek out diverse perspectives within the group. Recognizing that these differences enrich our collective knowledge and problem-solving capabilities, I will strive to create an inclusive environment where everyone feels valued and heard.
3. **Conflict Resolution Skills:** Building on the experience of managing disagreements, I intend to further develop my conflict resolution and compromise skills. These abilities are crucial in turning conflicts into opportunities for growth and consensus-building.
4. **Effective Feedback:** Constructive feedback was invaluable in our collaboration. I plan to maintain a culture of providing and receiving feedback in future projects, as it significantly contributes to personal and group development.
5. **Continuous Learning:** Finally, I will carry forward the lessons learned from this experience to continually improve my collaborative skills. I believe that each collaborative project offers a unique opportunity for growth and development, and I am committed to applying these insights in future academic and professional endeavors.

It is possible to adjust the prompt to get a better response, for example the reworded question below, specifying that the reflection should be about lack of effort from peers.



A

3. Using Gibbs model, write a deep reflection on lack of effort from peers in a group analysis of the case study.

Certainly, I can help you write a reflective essay using the Gibbs Reflective Cycle model to explore the situation of lack of effort from peers in a group analysis of a case study. The Gibbs Reflective Cycle consists of six stages: Description, Feelings, Evaluation, Analysis, Conclusion, and Action Plan.

Description:
In the context of a group analysis of a case study, I experienced a notable lack of effort from some of my peers. The task involved collaboratively analyzing a complex case study that required in-depth research, critical thinking, and the application of course concepts. However, it became evident that not all members of the group were contributing equally or putting in the expected level of effort.

Feelings:
The initial emotions I felt in response to the lack of effort from some peers were a mix of frustration, disappointment, and concern. I was frustrated because I believed that our group's success depended on the collective effort and commitment of all members. I felt disappointed because I had high expectations for our collaboration, and the lack of effort threatened to undermine our potential. Moreover, I was concerned about the impact on our final analysis and the fairness of the group's grading.

As many Computer Science students find it difficult to complete a reflective piece and perform poorly in experiential assessments, if the first example given above were submitted for assessment it would likely be graded around pass level. However, if students were to use the model's output as a structured starting point and inspiration, by expanding and adding details this could assist them in reflecting on their learning. By rewording and adding detail to the prompt, students can produce much better answers, but the argument is whether this requires some actual reflection on behalf of the student first, and if so, are models such as ChatGPT just aiding their written communication as some parents might?

8.2. Key Threats

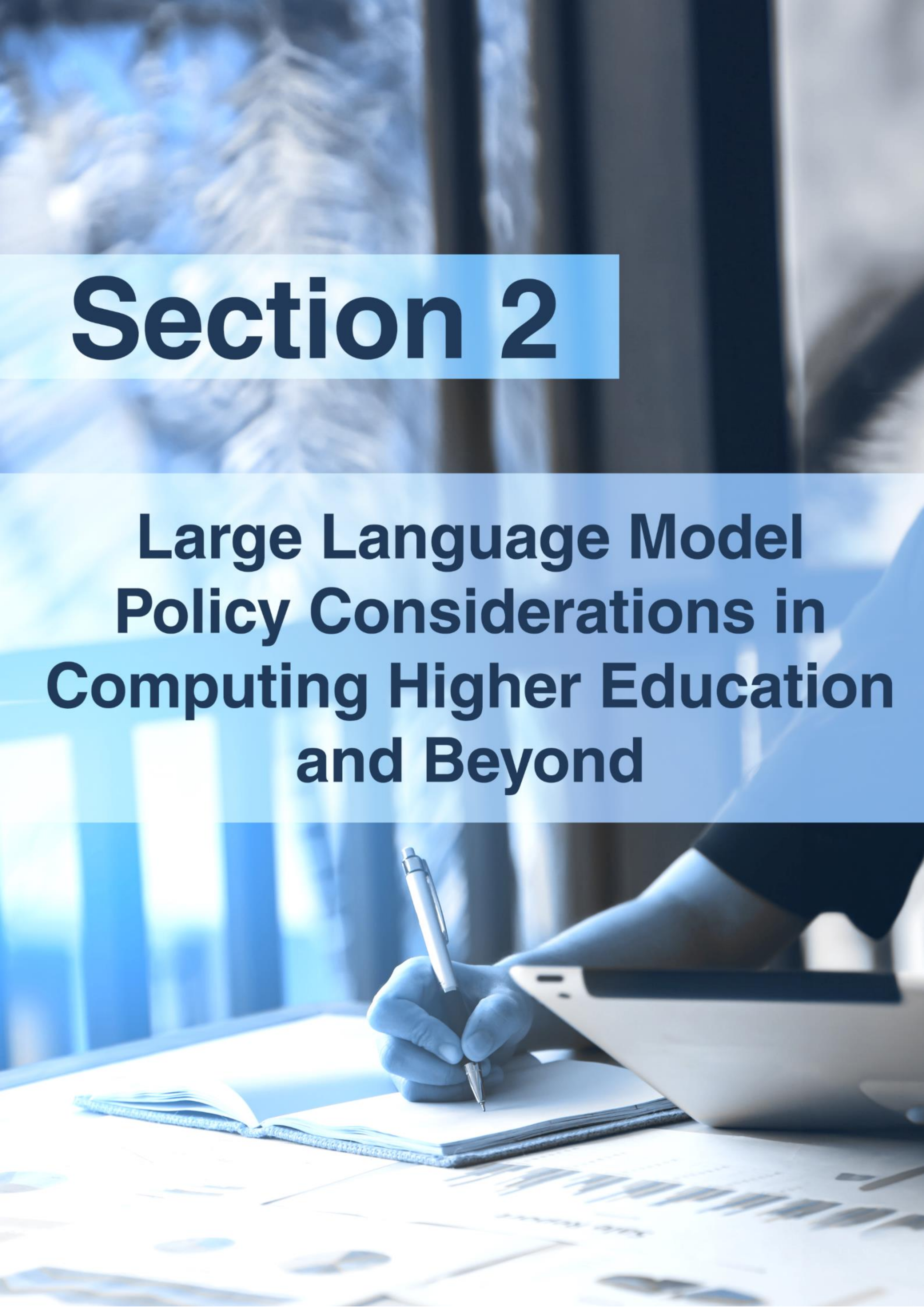
- Despite widely held beliefs that LLMs cannot tackle experiential reflections, they in fact produce a reasonable reflection that could be submitted without amendment.
- Small changes that make the answer context specific produce a good reflection that may be undetectable as an LLM output, especially if the language used is adjusted.
- Individual experiences cannot be verified.
- Refining the output through student interactions can improve and vary the output, getting closer to actual student experiences - answers are infinitely reworkable.

8.3. Mitigation Strategies

- Assessors could triangulate and group member experiences, considering scalability.
- Request the submission of lab diaries covering an extended period of time as evidence, carrying some of the assessment marks.
- Request screenshots of documentation such as meeting minutes, action lists.
- Require strong links to specific student/group work
- Longer activities with reflective assessment, creating a consistent chain of evidence.

Section 2

Large Language Model Policy Considerations in Computing Higher Education and Beyond



9. Employability

Students in technology-based degrees tend to be heavily motivated by the prospects of securing a job in tech, and (hopefully) the financial reward that comes with it. Higher Education is also heavily motivated toward producing “industry-ready” graduates given the impact that a focus on employability has on graduate outcomes.

Recent changes to the early years curricula means that primary schools place greater focus on Computer Science and computational thinking. As this increase in knowledge feeds through to Higher Education, universities will need to adapt and evolve in order to build on this standard foundational knowledge. At the opposite end of Higher Education, generative AI tools which support industry practitioners are becoming more popular. Many practitioners in roles such as development, project management, and test engineering, have adopted such tooling as part of their standard workflow. For example, GitHub[1] found that users of Co-pilot felt 88% more productive when developing, and 59% of respondents said that they were less frustrated when coding. In addition, 60% of respondents said that co-pilot made them feel more fulfilled in their job.

There are a number of anticipated effects that increased adoption of generative AI will have on industry. For example, given that models tend to be trained on public content, we can expect content to become more similar internationally. Thus, leading to a globalisation of standards. The RedHat blog lists some clear benefits that generative AI will have on industry[2]:

- Personalised customer experiences
- Streamlining operations and efficiency
- Enhancing decision making
- Preserving privacy and security
- Fraud detection and cyber security

In the technology industry, there are a number of applications where generative AI tools are expected to have impact:

- Automated unit test generation
- Document summarisation (e.g., summarising reports/survey data)
- Improvement of general writing (e.g., emails, references, technical documentation, client communications and marketing material)
- Forecasting (e.g., financial, trend analysis)
- Product design (e.g., software design, physical products)
- Code generation
- Architecture generation

This adoption of tooling means that the proper use and application of such tools will become an expectation for prospective employees. In order to bridge the well documented gap between academia and industry, universities should train students in

[1] https://github.blog/2023-05-09-how-companies-are-boosting-productivity-with-generative-ai/
[2] https://www.redhat.com/en/blog/generative-ai-business-applications

the proper usage of generative AI tools. However, it is imperative that universities discourage blind faith in generative AI tools and ultimately students need to understand the work that they produce, and that they have responsibility for the work that they produce. Some companies, such as JPMorgan, Chase, Amazon, and Accenture, have reportedly banned the use of ChatGPT within their workforce[3].

9.1. Policy Points

- Higher Education should train students on how to properly use generative AI to support industry tasks.
- Higher Education should train students on the risks of using generative AI, including issues surrounding copyright, data protection, and how they are ultimately responsible for the work which they produce.
- Higher Education should accept and adopt the use of generative AI in assessments where appropriate i.e., Universities should incorporate generative AI tools into authentic assessment.
- Higher Education should continue to emphasise the ‘soft skills’ required in industry and the value of a ‘human touch’ in projects. Automating everything is not always the answer.
- Higher Education should continue to reduce the gap between academia and industry, with a focus on producing industry-relevant graduates. This is often done through industrial feedback on curricula (e.g., Industry Advisory Boards)
- Higher Education should place higher emphasis on students’ understanding of security and secure software development practices.

[3] https://edition.cnn.com/2023/09/22/tech/generative-ai-corporate-policy/index.html

10. Academic Misconduct

With the advent of multi-topic faithful text generation through the modern era of chat-based AI systems, many in the HE sector are rightly concerned about the potential impact of this technology on academic misconduct during assessments. Indeed, since the introduction of ChatGPT several universities have prohibited access to this technology on their campuses (Duffy, 2023). A recent article, however, claimed that students are widely using ChatGPT and other similar AI chatbots in a variety of capacities to help with generating ideas, essay plans, and even submissible outputs for summative assessment (Terry, 2023).

Academic misconduct existed before the advent of readily available artificial intelligence systems. Students have always had opportunities to copy shared work, ask friends for help, or even pay for a readymade answer from an essay mill. Every institution has academic misconduct policies that handle these scenarios, and in many cases the nefarious use of AI in student work is already covered. For example, if a student submits work that is wholly generated by an AI, then they should receive the same penalty as a student who (1) copies work from another, or (2) pays for an answer from an online service. However, using AI is different from these two scenarios and presents a lower barrier for the students to cross. If a student copies from a friend, then someone else knows of their transgression - they may reveal this at any time. If they are caught, the person whose work was copied may admit to sharing the work. Cheating with AI is hidden. A student can generate an answer with no external involvement. Worse, if they are questioned, they may deny having used AI. Unless the tutor has strong evidence, the transgression may go unpunished. Similarly, using an essay mill presents a high barrier of cost. Cheating with AI, however, is free. A student can generate multiple examples and submit the one they think will do best across multiple assessment points, with no financial outlay.

Of course, many students will not directly use AI to create their final output, but may still use AI as part of the process of generating ideas, or as an initial knowledge base. We must develop sensible policies that allow students to engage with the technology in a constructive way, whilst remaining honest about what is representative of their own work, and what is generated by AI. The policies surrounding this in a HE is setting are likely to develop as AI regulation is developed at a national and international level.

Whilst this may paint a bleak picture, there are many opportunities to overcome AI-based academic misconduct. The first step in addressing this is to educate those in positions of power at all levels within our higher education institutions to the dangers, but also opportunities, of the integration of AI into education. There are many possibilities for students to constructively engage with AI technology as part of their learning journey. This is only possible if academics are able to curate valuable learning experiences that integrate AI technology. In the same way, modern degree courses integrate personal computing, the internet or smart devices as learning tools, we must also seek ways to integrate AI into our student's learning experiences.

10.1. AI Detection

Since the release of ChatGPT, a number of pieces of AI detection software have been released for use by the public. Most notably, TurnItIn released a tool, which was integrated into their existing software to score a piece of writing according to its likelihood of being generated by AI. This software is developed by training a machine learning algorithm to distinguish between examples of AI written text and human written text, based on common patterns found in each. The machine learning algorithm does not get a perfect rate of classification and may exhibit false positives (human-written work detected as generated by AI), as well as false negatives (AI-written work not detected). In fact, Turnitin aim for their tool to have a 1% false-positive rate:

"We might flag a human-written document as AI-written for one out of every 100 fully human written documents." - (Turnitin, 2023)

Ultimately, AI detection software is not sufficiently mature to be relied on at this point. Falsely accusing 1% of a cohort with allegations of AI-based academic misconduct is clearly not in the best interests of any institution. Whilst AI detection software may be useful as an indicator tool, it should not be relied on or used to make final judgments overriding those of a human.

There are a number of strategies that educators can use to manually detect AI-generated text. Whilst these may not give direct evidence of academic misconduct through AI, they may help to form a picture and allow an academic to build a case for an academic misconduct panel.

- Mismatch to student style: If the work submitted by a student is particularly different than their usual style (i.e., overly formal, beyond their usual academic level), this may be an indicator of the use of AI. This may also occur within an assessment, where a student has written some areas themselves and generated other areas, with the style varying drastically.
- Lack of depth: AI generated answers typically do not contain the level of depth or analysis that a human-generated answer may contain, and that is sought for at degree level. The answers are typically formulaic, consisting of a brief summary, a list of important topics and a summary/conclusion paragraph at the end. The answers typically contain heavy repetition and prioritise form over function.
- AI artefacts: Students may not realise that the answer they have automatically generated contains both subtle and obvious clues to indicate that it was generated by AI. An obvious clue is the use of the phrase "As an AI language model", which ChatGPT has been trained to produce to indicate that its response is the result of a machine generation. A more subtle clue is the use of the second person in answering the question. Chatbots are designed for conversation and often answer in a discursive manner (your question, etc.). Finally, in the UK context, tutors may notice Americanisms introduced into their student’s work. The major chatbots are trained on web-based text, which typically uses American-English spelling, so the text they produce also follows this convention.

- Hallucinations: Chatbots are not grounded in factuality. Whilst generating a response to a question, there is no check to determine whether the answer is truthful or not. This can lead to invented references and facts. References that cannot be resolved to texts, or are a recombination of authors, journals and titles from valid sources are a strong indicator of the use of AI.

10.2. Policy Points

- Include an element of assessment that is less vulnerable to misconduct (i.e., image/data-based analysis)
- Use automated detection software as a 'first-flag', but not as a 'final-flag'.
- Train staff to be aware of signs of LLM-based misconduct.
- Train students to be aware of how to (and how not to) use LLMs.
- Consider invigilated assessments, or elements thereof where text-based submissions are necessary.

11. New Assessments

If the basic problem of the use of LLMs in assessments is that students may be tempted to pass off (uncritically) the output of LLMs as their own, then the policy should be to look to ways that prevent that action and incorporate the use of LLMs in the assessment work itself.

There are a few areas where the temptation to use LLMs uncritically exists – code production and long essays. The policy for assessment should match a policy that prevents the use of essay mills and other forms of contract cheating.

11.1. Policy

- In all cases, ensure that a session on risks associated with the use of AI tools is included, with specific attention paid to areas where there is an ethical, privacy, security, or legal concern.
- Ensure a regulation around the citation of AI use.
    - This needs to match the institution's citation policy for general referencing.
- Make it clear which elements of an assessment may permit the use of LLM material but that the student must make it clear which parts are created by LLM and for there to be a defined maximum amount of content to come from AI tools.
- For project work where the use of AI tools might be encouraged (such as coding examples), ask for the input parameters, prompts, and preferences to be provided as appendices or part of the assignment response.
- Create assessments that actively develop and assess AI literacy.
    - This should be amongst the first assignments that students encounter and will prevent poor practice later in the course.
- Consider the use of live, in-person assessment events. This can include oral presentations, interviews, vivas or observed discussions.
    - Asking students to present or respond to questions will separate authentic learning from assessment responses that merely take output from AI tools.
- Develop assessments where there is an ongoing 'development' element with milestones.
    - Milestones can be used to test the development of a student's response over time.
- Develop assessments where students need to submit reference material at milestones.
    - Reference material submission can be used to see the development of a student's learning over time.
- Create submissions where students must complete a journal of time and/or activities involved in the completion of an assessment.

   - This can be useful where AI tools are used and required in the work. Students can report regularly on their progress and is useful when combined with reflection.
- Develop assessments based on action-learning cycles.
   - Action learning cycles rely on periodic reflection of an experimental process of intervention, which would be unique for each student.
- Develop just-in-time assignments where the task is made available for an in-person day-long assessment centre.
   - Just-in-time assessments can be used to test immediate responses from students to a particular problem, such as through a hackathon.
- Develop assessments where the grading is focused on a critical reflection and introspection element rather than the production of assets.
   - Rather than 'mark the work', the grades are assigned to how the student reflects on results and the process.
- Develop assessments that may test the student's ability to develop prompts to resolve problems and for the students to reflect on the relative success or failure of prompts to solve problems.
- Make sure that submission deadlines are not bunched so that students are not tempted to use LLMs uncritically if they are struggling to manage several assignments submissions.

12. Adoption of LLMs

Deciding whether to use LLMs is influenced by various factors, including the expectations of students, university management, employers, regulatory bodies, and professional associations. As we still need to have a detailed understanding of these groups' perspectives, there is uncertainty about the extent to which LLMs should be adopted for individual use and integration into curricula. Concerns were expressed about the fear held by many regarding artificial intelligence more generally (such as the impact on jobs), which was attributed to press and social media coverage. Risks and uncertainty about accountability are factors in LLM adoption.

A range of issues were discussed relating to the current landscape of AI tools available and equitable access to them. The affordability of tools was highlighted, noting that the widely used freemium pricing model links access levels to payment. Issues related to the data used to train LLMs were also frequently raised, with calls for greater transparency from leading developers of these tools. More certainty about the source of training data is needed to evaluate quality and potential biases, as well as address concerns about copyright. There are also worries about the impact of increasing amounts of synthetic text generated by LLMs. Without reliable methods of distinguishing between human-authored and AI-generated text, information literacy skills will become increasingly important for students and educators.

12.1. Developing AI Literacy

Developing AI Literacy involves familiarity with the LLMs themselves, as well as the broader AI landscape. Knowledge of the specific AI terminology, techniques, and tools is necessary for making informed decisions about use. For some, a basic understanding of how LLMs work and their capabilities was sufficient, but others felt that using them confidently and competently was also required. Writing effective prompts was noted as a specific skill needing development, with parallels drawn to performing an effective internet search. Individuals should be aware of the tools available and understand their limitations and ethical considerations to make informed choices about using them.

Building staff and student AI literacy will require adequate time, space, and support. Having opportunities to interact with these new technologies and forums to discuss implications was seen as important. Guidance and training should be differentiated based on needs and confidence levels. There were differing perspectives on whether developing AI literacy should be part of broader digital skills initiatives or a separate focus and whether to use existing or new support teams. Sensitivity is needed regarding the wider impacts of LLMs, as some may need to reskill, and there are varying comfort levels with new technologies.

12.2. Policy Points:

- Develop guidelines for responsible use of LLMs in educational settings that consider equity, transparency, copyright, fact-checking, and ethical implications. Engage stakeholders like students, staff, management, and regulatory bodies in drafting guidelines.

- Provide training and support for educators on AI literacy, including LLM capabilities, limitations, terminology, prompt writing, and ethical use cases. Give time and space for hands-on experience and discussions. Offer differentiation based on educator needs and confidence.
- Advocate for transparency from LLM developers regarding training data sources, potential biases, and ownership. Educators need this information to evaluate suitability.
- Consider affordability issues with widely used freemium pricing models.
- Offer opportunities to develop information literacy skills, such as fact-checking.
- Amend academic integrity policies to address the use of generative AI expressly. Define appropriate vs. inappropriate use cases.
- Foster broad awareness of the broader AI landscape among staff and students, not just LLMs.

13. Timescales

Academia can be very slow to adopt and adapt to new technologies and it is likely that LLMs will be similar. One of the main causes for slow adoption of changes is the regulations and restrictions that are in place regarding making changes to unit specifications, learning outcomes, assessments etc. Some of these are internal regulations designed to provide more certainty to students and ensure adequate oversight. Others are external regulations pertaining to consumer protection laws and other relevant legislation and guidance from governmental bodies such as the Office for Students (Protecting students as consumers, 2023).

Nevertheless, there are uses of LLMs that can be adopted in the very short term and others that can be prepared for now to take effect over the course of a few years.

13.1. For Immediate Adoption

One immediate use of LLMs that could prove extremely valuable is the testing of assessments. This falls into a few different areas. In some cases, particularly around coursework, LLMs can be fed assignment briefs and tasked with producing a submission. An expert marker can glean important information from these submissions in terms of understanding the level of difficult of the assignment and whether it is assessing all the knowledge and skills intended. Another use of LLMs in this context is to help the assessor check the readability and possible ambiguities of the assignment brief itself. An LLM can be asked questions regarding an assignment brief to ensure that it has been understood correctly and this can serve as a proxy to ensure, ahead of release, that an assignment brief is clear in its requirements and instructions.

A more general use of LLMs in the area of assessment is in actually generating components of an assessment. For example, LLMs can be tasked with generating example scenarios and/or assessment questions related to those or other scenarios. LLMs can also be tasked with generating potential project prompts for students for more open-ended project units. For formative work, LLMs can certainly be used to generate lab exercises. For example, an existing lab exercise could be used as the basis for generating new exercises that are similar in terms of the skills and knowledge they cover but use different scenarios or ask different questions. This can allow a unit to retain a freshness of content without the teaching team needing to spend a lot of time redesigning the exercises that have worked well previously.

LLMs can also be used in the initial teaching as well. Where there is scope within a unit, LLMs can be used to help plan entire lecture courses by suggesting topics to cover. These can be based on prompts or even a list of required learning outcomes. LLMs would be able to propose appropriate topics matching the number of teaching sessions. This kind of co-generation can be extended to other areas. For example, one use of LLMs is to generate texts or scripts from existing lecture slide decks. These scripts can be made available to students as additional sources of information and with LLMs they can be generated at very little extra cost.

However, there remain some concerns over whether the content generated by LLMs may be subject to copyright restrictions from the original owners of the data. The owner of the LLM may pass on responsibility for the copyright status of the generated text to the user who must then consider whether or not their use is in breach of copyright laws but could, under UK law, become the copyright holder of the generated text (Sekhon, Ozcan and Ozcan, 2023). It should be noted in this context that exceptions to copyright law exist for the purposes of teaching under UK law (Exceptions to Copyright, 2021).

Another possible use of LLMs which can be immediately adopted and with some important impact is the ability of LLMs to rewrite text changing its tone and feel. For example, LLMs have been used successfully by academics to mellow the tone of an email to a student who has not performed at the required level. This additional step in the emailing process can also help bridge language gaps between lecturers and students. One use, for example, may be to tailor text to or from non-native English speakers to avoid or adopt certain idioms and phrasings.

Whether or not LLMs are used in the generation of assessments or teaching, the existence and usefulness of LLMs must be recognised. It is therefore important that, with immediate effect, its existence is acknowledged during teaching and students are taught how to make proper and appropriate use of such systems. Universities and individual lecturers should not ignore them but should embrace them. Much as we may routinely advise students to Google for help when they are stuck on some particular syntax, we can now also advise them to ask an LLM for help. However, care must be taken to ensure that the LLMs output is carefully considered and not accepted uncritically by students. Unfortunately, this is a problem that still exists with search engine results where the first result's contents may be adopted uncritically and unthinkingly by students leading to detrimental effects.

13.2. Longer-Term Changes

In the longer term, LLMs may become more integrated and central to our teaching. Over the next 1 or 2 years, some learning outcomes may have to change to either explicitly or implicitly acknowledge and incorporate the use of LLMs. Changes to learning outcomes must follow due processes to ensure compliance with both internal and external regulations and therefore they will require a longer lead-in time. It is also important that changes are not too rigid given the fast pace of change at the moment in this area.

Over an even longer period, changes at the overall course level should be implemented. Courses must be taught in a way that does not fight against LLMs but embraces them as helpful tools and teaches students how to make proper use of them. Whilst the majority of students will not need to understand how LLMs work, students will need to understand enough to ensure they can make the best use of them as tools. Unlike search engines which students are assumed to have grown up with and know how to use, LLMs being new and more complex will likely require some explicit instruction.

14. Conclusion

When we were invited to host a workshop on the use of ChatGPT in Computing Education for the Council of Professors and Heads of Computing, we did not anticipate the interest that we would receive, nor the scale of the task that we had committed to. As we planned for the workshop, we realised that so little policy currently existed, and that we were able to meet a timely gap in the market by producing this report. The experiences in preparing, hosting and participating in the workshop, have allowed us to develop valuable insights into the AI future of higher education.

In this white paper, we have explored two key areas concerning the use of Large Language Models in Computing Higher Education. Firstly, we considered the threat level to a number of potential assessment modalities. Educators must consider the ways in which they assess students and the types of questions that they use. Students are already using Large Language Models in their learning and educators must address this to stay ahead of the curve. Secondly, we considered timely policy considerations across four key areas of concerns for decision makers in higher education institutions. By considering these policy areas we hope to be able to add to and shape the existing narrative around the use of this technology within our institutions.

We do not consider this report to be the last word on the matter, but rather the first – opening up the future conversation within the HE sector. We stand at an unfolding moment in the history of education, where rapidly evolving technology is changing the way we work and learn. We must make wise and conscious decisions today, to the benefit of our future practice and that of the learners and educators that will follow in our paths.

Acknowledgments

The '**ChatGPT in Computing Education: A workshop to Co-Develop Guidelines for Assessment Policy'** was supported by funding from the Council of Professors and Heads of Computing (CPHC) via a 2023 Special Project Fund grant and by the Department of Computing and Mathematics at Manchester Metropolitan University.

The editors would like to thank all the participants of the workshop, whose valuable contributions helped shape the work in this report. Particularly we thank the contributing authors named below.

Authors:

Matthew Shardlow, Manchester Metropolitan University, *Long Essay, Code Production, Academic Misconduct*

Annabel Latham, Manchester Metropolitan University, *Executive Summary, Experiential*

Daniel Gooch, Open University, *Introduction, Knowledge Recall*

Kevin Waugh, Open University, *Introduction, Knowledge Recall*

Mike Richards, Open University, *Introduction, Knowledge Recall*

Mark Slaymaker, Open University, *Introduction, Knowledge Recall*

Alistair Willis, Open University, *Introduction, Knowledge Recall*

John Halloran, Coventry University, *Critical Analysis, Code Analysis*

David Edmundson-Bird, Manchester Metropolitan University, *New Assessments*

Leanne Fitton, Manchester Metropolitan University, *Adoption of LLMs*

Anthony Kleerekoper, Manchester Metropolitan University, *Timescales*

Ashley Williams, Manchester Metropolitan University, *Employability*

Workshop Participants

This white paper reports and extends the discussions of participants who took part in the '**ChatGPT in Computing Education: A workshop to Co-Develop Guidelines for Assessment Policy'** on 17 July 2023. Participants from the following universities and organisations took part.

Abertay University

Aberystwyth University

ARC, UCL

BCS

Bournemouth University

City, University of London

Coventry University

Durham University

Edge Hill

King's College London

Lancaster University

Liverpool Hope University

Liverpool John Moores University

Loughborough University

Manchester Metropolitan University

MIT Academy of Engineering

Northeastern University

Northumbria University

Open University

Sheffield Hallam University

The Open University

The University of Sheffield

UEA

Umm Al-Qura University

University Academy 92

University of Chester

University of Derby

University of East Anglia

University of Hertfordshire

University of Hull

University of Manchester

University of Sheffield

University of Southampton

References

Adobe Inc. (2023) Generative AI – Adobe Sensei. Available at: https://www.adobe.com/sensei/generative-ai.html (Accessed: 21 August 2023).

Anderson, L.W. and Krathwohl, D. (2001) A Taxonomy For Learning, Teaching, And Assessing, A Revision Of Bloom's Taxonomy Of Educational Objectives. New York, NY, US: Addison Wesley Longman, Inc.

Baker, Catherine M., Yasmine N. El-Glaly, and Kristen Shinohara. 2020. A Systematic Analysis of Accessibility in Computing Education Research. In Proceedings of the 51st ACM Technical Symposium on Computer Science Education (SIGCSE '20). Association for Computing Machinery, New York, NY, USA, 107–113. https://doi.org/10.1145/3328778.3366843

Bender, E.M. et al. (2021) 'On the Dangers of Stochastic Parrots: Can Language Models Be Too Big? 🦜', in Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency. New York, NY, USA: Association for Computing Machinery (FAccT '21), pp. 610–623. Available at: https://doi.org/10.1145/3442188.3445922.

Bergmans, L. et al. (2021) 'On the Efficacy of Online Proctoring using Proctorio.', in. CSEDU (1), pp. 279–290. Available at: https://ris.utwente.nl/ws/files/275927505/3e2a9e5b2fad237a3d35f36fa2c5f44552f2.pdf (Accessed: 16 March 2023).

Bloom, B.S. and Krathwohl, D.R. (1956) Taxonomy of educational objectives: The classification of educational goals. Book 1, Cognitive domain. New York, NY, US: Longmans. Bolin, A.U.,

Bloxham, S., Boyd, P. (2007). Developing Assessment in Higher Education: A Practical Guide. United Kingdom: McGraw-Hill Education.

Bowman, E. (2022) 'Scanning students' rooms during remote tests is unconstitutional, judge rules', NPR, 26 August. Available at: https://www.npr.org/2022/08/25/1119337956/test-proctoring-room-scans-unconstitutional-cleveland-state-university (Accessed: 17 March 2023).

Chamberlain, C. et al. (2011) 'The role of short answer questions in developing higher-order thinking', Per Linguam, 20(2), pp. 28–45. Available at: https://doi.org/10.5785/20-2-83.

John Clanchy & Brigid Ballard (1995) Generic Skills in the Context of Higher Education, Higher Education Research & Development, 14:2, 155-166, DOI: 10.1080/0729436950140202

Cockcroft, W.H. (1982) Mathematics counts Report of the Committee of Inquiry into the Teaching of Mathematics in Schools under the Chairmanship of Dr WH Cockcroft. London: Her Majesty's Stationery Office. Available at: https://trulscronberg.se/teacherstudies/1982_Mathematics_Counts.pdf (Accessed: 21 August 2023).

Cox, K., Imrie, B. and Miller, A., 2014. Student assessment in higher education: a handbook for assessing performance. Routledge.

D'Agostino, S. (2022) Wikipedia, Once Shunned, Now Embraced in the Classroom, Inside Higher Ed. Available at: https://www.insidehighered.com/news/2022/11/09/more-professors-now-embrace-wikipedia-classroom (Accessed: 21 August 2023).

Dick, M. (2005) 'Student interviews as a tool for assessment and learning in a systems analysis and design course', in Proceedings of the 10th annual SIGCSE conference on Innovation and technology in computer science education. New York, NY, USA: Association for Computing Machinery (ITiCSE '05), pp. 24–28. Available at: https://doi.org/10.1145/1067445.1067456.

Duffy, C., "**Public school bans on AI tools like ChatGPT raise fears private school kids are gaining an unfair edge and widening a digital divide**", ABC News, Jan 2023. Retrieved from: https://www.abc.net.au/news/2023-05-26/artificial-intelligence-chatgpt-classrooms-schools/102356926 on 15/09/2023.

East, J.P. and Schafer, J.B. (2005) 'In-person grading: an evaluative experiment', in Proceedings of the 36th SIGCSE technical symposium on Computer science education. New York, NY, USA: Association for Computing Machinery (SIGCSE '05), pp. 378–382. Available at: https://doi.org/10.1145/1047344.1047472. Gibbs, G. (1988). *Learning by doing: A guide to teaching and learning methods.* London: Further Education Unit.

Gibbs, G. (1988). *Learning by doing: A guide to teaching and learning methods.* London: Further Education Unit.

Gilson, A. et al. (2022) 'How Does ChatGPT Perform on the Medical Licensing Exams? The Implications of Large Language Models for Medical Education and Knowledge Assessment'. medRxiv, p. 2022.12.23.22283901. Available at: https://doi.org/10.1101/2022.12.23.22283901. Google (2023) Try Bard, an AI experiment by Google. Available at: https://bard.google.com (Accessed: 21 August 2023).

Gooch, Daniel. Kevin Waugh, Mike Richards, Mark Slaymaker and John Woodthorpe. Exploring the rate and demographic profile of university assessments flagged as containing AI-generated material. ACM SIGCSE under review.

Guangul, F.M., Suhail, A.H., Khalit, M.I. et al. Challenges of remote assessment in higher education in the context of COVID-19: a case study of Middle East College. Educ Asse Eval Acc 32, 519–535 (2020). https://doi.org/10.1007/s11092-020-09340-w

Haigh, M.J. (2001) 'Constructing Gaia: Using journals to foster reflective learning', Journal of geography in higher education, 25(2), pp. 167–189. Hussein, M.J. et al. (2020) 'An Intellectual Property Office. Exceptions to Copyright. (2021). [online]. Available at: https://www.gov.uk/guidance/exceptions-to-copyright#full-publication-update-history [Accessed 12 Sep. 2023].

Joint Council for Qualifications (2023) 'AI Use in Assessments: Protecting the Integrity of Qualifications', JCQ Joint Council for Qualifications. Available at: https://www.jcq.org.uk/exams-office/malpractice/artificial-intelligence/ (Accessed: 22 August 2023).

Khramtsova, I. and Saarnio, D. (2005) ‘Using student journals to stimulate authentic learning: Balancing Bloom’s cognitive and affective domains’, Teaching of Psychology, 32(3), pp. 154–159.

Lee, N.T.S., Kurniawan, O. and Choo, K.T.W. (2021) ‘Assessing Programming Skills and Knowledge During the COVID-19 Pandemic: An Experience Report’, in Proceedings of the 26th ACM Conference on Innovation and Technology in Computer Science Education V. 1. New York, NY, USA: Association for Computing Machinery (ITiCSE ’21), pp. 352–358. Available at: https://doi.org/10.1145/3430665.3456323.

Lejk, Mark., Michael Wyvill & Stephen Farrow (1997) Group Learning and Group Assessment on Undergraduate Computing Courses in Higher Education in the UK: results of a survey, Assessment & Evaluation in Higher Education, 22:1, 81-91, DOI: 10.1080/0260293970220106

Lu, C., Macdonald, R., Odell, B. et al. A scoping review of computational thinking assessments in higher education. J Comput High Educ 34, 416–461 (2022). https://doi.org/10.1007/s12528-021-09305-y

McIntosh, A. (1990) ‘Becoming numerate: Developing number sense’, in Being numerate: What counts? A Fresh Look at the Basics. ACER, pp. 24–43.

Mehdi, Y. (2023) Reinventing search with a new AI-powered Microsoft Bing and Edge, your copilot for the web, The Official Microsoft Blog. Available at: https://blogs.microsoft.com/blog/2023/02/07/reinventing-search-with-a-new-ai-powered-microsoft-bing-and-edge-your-copilot-for-the-web/ (Accessed: 20 March 2023).

Meseguer-Artola, A. et al. (2020) ‘Academic impact and perceived value of Wikipedia as a primary learning resource in higher education’, 39, p. e290329. Available at: https://doi.org/10.3145/epi.2020.may.29.

Office for Students. Protecting students as consumers. (2023). [online] Available at: https://www.officeforstudents.org.uk/publications/protecting-students-as-consumers/ [Accessed 12 Sep. 2023].

Ohmann, P. (2019) ‘An assessment of oral exams in introductory cs’, in. Proceedings of the 50th ACM Technical Symposium on Computer Science Education, pp. 613–619. Open University (2020) ‘The PROMPT Criteria’, The Open University Library. Available at: https://www.open.ac.uk/libraryservices/beingdigital/activity/XK1087#page1 (Accessed: 21 August 2023).

Open Praxis. Evaluation of Online Proctoring Tools’, 12(4), p. 509. Available at: https://doi.org/10.5944/openpraxis.12.4.1113.

Pérez, J., C. Vizcarro, J. García, A. Bermúdez and R. Cobos, "Development of Procedures to Assess Problem-Solving Competence in Computing Engineering," in IEEE Transactions on Education, vol. 60, no. 1, pp. 22-28, Feb. 2017, doi: 10.1109/TE.2016.2582736.

Richards, Mike; Waugh, Kevin; Slaymaker, Mark; Petre, Marian and Gooch, Daniel (2023). Bob or Bot: Exploring ChatGPT’s answers to University Computer Science Assessment. ACM Pre-Print. https://oro.open.ac.uk/89325/1/chatGPT_pre-print.pdf

Ruthven, K. (2009) 'Towards a calculator-aware number curriculum', Mediterranean Journal of Mathematics Education, 8(1), pp. 111–124.

Sabin, M., Jin, K.H. and Smith, A. (2021) 'Oral Exams in Shift to Remote Learning', in Proceedings of the 52nd ACM Technical Symposium on Computer Science Education. New York, NY, USA: Association for Computing Machinery (SIGCSE '21), pp. 666–672. Available at: https://doi.org/10.1145/3408877.3432511.

Schultz, M.C., Schultz, J.T. and Gallogly, J. (2007) 'The Management of Testing in Distance Learning Environments', Journal of College Teaching & Learning, 4(9), pp. 19–26. Sellgren, K. (2014) Calculator test ban 'backward step', claim academics, BBC News. Available at: https://www.bbc.com/news/education-27391683 (Accessed: 21 August 2023).

Sekhon, J., Ozcan, O. and Ozcan, S. (2023). *ChatGPT: what the law says about who owns the copyright of AI-generated content*. [online] The Conversation. Available at: https://theconversation.com/chatgpt-what-the-law-says-about-who-owns-the-copyright-of-ai-generated-content-200597 [Accessed 22 Apr. 2023].

Sielicka-Barylka, K. (2021) 'A Cross-cultural Review of the Impact of Wiki Education in Learning and Teaching', Diff, 3 February. Available at: https://diff.wikimedia.org/2021/02/03/a-cross-cultural-review-of-the-impact-of-wiki-education-in-learning-and-teaching/ (Accessed: 21 August 2023).

Soler-Adillon, J., Pavlovic, D. and Freixa, P. (2018) 'Wikipedia in Higher Education: Changes in Perceived Value through Content Contribution', Comunicar: Media Education Research Journal, 26(54), pp. 39–48.

Sparrow, L., Kershaw, L. and Jones, K. (1994) Issues in primary mathematics education: calculators: Research and curriculum implications. Perth, Australia: Mathematics, Science & Technology Education Centre, Edith Cowan University. Available at: https://ro.ecu.edu.au/ecuworks/6812 (Accessed: 21 August 2023).

Spataro, J. (2023) Introducing Microsoft 365 Copilot – your copilot for work, The Official Microsoft Blog. Available at: https://blogs.microsoft.com/blog/2023/03/16/introducing-microsoft-365-copilot-your-copilot-for-work/ (Accessed: 21 August 2023).

Swauger, S. (2020) Software that monitors students during tests perpetuates inequality and violates their privacy, MIT Technology Review. Available at: https://www.technologyreview.com/2020/08/07/1006132/software-algorithms-proctoring-online-tests-ai-ethics/ (Accessed: 17 March 2023).

Terry, O., **"I'm a student you have no idea how much we're using chatgpt",** The Chronicle of Higher Education, May 2023. Retrieved from: "https://www.chronicle.com/article/im-a-student-you-have-no-idea-how-much-were-using-chatgpt on 15/09/2023.

Turkle, S. (2007) 'Authenticity in the age of digital companions', Interaction studies, 8(3), pp. 501–517. University of Edinburgh (2020) New case studies publication on using Wikipedia in education, The University of Edinburgh. Available at: https://www.ed.ac.uk/information-services/about/news/2020/using-wikipedia-in-education (Accessed: 21 August 2023).

Turnitin, "**AI Writing Detection Capabilities - Frequently Asked Questions",** Turnitin, 2023. Retrieved from: https://www.turnitin.com/products/features/ai-writing-detection/faq on 15/09/2023.

Yeadon, W. et al. (2022) 'The Death of the Short-Form Physics Essay in the Coming AI Revolution'. arXiv. Available at: https://doi.org/10.48550/ARXIV.2212.11661.

FIND OUT MORE:

For more information about the workshop that led to this white paper, please see our website.

https://chatgpt-in-he.github.io/

Manchester Metropolitan University All Saints
Manchester
M15 6BH

@ManMetUni

@manmetuni

@manmetuni

The Manchester Metropolitan University

ManMetUni

Find more contact details at mmu.ac.uk/contact-us
We are committed to ensuring that all of our materials are accessible.

Manchester Metropolitan University

CPHC The Council of Professors and Heads of Computing