Online Examinations in a Large Australian CS1 Course

Bryn Jeffries*
Grok Academy
Sydney, New South Wales, Australia
bryn.jeffries@grokacademy.org

Timothy Baldwin
The University of Melbourne
School of Computing and Information Systems
Melbourne, Victoria, Australia
tbaldwin@unimelb.edu.au

Marion Zalk
The University of Melbourne
School of Computing and Information Systems
Melbourne, Victoria, Australia
mzalk@unimelb.edu.au

ABSTRACT

Due to the Covid-19 pandemic, universities were required to shift teaching online in 2020, including exams. This rapid move required many changes and compromises to be made in order for students to take their exams remotely. We report on the experiences of running online exams for a large CS1 course at an Australian university, describing the changes made to the online learning platform to support this task, as well as the authoring of the exam and the support provided to students. Several clear instances of academic misconduct are presented to illustrate some of the issues endemic to online exams. Broadly, this approach to examination was a success that we plan to continue in future years, and we reflect on how the approach can be improved.

CCS CONCEPTS

• **Social and professional topics** → **CS1**; **Student assessment**; • **Information systems** → *Near-duplicate and plagiarism detection.*

KEYWORDS

CS1, online exams, academic honesty, cheating, Covid-19

ACM Reference Format:
Bryn Jeffries, Timothy Baldwin, and Marion Zalk. 2022. Online Examinations in a Large Australian CS1 Course. In *Australasian Computing Education Conference (ACE '22), February 14–18, 2022, Virtual Event, Australia.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3511861.3511864

1 INTRODUCTION

The Covid-19 pandemic caused a great deal of strain on educators and students, as universities around the world reacted to developments throughout 2020 by closing campuses and moving courses online. While approaches varied between countries and the institutions therein, disruption was prevalent everywhere, often with students suffering the most, and with existing inequalities exacerbated [1]. This transition has also spurred innovation that would have hitherto been hard to justify. In Australia, which initially had the good fortune to have far fewer coronavirus cases than most other countries, a large challenge was maintaining numbers of overseas students, who form a significant fraction of enrolments, despite restrictions on international travel. A critical issue was how to conduct summative assessments that would traditionally have taken place in pen-and-paper mode, when students were unable to visit campus and may not even have been in the same country. In both Australia and New Zealand the success of a zero-tolerance approach to Covid-19 cases initially suggested that the changes in assessment would be temporary. More recently the frequent localised lockdowns caused by the Delta variant have led to emergency measures becoming the new normal.

Decisions varied widely across Australasian universities on how exams should be managed [13]. While some universities abandoned exams altogether in favour of alternative assessment (similar to the approach successfully demonstrated in Singapore [10]), most opted to run exams in an online form. Many exams used the existing quiz facilities of the institution's Learning Management System (LMS), or external submission systems such as Turnitin. Institutions were evenly divided on their use of remote proctoring tools [8], especially since they often proved to be very unpopular with students. After the first semester's offering, most institutions planned to re-run with the same approach in the following semester, but intended to review requirements for an improved solution for the longer term. This was consistent with experiences in North America, where online assessment was found to be consistent with in-person exams [15].

A major concern in running exams online is the increased risk of cheating, which is acknowledged to be common in university courses. Recent anonymous self-reporting in a large algorithms course at an American university found 42% of students admitted to cheating [4], prior to the move to online delivery/assessment. Students are more likely to cheat off each other [2], and exams are more prone to collusion when communication between students cannot be controlled, regardless of factors such as gender or academic calibre [9]. Furthermore, estimates based on surveys suggest 5–6% Australian students make use of contract cheating services [5]. In 2020 the Australian government made it illegal to provide cheating services, in both commercial forms and "friends, family or community members" [11], but the impact of this legislation will take some time to be evident. The cost of academic honesty investigations is also very high, both for students who often view them as akin to criminal trials [12], and educators in terms of time spent in examining submissions for indications of cheating, and the preparation of evidence to submit to third-party committees (who are often not experts in the subject matter) to support the academic integrity charges.

*Also with The University of Sydney, School of Computing Science.

ACE '22, February 14–18, 2022, Virtual Event, Australia

ACM ISBN 978-1-4503-9643-1/22/02…$15.00
https://doi.org/10.1145/3511861.3511864

This article reports on the experience of teaching Foundations of Computing[1], a large CS1 course at the University of Melbourne, with a focus on the mid-semester test and final examinations in 2020. Prior to the Covid-19 pandemic, the course was already designed to have a strong online component, with interactive online worksheets, online submission of student projects, an active discussion forum, and a one-on-one remote tutoring facility. However, examinations had previously always been run on-campus as paper-and-pen activities under invigilated conditions. The University made the decision for exams to be hosted online, without remote proctoring. Since the questions asked in the exam were predominantly code-based, the decision was made to offer the exam within the existing online platform[2] with which students were already familiar, rather than present students with another system.

This paper continues in Section 2 with an overview of the course, including some of the changes made to the structure in response to the demands of the pandemic. Section 3 focuses on the delivery of the mid-semester test and final exam, describing the technical changes implemented to the online learning platform, and the redesign of the exam questions to make them suitable for the online format. A subjective assessment of the impact of running exams in this way is presented in Section 4. An analysis of the degree and forms of academic dishonesty conducted by students is presented in Section 5. Section 6 presents our conclusions and thoughts for how the course's examinations may be run in future.

2 COURSE OVERVIEW

The course presented herein is the first course for the University's computing majors, and assumes little or no background in computer programming. It is intended to teach students how to design and write basic programs in Python. The course is delivered over a 12-week period, and is run in both semesters of each year (in Australian universities, Semester 1 typically runs through March, April and May, while Semester 2 runs through August, September and October). In 2020 approximately 1100 students were enrolled in Semester 1, and 750 in Semester 2.Each week there are three 1-hour lectures, one 1-hour tutorial, and one 1-hour laboratory. The lecture is where students are introduced to the theory and practical applications of that theory. The tutorial is complementary to the lectures, and is primarily made up of more practical examples of concepts learned in lectures, that tutors work through with students. A tutorial is a small-group learning environment where there is one tutor for around 24 students. A laboratory is more informal, where students work through programming worksheets at their own pace in order to practice their programming. Each laboratory is staffed by around one demonstrator per 12 students, who are available for one-on-one assistance as required. In addition to this, we provide online support for students. Demonstrators monitor a tutor messaging chat application, and provide both asynchronous and synchronous support to students, largely in the context of completing programming exercises and projects. For more general questions and questions not relating to assessable material, a Discourse forum is run for the course. This provides the key means for communication between the students and teaching staff. These services were maintained with the move online.

Assessment is structured as follows. Students are assigned online worksheets, which include an explanation of the week's concepts and pose a variety of programming exercises. The worksheets are hosted in the course's online learning environment. Each worksheet problem provides automated marking, usually through a set of tests comparing output of the student's submitted code to the expected outcome under different conditions, and providing the student with immediate feedback from these tests. Students may attempt each problem as many times as they like within the 2-week availability period for each worksheet. These worksheets form 10% of a student's total mark for the subject. Once the deadline has passed the worksheets are still available (not for marks), and students can attempt them as many times as they like. Solutions are also released after the submission deadline for a given worksheet, so students can look over different approaches to the same problem. As mentioned, students were able to ask for help with questions through an in-platform tutor messaging service, and this means of communication became more heavily used after teaching moved wholly online. Additionally, tutors ran in-person Zoom sessions for students when it was felt that a more direct means of communication was required.

In addition to weekly worksheets to practice programming skills, students are required to complete two projects. Submissions are made through the course's online platform, and are marked through a mix of automated tests and manual (rubric-based) marking by tutors. These assessments each are worth 15% of a student's mark. Together with the weekly worksheets, this continual assessment forms 40% of the student's final mark.

Two summative tests are delivered in the course: a 1-hour mid-semester test worth 10% of the final mark; and a 2-hour final examination at the end of the semester, making up the remaining 50% of the subject mark. Prior to 2020, both of these were run in pen-and-paper mode under standard exam conditions without access to computers or online materials, and with standard invigilation. In 2020, the mid-semester test and exam were both offered online, as described in the following sections.

In the first semester's offering of 2020, it became obvious in the first couple of weeks of semester that we would need to make changes to the subject delivery due to the pandemic. We discussed the changes with the 46 tutors and demonstrators to understand our teaching staff and their needs. From this we gathered that some tutors were very keen to move to online teaching, but there was some trepidation in delivering the content online. Some of the concerns were related to logistics, namely a suitable location for delivering the tutorials and the desire for the University to provide a room/space that they could use for that purpose. The use of a computer or internet was not an issue for most of the staff, although a couple of tutors were concerned about the speed of their computers and having the appropriate tools (e.g., whiteboard type application to teach, or having access to a room with a whiteboard that they could video themselves teaching). It also emerged that the teaching staff wanted time to familiarise themselves with Zoom and online teaching tools. We provided tablets for staff that required a more suitable computer, and offered Zoom workshops through the University to prepare staff for the move to online learning. The

[1] https://handbook.unimelb.edu.au/2020/subjects/comp10001
[2] Grok Academy, https://launch.grokacademy.org/

move to fully online delivery occurred by the end of the third week of semester, by which time all logistics was set up for staff involved with the delivery of the subject.

3 ONLINE EXAMINATION ENVIRONMENT

Since students were already very familiar with the online learning environment used for weekly worksheets and project submissions, it was a natural step for students to also undertake their exams within the same platform. From previous experience it was known that the platform was resilient to the sudden surges in usage expected from an exam, and its scale-out architecture ensured sufficient infrastructure resources could be easily allocated in advance of busy periods. Furthermore, since many students were stranded overseas, it was essential that students could attempt the exams from countries such as China that blocked access to many websites: reports from enrolled overseas students verified that all students could access the platform without resorting to VPN or similar workarounds.

Although several Australasian institutions opted to use remote proctoring services [13], the decision at this university was to avoid using any such service. Because students were free to take the exam from an uninvigilated environment, several key challenges for academic integrity were identified:

(1) a student could give their credentials to another individual to attempt the exam in their stead;
(2) two or more students could attempt the exam together, colluding on their answers; and
(3) a student could make use of unapproved resources to assist in answering questions.

These issues needed to be considered in the technical changes to the platform to support the hosting of exams, described below, and in the design of the exams as described in Section 3.3.

In order to familiarise students with the new format of exam, they were provided with several practice versions of the exam problems (with different questions to those in the actual exam, but of similar question style). Students were also offered two trial runs of the practice questions under timed conditions, to give them experience of the new exam environment.

There were additional concerns regarding student equity in holding the exam online, which needed to be taken into consideration in planning the logistics:

(1) how to deal with students with consistently unreliable internet who lacked the financial means to source alternative internet connectivity;
(2) how to deal with possible short-term internet outages on the student end; and
(3) how to offer flexibility in the timing of the exam (especially for students based overseas in regions such as Europe and the Americas, in substantially different timezones) without potentially exacerbating collusion through exam question "leakage".

For the small number of students in the first case, special provision was made to provide University-funded 4G internet dongles ahead of time, solving the issue. For the second issue, the decision was made to allow all students an additional 10 minutes' exam time specifically to deal with such short-term local outages, and students were informed to document any outages (as well as continually monitoring the connection via the exam platform), and that if the total affected time was less than 15 minutes they would not be eligible for special consideration. For the third issue, we determined what timezones all students were based in, and determined that we could offer one earlier sitting for a small number of students based in Europe/Eastern USA, and have the remainder of students complete the exam at one time to cover the remainder of the timezones.

3.1 Platform Changes

To make the platform suitable for examinations, a new timed exam mode was developed. In this mode, students are only able to access the assigned problems within a specified time window (controlled via a link from the LMS), for up to a pre-defined time. For the mid-semester test, for example, students were allowed up to 50 minutes to complete the test, starting any time within a 3.5 hour window. A countdown timer is visible throughout the exam period, with alerts as the end time approached. Since it was anticipated that some students would encounter unforeseen technical problems (e.g., major network outages), the exams could be configured to go over-time, such that students could make late submissions to problems that would be marked if a student's appeal was successful (and to deal with students with extra time allocations awarded through the University's Student Equity and Disability Support team).

To facilitate auditing of student activity, a log is recorded of when the student commences an exam session, including details of the browser, IP address, and IP-based estimated geographical location. The exam mode allows students to explicitly end their exam, with details of this event also logged. Anticipating that some students may accidentally end their session, or have it terminated due to network issues, the platform permits students to resume a previously ended session, provided the available time has not already ended: lecturers can use the session data to determine whether submissions in resumed sessions should be investigated after the exam.

The platform preserves a record of all submissions made by a student for a problem, including the time of submission, all files in the submitted workspace, and the outcome of all tests run on it. Additionally, the state of code before and after any paste event is saved, to assist investigations of plagiarism. For the exam mode, these saves were augmented by regular snapshots, once each minute, so that the progress of any suspect submission could be reviewed.

Marking of submissions for each problem was performed within the platform, using the platform's rubric-based manual marking facility and automatic testing as used for the worksheets (with the major difference that the results of the testing were not visible to students at the time of the exam). Coordination of marking was managed through the platform's marking dashboard. The dashboard was modified to provide tutors with better tracking of marked and unmarked exam problems. These improvements were implemented throughout both semesters in response to feedback from lecturers and tutors.

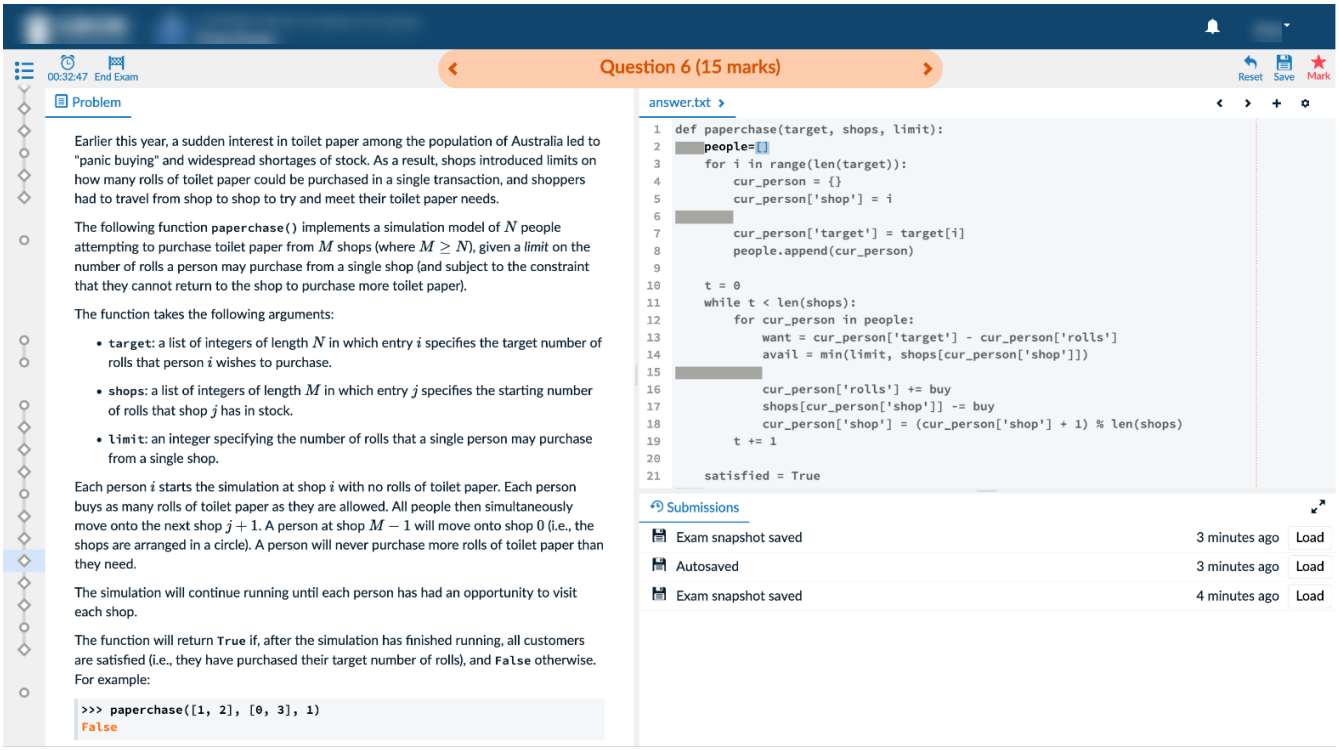

Figure 1: Example of a fill-in-the-blanks exam question. The problem is described in text in a panel on the left. The panel on the right has code to be edited by the user. Regions are highlighted in grey to show where the user should add code (including specification of the indentation level).

3.2 Support Channels

In the first 2020 offering of the subject, a Zoom session ran continually throughout the mid-semester test and exam period, for which students could join to ask for clarification of exam questions (as per standard exam sittings) or technical assistance with the environment at any time during the exam. In the second offering, Zoom was used in a similar way for the mid-semester test, however, this was replaced for the exam with a University-managed service that performed the same function.

3.3 Design of exam

The questions asked in the online assessments largely followed the style used in previous paper-based assessments. The question types included:

Code Re-writing The student is provided with a working function and must rewrite it to use a particular technique (e.g., replacing a `for` loop with a `while` loop).

Parsons Problem The student must re-arrange the provided lines of code to create a function that meets given criteria [6].

Example Input A function is presented to the student, who must suggest arguments that will result in certain outcomes (e.g., an instance of an input which exposes a particular class of bug in the code implementation).

Expressions The student must construct a single-line expression that evaluates to the given value, using specified operators.

Debugging For a given defective piece of code, the student must identify a number of errors and propose solutions for them.

Fill in the Blanks The student must complete a piece of code to operate in a particular manner, by adding code in specified regions.

Code writing The student must write a function from scratch, meeting specified criteria.

Long Answer The student must answer open-ended questions reviewing topics from the course, in plain English.

The mid-semester tests consisted of 4 questions, avoiding longer question types (code writing and long answers), while the final exam had 9 questions with a broader range of difficulty.

All problem types were implemented using the standard layout provided by the online platform, which presents each question as a separate "slide", containing a problem description alongside a code editor, into which the student could submit their answer. An example of a fill-in-the-blanks question is shown in Figure 1.

Since students had access to unrestricted resources at home, it was impractical to expect students could be prevented from testing their code by running it locally. The questions were therefore developed while keeping in mind that some students could test

code outside the provided environment. To keep things equitable, problems for which such access would provide a strong advantage included a scratch-pad interpreter to run code within the learning environment, and guidance was included in the question text to encourage students to make use of this facility.

4 ASSESSMENT OF IMPACT

In this section we reflect on how the online exams ran in both semesters of 2020. Discussion of academic integrity is reserved for Section 5.

For students, the change from written to typed exams provided some obvious advantages. Students were already comfortable with typing as part of their usual activities and as such the format was more familiar than hand-writing, especially of code. This was supported by the use of the same online learning environment for assessments and worksheets during the semester, ensuring that students were familiar with the interface and functionality. The main vehicle for students to provide feedback of their experience in a subject offering is through the institutional end-of-semester surveys. Unfortunately these surveys are conducted before the exam period, and so do not capture useful information for this work. Some students do discuss their experiences in public forums such as Reddit, which has channels (*subreddits*) specific to the institution and the course). A search of these channels revealed several complaints to the effect that some students would have liked more time, and that certain code-writing questions were very hard — both of which could be said of previous paper-based exams. Several students also commented positively on not having to hand-write code.

For the teaching staff, it is easier to read students' typed answers, and therefore online exams remove the hassle of reading and deciphering pen and paper answers, especially in instances where the response has been heavily edited. The online marking process was considerably easier to perform and coordinate. Since there were no paper scripts, there wasn't a need for markers to physically exchange scripts, and multiple tutors could mark different parts of one student's 'paper' simultaneously. This was particularly advantageous given Covid-19 protocols that prevented coordination of marking in a shared physical space. With a large student cohort, it is easy for some exam papers to be misfiled, resulting in stress and disruption until the missing papers are found. The use of the marking dashboard made it clear what was being marked and a paper could not be 'misplaced'. The exam marking process was further simplified in terms of data entry: with paper exams, the transfer of marks and data entry can be quite time consuming, especially when done carefully with error-checking protocols. In this regard, online exams reduced the overall time required to mark the exam papers and process the marks.

The use of an electronic marking rubric for each problem streamlined the process of marking answers and made it easier to keep marks consistent. Marking was also assisted by the results of automated tests that were applied to most problems. For instance, the code writing questions included unit tests to check a student's submission worked for all corner cases. The results of these tests were not made visible to students during the exam, but were available to tutors marking the submission, allowing complete solutions to be marked rapidly and attention to be focused on submissions with more nuanced problems which failed the unit tests (where partial marks were often still warranted).

Table 1: Code Writing exam questions

Question	Length[1]	Students[2]	Attempts[3]
Semester 1 Q7	10 lines	1132	6801
Semester 2 Q7	10 lines	757	3978
Semester 2 Q8	21 lines	697	4950

[1] Size of reference solution
[2] Number of distinct students attempting question
[3] Total number of submissions and saves analysed

Some students wrote complete answers to exam questions but neglected to submit the answer for marking. This was surprising, since the submission idiom was used throughout the semesters' teaching, both for worksheets and project submissions. Some students may simply have overlooked this step under the pressure of exam conditions, while others may not have engaged with the previous course activities and slipped through the system. In response to this issue, a change was introduced in Semester 2 to allow the tutor to change the state of a response from a saved snapshot to a submitted snapshot. This allowed the student's response to be marked if a student had a technical issue or forgot to submit.

Inevitably, some students encountered technical issues. Many of these were due to network problems, which were impossible to control when students were attempting the exam remotely. Importantly, many issues which impacted on student performance could not be diagnosed and resolved during the exam, some of which were the result of technical glitches (especially for the early implementation of exam environment in Semester 1), and some of which were due to network issues or user error. This course was far from alone in encountering such issues, however, and the University introduced a process for students to request to re-take the exam on the grounds of technical issues. In the first semester of 2020, 1.3% of students sat a supplementary exam, and 2.2% did so in the second semester. These numbers were not noticeably greater than in previous years, and they include reasons (such as illness) other than just technical issues.

5 ACADEMIC DISHONESTY

The key means of identifying candidates for academic dishonesty was plagiarism detection. The platform provides plagiarism detection via similarity checking of all submissions for a problem. This process includes syntax normalisation of the (Python) code, followed by an implementation of the winnowing local code fingerprinting algorithm used by the popular MOSS tool[14]. This comparison is applied across the entire submission histories of all participating students. Deeper inspection can also be performed by comparing pairs of students' submission and saved code histories. This process was used to explore evidence of some of the forms of academic dishonesty described in Section 3. This could be identified most clearly in Code Writing problems, where students' submissions typically varied sufficiently that similar submissions were likely to indicate some form of cheating. There were three problems of this type across the final exams of both semesters in

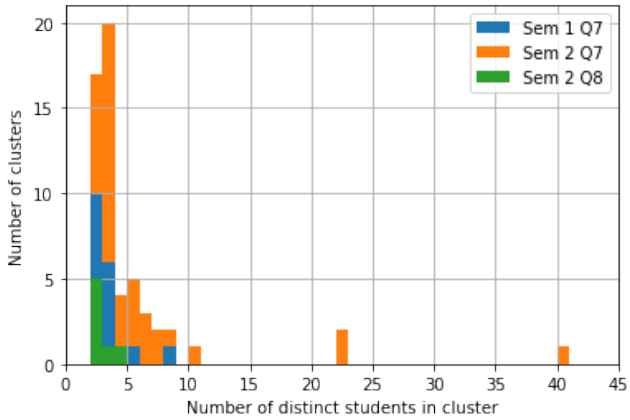

Figure 2: Histogram of cluster sizes of matching submissions

2020, summarised in Table 1. As can be seen from the table, not all students attempted all questions, and each student's code history for a question contained multiple submissions and saves.

5.1 Collusion

The MOSS-style similarity threshold at which a pair of code submissions should be judged suspicious depends upon the size of the submissions [3]. Often programming exercises have very small solutions, in which case only 100% matches after syntax normalisation and comment removal need be considered. To this end, the platform automatically identifies clusters of such submissions.

Of the 110 clusters identified for the selected exam questions, manual inspection revealed 24 across the three problems to be incomplete implementations, and were therefore excluded from further investigation. Of the students making any submissions to the respective exam question, 1.9% appeared in one of the remaining clusters for Semester 1 Q7, 7.5% for Semester 2 Q7, and 1.0% for Semester 2 Q8. Figure 2 shows the histogram of the cluster sizes for each of the three exam questions — note that a student can appear in multiple clusters, since their submission history may contain multiple matches.

Deciding whether any such clusters is evidence of collusion is a subjective task, but a number of traits can inform the process. A sudden change in approach, or a jump from no code to a complete solution, suggests that a student may have pasted code from another source. Collusion is very likely when this pasted code matches characteristics other than syntactic equivalence, such as variable names, or comment text. Sometimes it is something small, like a comment with an incorrect spelling, noticed later and fixed by the student, but preserved in the history showing that both versions of the code came from the same source. Matches in syntax errors are also strong indicators, since these tend not to be in the same place in independently-written code.

Students appear to be aware that submitting code identical to a peer's (or website) may look suspicious, and the code history for several candidate cases of plagiarism showed evidence that a pasted identical solution was modified to mask its origins, through changes in comments and variable names, or sometimes changing between `while` and `for` loop structures (Level 1, 2 and 5 code changes, respectively, as characterised in [7]).

Table 2: Clusters containing externally available solutions

Question	Chegg requests	Chegg submissions	Matching clusters	Cluster size(s)
Semester 1 Q7	2	2	2	8, 4
Semester 2 Q7	7	6	1	40
Semester 2 Q8	5	4	0	-

5.2 External Resources

There are many websites and commercial services providing assistance for assessments in general, including code-writing activities. Surveying all such services is infeasible, but the teaching team were aware that many students have made use of one service in particular, Chegg[3]. This website allows paying members to ask for help on a question, to which an active team of 'tutors' provide answers sufficiently rapidly that they could be used during the course's exams. Answers are available to all members, not just to the original asker. Questions are posted anonymously, while solutions can be posted anonymously or have the tutor's username optionally declared.

After each exam, the Chegg website was manually searched by the authors for any questions and associated solutions to the exam problems listed in Table 1. Details of the pages relating to each exam question are shown in Table 2. Each post received 1 solution, with the exception of Sem. 2 Q8 for which one post received a solution in the wrong language (C++ instead of Python), and one received no answer. For Sem. 2 Q7 four of the posted submissions were all provided by the same member, and were syntactically identical.

The posted solutions were all included as artificial student submissions for the purposes of plagiarism detection. In order to better understand the timeline of student activity, the submission times of these artificial submissions were backdated to the respective timestamps found in the website.

Only 3 clusters matched any of the Chegg submissions, as summarised in Table 2. Manual inspection of the submission history of students in these clusters revealed several unambiguous cases of copying from the external sources. A quirk of the Chegg website is that code is often displayed without indentation, losing important whitespace semantic data for Python code. Several student submissions reproduced these distinctive syntax errors, which were preserved in their development history even if they were fixed ahead of their final submission. In addition, we noted that in several of these cases, students added a surfeit of comments, perhaps with a view to showing the marker that they really understand the solution and therefore it must be their own.

It is important to highlight that although we may suspect that code is copied — either from another student or from a website — these indicators become important for an Academic Honesty panel and demonstrating collusion or plagiarism. For this reason, we thoroughly consider previous saves/submissions of the code

[3] www.chegg.com

to gain a complete picture of how and when the code was developed. Although the submission of interest is the code in the final submission (as that is the work submitted for examination), these 'red flags' are integral in demonstrating and explaining to a person (who may not have a programming background), why in fact the code is not the student's own.

5.3 Shared Credentials

Beyond the cases observed through plagiarism, there were indicators of suspicious behaviour in the access logs of some students. Several students accessed the same exam from multiple IP addresses. This can occur innocently when their device is connected to a complicated or unstable network, or if the student switches from using a VPN. More suspicious is when the access sessions use different web browsers and operating systems, and when the IP addresses relate to different geographical locations. These cases suggest that students had shared their login credentials with other individuals, potentially with the aim of getting assistance in completing the exam.

6 CONCLUSIONS AND FUTURE WORK

Despite the difficulties encountered by the move to run examinations online, it has presented an opportunity to experiment with a change of approach that will likely be continued in the future. Delivering exams online for this CS1 course brings many benefits, including a more comfortable/naturalistic experience for students, and a faster and more scalable marking experience for the teaching team. The chief disadvantage is the greater risk of cheating by students, of which we have found significant evidence. Even so, the number of suspicious submissions is a small proportion of such a large cohort, and since the final exam forms 50% of the overall grade, there are other activities that will affect the end result.

When the pandemic eventually passes, and 'normal' teaching resumes, online exams could be conducted on campus under invigilated conditions. This would preserve all the benefits of the new exam format, while minimising the risks of cheating. Alternatively, there may be renewed interest in using remote proctoring services to address these issues. A middle-ground could be to place some limited restrictions on where a student accesses their exam, by requiring them to register their intended location and system details ahead of the exam. However, such a step would not prevent collusion by students via other channels.

Based on the experience from 2020, there are numerous improvements planned for future exams. The use of automation in marking exams was a learning process in 2020, and there is significant room to extend this. Some problems might be marked completely automatically, or in a semi-automated fashion such that manual marking is only performed for submissions where some tests fail. It would also be possible to provide limited automated feedback to students for some questions, for instance to inform students that their submission contains syntax errors, to allow students to submit a better (and more easily marked) version.

Within the platform that are some question types, such as Parsons problems, that could be better presented using a dedicated presentation format, and work to implement these is already underway. Some of the plagiarism analysis conducted on the 2020 exam submissions has identified reporting processes that can be automated in the future.

REFERENCES

[1] Cristina Alarcón López, Mathias Decuypère, Joyeeta Dey, Radhika Gorur, Mary Hamilton, Christian Lundahl, and Elin Sundström Sjödin. 2021. Dancing with Covid: Choreographing examinations in pandemic times. *European Educational Research Journal* 20, 4 (2021), 403–422. https://doi.org/10.1177/14749041211022130

[2] Rebecca Awdry and Bob Ives. 2020. Students cheat more often from those known to them: situation matters more than the individual. *Assessment & Evaluation in Higher Education* 0, 0 (2020), 1–15. https://doi.org/10.1080/02602938.2020.1851651

[3] Kevin W Bowyer and Lawrence O Hall. 1999. Experience using "MOSS" to detect cheating on programming assignments. In *FIE'99 Frontiers in Education. 29th Annual Frontiers in Education Conference. Designing the Future of Science and Engineering Education. Conference Proceedings (IEEE Cat. No. 99CH37011* (San Juan, Puerto Rico, USA), Vol. 3. IEEE, New York, NY, USA, 13B3/18–13B3/22. https://doi.org/10.1109/FIE.1999.840376

[4] Nathan Brunelle and John R. Hott. 2020. Ask Me Anything: Assessing Academic Dishonesty. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education* (Portland, OR, USA) *(SIGCSE '20)*. Association for Computing Machinery, New York, NY, USA, 1375. https://doi.org/10.1145/3328778.3372658

[5] Phillip Dawson. 2020. *The prevention of contract cheating in an online environment.* Technical Report. TEQSA. Downloaded from https://www.teqsa.gov.au/sites/default/files/prevention-contract-cheating-in-online-environment-web.pdf?v=1587691121.

[6] Yuemeng Du, Andrew Luxton-Reilly, and Paul Denny. 2020. A Review of Research on Parsons Problems. In *Proceedings of the Twenty-Second Australasian Computing Education Conference* (Melbourne, VIC, Australia) *(ACE'20)*. Association for Computing Machinery, New York, NY, USA, 195–202. https://doi.org/10.1145/3373165.3373187

[7] J.A.W. Faidhi and S.K. Robinson. 2020. An empirical approach for detecting program similarity and plagiarism within a university programming environment. *Computers & Education* 11, 1 (2020), 11–19. https://doi.org/10.1016/0360-1315(87)90042-X

[8] Mohammed Juned Hussein, Javed Yusuf, Arpana Sandhya Deb, Letila Fong, and Som Naidu. 2020. An Evaluation of Online Proctoring Tools. *Open Praxis* 12, 4 (2020), 509–525.

[9] Maria Krambia Kapardis and George Spanoudis. 2021. Lessons learned during Covid-19 concerning cheating in e-examinations by university students. *Journal of Financial Crime* ahead-of-print, ahead-of-print (June 2021). https://doi.org/10.1108/jfc-05-2021-0105

[10] Norman Tiong Seng Lee, Oka Kurniawan, and Kenny Tsu Wei Choo. 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* (Virtual Event, Germany) *(ITiCSE '21)*. Association for Computing Machinery, New York, NY, USA, 352–358. https://doi.org/10.1145/3430665.3456323

[11] The Parliament of the Commonwealth of Australia. 2020. Tertiary Education Quality and Standards Agency Amendment (Prohibiting Academic Cheating Services) Bill 2020. Downloaded from https://parlinfo.aph.gov.au/parlInfo/search/display/display.w3p;query=Id%3A%22legislation%2Fbillhome%2Fr6483%22.

[12] Penelope Pitt, Kevin Dullaghan, and Wendy Sutherland-Smith. 2020. 'Mess, stress and trauma': students' experiences of formal contract cheating processes. *Assessment & Evaluation in Higher Education* 0, 0 (2020), 1–14. https://doi.org/10.1080/02602938.2020.1787332

[13] Michael Sankey. 2020. *COVID-19 Exam Software Survey – 2020.* Technical Report. ACODE. Downloaded from https://www.acode.edu.au/mod/resource/view.php?id=3874.

[14] Saul Schleimer, Daniel S. Wilkerson, and Alex Aiken. 2003. Winnowing: Local Algorithms for Document Fingerprinting. In *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data* (San Diego, California) *(SIGMOD '03)*. Association for Computing Machinery, New York, NY, USA, 76––85. https://doi.org/10.1145/872757.872770

[15] Larry Yueli Zhang, Andrew K. Petersen, Michael Liut, Bogdan Simion, and Furkan Alaca. 2021. A Multi-Course Report on the Experience of Unplanned Online Exams. In *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education* (Virtual Event, USA) *(SIGCSE '21)*. Association for Computing Machinery, New York, NY, USA, 17—-23. https://doi.org/10.1145/3408877.3432515