DON'T LET COVID-19 RUIN OUR COMPETENCY-ORIENTED EXAMS: LESSONS LEARNED FROM AN AD-HOC TRANSFORMATION OF PRACTICAL PROGRAMMING EXAMS FROM LAB TO ONLINE

Manfred Meyer

Westfälische Hochschule – Westphalian University of Applied Sciences
Bocholt (GERMANY)

Abstract

Competency-oriented exams offer a wide range of advantages, especially where the use and mastery of third-party applications and tools play an important role. Therefore, we developed a competency-oriented setup for both our programming classes and exams ensuring their constructive alignment. Exams were moved to the computer lab and designed to test both conceptional skills as well as the use of state-of-the-art programming tools. At the peak of the COVID-19 pandemic, when exams had to be moved from lab to online, we needed to design an online setup for our practical programming exams preserving the competency-oriented approach and its constructive alignment as well as the validity, reliability and fairness of the exams. The key was to use the same online tools that have been introduced for running lectures and practical classes offering almost the same learning experience as before the pandemic. However, to ensure the validity and fairness of the exams, some kind of online supervision needed to be implemented as technical solutions were found to be either unusable or not working properly in our case. This paper discusses the driving factors, the resulting technical and organizational setup as well as students' feedback and lessons learned for further improvements. Therefore, COVID-19 has not been able to ruin our competency-oriented programming exams.

Keywords: Competency-Oriented Exams, Online Programming Exams, Constructive Alignment, Exams with Third-Party Applications, Online Supervision.

1 INTRODUCTION

Almost a decade ago, we redesigned our first-year programming courses (GDP1 and GDP2 at our university, internationally often referred to as CS100/101 courses) in a stringent competency-oriented manner. We now strictly focus on the key competencies of (1) understanding a problem statement, (2) modelling the required data and deriving an algorithm to solve it, and (3) implementing and testing the resulting Java program using Eclipse as state-of-the-art Integrated Development Environment (IDE).

In addition to introducing the social learning platform Perusall [1,2] in order to get students better prepared for the lectures, enforce social interaction between peers (which later became quite valuable and important during the pandemic) and leave more of the precious contact time for in-depth discussion of "hard topics" (as identified by Perusall's "Confusion Report") and the use of "Just-in-Time Teaching (JiTT)" methods [3] like Peer Instruction or Flipped Classroom, the weekly practical classes were designed so as to apply the newly introduced programming concepts (from data types, control structures, functional abstraction through object-oriented design and programming to GUI development using JavaFX, concurrency/threads etc.) to given programming problems of increasing complexity but at all times focusing on the full-range task from problem understanding to successful implementation. According to Bloom's Revised Taxonomy of Learning Domains [4], the required cognitive task falls into the highest category ("creating") – or level 5 ("synthesis") in Bloom's original taxonomy [5].

Consequently, in order to ensure the 'constructive alignment' of learning and assessment according to Biggs & Tang [6], students are informed from the very beginning of their classes, that it is exactly this competency that is getting tested in the final exams [7]. There are no questions like naming all primitive data types in Java or transforming a for-loop in an equivalent while-loop, but knowledge about data types or loop-statements is nevertheless essential to solve the given programming exercise.

Such competency-oriented exams offer a wide range of advantages over more abstract or more theoretical exam setups. They improve students' learning motivation, as well as their perception of the relevance and fairness of an exam. Additionally, competency-oriented exams usually do a better job of making students focus on acquiring relevant skills, rather than mere rote learning "for the exam" [8].

This approach, of course, requires that the final exams can be held in almost the same way as the practical classes, thus using state-of-the-art programming tools instead of “coding on paper” as done previously and was formally required until recently at our university.

2 COMPETENCY-ORIENTED PRACTICAL PROGRAMMING EXAMS

In 2015, we received funding to buy a set of 30 identical notebook computers mainly for the practical exams but also to be used in practical classes when students forgot or were not able to bring their own device. Thus, we were able to offer an identical environment to all students taking part in the exam. However, there were a few more requirements we had to respond to:

- The examination regulations for our course at that time did not allow any type of e-assessment. Therefore, we had to develop a setup that could be finally accepted as a “written exam” though: students were still given special exam paper on which normally they have to write down all their work and submit for marking. However, we formally accepted the use of the given notebook computer with the pre-installed programming environment (Java, Eclipse) and other tools and provided a print-out service for them during the exam session, such that they could finally still submit the required bunch of paper thus fulfilling all formal requirements as set by the examination regulations. Therefore, any hand-written remarks on or changes to the printouts were of course regarded as part of their submission and taken into account in the marking.
- Additionally, the use of computers for the exams may lead to cheating by accessing external sources on the Web (however to some extent that would be acceptable in a truly competency-oriented exam, e.g. using web resources like stackoverflow.com as programmers do in practice) or communicating with peers, friends or others.

Thus, we came up with the following technical setup for the programming exams held in our lab on campus as shown in Fig. 1.

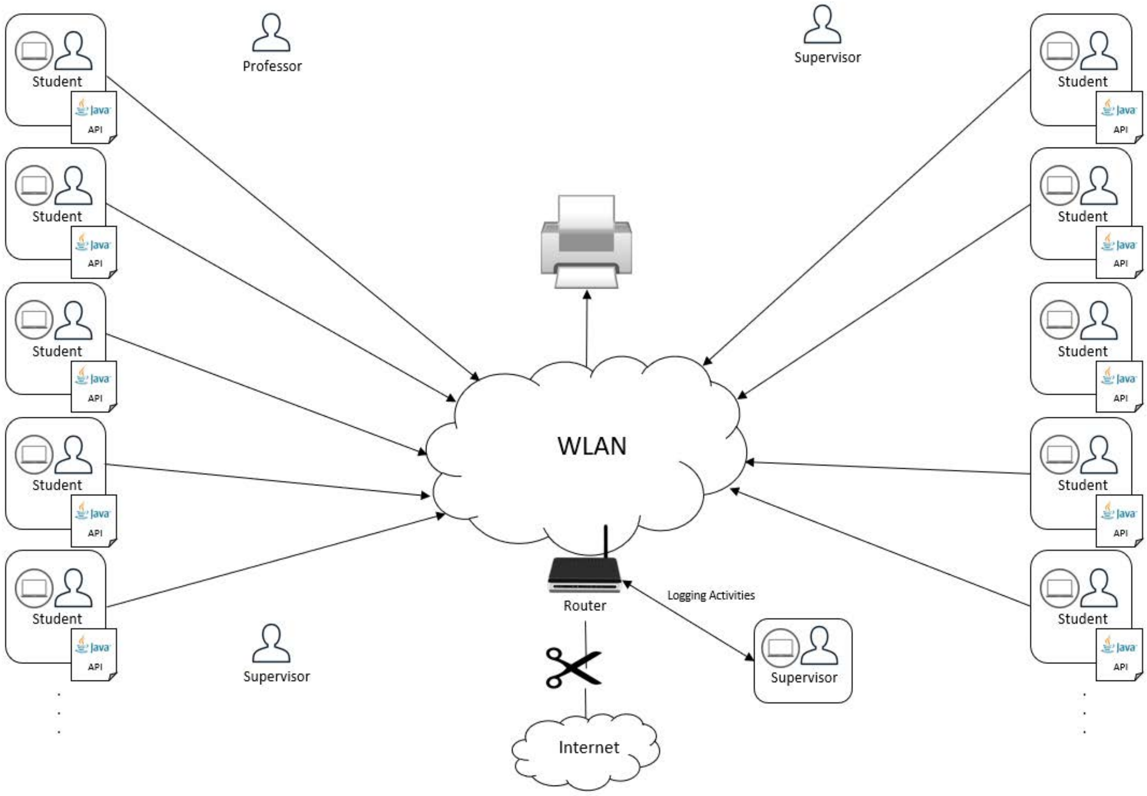

Figure 1. Technical setup for our programming exams on campus.

Students are using identical notebook computers with pre-installed programming tools (Java, Eclipse) and a local copy of the Java API Documentation. All computers are connected to a dedicated WLAN router throughout the exam. Students are advised not to disconnect at any time which would be regarded as cheating as it may allow connecting to a different network, e.g. through a personal smartphone. Consequently, the access protocol of the WLAN router is being saved for further investigations if needed. Through the WLAN, students can connect to a dedicated printer also located in the lab in order to print out relevant program files together with a header carrying their name and seat number such that the exam supervisors can deliver the printout to them for later inclusion in the official paper submission. Additionally, the computer's workspace is being stored on separate USB devices for easier checking of the submitted work for syntax, executability and correctness.

This setup was first used for programming exams in 2016 and since then it allowed to run proper competency-oriented exams ensuring both validity and fairness with a reasonable number of staff (supervisors in the lab) required. Additionally, the feedback from students was also very positive ("we don't need to learn for the exam"). However, the Gaussian grade-distribution curve somehow got inverted as is often observed with competency-oriented exams on a higher taxonomic level. While the number of fails did not change much (actually, it slightly increased), a general shift from average to good in marks could be observed.

3 COVID-19 AND THE TRANSFORMATION TO ONLINE EXAMS

When the COVID-19 pandemic hit the world in Spring 2020, both lectures and practical classes had to be moved online. With the use of video conferencing (Zoom) and other tools for social-learning (Perusall), audience response (Mentimeter) or visual collaboration (Miro) in addition to the learning-management system (Moodle which has been in use since long ago), we were able to offer almost the same learning experience as before the pandemic. This also holds true for the practical classes where groups of students are now collaborating in various Zoom breakout rooms sharing screens with their peers instead of sitting around group tables in the lab while the lecturer can walk around the breakout rooms as between tables in the lab. Students can even call for help in Zoom and give online control over their computers to our teaching staff for assistance just as they would do in person in the lab.

Thus, the transition of classes to online, including the practical programming classes from the lab, worked well in our case as the target of our classes (computer programs in Java) is virtual anyway, classes are not too large (around 30-50 students in lectures) to allow interactions even in Zoom, and we found suitable online tools offering the same (or even more) functionality as their physical counterparts, e.g. Mentimeter instead of hand-held audience-response systems ('clicker') we used before [9] or Miro instead of wall-mounted whiteboards in lecture theatres and labs.

As the practical exams could still take place in the lab on campus during the first year of the pandemic (except for the very first weeks in March/April 2020 when universities were closed completely and thus no exams could be offered at all), we were able to maintain our competency-oriented teaching and assessment approach using the setup as described above and shown in Fig. 1.

However, when COVID-19 incidence rose dramatically in early 2021, state regulations hardly allowed any exams being held on campus. Also, since some of our students regarded themselves as being more vulnerable to COVID-19 due to some medical conditions, it became clear that we needed to transform exams from labs to online at students' homes at a very short notice.

3.1 Requirements for Online Programming Exams

While exams for other courses have been moved to open-book or take-home exams, automated tests and e-assessments or just any kind of unsupervised homework, these options cannot be used for our programming course as we wanted to preserve the following main characteristics of our exams:

- Constructive alignment: As the entire course was organized around training the students' problem-solving and programming competencies, we wanted to preserve its constructive alignment meaning that in the exam we keep the promise and students will not be asked now to answer questions on theoretical concepts or even some isolated practical aspects but to practically apply what they have learned, namely to demonstrate the competencies they have achieved in order to solve some given programming problems using state-of-the-art tools.
- Validity: Of course, the exam needs to be still valid. Additional to focusing on appropriate content (from the classes) and asking for relevant skills as mentioned above, we need to ensure that the

exam can discriminate those students who have mastered the course from those who have not. Only those students should pass the exam that have demonstrated the required competencies relevant for their further courses and professional career later on.

- Reliability: This aspect usually refers to how dependably or consistently an exam measures the required competencies: If a student takes the exam again, will he or she get a similar score, or a much different score? As students cannot retake an exam that they have passed already, this cannot be answered easily from only the exams. On the contrary, that means that students who failed the exam and take it again without much training before, are likely to fail again. Unfortunately, that in fact had to be witnessed in more cases now with the competency-oriented exams than before. Anyway, if we also include the student's performance in the latest practical classes before the exam, it can well be observed that those students performing well on these exercises do also pass the exam with better scores then.
- Fairness: Finally, the exam itself needs to be equally fair which means that all students are given the same opportunity to achieve the right grade, irrespective of their social or economic background, their technical equipment or personal abilities or disabilities. In case of the latter, students get granted a different exam setup to encompass their disadvantages (NTA), e.g. using special tools or extended time to work on the exam, that may also lead to an entirely different type of exam, e.g. an oral exam with more interaction instead of a written or a lab exam. As regards the other aspects like social or economic background, this becomes quite crucial for the design of our new online exams setup: In contrast to other exams covering very special topics or referring to some special notation or a given textbook, our exams "only" require state-of-the-art Java programming skills that may get easily found within the students' family or friends.
  Thus, failing to effectively prevent student interaction with peers or externals during the exam could well lead to unfairness, i.e. serious disadvantages for those who don't have an academic family background or programmers as friends to ask or those socially less integrated in their peer group and thus being excluded from a "cheater's network". On the other hand, a strong economic background would unfairly help those students who can afford to pay for such external exam support now widely offered on eBay and other platforms.
- Comparability: Students can choose the time when to take the exam for a course they attended as exams are normally offered three times a year. Some students even go for their first try more than a year after the respective course. As we could only offer the exams for our GDP classes online, we had to ensure that conditions for the students are as much the same as possible for those taking the lab exam as described in the previous section and those taking the online variant in their own homes. Although state regulations would allow for a completely different exam setup, e.g. moving to unsupervised homework, from the academic point of view this wasn't an option for us at all.

In order to preserve these properties of our programming exams, we had to design the exam setup in a way:

- that students will still be asked to solve given programming problems using state-of-the-art tools (for constructive alignment and validity) now on their own computers rather than on those in our lab and
- that synchronous communication with peers or others needs to be prevented (for fairness and also reliability and comparability).

Additionally, when faced with this challenge we had to quickly come up with a solution that also needs to be practicable as regards time for preparation and introduction as well as needed personnel and technical resources for running the exams.

3.2 A Setup for Fair Competency-Oriented Online Programming Exams

As discussed, the main goal was to keep the competency-oriented approach having students demonstrating their competency in developing and testing a computer program to solve a given problem using state-of-the-art tools while at the same time preserving fairness, validity and comparability of the exams. Therefore, we needed to have the exam setup at the student's home as close as possible as in the lab at university.

As the programming environment (Java, Eclipse/IDE etc.) is already available on the students' computers from the practical classes, the main issue came with uncontrolled use of internet access in order to prevent communication with peers or others:

- Commercial online proctoring services with video capturing and external monitoring, no matter whether by third-party personnel or through algorithms from machine learning and artificial intelligence, have not been an option due to serious privacy concerns [10].
- Also, the use of dedicated software to be installed on the students' computers to temporarily turn it into a secure workstation by controlling access to resources like system functions, other websites and applications like "Safe Exam Browser" [11] or "Respondus LockDown Browser" [12] would not work in our setup as the "Third Party Application" (Eclipse IDE in our case) would still allow uncontrolled web access.

However, using the same tools as in the online classes we have been running since the begin of the pandemic in Spring 2020, esp. Moodle plus Zoom with break-out rooms and screen sharing, together with online audio and video supervision of the students by our own staff, we have been able to design the setup and the process to conduct true competency-oriented programming exams that are both fair and as cheat-resistant as on campus. Fig. 2 shows the technical setup for our programming exams run online on the students' computers while being supervised by university staff via Zoom.

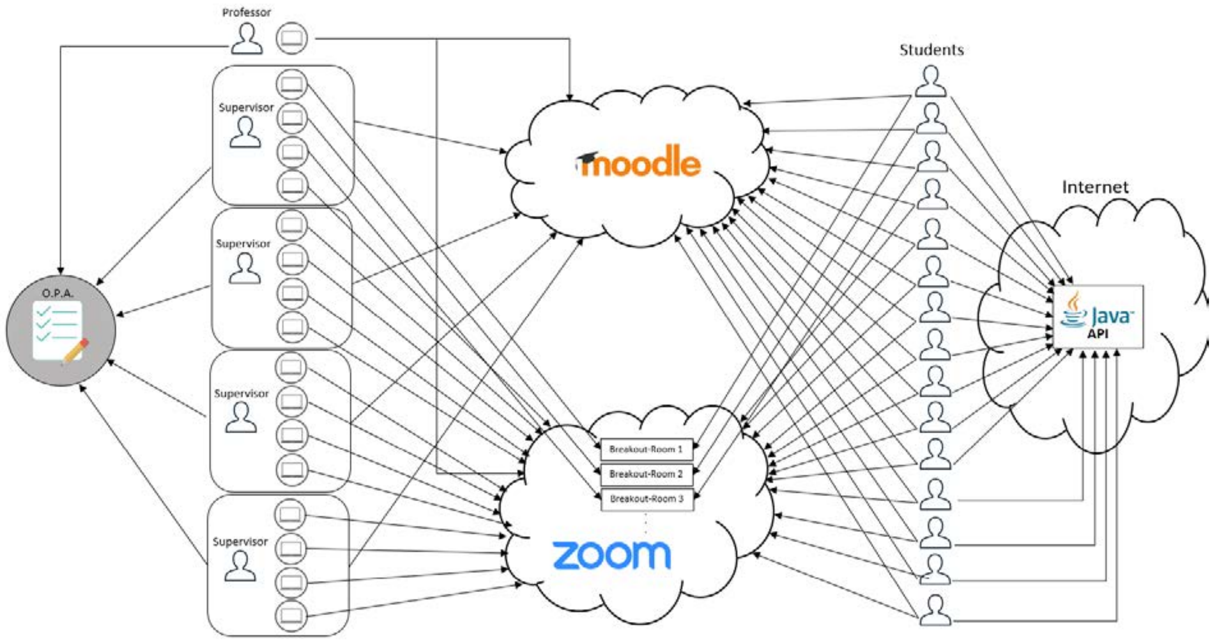

Figure 2. Technical setup for our online programming exams on students' computers

Our online programming exams are run using the same tools the students already know from their online lectures (Zoom with webcam, audio and screen-sharing), practical classes (Java and Eclipse) or even before (Moodle).

Each exam is accompanied by a dedicated course on a separate Moodle server that was setup and tested for load and performance by our university's support team. Prior to the exam students are invited to register for the exam course on Moodle where they find detailed information about how the exam is being organized and about the technical requirements, e.g. the third-party applications and resources to be used including download links to the Java JDK, Eclipse IDE and to the appropriate online documentation (Java API) that can also be used during the exam.

At the scheduled time for the exam, students connect to the "virtual lab" via Zoom where again they have the opportunity to get any questions answered about the organization of the exams and especially what to do in case of any technical issues like loss of internet connection etc. Once all questions are answered, students are sent to individual breakout rooms in Zoom where they meet with their supervisor, a staff member from our university. In fact, one supervisor is being connected to up to four different breakout rooms and thus to up to four students at the same time using different computers from our lab, actually those notebook computers which were used by the students for their exams in the lab before. As the supervisor can only interact and talk with one student at a time, the next steps may need some minutes and students are requested to be patient until being attended by their supervisor.

The supervisor first checks the student’s identity card for authentication, a process that needs to be done individually in the breakout session for data privacy reasons, and then asks the student to share the entire screen of her or his computer and checks the student’s environment (closed doors, no other screens connected, no other software running or browser tabs opened). Audio, video and screen-sharing remain active throughout the entire exam allowing supervisors to virtually “walk around” and monitor students’ activities and recognize the unauthorized use of other tools or websites. The supervisors themselves obviously only share their video and audio when interacting with one of the students assigned to them.

In order to coordinate activities and monitor progress among the supervisors and the exam coordinator, we have developed a small dashboard as Google Spreadsheet (named “*Online-Prüfungs-Aufsichten*” in German or simply “OPA”, see Fig. 3) where we share information about status and progress in each breakout-room. Once all the supervisors have marked their sessions to be ready (“green flag” on OPA), the exam gets started and students find their exam paper online on Moodle (organized as a Moodle Test instance, no PDF download) together with a code framework (as a Java Archive .jar file) which they can download, import to their Eclipse IDE, complete with their solution, get it tested and finally export and upload on Moodle at the end of the exam session. After answering some final questions and ticking off (“signing”) the formal Declaration of Originality, they are done with the exam.

Westfälische Hochschule | **Online Prüfungs Aufsichten (OPA) Dashboard** | *Aufgaben:* | *1 - 5* | *1 - 6* | *1 - 7* | **Zeit für Upload** | **Aktuelle Zeit:**

Grundlagen der Programmierung 1 (GDP1 *8.4.2021* | **Time:** | ***01:26*** | ***01:43*** | ***02:00*** | ***00:10*** | **10:31**

Breakout Room	Zoom Name	Aufsicht	Studierender Name, Vorname	Start-bereit?	Beginn (Moodle)	Zusätzliche Zeit	Ende bei 5 CP	Ende bei 6 CP	Ende bei 7 CP	CP	Abgabe (Moodle)	Bemerkungen
1	Aufsicht_1	M [redacted]	[redacted]	Ready!	08:37		10:03	10:20	10:37	5 CP	10:07	
2	Aufsicht_2	M [redacted]	[redacted]	Ready!	08:39		10:05	10:22	10:39	6 CP	00:01	
3	Aufsicht_3	M [redacted]	[redacted]	Ready!	08:40			10:23	10:40	6 CP	10:30	
4	Aufsicht_4	M [redacted]	[redacted]	Ready!	08:37	00:24			11:01			NTA (20% mehr Bearbeitungszeit)
5	Aufsicht_5	T [redacted]	[redacted]	Ready!	08:38		10:04	10:21	10:38			
6	Aufsicht_6	T [redacted]	[redacted]	Ready!	08:39		10:05	10:22	10:39	5 CP	10:11	
7	Aufsicht_7	T [redacted]	[redacted]	Ready!	08:37			10:20	10:37	6 CP	10:27	
8	Aufsicht_8	T [redacted]	[redacted]	Ready!	08:36		10:02	10:19	10:36	6 CP	10:25	Toillettengang 9:16-9:20
9	Aufsicht_9	O [redacted]	[redacted]	Ready!	08:39		10:05	10:22	10:39	7 CP		
10	Aufsicht_10	O [redacted]	[redacted]	Ready!	08:42		10:08	10:25	10:42	6 CP	00:04	
11	Aufsicht_11	O [redacted]	[redacted]	Ready!	08:39				10:39	7 CP	10:30	
12	Aufsicht_12	M [redacted]	[redacted]	Ready!	08:37		10:03	10:20	10:37	5 CP	10:09	
13	Aufsicht_13	M [redacted]	[redacted]	Ready!	08:40	00:05		10:28	10:45			Internet-Problem, Neustart Router
14	Aufsicht_14	M [redacted]	[redacted]	Ready!	08:41		10:07	10:24	10:41	5 CP	10:12	

Figure 3. Dashboard (OPA) used to coordinate supervisors (in German)

The dashboard has also become very helpful as we allow students to take only a part of the exam, e.g. the first 6 out of 7 exercises and substitute the last exercise by the score of some programming project they have submitted earlier during the course. When doing so, they are only allowed the time for the first 6 exercises and need to submit their result earlier than if working on all exercises, in which case no score of any prior programming project would go into the final score of the exam. The dashboard monitors the remaining time for each student to submit the exam depending on his/her choice of how many exercises to skip in exchange for the scores of one or two programming projects. In the exam shown in Fig. 3 students could choose to go for 5, 6, or 7 credits (CP) meaning they skip two, one or none of the exercises, however some students may have no choice at all as they haven’t done any programming project during the course. This is also reflected in the dashboard when the respective submission times, e.g. “*Ende bei 5 CP*”, are left blank.

Also, as mentioned before, some students may have been granted more time (“*Zusätzliche Zeit*”) to work on the exam, either beforehand to encompass any disadvantages (NTA) or because of some technical issues which also gets added to the dashboard. This then monitors the remaining time left for each student also taking into account the 10 minutes additional time allowed for the final export, upload and submission on Moodle. In the exam shown in Fig. 3, at the time of the screen shot being taken, two students who decided to submit their exam for six credits (“6 CP”) were still working on the upload with only a few minutes left, thus the dashboard shows their remaining times in yellow.

4 RESULTS AND LESSONS LEARNED

With the exam setup as discussed in the previous section and shown in Fig. 2, we have been able to respond to the need of moving our competency-oriented exams from lab to online while still preserving the most important properties of our exams.

A survey with students after they had taken such an online programming exam and prior to the release of the scores showed that they were particularly happy with the opportunity to do the exam from home (96%) and to do it almost the same way as they prepared for, namely using the same tools as in the practical classes (92%). While the vast majority of the students regarded the exam to be "very fair" (46%) or "fair" (38%), some mentioned that they felt "limited" (8%) but not "very limited" (0%) by the fact they had to use their own technical equipment and environment, an aspect that we will touch later in this section. As one of the major concerns being raised when it comes to online-supervised or even online-proctored exams is about data privacy or privacy infringement in general, this was only raised by a minority of the students as a "concern" (20%) or "serious concern" (4%).

Compared to the results from an ad-hoc poll done during one of the last lectures prior to the first online programming exams in February 2021, where a majority of the students said they were "worried" (56%) or "extremely worried" (16%) about the upcoming programming exam, a similar poll done in May 2021 revealed that students now feel much more relaxed about the next exams (only 8% still being "worried") which confirms our decision to keep the new setup until exams can be held absolutely safely for everyone on campus again.

Having used the new setup for a number of programming exams now, we identified some aspects that we needed to improve on as they appeared to be crucial for the success of this approach:

- Especially for the first time when we asked the students to take their exams at home, it was important to offer a test session at least one week before the exam to let students test their technical infrastructure (internet bandwidth, video/audio, setup of programming environment: Java JDK and Eclipse IDE). This turned out to be essential also to reduce anxiety and stress for the students even if they had done some online exams already but most likely none using the setup as developed for our programming exams.
  The same opportunity can also be used for doing a test run of the entire exam process with the supervisors from the initial Zoom session with all participants through the individual breakout sessions focusing not only on organizational aspects but also on how to deal under stress with concurrent requests by students from different breakout rooms asking for some help but needing to wait for supervisor's attendance.
  In our case, we included the test run into the final practical class thus asking the students to work on a given exercise under exam conditions including individual breakout rooms, using Moodle for downloading the exercise, importing files to and exporting the results from Eclipse for upload on Moodle at the end. Especially with the export and uploading part some students struggled as they were never asked this in the practical classes before, thus doing this for training was very important.
- Even with the best preparation on our and the student's side, technical, e.g. internet failure, or other issues, e.g. medical conditions, may come up during the exam and need to be dealt with properly. Thus, students are provided a dedicated phone number prior to starting the exam and asked to call this in case of connectivity problems. Both this number as well as the "Call for Help" button that is available in each Zoom breakout session connect to the lecturer responsible for the exam. As soon as the assigned supervisor becomes aware of the issue, it will also be monitored on OPA and appropriate action can be taken, e.g. continuing the exam with some additional time allowance after resolving the connectivity issue or cancelling the exam with no negative consequences for the student (as if not taken at all) in case that the connection could not be properly established again. However for the latter option, caution needs to be applied as internet failure may also have been caused deliberately by the student to avoid a fail or even a low score but getting another opportunity to take the exam again.
- Obviously, the scalability of our approach can become an issue as the number of staff needed for the online supervision quickly becomes the limiting factor. As regards the number of students that each staff member is supervising, it showed to work quite well with a ratio of 1:3 (one supervisor for three students, thus monitoring three shared screens and video channels in parallel) while a ratio of 1:4 requires some experience with the setup, i.e. from previous exams, but then also worked well still. However, from our experiences we strongly recommend not to increase the ratio any further. Instead, students may get divided into two (or more) groups taking the exams

sequentially (with modified exercises) if staff is limited. Using this approach, we have been able to run our programming exam with 25 to 30 students even while having only access to four staff members for supervision.

- It is also often being argued that different yet individual sets of exam questions, whether they are generated automatically according to programmatically controlled variability as Fully Individualised Programmable Exams (FIPE) [13] or designed manually, can help prevent students from cheating by sharing solutions with peers. However, as we have discussed in section 3.1, this does not help when students can communicate with external “experts” in the field of the exam who, in the case of Java programming expertise, can be found quite easily. Thus, our focus through supervision is on preventing students from communicating with others during the exams at all – or, at least recognizing if they do, hence appropriate action can be taken. While smart students will still be able to communicate with others, e.g. using a smartphone or tablet outside the range of the webcam, they can not simply copy/paste answers into their IDE which makes this way of cheating quite cumbersome thus less effective.

As raised by some of the students in the after-exam survey, the various and different technical equipment that students are required to use for their exams, is still an issue to be discussed, especially as this is almost the only aspect that sets our online programming exams apart from the exams held in our labs before, where students were using a uniform environment including the notebook computers provided by our department. While the different network environments at the student’s home as regards internet speed or stability etc. cannot be changed, we will need to investigate how the programming environment itself (Java and Eclipse IDE) can be virtualized and made equal to all and thus fair amongst the students, e.g. by introducing a virtual desktop infrastructure (VDI) [8]. However, as this is of great importance for exams with more resource-demanding third-party applications like CAD or complex simulation software, differently powerful computers on the student’s desk may not make such a difference for our programming exam.

5 CONCLUSIONS

Competency-oriented exams have gained increasing interest from educators especially as they support constructive alignment of classes and exams. Applied to programming classes this requires performing exams on computers using state-of-the-art tools as in the practical classes. While validity, reliability and fairness could be established for such computer-based practical programming exams in the lab on campus, the setup could not easily be transferred online as demanded by the COVID-19 pandemic. Instead, the supervised and technically controlled environment in the lab asks for an equivalent setup for online programming exams performed on the student’s computer at the student’s home. With the supervision of students by staff members through Zoom video and screen-sharing, we were able to implement a setup that offers competency-oriented programming exams to be transformed from lab to online providing almost the same level of supervision and cheat resistance while preserving their validity, reliability and fairness even during the pandemic.

Hence, we did not let COVID-19 ruin our competency-oriented programming exams!

ACKNOWLEDGEMENTS

The work on introducing Perusall in our classes was supported as part of my “Fellowship for Innovations in Digital Teaching” (“Digi-Fellowhip”) grant No. 214-5.01.03.02 (143061) by the Ministry for Culture and Science of the State of North Rhine-Westphalia.

I would like to thank our university’s administration for the opportunity to run the exams as described and also Michael Engelmann for preparing the graphics for this paper as well as supporting the organization of the exams together with Thomas Müller. Special thanks go to Gayatri Das for repeatedly proofreading parts of the paper and valuable discussions even down to the single words which definitely improved the clarity and readability of the paper a lot, however the author takes full responsibility for any remaining stylistic or even factual mistakes.

REFERENCES

[1] K. Miller, B. Lukoff, G. King, and E. Mazur, “Use of a Social Annotation Platform for Pre-Class Reading Assignments in a Flipped Introductory Physics Class”. *Frontiers in Education,* vol. 3:8, 2018

[2] M. Meyer and T. Müller, “If it were that easy: First Experiences of Introducing a Social Learning Platform in an Undergraduate CS Course”, *Proc.12th International Conference of Education, Research and Innovation* (ICERI 2019), pp. 10688-10697, Seville (Spain): IATED, 2019

[3] M. Camp, J. Middendorf, and C. Sullivan, "Using Just-in-Time Teaching to Motivate Student Learning", in *Just-in-Time Teaching: Across the Disciplines, Across the Academy* (S. Simkins and M. Maier eds.), Sterling: Stylus Publishing, 2010

[4] L. Anderson, D. Krathwohl, P. Airasian, K. Cruikshank, R. Mayer, P. Pintrich, J. Raths, and M. Wittrock, “A Taxonomy for Learning, Teaching, and Assessing: A revision of Bloom's Taxonomy of Educational Objectives”, New York: Pearson, Allyn & Bacon, 2001

[5] B. Bloom (Ed.), M. Engelhart, E. Furst, W. Hill, D. Krathwohl, „Taxonomy of Educational Objectives, Handbook I: The Cognitive Domain”, New York: David McKay, 1956

[6] J. Biggs and C. Tang, “Teaching for Quality Learning at University”, Maidenhead: The Society for Research into Higher Education and Open University Press (McGraw Hill Education), 2011

[7] S. Hammer, M. Hobelsberger, and G. Braun: “Using Laboratory Examination to Assess Computer Programming Competences - Questionnaire-based Evaluation of the Impact on Students”, in *Proc. 2018 Global Engineering Education Conference* (EDUCON 2018), pp. 355-363, Santa Cruz de Tenerife (Spain), 2018

[8] T. Lüthi, M. Kern, K. Reuter, T. Halbherr, and T. Piendl, „Competence-Oriented Exams Using Virtual Desktop Infrastructure (VDI)”. Technical Report, 3rd ed., Zurich (Switzerland): ETH, 2019

[9] M. Meyer, “Who wants to be a Millionaire? Or: How Interactive Voting Systems help Activating Undergraduate Students and Improving Teaching Performance”, in *Proc. 3rd International Conference on Education and New Learning Technologies* (EDULEARN11), pp. 6912-6921, Barcelona (Spain), IATED, 2011

[10] L. Ackermann, “Exams in the Pandemic: Online Proctoring is no Solution”, *Hochschulforum Digitalisierung Blog*, Accessed 10 May, 2021. Retrieved from https://hochschulforumdigitalisierung.de/en/blog/exams-pandemic-online-proctoring-no-solution

[11] H. Nurwarsito, “Secure Online Assessment on E-Learning Moodle with SEB (Safe Exam Browser)”, in *Proc. 5th International Conference on Education and New Learning Technologies* (EDULEARN13), pp. 3232-3239, Barcelona (Spain), IATED, 2013

[12] Respondus, Assessment Tools for Learning Systems, Accessed 10 May, 2021. Retrieved from https://web.respondus.com/

[13] J. Spillner, “Examination Cheat Risk Reduction through FIPEs”, in *Proc. 13th International Conference on Computer Supported Education* (CSEDU 2021), Lisbon (Portugal): INSTICC, 2021