Introduction

It is common for students that learn programming, to go through courses in a practical way with assignments and programming exercises. Assignments and exercises usually consist of a number of tasks that the student will have to solve. When the student is finished, the code is submitted to the teacher. The teacher downloads the code, runs it and grades the student. This approach is time consuming, and by the time the teacher has graded all the exercises, the students could have already started on their next assignment. This can lead to students not **(fully understanding the curriculum - why)** and in worst case miss out on key parts of the course. When the courses get more advanced, and the code base increases, this problem gets even bigger. A lot of time is wasted on the overhead work, rather than providing students with help and feedback they need. It also requires a lot of work for the teachers, and can often result in students being left behind by the teacher because of pressure to get all the grading done. Even though the system discussed **(har ikke blitt nevnt ennå)** is in use now, it does not mean that it cannot be changed and improved. One way of solving this problem is by using a system of automatic testing and grading. Such a system is already implemented in other universities, but these systems usually lack key features **(what are these key features, unclear)**. With that in mind, former master student Heine Furubotten created a system called Autograder. It's a system for automatic testing, code reviewing and grading. The system collects student submitted code from GitHub - a version control system, and grades it based on tests written by the teacher. The system has already been in use in some of the courses held at bachelor and master level at UiS. Autograder requires some basic knowledge of Github and Git, however, not all the students are familiar with those systems, these technologies will have to be taught at some point later in the curriculum on anyway, so it can be introduced to them earlier. The process of submitting code is easy, as well as the process of getting feedback.

Autograder works the same way many systems in the industry do, like automatic code testing of submitted work, and mimics the the real world scenarios that the student will encounter later. When students graduate and start working in the industry, a similar process of code review and test-driven development will probably be the most common way of programming. It is therefore not an issue to rely on the students learning version control system. As mentioned earlier Autograder has already been used on some of the advanced programming couses at UiS, and has proven a valuable asset. The system worked as expected and made the job of the teachers easier, by showing all the students in a web-based system and their progress. Automatic testing was done though code submitted to Github. The code was downloaded and run on computers at UiS, showing the results and build log in the browser. This made it easy for both students and teachers to monitor progress, and evaluate the work. **(Feedback was given along the way, and helped improved the system with direct feedback from the users.)** According to Heine, the system improved how the assignments and grading process is done. It also made it simpler for the **(teacher or students ->?)** students to focus on the tasks at hand, rather than having to think about a cumbersome and slow grading process.

Automatic testing of code and grading is not a new idea. Many similar systems are in use at other universities. The biggest difference is that they rely on third party software for testing, and can in many cases not fit the need of the teachers and students. The Autograder system is developed at UiS with direct student and teacher feedback. This gives an advantage when it comes to implementation of useful functions, and is therefore tailored for the needs of the advanced programming courses at UiS. Nevertheless, this does not limit the functionality of the program, which can be implemented in other universities. Autograder is not limited to programming languages design-wise, however, it has been mostly tested with the Go programming language. Both student and teachers have been very pleased with the system, and now that it has been used for some time, relevant user-data can be gathered to improve the system even more.