Software quality and testing (DAT560 / DIT636)

Jump to today ✎ Edit

Course-PM: Software Quality and Testing (DIT636 / DAT560), LP3 VT26 (7.5 hp)

Click here for detailed schedule, slides, and resources

(Last Updated: January 2)

Course is offered by the department of Computer Science and Engineering.

(In case of issues, all course content is also available at https://greg4cr.github.io/courses/spring26dit636/index.html, with a short delay).

Contact Details

Examiner/Course Responsible
Gregory Gay (ggay@chalmers.se)

Teaching Assistants
Lirong (Esme) Yi (lirongy@chalmers.se)
(To Be Added)

Student Representatives
Interested in volunteering? Contact me (ggay@chalmers.se).

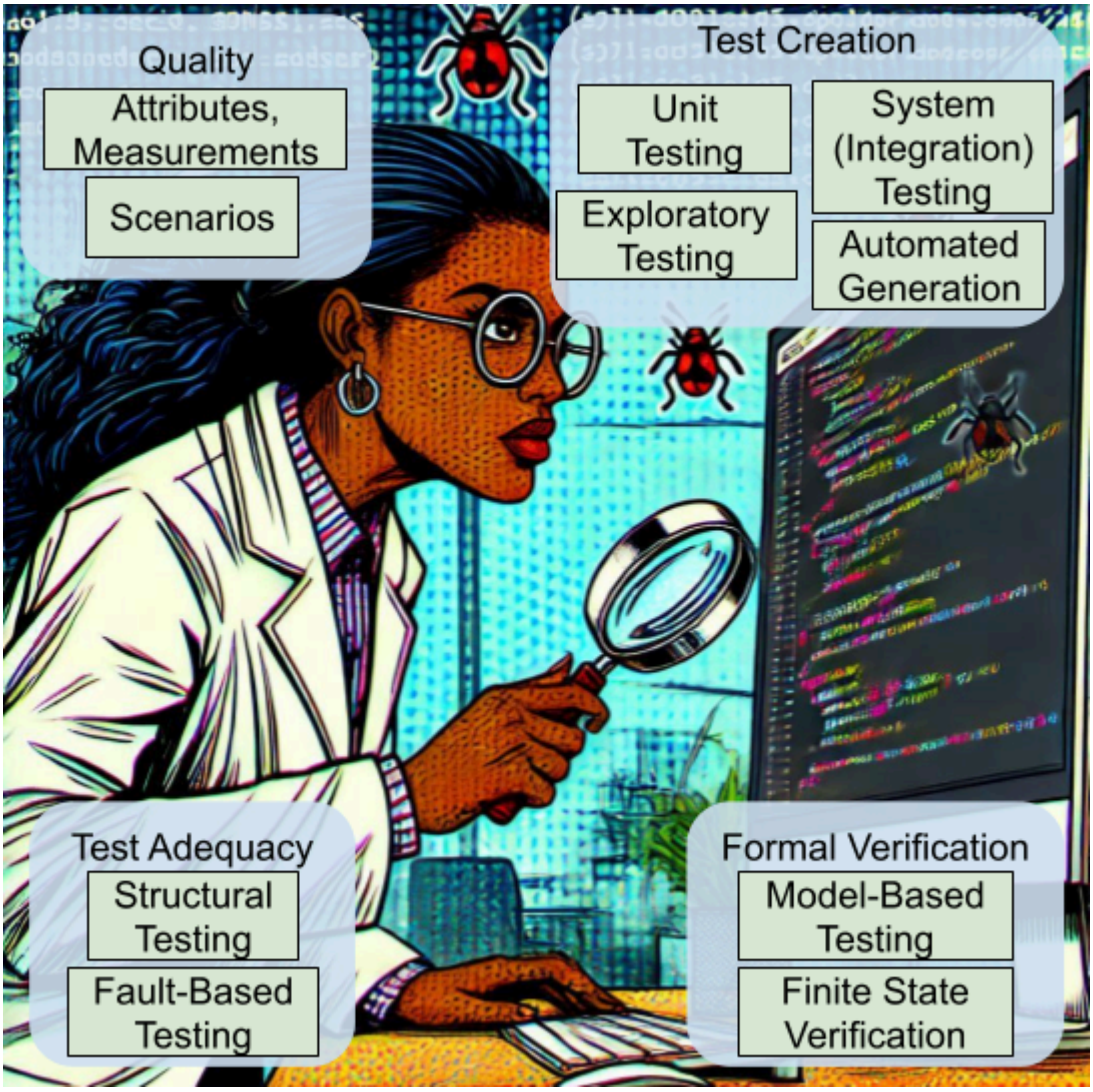
Study counsellor: studycounselling@cse.gu.se (questions related to study and career planning)
Student office: studentoffice@cse.gu.se (questions related to the course administration - e.g., registration, signup, grades in LADOK)
Student portal: https://studentportal.gu.se/english/my-studies/cse

Communication and Course Feedback

We understand that it can be difficult to get answers to your questions, and want to make sure you get any help you need. We recommend the following methods for contacting the teaching staff:

- Any questions related to the lectures or assignments can be asked in the **Canvas discussion forum**. This forum will be monitored by the teacher and teaching assistants. You may also e-mail any of us directly. However, we recommend posting questions to the forum to help other students who may be having similar issues.
- Private or otherwise sensitive questions may be sent directly to the teacher.
- Course feedback can be presented to the student representatives, who will pass it on to the teacher and teaching assistants, or you can contact us directly (especially if you have any issues that are time-sensitive or private). We welcome course feedback, and are happy to adjust the course to correct issues affecting a majority of the students.

Course Purpose



Our society is built on software. It powers our homes, it manages our private information, it controls our cars, it automates our factories, and it even regulates our bodies. It is incredibly important that we construct robust, operational systems, especially given growing demand for features, limited development budgets and strict time constraints.

The key to delivering robust software is through a thorough verification and validation (V&V) process. In this course, we will explore the V&V process and examine a variety of methods to test systems, prove their correctness, and provide evidence that the software we build is reliable and safe to use. In

this course, we introduce the concepts and best practices of quality assurance and testing in software engineering.

In this course, we first introduce the notion of software quality and quality assurance. We present methods and techniques to assure quality of both the end product (a system or application), and for the software process itself. We then cover testing tools, techniques and methods that can be used to assess the quality and correctness of software systems. We conclude with formal methods of proving certain properties of software, and contrast those techniques to testing in terms of completeness, soundness, and scalability. Together, the goal of the course is to prepare students to use these methods, techniques and tools in a software development project to increase software quality.

Schedule

Detailed Schedule Page

TimeEdit

Course Literature

There is no **mandatory** course literature.

The following text books are **optional**:

- Mauro Pezze, Michal Young. Software Testing and Analysis: Process, Principles, and Techniques.
  - **Students can request a free copy of the textbook from https://ix.cs.uoregon.edu/~michal/book/free.php.**
- Maurício Aniche. Effective Software Testing: A Developer's Guide.
  - $32.49 (physical), available at **https://www.effective-software-testing.com/**.
  - E-book available for free from the university library.

Both books provide additional material on some of the course topics and cover additional topics that we do not have time to get to in the course. Neither are required, but both will enhance your understanding of the subject.

Additional optional readings will be provided for individual topics.

Course Design

The teaching consists of lectures, group work (assignments), class exercises, as well as supervision in connection to the exercises and assignments. The course emphasizes problem-based learning. Basic concepts of are presented in the lecture, applied in exercise sessions, and then extended in the context of integrated, graded assignments. Assignments are developed in teams of **three** students.

Language of instruction: English

Teaching and Learning Activities

Lectures: There will generally be two lectures per week (see **Schedule**). Attendance in the lectures is highly recommended, but not mandatory. Note that any material covered in lectures can be referred to in the project and exam (i.e., not just material written in the slides).

Exercises: We will work through a specific set of exercises in formal exercise sessions, held once per week (see **Schedule**). This gives students the chance to apply class concepts, as well as to see demonstrations. The teacher and the TAs will be present to answer questions. Attendance in supervision sessions is, again, highly recommended but not mandatory. These sessions are also used to provide assignment guidance.

Forum: Any questions related to the lecture, exercises, or assignment can be asked in the **Canvas discussion forum**. If you have any questions regarding the course material, this is a good place to express them.

Learning Objectives and Syllabus

Learning objectives:

Knowledge and understanding

- Explain quality assurance models in software engineering and the contents of quality assurance plans
- Describe the distinction between software verification and software validation
- Name and describe the basic concepts on testing, as well as different testing techniques and approaches
- Describe the connection between software development phases and kinds of testing
- Exemplify and describe a number of different test methods, and be able to use them in practical situations
- Exemplify and describe tools used for testing software, and be able to use them and interpret their output

Competence and skills

- Exemplify and describe the area of formal verification in general, including model checking and runtime verification, and its relationship to software quality
- Define metrics required for monitoring the quality of projects, products and processes in software engineering
- Construct appropriate and meaningful test cases, and interpret and explain (to stakeholders) the results of the application of such test cases (using appropriate tools) to practical examples
- Write models in at least one formal specification language plan and produce appropriate documentation for testing

- Apply different testing techniques on realistic examples

Judgement and approach

- Identify emerging techniques and methods for quality management using relevant information sources
- Identify and hypothesize about sources of program failures, and reflect on how to better verify the correctness of such programs

Link to the syllabus:

- **DIT636 (GU)**
- **DAT560 (Chalmers)**

Examination Form

Sub-Courses

1. Written examination (Skriftlig tentamen), 4.5 higher education credits
   Grading scale: Fail (U), 3-5
2. Assignments (Inlämningsuppgifter), 3 higher education credits
   Grading scale: Fail (U), 3-5

Assessment

The course is examined by an individual written exam carried out in an examination hall at the end of course and written assignments normally carried out in groups of three students. The assignments part is examined on the basis of solutions to compulsory problems handed in during the course and on the basis of individual contribution to the group work.

There will be five written group assignments. Each assignment is equally weighted. Specific requirements for the assignments will be provided on Canvas.

Students are required to complete written self- and peer-assessment forms during the course which will be part of the assessment of the student's individual contribution to the project. The teacher may adjust the individual grades of a student depending on this evaluation.

Grading Scales

Written assignments and the final exam will be graded on a numeric scale, converted to scaled grades. The final percentage grade for the assignments and exams will be converted as follows:

% Grade	Grading Scale
0-49%	Fail (U)
50-69%	3

70-85%	4
86-100%	5

The final grade for the assignment part of the course (3 credits) will be an average of the grade (0-100) for each of the four assignments. All four assignments must be submitted. Failing one of four assignments does not necessarily mean there will be a failing grade for the assignments part of the course.

The final grade of the course is calculated as follows:

(right) Exam Grade **(down) Assignment Grade**	**U**	**3**	**4**	**5**
U	U	U	U	U
3	U	3	4	4
4	U	3	4	5
5	U	4	4	5

How Groups are Formed

Students may form their own groups (see Assignment 0). You may request placement on a team if you do not want to find your own group.

Group Grades

Note that although the grade is given for a group assignment, this grade is then assigned individually to students, and may be adjusted depending on the peer evaluation form. Thus, not all students in the same group are guaranteed to get the same grade. Such situations are rare.

Late Assignments

Up to One day late: - 20% reduction of final grade

Up to Two days late: - 40% reduction of final grade

Two or more days late: 0% on assignment

Failing Assignments

If the final average grade of all assignments is a failing grade, **all** five assignments must redone and resubmitted. The redone assignments should handed in again at a date after course completion. Modified assignments may be provided. Redone assignments can be done in groups or individually.

Assignment Re-submission

If an assignment is failed, students have **up to two** chances to resubmit an improved version. **Re-submissions will only be accepted until one month after the (first) written exam.**

Failing the Exam

If a student, who has failed the same examined component twice, wishes to change examiner before the next examination, a written application shall be sent to the department responsible for the course and shall be granted unless there are special reasons to the contrary (Chapter 6, Section 22 of Higher Education Ordinance).

In cases where a course has been discontinued or has undergone major changes, the student shall normally be guaranteed at least three examination occasions (including the ordinary examination) during a period of at least one year from the last time the course was given.

Examination Dates

- First round: March (TBA)
- Second round: June (TBA)
- Third round: August (TBA)

See the following for any updates:

https://studentportal.gu.se/english/my-studies/cse/Examination

---

Changes made since the last occasion

Based on feedback from the last occasion, the following changes have been made:

- Slides have been updated and some examples have been added or reworked.
- Assignments have been updated and reworked. In particular, the final assignment has been split into two assignments (Assignments 4-5) so that students get feedback on Assignment 4 before the exam.

Additional Information

This section contains some general rules that will be enforced during this course. Please review these guidelines carefully. Violations of conduct guidelines will be taken seriously and will lead to disciplinary action.

Integrity and Ethics

The homework and programs you submit for this class must be entirely your own. Any text, images, or other resources created by others and used in a submission must be properly cited. If this policy is not absolutely clear, then please contact me. Collaboration of any type with students outside of your group on any assignment is not permitted. It is also your responsibility to protect your work from unauthorized access. Any violation of this policy will result - at minimum - in a failing grade on the assignment and a formal report to the university. Further infractions will result in a failing grade in the course and additional disciplinary action. More information on plagiarism will be provided on the Canvas page. We recommend referring to this material.

Generative AI: Generative AI tools (e.g., ChatGPT, GitHub Copilot) may be used to support your learning (e.g., to help you understand concepts or brainstorm). However, all code, text, or other material you submit in assignments must be your own original work.

- You may **not** submit generated test cases or scenarios as your own. You must create your own test cases or scenarios in assignments.
- You may use tools to improve your grammar (i.e., in a form similar to tools like Grammarly), but you must write all text yourself initially.
- You may use tools to generate plots or images, based on data that you gathered or provided yourself.

If you plan to use Generative AI in an assignment:

- Ask the teacher first!
- Disclose in the assignment which tools you used and how you used them.

Bear in mind that the answers provided by Generative AI tools may not be correct. Perform your own evaluation of the output of these tools.

We reserve the right to use AI detection tools to check assignments for plagiarism.

Classroom Climate

All students are expected to behave as scholars at a leading institute of technology. This includes arriving on time, not talking during lecture (unless addressing the instructor or as part of a group discussion), and not causing disruption in the classroom. Disruptive students will be warned and potentially dismissed from lectures, with multiple instances leading to potential consequences on your course grade.

Special Needs

It is university policy to provide, on a flexible and individual basis, reasonable accommodations to students that have conditions that may affect their ability to participate in course activities or to meet course requirements. Students are encouraged to contact their teacher early in the semester to discuss their individual needs for accommodations.

Diversity

Someday you will graduate, and in the real world, you will have to work with a wide variety of people. Now is the time to abandon preconceived prejudices about others. Students in this class are expected to respectfully work with all other students, regardless of gender, race, sexuality, religion, or any other criteria. There is a zero-tolerance policy for any student that discriminates against other students for any reason, including potential disciplinary action from the university, dismissal from the course, and a failing course grade.

Course summary:

Date	Details	Due
Mon, 19 Jan 2026	Lecture 1 - Software Quality, Verification, and Validation	10:15 to 12:00
Wed, 21 Jan 2026	Lecture 2 - Quality Attributes and Measurement	10:15 to 12:00
Sun, 25 Jan 2026	Assignment 0 - Team Selection	due by 23:59
Mon, 26 Jan 2026	Lecture 3 - Quality Scenarios	10:15 to 12:00
Wed, 28 Jan 2026	Lecture 4 - Testing Fundamentals	10:15 to 12:00
	Exercise Session 1 - Quality Scenarios	15:15 to 17:00
Mon, 2 Feb 2026	Lecture 5 - Test Case Design	10:15 to 12:00
Wed, 4 Feb 2026	Lecture 6 - Test Case Design and Unit Testing	10:15 to 12:00
	Exercise Session 2 - Test Case Design	15:15 to 17:00

Date	Details	Due
Mon, 9 Feb 2026	Lecture 7 - Integration Testing and Test Automation	10:15 to 12:00
Wed, 11 Feb 2026	Lecture 8 - Exploratory Testing	10:15 to 12:00
	Exercise Session 3 - Unit Testing	15:15 to 17:00
Mon, 16 Feb 2026	Lecture 9 - Test Adequacy and Structural Testing	10:15 to 12:00
	Lecture 10 - Structural Testing - Paths and Data Flow	13:15 to 15:00
Fri, 20 Feb 2026	Exercise Session 4 - Structural Testing	10:15 to 12:00
Mon, 23 Feb 2026	Lecture 11 - Mutation Testing	10:15 to 12:00
Wed, 25 Feb 2026	Lecture 12 - Testing in Industry	10:15 to 12:00
	Exercise Session 5 - Mutation Testing	15:15 to 17:00
Mon, 2 Mar 2026	Lecture 13 - Model-Based Testing	10:15 to 12:00
Wed, 4 Mar 2026	Lecture 14 - Finite State Verification	10:15 to 12:00
	Exercise Session 6 - Finite State Verification	15:15 to 17:00
Mon, 9 Mar 2026	Lecture 15 - Automated Test Generation	10:15 to 12:00
Wed, 11 Mar 2026	Lecture 16 - Course Summary and Review	10:15 to 12:00