Course Outline

Course Code	SENG2011
Course Title	Software Engineering Workshop 2A
Units of Credit	6
Course Website	https://webcms3.cse.unsw.edu.au/SENG2011/21T3/ (https://webcms3.cse.unsw.edu.au/SENG2011/20T3/)
Current hand-book entry	http://www.handbook.unsw.edu.au/undergraduate/courses/current/SENG2011.html (http://www.handbook.unsw.edu.au/undergraduate/courses/current/SENG2011.html)

Course Summary

This course teaches practical techniques in reasoning about computer program development. 'Reasoning' involves determining what program code is expected to achieve, and what determines correct behaviour as specified in logic. In this approach, students will need to think about the behaviour of a algorithm separately from the code. Proving that an algorithm satisfies a behavioural description is referred to as verification and forms the backbone of the course.

Course Aims

Engineering: learn how to construct a behavioural specification of a software system, using a formal tool, and how to implement this specification in code.

Student Learning Outcomes

If you successfully complete this course, you should have acquired a bunch of new skills:

- be able to build a formal specification of the behaviour of a program
- be able to engineer code, not write it in an ad hoc fashion
- be able to reason abstractly about requirements and be able to model them using formal methods
- be aware that proving code is correct is crucial for system reliability

Assumed Knowledge

- **essential** is that you are competent in programming at least one procedural language (e.g. C, Python, Java)
  - e.g. *can you write a program in your favourite procedural language that does linear search in an array?*
- discrete mathematics: familiarity with propositional and predicate calculus, and formal proof techniques
  - e.g. p→ ¬ q *is equivalent to which of* [A] ¬ p→q [B] ¬ (p→q) [C] q→ ¬ p [D] ¬ q ∨ ¬ p
  - e.g. *prove by induction that* n! ≥ 2^(n−1)
- competence with Unix (e.g. Linux) commands, and interacting with the operating system

Answers shown at the end.

Teaching Strategies

As SENG2011 is a workshop, it relies less on formal lectures and tutorials than a conventional lecture course, and more on individual effort. The lectures however are very important to understanding, particularly conceptual of issues in verification. There are no scheduled tutorials or lab sessions, but there is a very active forum. On a regular basis there will be exercises for students to practice their skills.

Assessment

The total course mark will be the sum of the marks for the following components:

- Assessment 1: 6 'weekly' quizzes (multiple choice) (10 marks)
- Assessment 2: Assignment 1 (15 marks)
- Assessment 3: Assignment 2 (25 marks)
- Final Examination (50 marks)
  - If your final exam mark is below the pass mark, your mark for the exam will be set to 0. Your final course mark will be the total of your 3 assessments in that case.

The final quiz mark will be calculate by the formula *10 * total-number-of-correct-answers / total-number-of-questions.*

All the assessments should be done on an individual basis . A tentative schedule of when assessments are due can be found on the course website.

Late submission

Handing an assignment in late by:

- up to one day (i.e. 24 hours) will be scaled by 85%
- for two days it is 75% (that is total scaling, not in addition to the 85%)
- for three it is 50%
- after three days the submission will not be accepted at all (i.e. is scaled by 0%)

If you think you have sound reasons to request a waiver of these rules, e.g. illness or misadventure, you must submit an official request for special consideration (https://student.unsw.edu.au/special-consideration) , with supporting documentation (e.g. medical certificates) through the formal UNSW central channels (*not* by direct request to the convenor or system administrator.)

Supplementary Exam

The document **"Essential Advice for CSE Students" (https://www.engineering.unsw.edu.au/computer-science-engineering/about-us/organisational-structure/student-services/policies/essential-advice-for-cse-students)** states the supplementary assessment policy for the school of CSE.

I have no control over who is eligible to take a supplementary exam. This is handled by Student Administration and requires you to apply for Special Consideration.

If you are granted a Supplementary examination, then it will be held on a date specified by the university, or, if it is an on-line examination, specified by the school. It is your responsibility to check at the School Office for details of Supplementary examinations.

Student Conduct

The **Student Code of Conduct** (Information (https://student.unsw.edu.au/conduct) , Policy (https://www.gs.unsw.edu.au/policy/documents/studentcodepolicy.pdf)) sets out what the University expects from students as members of the UNSW community. As well as the learning, teaching and research environment, the University aims to provide an environment that enables students to achieve their full potential and to provide an experience consistent with the University's values and guiding principles. A condition of enrolment is that students *inform themselves* of the University's rules and policies affecting them, and conduct themselves accordingly.

In particular, students have the responsibility to observe standards of equity and respect in dealing with every member of the University community. This applies to all activities on UNSW premises and all external activities related to study and research. This includes behaviour in person as well as behaviour on social media, for example Facebook groups set up for the purpose of discussing UNSW courses or course work. Behaviour that is considered in breach of the Student Code Policy as discriminatory, sexually inappropriate, bullying, harassing, invading another's privacy or causing any person to fear for their personal safety is serious misconduct and can lead to severe penalties, including suspension or exclusion from UNSW.

There is a document explaining the procedure for student misconduct. (https://www.gs.unsw.edu.au/policy/documents/studentmisconductprocedures.pdf)

If you have any concerns, you may raise them with your lecturer, or approach the School Ethics Officer (mailto:ethics-officer@cse.unsw.edu.au) , Grievance Officer (mailto:grievance-officer@cse.unsw.edu.au) , or one of the student representatives.

Plagiarism

Plagiarism is defined as (https://student.unsw.edu.au/what-plagiarism) *using the words or ideas of others and presenting them as your own* . UNSW and CSE treat plagiarism as *academic misconduct* , which means that it carries penalties as severe as being excluded from further study at UNSW. There are on-line sources to help you understand what plagiarism is and how it is dealt with at UNSW: Academic Integrity and Plagiarism (https://student.unsw.edu.au/plagiarism) , and the very recent (effective 1st June 2020) Plagiarism Management Procedure (https://www.gs.unsw.edu.au/policy/documents/plagiarismprocedure.pdf) .

- Make sure that you read and understand these documents.
- You are also responsible that your course assessment solutions are not accessible by anyone.
- Plagiarism includes paying or asking another person to do a piece of work for you and then submitting it as your own work.
- Posting exercises from the course assessments on the internet is considered an attempt to plagiarise, and accordingly such an action will result in penalties.

Resources for Students

The lecture notes are comprehensive, detailed and should be closely studied. Links to supporting documents such as the Reference Manual, excerpts from useful documents and published articles will also be provided on the website.

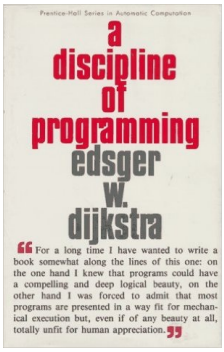
The book on this topic is *Program Proofs* by *Rustan Leino* .

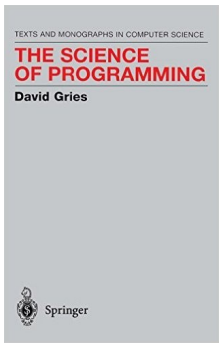


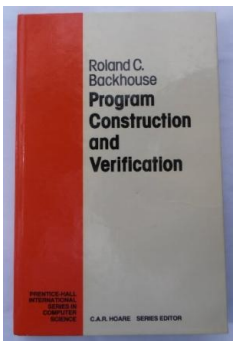
Rustan Leino is an acknowledged leader in the field of formal methods of software engineering, and works at the cutting-edge of compiler development. The book is still in draft form, and is available from https://www.lulu.com/en/gb/shop/k-rustan-m-leino-a... (https://www.lulu.com/en/gb/shop/k-rustan-m-leino-and-kaleb-leino/program-proofs/paperback/product-wqy8w5.html?page=1&pageSize=4) The main difference between the book and the course is that the course focusses on the imperative language Dafny, and how to verify code using the language, whereas the book takes a broader view that first considers issues in program proving and the functional side of the language before treating its imperative side. The book is not a required text (yet).

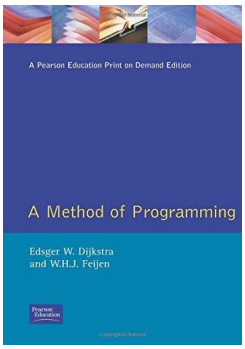
References that explain the origins and form the theory of program verification are the following.

- Edsger W. Dijkstra : *A discipline of programming*
- David Gries : *The science of programming*
- Roland Backhouse: *Program construction and verification*
- Edsger Dijkstra and Wim Feijen : *A method of programming*









Course Evaluation and Development

Survey of Student Experience 2020T3:

- Felt part of community: 72% in agreement
- Feedback helped me to learn: 77% in agreement
- Course resources helped me to learn: 80%
- Assessment tasks were relevant: 92%
- Course encouraged self-directed learning: 93%
- Assignment gave me opportunity to demonstrate knowledge: 93%
- Best things about the course:
  - Slides are well structured and fun
  - Videos were short and understandable, and narration was useful
  - Active piazza forum
  - Prompt response to emails
  - Weekly quizzes
  - Assignment were fun and relevant to notes
  - No tutorials
  - Course was punctual, well organised
  - The course is just hard, it's a new way of thinking
- What needs improvement:
  - Lack of information on Dafny. Changes:
    - the first book on Dafny (author Rustan Leino), described above, has recently been published, for those students that require in-depth information
    - more than 100 new lecture slides have been added
    - for the first time there will be so-called 'play' videos that show the lecturer solving programming exercises
  - Lack of interaction, tutorials, coding practice. Changes:
    - Face-to-face tutorials are not possible this session, very unfortunately
    - It is hoped the 'play' nature of the new videos will 'mimic' the interactive style of a tutorial
    - The new videos should answer a lot of 'practical' questions that students have
    - New quiz questions will be included this year that require weekly programming
    - The lecturer will be more present on the forum this year to encourage interaction
  - 50% for exam hurdle is too much. The lecturer's response:
    - the final exam is very important as a motivator and test of individual skills, not guaranteed by assignments and quizzes

Answers

1. Linear search in an arbitrary language

linearSearch.js

```
function linearSearch(values, target) {
  for(var i = 0; i < values.length; ++i){
    if (values[i] == target) { return i; }
  }
  return -1;
}
linearSearch([7, 3, 6, 1, 0], 6)
```

2. $p \rightarrow \neg q$ is equivalent to [C] q→ ¬ p and to [D] ¬ q ∨ ¬ p

3. Proof by induction that $n! \geq 2^{n-1}$ for $n \geq 1$

Base case $n = 1$: $1! \geq 2^0$ is true
Assume true for $n = k$: $k! \geq 2^{k-1}$
Inductive step: prove true for $n = k + 1$, i.e. $(k + 1)! \geq 2^k$
$(k + 1)! = (k + 1) \times k! \geq (k + 1) \times 2^{k-1} \geq 2 \times 2^{k-1} = 2^k$ **for** $k \geq 1$ ***QED***
(justifications not shown)

Resource created 5 months ago (Tuesday 22 June 2021, 04:59:32 PM), last modified 3 months ago (Monday 06 September 2021, 10:27:50 AM).

Comments (/SENG2011/21T3/forums/search?forum_choice=resource/63425) (/SENG2011/21T3/forums/resource/63425)

There are no comments yet.