

SE 2AA4, CS 2ME3 (Introduction to Software Development)

Winter 2017

01 Introduction to the Course

Dr. Spencer Smith

Faculty of Engineering, McMaster University

January 5, 2017



Introduction to Course

- Administrative details
- Course outline
 - ▶ Introduction
 - ▶ Learning objectives
 - ▶ Outline of topics
 - ▶ Grade assessment
 - ▶ Policy statements

Administrative Details

- Tutorials start next week
- This course uses Avenue
 - ▶ <http://avenue.mcmaster.ca/>
 - ▶ Please put a picture up on Avenue!
- We'll also use git on GitLab
 - ▶ <https://gitlab.cas.mcmaster.ca/>
 - ▶ Create your account by logging in
 - ▶ Course material and issue tracking at
https://gitlab.cas.mcmaster.ca/smiths/se2aa4_cs2me3

Linux Tutorial for Beginners

The Linux Tutorial for beginners will be given on Tuesday, Jan 10th, from 1:30pm to 3:20 pm in lab ITB237. The second part of the tutorial will be given in the following week at the same time and location (on Tuesday, Jan. 17th, from 1:30pm to 3:20 pm in lab ITB237)

The first part of the tutorial covers some basic Linux commands and utilities. The second part is to show students to use text editors, including VI(VIM), and to compile and run simple C and Java programs and projects.

The first part of the same tutorial will be repeated on Wednesday, Jan. 11th, from 12:20 pm to 2:20 pm in lab ITB237 for students who can not attend the Tuesday session of tutorial. The second part of the tutorial will be repeated on Wednesday, Jan. 18th, from 12:20 pm to 2:20 pm in lab ITB237.

Instructors

- Instructor

- ▶ Dr. Spencer Smith (smiths@mcmaster.ca)
- ▶ ITB/167
- ▶ Drop in or make an appointment

- TAs

- ▶ Steven Palmer (palmes4@mcmaster.ca)
- ▶ Rober Boshra (boshrrar@mcmaster.ca)
- ▶ Gurankash Singh (singhg47@mcmaster.ca)
- ▶ Owen Huyn (huyno@mcmaster.ca)
- ▶ Teaching assistants will
 - ▶ Give tutorials
 - ▶ Mark assignments
 - ▶ Provide programming assistance
 - ▶ Answer questions on the course material

Introduction of Instructor: Dr. Spencer Smith

- Associate Professor, Department of Computing and Software.
- B.Eng.C.S, Civil Engineering Department, McMaster University.
M.Eng., Ph.D., Civil Engineering Department, McMaster University.
- P.Eng. (Licensed Professional Engineer in Ontario).
- **Teaching:** Software design, scientific computing, introduction to computing, communication skills, software project management.
- **Research:** Application of software engineering methodologies to improve the quality of scientific computing software.

Introduction

- Calendar description
 - ▶ Development of small software units
 - ▶ Precise specification expressed using logic and discrete math
 - ▶ Design methods and design patterns
 - ▶ Implementation and testing
- Mission
 - ▶ Introduction to profession of software engineering
 - ▶ Strategies for large applications with multiple developers
 - ▶ Java and Python

Learning Objectives

<https://gitlab.cas.mcmaster.ca/.../LearningOutcomes>

Resources

- Ghezzi et al. (2003) (required)
- Hoffman and Strooper (1995) (other)
- VanVliet (2000) (other)

Outline of Topics

1. Introduction to Course
2. Software Engineering as an Engineering Discipline [Chapter 1]
3. Software Qualities [Chapter 2]
4. Software Engineering Principles [Chapter 3]
5. Software Design [Chapter 4]
6. Modularization [Chapter 4]
7. Specification [Chapter 5]
8. Verification [Chapter 6]
9. The Software Development Process [Chapter 7]
10. Design Patterns

Grade Assessment

1. Participation 5%
2. Assignments 20%
3. Midterm 25%
4. Final Exam 50%

Participation

- Based on participation in lectures
 - ▶ Answer questions
 - ▶ Ask questions

Assignments

- Four equally weighted assignments
- Assignments must be your own work
- Do not allow other students to copy your work
- Use any available resources, but explicitly cite all sources
- Keep all of your working notes and files used to prepare your solutions
- If there is a problem with a grade
 - ▶ Report it first to the TA and if necessary to the instructor
 - ▶ Report it within two weeks of receiving your grade
- The assignment grade will only be counted in the final grade if the weighted average of the participation, midterm and final is greater than 50 %

Examinations

- Midterm
 - ▶ 90 minutes
 - ▶ Time and date on course outline
 - ▶ Multiple choice
 - ▶ Bring your student card
 - ▶ Bring pencil
- Final examination
 - ▶ 2.5 hours
 - ▶ Multiple choice
 - ▶ Scheduled by registrar
 - ▶ Will cover entire course
- Both tests are open book
 - ▶ You may bring any paper based resources
 - ▶ Any textbook is fine
 - ▶ Any notes you have created are fine

Policy Statements

- No calculators
- Ideas to improve the course are welcomed
- Missed/late work use MSAF, or a penalty of 20 % per working day
- If there is a problem with discrimination please contact the Department Chair, or other appropriate body

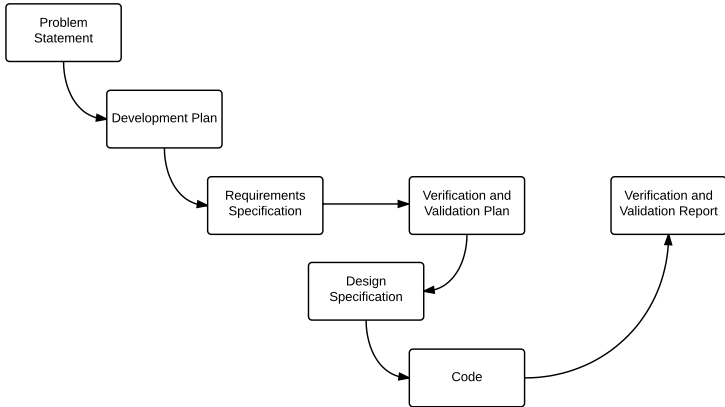
Academic Dishonesty

- Academic dishonesty consists of misrepresentation by deception or by other fraudulent means
- Can result in serious consequences, e.g. the grade of zero on an assignment, loss of credit with a notation on the transcript, and/or suspension or expulsion from the university.
- It is your responsibility to understand what constitutes academic dishonesty
- Three examples of academic dishonesty
 - ▶ Plagiarism
 - ▶ Improper collaboration
 - ▶ Copying or using unauthorized aids in tests and examinations
- Academic dishonesty will not be tolerated!

Course Evaluations

Class Participation	Bonus
80–84%	0.75
85–89%	1.00
90–94%	1.25
95–100%	1.50

“Faked” Rational Design Process



See Parnas and Clements 1986 about “Faking It”

Software Engineering versus Computer Science

What is the difference between Software Engineering and Computer Science?

What is Software Engineering?

- An area of engineering that deals with the development of software systems that
 - ▶ Are large or complex
 - ▶ Exist in multiple versions
 - ▶ Exist for large period of time
 - ▶ Are continuously being modified
 - ▶ Are built by teams
- Software engineering is “application of a systematic, disciplined, quantifiable approach to the development, operation and maintenance of software” (IEEE 1990)
- D. Parnas (1978) defines software engineering as “multi-person construction of multi-version software”
- Like other areas of engineering, software engineering relies heavily on mathematical techniques, especially logic and discrete mathematics

The PEO

- Degree from an accredited program
- Experience requirement
- Law exam
 - ▶ Contracts
 - ▶ Torts
 - ▶ Exculpatory evidence
 - ▶ ...
- Ethics
 - ▶ Duty to society
 - ▶ Duty to employer
 - ▶ Duty to profession

Software Engineering in System Design

- A physical system is often controlled by a software system called an embedded system
- As a result, software engineering is often a crucial part of system design
- Examples of embedded systems
 - ▶ Cell phones
 - ▶ Nuclear power plants
 - ▶ Automobiles
 - ▶ Aircraft
 - ▶ Pacemakers
 - ▶ mp3 players
 - ▶ Programmable household devices
- Embedded systems are rapidly appearing everywhere
- The developers of software for an embedded system needs to understand both the software and the physical device.