About ENGF0002.

Design and Professional Skills

Mark Handley, University College London

Term 1, 2019

Notice of Recording & Questions.

This course is **video recorded**, and your questions may be recorded too.

- Feel free to not provide your name in questions.
- We intend to make recordings available through moodle and lecture cast.

Please raise your hand at **any time if you have a question**.

- Ask for clarifications about the aims of teaching you anything.
- Or to help the understanding of the content of the course.
- Ask lecturers, and TAs, to speak slowly or write things down to help comprehension.
- More will follow . . .

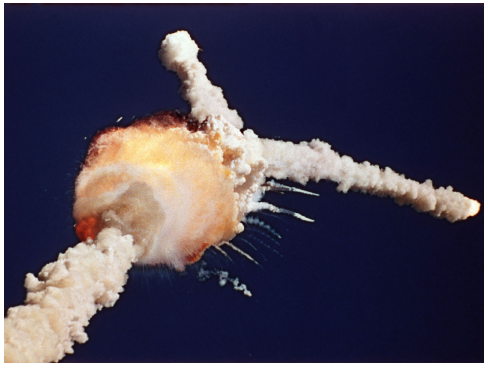
Welcome to UCL Engineering & Computer Science!

Our aim is to prepare you to be the **best engineers in the world**!

Engineering is the discipline of **shaping the world** around us for the **benefit of humanity**.

The cost of bad engineering.

Never forget the **human cost** of bad engineering,
and your **personal professional responsibility**.



Overall aims of the ENGF0002.

The essence of good engineering is **quality**. This course aims to:

- Teach you professional skills important in all engineering.
- Teach you computer science specific professional skills.
- Expose you to important social and ethical discussions.
- Provide opportunities to practice all those skills.

Most of the teaching is by Computer Science faculty, but this is an Engineering module, so some of the non-CS parts will be assessed by Engineering staff.

Who are we? The course leaders.

Mark Handley is Professor of Networked Systems at UCL, with interests in Internet technologies, network protocols, and a bit of robotics.

Stefano Vissicchio is a Lecturer at UCL, with interests in in theory, algorithms and systems for efficiently and reliably managing communication networks.

Sarah Sanders is a Teaching Fellow at UCL, with interests in the use of technology to support the teaching and learning of computer science.

Who are we? Our teaching assistants.

Gerco Van Heerdt

Stefan Zetszsche

Jana Wagemaker

Tobias Kappe

Tao Gu

They will provide you with one-on-one help in our programming clinics.

What Engineering Skills?

You will spend a significant part of your career **communicating** with other engineers, other professionals and the public.

To help you we teach:

- Technical writing.
- Engineering visualization.
- Technical Presentations.
- Effective team work & dynamics.

What Computer Science Skills?

Computer Science lies at the intersection of **mathematics** and **engineering**. It solves problems through the manipulation of **information**, and **computation**.

This module will cover:

- The **principles of programming** illustrated in Python.
- Techniques for developing and reasoning about **software correctness**.
- Techniques for **debugging software**.
- The principles of **data structures** and **algorithms**, their implementation and performance.
- The organization of **software production** in teams.
- Key tools and technologies for **rapid prototyping**.

Topics Covered

Computer Science	Engineering
Python Basics	Pebble in the Pond
Debugging and Testing	Visualization
Code Structure	Technical Argument
Data Structures	Teamwork
Algorithms	Technical Writing
Development Practices	Presentation
Networked Apps	Ethics
Real-world Systems	

Lectures

Familiarize your self with the timetable:
`https://timetable.ucl.ac.uk`.

- Tuesday 09:00–11:00 – Lectures.
- Friday 11:00–13:00 – Lectures.
  All lectures are in Bentham House Denys Holland Lecture Theatre.

CS Practical Work

Aim is to have practical work every week - most of it programming.

Most of the practical exercises are **not assessed**. You must hand in a reasonable attempt at solution though - will receive a **binary mark**.

For unassessed exercises, you may work with your friends, or in small groups. You can learn a lot by discussing ideas together.

Generally, working in pairs works best. In larger groups you learn less.

For assessed exercises, your work **must be your own**. We will make it clear which these are, and will run plagiarism detection software on the submissions.

Practical Exercises

Goal is to learn by doing

And by having some fun

Will give you software for some simple retro games.
Your tasks:

1. Play game, find bugs.
2. Understand bugs by reading and instrumenting code.
3. Fix the bugs.

Option: improve the games.

Why Debugging?

- You can't write bug free code.
- You can't write non-trivial code unless you can debug it.
- You can only learn debugging by doing it.
- Debugging trivial code doesn't teach you much.

Forces you to confront code that you don't understand, and progressively learn about it.

Teaches you programming by example.

Teaches you about code structure.

Assessment

Course has no exam.

- Binary marks from practical exercises and classroom tests 5%.
- Mid-term coursework (week 5): 15%
- Technical writing coursework (week 8): 15%
- Ethics report (400-500 words, due end of Feb 2018): 15%
- Ethics report presentations (during term 2 reading week): 10%
- CS Scenario week (term 2): 40%

We will provide more information during term.

Pebble in the Pond activity

Friday 5th October, 11am-1pm. Jeffery Hall.

- You are working in groups of 5-6, randomly assigned on arrival.
- Each group is in charge of a segment of table.
- You are provided materials, and need to design a machine.
- That takes a pebble (rock) from one end of the set of tables to the other. The pebble must only touch the machine.
- Aims: fun, opportunity to meet & reflections on team work.

Arrive promptly at 11am, find your group, and pick a CS table at random quickly! Then listen to the instructions carefully.

2017 Taster: `https://mediacentral.ucl.ac.uk/Player/95966042`

Course material.

All course material is available on-line, on github and moodle.

Computer Science specific **syllabus, slides & code**:
`https://github.com/mhandley/ENGF0002-2019`

To support you while learning the Python programming language:

Allen Downey. **Think Python**. Green Tea Press. (free e-book)
`http://greenteapress.com/thinkpython/thinkpython.pdf`

Bruce Eckel. **Thinking in Python**. (More advanced)
`http://docs.linuxtone.org/ebooks/Python/Thinking_In_Python.pdf`

How to seek help with the course.

Use the Piazza discussion board first for questions:
`https://piazza.com/ucl.ac.uk/fall2019/engf0002/home`

You must sign up to Piazza, or you won't receive announcements!

Our excellent team of TAs help with questions and provide feedback:

- Gerco Van Heerdt — `gerco.heerdt.16@ucl.ac.uk`
- Stefan Zetszsche — `stefan.zetzsche.18@ucl.ac.uk`
- Jana Wagemaker — `j.wagemaker@ucl.ac.uk`
- Tobias Kappe — `tobias.kappe.16@ucl.ac.uk`
- Tao Gu — `tao.gu22@gmail.com`

We plan to run ENGF0002 programming clinics daily, whenever you have practical exercises running.

Daily Programming Clinics.

Current plan:

- Monday, 3-5pm
- Tuesday, 12am-2pm
- Wednesday, 3-5pm
- Thursday, 4pm-6pm
- Friday, 3-5pm

Malet Place Engineering Building, 7th floor lift lobby.
No need to book, but if you all turn up at once, won't work.

How to communicate issues about the course.

We always welcome your feedback to improve courses!

Provide fixes to course material using the **Github issue tracker**:
`https://github.com/mhandley/ENGF0002-2019/issues`
Use **Piazza** for general questions and discussion.

You are welcome to bring up any issues to:

- the **course leaders or TAs**.
- the **Undergraduate coordinator** (`E.Barr@ucl.ac.uk`).
- our Dept. **teaching team** (`Louisa.Ball@ucl.ac.uk`).
- the **Director of Studies** (`L.Griffin@cs.ucl.ac.uk`).
- the **Head of Department** (`S.Hailes@cs.ucl.ac.uk`)

Professionalism & Code of Conduct (I).

We expect you to keep a very high standard of **professionalism** in this course. Including a high degree of **courtesy** in all your interactions, and **integrity** when it comes to assessment.

The **British Computing Society** (BCS) defines a code of conduct:

The Public interest: Respect for public health, privacy, security and wellbeing of others and the environment. Respect the rights of others. Conduct activities without discrimination, and extend to all the benefits of computing.

Competence and Integrity: do not overclaim your competence, and only accept work within it; keep up professional development; respect all view points and offer honest opinions; do not be malicious or neglectful; do not bride or be bribed.

Professionalism & Code of Conduct (II).

Duty to Relevant Authority: Implement diligently requirements, while exercising your judgment; avoid conflicts of interest; accept personal responsibility for your work and reports; keep confidences; and do not withhold relevant information.

Duty to the profession: improve standards in the profession; support the BCS activities and members, and do not bring it into disrepute.

Full code of conduct: `http://www.bcs.org/category/6030`.

If ever in doubt please approach us for discussion or advice.