

COMP10001 Foundations of Computing

Introduction to Computing

Semester 1, 2019

Tim Baldwin, Nic Geard, Farah Khan, and Marion Zalk



— VERSION: 1460, DATE: MARCH 7, 2019 —

© 2019 The University of Melbourne

Lecture Outline

- 1 Reminders
- 2 Subject Intro
- 3 Foundations of Computing

Female-only Workshop

- We will be running a **female-only** workshop again this semester (with teaching staff also all-female), in response to student/student club feedback, and building off a successful pilot in 2018, on **Tue 3:15–4:15 + Thu 2:15–3:15**
- Yes, this has legal signoff from the university ... and no, this is not in any way a patronising gesture — same content, same pace as other workshops, just different sub-cohort of students
- If you **identify as female** and are interested, email us and we will manually enrol you:
`comp10001-lecturers@lists.unimelb.edu.au`

Lecture Agenda

- Tips from Uncle Tim
- What did last year's students think? (SES)
- Will you fail this subject?
- What is a computer, and how do we talk to it?
- Python
- Grok

Proficiency Test

- Available for those who have a strong computational/programming background (in any language)
- Successful completion of the test will allow you to go straight into COMP10002 Foundations of Algorithms, or equivalent
- Email the lecturers **today** if you are interested in sitting the test:

`comp10001-lecturers@lists.unimelb.edu.au`

Lecture Outline

- 1 Reminders
- 2 Subject Intro
- 3 Foundations of Computing

Secret to Subject Success

- Balanced workload of 10–13 hours per week, e.g.:
 - Workshop (2 hours attendance + 2 hours follow-up)
 - Lectures (3 hours attendance + 3 hours review)
 - Study (2 hours review/reading/study group)
 - Form an informal study group and copiously share ideas, and **non-embargoed** code
- Attend, listen, ask, and share, but above all do, do, do!

Tim's Secret Checklist of Things to Point out

- All “tutesheets” from workshops will be made available (complete with a solution) at the end of each week, on completion of all workshops
- Within Grok Learning, you will be able to access the COMP10001 materials, and also see materials for other courses offered by Grok Learning. Note that you have free access to all Grok Learning courses, but none of it is examinable or required for this subject.

Tips from Uncle Tim

- When learning programming, constant “practice” is the key to success
- Never share any **examinable** code with your fellow students (not on the forums, not via email, not via shared machines, ...)
- If you need help, avail *yourself* of the various sources of assistance; don't expect us to come chasing you

Tim's Secret Checklist of Things to Point out

- Paired lectures (on Tue/Thu/Fri) will have the same content; attend the one you are enrolled in
- Guest lectures will be examinable
- Log in to Grok Learning using your unimelb credentials (@student.unimelb.edu.au account name, and unimelb password):
groklearning.com/course/unimelb-comp10001-2019-s1/

Tips from Uncle Tim

- All lecture slides will be made available on the LMS ahead of time (but may be tweaked slightly leading up to/after the lecture; watch the version numbers)
- Lectures and workshops start 5 mins later and end 5 mins earlier than advertised
- All lectures will be recorded (audio and screen scrape) ... but try to come along to ask questions and get the full lecture experience
- Expect things to move faster and marks to be harder to get than in high school

We Need You!

- We need 2–3 volunteers per lecture stream to act as “student representatives” for the subject, with the following responsibilities:
 - keep finger on pulse of student body
 - (possibly) act as go-between between students and teaching staff
 - attend a Staff–Student Liaison Committee meeting later in the semester to report on issues with the subject, and run a feedback session immediately beforehand to poll the student body

SES – Student Experience Survey

- At the end of each semester in each subject you will be asked to fill out an SES survey.
- Summary of last year's feedback:
 - Awesome subject. The lecturers are awesome. Tutors are awesome. etc
 - Grok is even awesomer
 - Room for improvement with feedback to students
 - This subject is very challenging and rewarding OR too hard

Academic Honesty

- The only possible exception to this for COMP10001 is where you have been provided with “skeleton” code as part of a project, in which case you should clearly attribute the code in comments, e.g.:

```
#####
# The following block of code was taken from the skeleton
# code provided by Tim Baldwin
:
# End of provided skeleton code
#####
```

Academic Honesty

- Common attempts to escape undetected are:
 - changing the comments but not the code
 - changing variable names
 - rearranging blocks of code (sometimes breaking the logic in the process!)
- It is all too easy to automatically pick up on all of these, and many, many more, approaches using software plagiarism detection software ... and we **do** check

Academic Honesty

- In accordance with the University's Academic Honesty and Plagiarism Policy (which you should familiarize yourself with!): <https://academichonesty.unimelb.edu.au/>
All examinable work (Grok worksheet answers and all project work) that you submit for COMP10001 must be **your own**

Academic Honesty

- Common causes of breaches in the past have been:
 - friends asking to look over your code to “get hints” for their own project
 - flatmates accessing your code via a shared computer with saved login details
 - study groups where the facilitator has overstepped the line and provided sample code to help people along
 - students posting coding online for feedback from others
 - students posting questions to programming forums to source answers directly

Sobering Statistics

- In 2018, meetings were held with 162 students relating to concerns over Academic Honesty
- Attempts at plagiarism get caught, and lead to un-fun academic honesty hearings
- Don't fall into any traps, and don't let your friends make a mistake
- Never share any **examinable** code with your fellow students (not on the forums, not via email, not via shared machines, ...)

So What *is* Appropriate?

- You are encouraged to share/collaborate directly on code for any non-examinable items (notably the tutesheet questions and the practice project) ... and you will learn a lot from reading the code of others (including the sample solutions in the worksheets)
- You are very welcome to discuss with fellow classmates your *approach* to worksheet questions and the projects, in conceptual terms, or in terms of key data types or programming constructs used (just **not** with the aid of raw code)

What is a Computer?

- A big grid/matrix of cells (memory locations)
- Can add, multiply and compare cells really fast (instructions)
- Can run a “program” (list of instructions = machine code)
- At the most basic level, computers use binary (one or zero)
- E.g. to turn top left pixel on the screen red ...

```
10010010 00000001 11111111000000000000000000000000
10001010 00000001 11010010010100101001010100001001
```

There are Lots of Programming Languages

- http://en.wikipedia.org/wiki/List_of_programming_languages
- We will use Python 3.6
- You just write it like a text file, and the Python “interpreter” turns it into machine code for you

Lecture Outline

- 1 Reminders
- 2 Subject Intro
- 3 Foundations of Computing

Obviously not human friendly

- Easier (assembly language)

```
LDC r1 0xFF000000
ST0 r1 #D2529509
```

- Even better (Python-like)

```
screen[0, 0] = (255, 0, 0, 0)
```

- Best?

Hey Sirixa, make the pixel at the top left of the screen red!

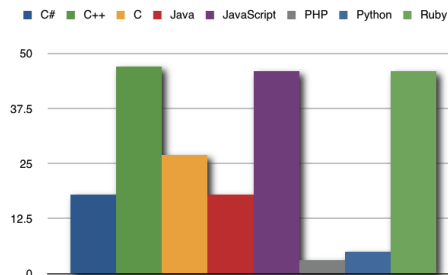
A Message from the Python Creator



“Actually, my initial goal for Python was to serve as a second language for people who were C or C++ programmers, but who had work where writing a C program was just not effective.”
— Guido van Rossum (the Benevolent Dictator for Life)

And Another Thing ...

The relative proportion of profanities per line in code written in different languages:



Source(s): <http://ilovecharts.tumblr.com/post/3463898034/amount-of-profanity-in-git-commit-messages-per>

Lecture Summary

- Computers speak binary, but we don't
- High level programming languages make life easier
- We will use Python inside Grok

Grok Learning

- Grok Learning is the web-based programming environment we will be using for the duration of this subject:
 - groklearning.com/course/unimelb-comp10001-2019-s1/
- All you need to access the system is a browser, an internet connection and your unimelb account
- Different modes of working in Grok:
 - code, run, mark
 - terminal

A Reminder about Workshops

- Just to make sure you got the message:
 - ONLY** go to a workshop (W01 or W02) this week if it is on **FRIDAY**; other workshops start from **NEXT WEEK**