COMP10001 Foundations of Computing
Introduction to Computing

Semester 2, 2021

Chris Leckie, Marion Zalk and Farah Khan

THE UNIVERSITY OF MELBOURNE

— VERSION: 1461, DATE: MARCH 7, 2019 —

Lecture Agenda

- Tips for winning at COMP10001
- What did preious students think? (SES)
- Will you fail this subject?
- What is a computer, and how do we talk to it?
- Python
- Grok

Lecture Outline

Female-only Workshop

- We will be running a **female-only** workshop again this semester (with teaching staff also all-female)
- This initiative was prompted by student/student club suggestions in 2018, and has received positive feedback to date
- This workshop has exactly the same content, and proceeds at exactly the same pace as other workshops, just different sub-cohort of students
- The workshop will run at 3.15pm Tuesdays; if you identify as female and are interested, email us and we will confirm details:

  `comp10001s2-lecturers@lists.unimelb.edu.au`

Lecture Outline

How do I Get Help?

- See the additional slides "Getting Help"

Tips for winning at COMP10001

- Balanced workload of 10–13 hours per week, e.g.:
  - Tutorial (1 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!

Tips for winning at COMP10001

- All lecture content and "non-optional" Grok worksheets are examinable
- Guest lectures will be examinable
- Advanced lectures and "optional" worksheets are **not** examinable
- Access Grok Learning using your unimelb credentials via the Grok Worksheets on the Assignments page of the LMS

Tips for winning at COMP10001

- 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 some other Grok Learning courses, but none of it is examinable or required for this subject.

Tips for winning at COMP10001

- All lecture slides will be made available on the LMS ahead of time (but may be tweaked slightly leading up to/after the lecture)
- 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

Tips for winning at COMP10001

- 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

SES – Student Experience Survey

- At the end of previous semesters in each subject students are 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
  - Chris is a sick dog
  - This subject is very challenging and rewarding OR too hard

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

- 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 Nic Geard

:

# End of provided skeleton code
############################################################
```

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

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

Sobering Statistics

- For example, 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)

Lecture Outline

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
```

Obviously not human friendly

- Easier (assembly language)

```
LDC r1 0xFF000000
STO 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!

There are Lots of Programming Languages

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

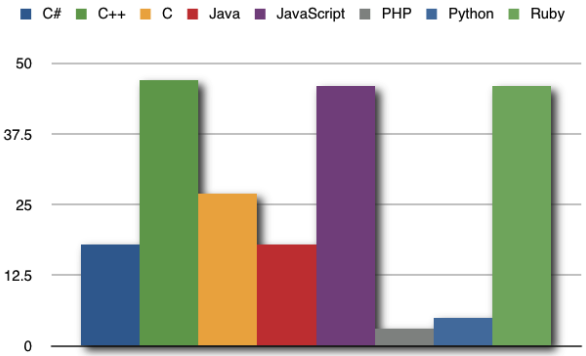
A Message from the Python Creator

"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)

Source(s): `goo.gl/mk5ou2`

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`

Grok Learning

- Grok Learning is the web-based programming environment we will be using for the duration of this subject
- 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

Lecture Summary

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

A Reminder about Tutorials/Workshops

- Just to make sure you got the message:

  DON'T go to workshops this week!
  Workshops start from **NEXT WEEK!**