Welcome!

COMP1511 17s2

Introduction to Programming

— Lecture 0 —
Hello, World!

Andrew Bennett
`<andrew.bennett@unsw.edu.au>`

course introduction
the big picture
many sights to C

the bigger picture…

what is programming?

doesn’t have to be in a language like C…

https://www.youtube.com/watch?v=FN2RM-CHkuI

why is programming awesome?

solving problems

What?

this is a course where you will…

learn to program

become a confident programmer

write code you’re proud of

discover the joys of programming

Who’s teaching?

Mr Andrew Bennett
is the lecturer of COMP1511/17s2

Dr John ‘jas’ Shepherd
is the course convenor

Mei Cheng Whale
is the course administrator

Zain Afzal, **Alex Hinds**, **Jashank Jeremy**, **Helena Kertesz**,
Stephen Leung, **Alex Linker**, **Curtis Millar**,
Riyasat Saber, **Mark Sonnenschein**, **Victoria Xu**
are your tutors…

they’ll be working alongside our cast of assistant tutors
Gal Aharon, **George Fidler**, **Colm Flanagan**, **Nicholas Hiebl**, **Peter Kerr**, **Carey Li**, **James O’Brien**, **Connor O’Shea**, **Zac Partridge**,
Ben Pieters-Hawke, **Keiran Sampson**, **Jack She**, **Alvin Sho**, **Jared Steiner**, **Adam Stucci**, **Ian Thorvaldson**, **Cadel Watson**,
Dean Wunder, **Anna Zhang**

Who’s learning?

…

You!

How?

Lectures
introduce theory and practice of programming

Tutorials and **Laboratories**
reinforce ideas and concepts with hands-on examples

Assignments
assess understanding of C, problem-solving skills and teamwork

Milestone Writeups
… but more on this later

Practice Practical Exams
two during semester assessing skill in using *arrays* and *linked lists*

Final Exam
a 3-hour theory and practical extravaganza, in CSE laboratories

Assessment

assignment 0
individual, worth 5%, due week ~4

assignment 1
group-work, worth 10%, due week ~6

assignment 2
group-work, worth 15%, due week ~12

7× milestone writeups
individual, worth 10% total, due fortnightly

lab exercises
groups, worth 5% total, due weekly

2× practice prac exams
worth 10% each, in weeks 7 and 13

final exam
3h theory+practical exam, worth 35%, in the exam period

Communications

official communications from the course
will come to your UNSW email address
make sure you can receive emails!

when you send emails,
send them from your UNSW email address
and include your zID…
don't email from personal email addresses!

to get in touch with the course urgently
email `<cs1511@cse.unsw.edu.au>`

Course Evaluation and Development

assessed with myExperience and the Sturep Survey

Conduct and Integrity

treat people with courtesy and respect
… we are all humans …

* * *

pretending someone else’s work is yours is not okay.

CSE is a bit different to other places…
we don’t care *how* you reference,
we just care *that* you reference

More information?

course material lives on WebCMS3

`webcms3.cse.unsw.edu.au/COMP1511/17s2`

How to do well in this course

Prepare for all tutorials and labs
by attempting the questions before your class.

Attend all tutorials and labs.
Ask your questions.
Use your resources.

Make a list of each compile error you get,
and how you fixed it.
(they will come back to haunt you repeatedly… this will be invaluable.)

Resources

Alistair Moffat
Programing, Problem Solving, and Abstraction with C
(Pearson Educational, 2003; ISBN 978 1 74103 080 3)

Google is your friend,
as is Stack Overflow,
especially when debugging compile errors

Getting Help

read the *Comments* section on WebCMS3…
ask your questions there, if they’re not answered

ask one of your COMP1511 peers

ask your tutor! (they are all very friendly :-)

contact me –
email andrew.bennett@unsw.edu.au
subject “COMP1511 *[rest of subject]*”

Getting Started

Before the end of this week, you should:

do Lab 1.

be able to write a ‘hello world’ program
from your CSE account at uni, and

be able to write the ‘hello world’ program
from your home computer.

The CSE Labs

CSE has lab computers…

unlike other workstations at UNSW,
these don’t run Windows;
they run Unix, which is very different

the easiest way to use these
(if you’re not in a lab)
is using VLAB

use your zID and zPass to log in
if you don’t have a zID/zPass,
you should fix that asap!

It’s All Text!

we write programs in a **text editor**
very different to (e.g.) Word or Pages

we’ll be programming in **C**
which has well-defined rules for how the language works,
which means we can use this to describe
something that can be turned into a program
that the computer can run

let’s try it!

```
// Prints out a friendly message.
// Andrew Bennett <andrew.bennett@unsw.edu.au>
// 2017-07-24

#include <stdio.h>
#include <stdlib.h>

int main (int argc, char *argv[]) {




    return EXIT_SUCCESS;
}
```

```c
// Prints out a friendly message.
// Andrew Bennett <andrew.bennett@unsw.edu.au>
// 2017-07-24

#include <stdio.h>
#include <stdlib.h>

int main (int argc, char *argv[]) {

    // Print out the famous 'hello world' message.
    printf ("Hello, world!\n");

    return EXIT_SUCCESS;
}
```

Navigating on Unix

pwd shows where you currently are

```
$ pwd
/import/ravel/2/andrewb
```

ls lists the items in the current directory

```
$ ls
17s2    bin     lib     public_html     tmp     web
```

mkdir makes a new directory

```
$ mkdir cs1511
$ ls
17s2    bin     cs1511  lib     public_html     tmp     web
```

Navigating on Unix

cd changes directory

```
$ cd cs1511
$ pwd
/import/ravel/2/andrewb/cs1511
$ ls
$
```

cd .. changes into the previous directory

```
$ cd ..
$ pwd
/import/ravel/2/andrewb
```

Writing a Program

to create a C program from the terminal,
open a text editor like gedit

```
$ gedit hello.c &
```

once the code is written and saved…
compile it with dcc!

```
$ dcc -o hello hello.c
```

Programming is a construction exercise

Think about the problem

Write down a proposed solutions

Break each step into smaller steps

Convert the basic steps into instructions in the programming language

Use an *editor* to create a *file* that contains the program

Use the *compiler* to check the *syntax* of the program

Test the program on a range of data

Compiling

A C program must be translated into *machine code* to be run.

this process is known as *compilation*,
and is performed by a *compiler*.

We will use a compiler named dcc for COMP1511
dcc is actually a custom wrapper aroung a compiler named clang.

Another widely used compiler is called gcc.