"Hello world! Welcome to the best term yet :)\n"
IN THIS LECTURE

TODAY....

- Welcome and Introductions
- Course Administration
- How COMP1511 works
- How to get help and the best ways to approach learning Programming
- What is programming?
- What is Linux and working in Linux
- A first look at C
JAX
Teaching Assistant
Loves long walks, treaties and pats, does not like deliveries

DR SASHA VASSAR
Lecturer in Charge/Course Convenor
Loves dogs, teaching, solving complex problems and having a good yarn...

JUNO
Teaching Assistant
Loves sleeping in random places, will bark randomly

WHO AM I?
Admin Extraordinaire

Always happy, sometimes forgetful, likes pink

Ben Briant

Admin Extraordinaire

Forum king (toppled by Paula in T2)
Now chief Sasha mind reader

Tammy Zhong
Mot
Livingfront
Your chat responder, lots of replies turn out to be just smilies...
Sunswift record setter

SOFIA DE BELLIS
Official Chocolate Thrower
Keeps the lecture chat well answered
Finds the best lecture tunes

TOM KILLINGBACK

THE LECTURE MODS
THE WONDERFUL TUTORING TEAM

https://cgi.cse.unsw.edu.au/~cs1511/23T1/team/
All course information can be found HERE (not Moodle!)

https://cgi.cse.unsw.edu.au/~cs1511/23T1/
ADMIN RELATED ISSUES:
Email the course email for all admin related enquiries:
cs1511@unsw.edu.au

FOR ANY ENROLMENT ISSUES:
UNSW Nucleus Student Hub
https://nucleus.unsw.edu.au/en/contact-us

ELP PLANS
If you have an ELP plan in place, please email me directly on a.vassar@unsw.edu.au
COMMUNICATION

FORUM
Post all your questions here and feel free to answer others' questions
https://edstem.org/au/courses/10623/discussion/

ASK QUESTIONS IN TUT/LABS

HELP SESSIONS
Schedule will be announced shortly

Good place to get help outside of normal lab/tutorial times
SO WHAT IS COMP1511?

- It is your intro to programming
- This is where the journey starts :)
- Computers can only follow instructions that we give them to solve problems
- Writing a program is providing the computer with a set of instructions
- Problem solving is a very important skill, can only be built up with practice!
COURSE FORMAT

• We assume no prior knowledge & zero previous programming experience
• We teach you the fundamentals of programming, how to approach and solve problems, and how to talk to computers in a common language
LECTURES
TWO HOUR SESSION TWICE A WEEK

• Live in CLB6 and streamed online via YouTube Live (recordings will be available)
  ○ Monday 11am-1pm (AEDT)
  ○ Thursday 12pm-2pm (AEDT)
• Week 6 is Flex Week, so no formal lectures!
• If you have a question, feel free to ask in live chat
• Please be respectful of others at all times - everyone is here to learn.
Theory - What are we trying to understand?
Demonstrations - Some live coding to show you how some things work
Problem Solving - How do we decide what to code?
Other stuff - Outside of programming, what’s important?
Lecture slides (and other materials) are available from the Course Website
https://cgi.cse.unsw.edu.au/COMP1511/23T1/
Lecture recordings will be in the YouTube playlist and linked via the Course Website
Go further in depth into the topics we're teaching
Actual practical working of tasks and problems we've given you
Learning how to solve problems before you write the code!
Tutorial Questions will be available in advance of the tutorials on the course website: https://cgi.cse.unsw.edu.au/COMP1511/23T1/
Online and face-to-face: please check your timetable for your enrolment details.

For online classes, use Teams.

Please turn on your cameras if you can.

We love seeing pets make an appearance.

Sample answers released after the last tutorial for the week.

"Tutorials are a good place for interactive learning. You’ll have time to discuss and work through problems there."

TUTORIALS
ONE HOUR
CLASSROOM ENVIRONMENT

"Tutorials are a good place for interactive learning. You’ll have time to discuss and work through problems there."

- Online and face-to-face: please check your timetable for your enrolment details
  - For online classes, use Teams
  - Please turn on your cameras if you can
  - We love seeing pets make an appearance
- Sample answers released after the last tutorial for the week
• Practical coding including working in small groups
• Time to have one on one conversations with your tutors
• Problem sets will be marked automatically and count towards your final marks (15% total over the term)
• There are challenge exercises for earning bonus marks (not necessary and some are hard enough that they'll eat up a lot of time)
• Tutorials and Labs do NOT run in Week 6
ASSIGNMENTS
LARGER SCALE PROJECTS

"Start the assignments early, so that you have time to chip away and get help as needed."

- Individual work
- These will take you a few weeks and will test how well you can apply the theory you've learnt
- There are two Assignments due:
  - Assignment 1 - 20% (Monday 8pm Week 7)
  - Assignment 2 - 25% (Friday 8pm Week 10)
- Late penalties of 5% per day late apply off the ceiling (maximum lateness is five days, after which time it is zero marks)
HELP SESSIONS
OPTIONAL SESSIONS SCHEDULED DURING THE WEEK

"A great place to ask questions and get help to fill any gaps."

- Held both in-person and some online, using Teams
- Face-to-face help sessions will have lab spaces allocated
- Some one-on-one consultation with tutors
- Time for you to ask individual questions or get help with specific problems
- Schedule will be up on the Course Website soon
- These are particularly busy around Assignment deadlines
PASS SESSIONS
PEER ASSISTED STUDY SESSIONS

- You can come to:
  - Ask questions about specific problems from lectures, tutorials and labs
  - Work on a variety of problems with friendly and experienced student leaders
  - Chat through study hacks and tips on managing time and assignments
  - Get to know other students in your course

- PASS begins in Week 3 and ends in Week 10.
- You can attend any class you like! It's great to come each week and you can also pop in only when you need help – it's up to you.
PASS SESSIONS
PEER ASSISTED
STUDY SESSIONS

- Visit student.unsw.edu.au/pass for more information or email pass@unsw.edu.au with any questions.
- In-person or online STARTING IN WEEK 3

<table>
<thead>
<tr>
<th>Day</th>
<th>Time</th>
<th>Place</th>
<th>PASS Leader</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wednesday</td>
<td>12pm</td>
<td>Quad G040</td>
<td>Adrian Lim</td>
</tr>
<tr>
<td>Wednesday</td>
<td>3pm</td>
<td>Teams PASS Channel</td>
<td>Danil Golovanov</td>
</tr>
<tr>
<td>Thursday</td>
<td>2pm</td>
<td>Teams PASS Channel</td>
<td>Adrian Lim</td>
</tr>
<tr>
<td>Thursday</td>
<td>5pm</td>
<td>Elect Eng G04</td>
<td>Danil Golovanov</td>
</tr>
</tbody>
</table>
We believe in the importance of feedback and helping you learn through that feedback!

- Running an optional module in Formatif
- Login using: https://formatif.cse.unsw.edu.au/
- We will add you to the module AFTER login
- There will be weekly OPTIONAL feedback modules
STEP 1: Upload code from your problem set
- Choose one problem from the past week’s problem set and upload the problem code that you found particularly interesting or challenging
- Use the task dropdown to select ”Ready for feedback” and upload your file.

STEP 2: Using the discussion panel, provide your tutor with a discussion prompt
- Talk about what you found interesting/challenging or an aspect of the code that you would like some feedback on
- Reflect on your solution to the problem, and discuss another potential way of solving this problem
FINAL EXAM
TAKE-HOME OPEN-BOOK EXAM

• IN-PERSON
• Expected workload of around 3 hours total
• You’ll be given a series of problems to solve in C
• You will also be expected to read some C and show you understand it
• There will also be some questions covering programming ideas

Exam Hurdles
• Parts of the exam are competency hurdles
• These questions must be answered correctly to pass the course
Labs = 15%
Assignment 1 = 20%
Assignment 2 = 25%
Final Exam = 40%

To pass the course you must:
- Score at least 50/100 overall
- Solve problems using arrays in the final exam
- Solve problems using linked lists in the final exam
Support for any issues that make it difficult for you to study

https://student.unsw.edu.au/special-consideration

You can apply now if you have existing reasons (or later if something comes up)

If you have an ELP plan, please email it directly to me:

a.vassar@unsw.edu.au
If you have an ELP plan, please email it directly to me:

a.vassar@unsw.edu.au
A Supplementary exam can be offered to students granted Special Consideration for the exam.

- Fit-to-Sit rule
- Identical in format to the main exam
- Held sometime in May (will update this as soon as dates are released, so you must make yourself available if you have been granted a supplementary exam)
CODE OF CONDUCT

This course and this University allows all students to learn, regardless of background or situation. Remember the one rule... you will not hinder anyone else's learning!

Anything connected to COMP1511, including social media, will follow respectful behaviour:

- No discrimination of any kind
- No inappropriate behaviour
  - No harassment, bullying, aggression or sexual harassment
- Full respect for the privacy of others
Plagiarism is the presentation of someone else’s work or ideas as if they were your own.

Any kind of cheating on your work for this course will incur penalties (see the course outline for details).

Collaboration on individual assessments like Assignments is considered plagiarism.

“If you don’t spend the time to learn and practice the content, the only person who loses is you.”
The internet has a lot of resources you should learn to use, just make sure you credit your sources.

No collaboration at all on individual assignments.

Your submissions are entirely your own work.

Don’t use other people’s code.

Don’t ask someone else to solve problems for you (even verbally).

Don’t provide your code to other people.

“Discussion of work and algorithms is fine (and encouraged).”
COLLABORATION VS PLAGIARISM

- At best, you'll lose the marks for the particular assignment
- At worst, you'll be asked to leave UNSW
- And even worse . . . you won't learn what you paid all this money and time to learn
IF YOU WANT MORE INFO . . .

- Course webpage
- Course forum
- Recorded Lectures (replay YouTube Streams or via Moodle)
- One on One
  - Ask your tutor during lab sessions
  - Help Sessions
- Serious Issues
  - Email: cs1511@unsw.edu.au
  - The Nucleus: nucleus.unsw.edu.au
  - CSE Help Desk: http://www.cse.unsw.edu.au/~helpdesk/
Student Support | I Need Help With...

**My Feelings and Mental Health**
Managing Low Mood, Unusual Feelings & Depression
- Mental Health Connect: student.unsw.edu.au/counselling
- Mind HUB: student.unsw.edu.au/mind-hub

**Uni and Life Pressures**
Stress, Financial, Visas, Accommodation & More
- Student Support
- Indigenous Student Support: student.unsw.edu.au/advisors
- Nura-gili-centre-indigenous-programs

**Reporting Sexual Assault/Harassment**
- Equity Diversity and Inclusion (EDI)
- ed.unsw.edu.au/sexual-misconduct

**Educational Adjustments**
To Manage my Studies and Disability / Health Condition
- Equitable Learning Services (ELS)
- student.unsw.edu.au/els

**Academic and Study Skills**
- Academic Skills
- student.unsw.edu.au/skills

**Special Consideration**
Because Life Impacts our Studies and Exams
- Special Consideration
- student.unsw.edu.au/special-consideration

**In Australia Call Afterhours**
UNSW Mental Health Support Line 1300 797 026 5pm-9am

**Outside Australia Afterhours**
24-hour Medibank Hotline +61 (2) 8905 0307
LEARNING IS HARD...

“Learning programming is a secondary skill (like many others!) — it is not intuitive like learning how to speak...”

Secondary skills are learnt slowly and with conscious and deliberate effort. It is not magic and it will not happen overnight, you have to keep practising and building up your knowledge base. Don’t feel disheartened if you do not understand something first go - try and try again, get help, let us know if there is something that is just not making sense. Make sure to attempt all your labs questions and assignments, working through these problems will help you build an understanding of how to solve similar problems, and how to use code to solve these.
Merlin has to let a potion rest for precisely 45 minutes, but he doesn’t have any instrument for measuring time. He does, however, have a flame and two fuses, which he knows each take an hour to burn, but not in a regular way (half of the fuse won’t be burned in 30 minutes). How can the wizard measure exactly 45 minutes?
WHAT IS A COMPUTER?
A TOOL . . . A MACHINE . . .
THE LOVE OF MY LIFE...

The ultimate tool in its ability to be reconfigured for different purposes.

The key elements:
• A processor to execute commands
• Memory to store information

Some trivia:
WHAT IS PROGRAMMING?

- Providing a computer with specific instructions to solve various problems
  - Using specific languages to write those instructions (code)

- At the core of it - problem solving!
  - You may go through many iterations before you get it right
    - mistakes are good!
WHAT IS AN OPERATING SYSTEM?

- An Operating System is the interface between the user and the computer hardware
- Operating Systems:
  - Execute user programs and make solving problems easier
  - Make the computer system convenient to use
- Basically, an Operating System sits between our code and the computer, providing essential services
WHAT IS LINUX?

- Linux is a Unix-based operating system:
  - Open source
  - More reliable
  - Lightweight
  - Faster, and
  - More secure
Terminal (command line driven) allow us to send simple text commands to our shell. It handles things like user input, displaying shell output.
The shell, (bash, zsh) is a program that executes commands, and has its own syntax. It returns output which the terminal can display, or can launch other applications.
The prompt is controlled by the shell, and is the line of text which displays some information.
SOME IMPORTANT TERMINAL COMMANDS

- Lists all the files in the current directory: `ls`
- Makes a new directory called `directoryName`: `mkdir directoryName`
- Changes the current directory to `directoryName`: `cd directoryName`
- Moves up one level of directories (one folder level): `cd ..`
- Tells you where you are in the directory structure at the moment: `pwd`
File operations on the command line

- Copy a file from the source to the destination
  `cp source destination`
- Move a file from the source to the destination (can also be used to rename)
  `mv source destination`
- Remove a file (delete)
  `rm filename`

The `-r` tag can be added to `cp` or `rm` commands to recursively go through a directory and perform the command on all the files

`cp -r COMP1511 COMP1511_backup`
(will copy all files from my COMP1511 directory to my COMP1511_backup directory)
USING CSE’S COMPUTING RESOURCES

Our labs are running Linux with the basic tools necessary to get started.

You will definitely want to get your own computer ready to code with:

- VLAB allows you to remotely use CSE’s resources - instructions on setting this up available in the first laboratory.
- There are other more advanced options that we can help you with also - check the Home Computing site or the guides on our course website.
WHAT THE BASICS LOOK LIKE

For COMP1511 we need:

- A development environment (we will use a minimal version of VSCode)
  - Run **1511 setup** to get everything ready (you will do this in your first Lab)
- A compiler (we use dcc)
  - A translator that takes our formal human readable C and turns it into the actual machine readable program
  - The result of the compiler is a program we can "run"
- You can use VLAB to access CSE's editor and compiler
PROGRAMMING IN C

PROGRAMMING IS LIKE TALKING TO YOUR COMPUTER

- We need a shared language to be able to have this conversation
- We'll be looking at one particular language, C and learning how to write it. C is:
  - A clear language with defined rules so that nothing we write in it is ambiguous
  - Many modern programming languages are based on C
  - A good starting point for learning how to control a computer from its roots
Let's C some C

Sorry can't help myself!

// A demo program showing output in C
// Welcome to COMP1511 :)
// Buckle in, you are in for a ride!
//
// Sasha, T123

#include <stdio.h>

int main(void){
    printf("Welcome to COMP1511!\n");
    return 0;
}

Words for humans

Half our code is for the machine, the other half is for humans! (roughly)

We put “comments” in to describe to our future selves or our colleagues what we intended for this code

// in front of a line makes it a comment

If we use /* and */ everything between them will be comments

The compiler will ignore comments, so they don’t have to be proper code
BREAKING IT DOWN INTO PARTS

#include is a special tag for our compiler (line 7)

- It asks the compiler to grab another file of code and add it to ours
- In this case, it’s the Standard Input Output Library, allowing us to make text appear on the screen (as well as other things)
- Almost every C program you will write in this course will have this line

```c
1 // A demo program showing output in C
2 // Welcome to COMP1511 :)
3 // Buckle in, you are in for a ride!
4 //
5 // Sasha, T123
6
7 #include <stdio.h>
8
9 int main(void){
10   printf("Welcome to COMP1511!\n");
11   return 0;
12 }
```
A function is a block of code that is a set of instructions that returns something.

Our computer will run this code line by line, executing our instructions.

The first line has details that we’ll cover in later lectures:

- **int** is the output (return) type - this stands for integer, which is a whole number.
- **main** is the name of the function.
- **(void)** means that this function doesn’t take any input.
BREAKING IT DOWN INTO PARTS

THE "MAIN" FUNCTION

- Between the { and } are a set of program instructions

```c
{} return 0;
```

- `printf()` makes text appear on the screen. It is actually another function from stdio.h which we included.

```c
printf("Hey!\n");
```

- `return` is a C keyword that says we are now delivering the output of the function. A main that returns 0 is signifying a correct outcome of the program

```c
return 0;
```
EDITING AND COMPILATION

LET'S TRY THIS IN OUR EDITOR AND COMPILE IT

- In the linux terminal we will open the file to edit
  `code hey.c`
- Once we’re happy with the code we’ve written, we’ll compile it
  `dcc hey.c -o hey`
  - The `-o` part tells our compiler to write out a file called ”hello” that we can then run
- The `./` lets us run the program ”hello” that is in our current directory
  `./hey`
AND WE ARE OFF!

WE NOW HAVE OUR FIRST WORKING PROGRAM...

• Try this yourself!
• Try it using VLAB via your own computer
• Try setting up a programming environment on your own computer (differing levels of difficulty depending on your operating system)
Did you know that the first computer in the world, ENIAC, weighed more than 27 tonnes and covered an area of about 1800 square feet?

Designing the correct configuration for each new problem, and then connecting the wires and setting the switches, took many days.

WHAT DID WE LEARN TODAY?

ADMIN
How COMP1511 is run

RESOURCES
Where to find resources (course webpage and forum)

HELP!
How to get help and best ways to approach learning programming

WHAT IS ...?
What is programming?
What is an Operating System?
What is Linux?

LINUX
Some basic Linux commands to get you started

C
Hello World!

...
Check out the forum

CONTENT RELATED QUESTIONS
Check out the forum

ADMIN QUESTIONS
cs1511@unsw.edu.au