Course Outline Public

Edit Resource (/ENGG1811/19T2/resources/27071/edit)

Contents

- Course Details
- Course Summary
- Assumed Knowledge
- Student Learning Outcomes
- Teaching Strategies
- Teaching Rationale
- Student Conduct
- Assessment
- Course Schedule
- Resources for Students
- Course Evaluation and Development

Course Details

Course Code	ENGG1811
Course Title	Computing for Engineers
Convenor	Ashesh Mahidadia (/users/z8970925)
Lecturer	Ashesh Mahidadia (/users/z8970925)
Admin	Mei Cheng Whale (/users/z8470184)
Classes	Timetable for all classes (https://webcms3.cse.unsw.edu.au/ENGG1811/19T2/timetable)
Consultations	To be announced in Timetable for all classes (/ENGG1811/19T2/timetable)
Units of Credit	6
Course Website	https://webcms3.cse.unsw.edu.au/ENGG1811/19T2/ (https://webcms3.cse.unsw.edu.au/ENGG1811/19T2/)
Handbook Entry	http://www.handbook.unsw.edu.au/undergraduate/courses/current/ENGG1811.html (http://www.handbook.unsw.edu.au/undergraduate/courses/current/ENGG1811.html)

Course Summary

Computing is an integral part of modern engineering. Computing is used in the design, automation, experimentation, monitoring, diagnosis, data collection, data analysis, visualisation and many other aspects of engineering. A very important skill for engineers is to be able to use computers to help them to solve problems efficiently. The aim of this course is to give an introduction to computing for engineers with an emphasis on

computational problem solving. In order to realise this aim, the students will learn to use the Python programming language and some its many packages to solve problems. Since the course is designed primarily for engineers, the computing examples are often presented together with an engineering context to help the students to appreciate the applications of computing in engineering. This course also includes a minor component in Matlab and spreadsheet.

Assumed Knowledge

This course assumes that the students have taken HSC level Maths. No prior knowledge on computing is required.

Student Learning Outcomes

After completing this course, students should be able to:

1. Design algorithms to solve computation problems
2. Implement algorithms using the Python programming language
3. Able to write Python programs to automate many different tasks
4. Have some knowledge on using numerical computation packages of Matlab and Excel

The assessment are designed to assess your skills in solving problems and implementing your solution in a programming language.

This course contributes to the development of the following graduate capabilities:

Graduate Capability	Acquired in
Scholars capable of independent and collaborative enquiry, rigorous in their analysis, critique and reflection, and able to innovate by applying their knowledge and skills to the solution of novel as well as routine problems	Lectures, labs and assignments
Entrepreneurial leaders capable of initiating and embracing innovation and change, as well as engaging and enabling others to contribute to change	Assignments
Professionals capable of ethical, self- directed practice and independent lifelong learning	Labs and assignments
Global citizens who are culturally adept and capable of respecting diversity and acting in a socially just and responsible way	-

Teaching Strategies

- Lectures: introduce concepts and principles, demonstrate with example code, students working to solve problems
- Lab Work: provides an opportunity for students to solve some well-defined problem in order to practice problem solving and programming skills
- Assignments: provide opportunities for students to work on an extended programming work

Teaching Rationale

Problem solving and programming are practical skills that can be acquired through practice. The aim of the lecture is to introduce problem solving strategies and programming concepts. Examples will be used in the lecture to illustrate the concepts discussed. Students are encouraged to bring their laptop to the lectures so that they can work on the problem together.

Laboratory exercises are provided so that students can apply what they have learn in the lectures to solve problems. Surveys consistently rate lab classes as the most valuable learning experience in the course. This is because problem solving and programming skills can only be acquired by the students actually attempting, and perhaps sometimes agonising over, the problems.

Students are supported in various ways. Labs are supported by tutors and assistant tutors. Help sessions supported in tutors are run at various times in the week for students to drop in to ask questions. Additional help sessions will be run for assignments.

Student Conduct

The **Student Code of Conduct** (Information (https://student.unsw.edu.au/conduct) , Policy (https://www.gs.unsw.edu.au/policy/documents/studentcodepolicy.pdf)) sets out what the University expects from students as members of the UNSW community. As well as the learning, teaching and research environment, the University aims to provide an environment that enables students to achieve their full potential and to provide an experience consistent with the University's values and guiding principles. A condition of enrolment is that students *inform themselves* of the University's rules and policies affecting them, and conduct themselves accordingly.

In particular, students have the responsibility to observe standards of equity and respect in dealing with every member of the University community. This applies to all activities on UNSW premises and all external activities related to study and research. This includes behaviour in person as well as behaviour on social media, for example Facebook groups set up for the purpose of discussing UNSW courses or course work. Behaviour that is considered in breach of the Student Code Policy as discriminatory, sexually inappropriate, bullying, harassing, invading another's privacy or causing any person to fear for their personal safety is serious misconduct and can lead to severe penalties, including suspension or exclusion from UNSW.

If you have any concerns, you may raise them with your lecturer, or approach the School Ethics Officer (mailto:ethics-officer@cse.unsw.edu.au) , Grievance Officer (mailto:grievance-officer@cse.unsw.edu.au) , or one of the student representatives.

Plagiarism is defined as (https://student.unsw.edu.au/plagiarism) using the words or ideas of others and presenting them as your own. UNSW and CSE treat plagiarism as academic misconduct, which means that it carries penalties as severe as being excluded from further study at UNSW. There are several on-line sources to help you understand what plagiarism is and how it is dealt with at UNSW:

- Plagiarism and Academic Integrity (https://student.unsw.edu.au/plagiarism)
- UNSW Plagiarism Procedure (https://www.gs.unsw.edu.au/policy/documents/plagiarismprocedure.pdf)

Make sure that you read and understand these. Ignorance is not accepted as an excuse for plagiarism. In particular, you are also responsible that your assignment files are not accessible by anyone but you by setting the correct permissions in your CSE directory and code repository, if using. Note also that plagiarism includes paying or asking another person to do a piece of work for you and then submitting it as your own work.

UNSW has an ongoing commitment to fostering a culture of learning informed by academic integrity. All UNSW staff and students have a responsibility to adhere to this principle of academic integrity. Plagiarism undermines academic integrity and is not tolerated at UNSW. Plagiarism at UNSW is defined as using the words or ideas of others and passing them off as your own.

If you haven't done so yet, please take the time to read the full text of

- UNSW's policy regarding academic honesty and plagiarism (https://student.unsw.edu.au/plagiarism)

The pages below describe the policies and procedures in more detail:

- Student Code Policy (https://www.gs.unsw.edu.au/policy/documents/studentcodepolicy.pdf)
- Student Misconduct Procedure (https://www.gs.unsw.edu.au/policy/documents/studentmisconductprocedures.pdf)

- Plagiarism Policy Statement (https://www.gs.unsw.edu.au/policy/documents/plagiarismpolicy.pdf)
- Plagiarism Procedure (https://www.gs.unsw.edu.au/policy/documents/plagiarismprocedure.pdf)

You should also read the following page which describes your rights and responsibilities in the CSE context:

- Essential Advice for CSE Students (https://www.engineering.unsw.edu.au/computer-science-engineering/about-us/organisational-structure/student-services/policies/essential-advice-for-cse-students)

Assessment

Item	Topics	Due	Marks	Notes
Labs	All topics	10 in-class labs during Weeks 1 to 10 in your lab classes. In addition, two virtual labs.	10%	• No marks for Week-01 lab. • In-class labs in Weeks 2 to 9 are marked out of 3 marks (2 marks from tutor assessment of your lab work, 1 mark for an on-line multiple choice question). • Week-10 lab is marked out of 3 marks. • Each virtual lab is marked out of 2 marks. There are two virtual labs. • We consider best 8 in-class lab marks (24 marks), plus 4 marks for two virtual lab marks. Total lab marks 28. They will constitute 10% of the final marks.
Mid-term	Weeks 1-4 (Lectures) and Weeks 1-5 (Labs)	Week 6	10%	Mid-term will be in your lab class in Week 6
Assignment 1		Week 8	10%	
Assignment 2		Week 10	10%	
Final Exam	All topics	Exam period	60%	

**** Please note that the above deadlines are subject to change.**

Labs : The lab assessment must be completed within the 2 hour period in the designated week. To avoid tutor overload, you must be ready to have some part of the work ready for assessment 30 minutes before the end of the class. Any work presented after this time may not be able to be assessed by the tutor. **Please note** that at the time of marking your lab exercises, your tutor may ask you to solve other similar problems. You need to demonstrate that you are able to solve lab exercises and related problems, in order to receive any marks for your lab work. In other words, using someone else's lab solution is pointless!

Assignments are to be completed in your own time. To maximise the learning benefits from doing assignments, it is essential that you start work on assignments early. Do not leave your assignments until the last minute. If you submit an assignment late, the maximum available mark is reduced by an amount (usually

15%) per day that it is late. Assignments are submitted using a link from the class website. Submissions will not be accepted 3 days after the assignment deadline.

Calculations of final mark :

You receive the sum of the component marks described above in the table.

Final mark = Labs (10) + Midterm(10) + Assignment-1 (10) + Assignment-2(10) + Final exam(60).

You need to get 50 or more marks to pass the course.

Course Schedule

The following course schedule is subject to change.

Week	Lectures	Notes
1	Course introduction. Introduction to Python. Assignment and variables. Operators and Precedence. Writing the first computer program. The math library.	Labs start from Week-01, Go to your Week-01 lab!
2	Data types. Control structure. Boolean. Software development.	-
3	Functions, lists, for-loop, list comprehension, plotting	-
4	List indexing and slicing. String processing and formatting. List data processing using for-loops	-
5	While loops. Program testing and debugging.	Assignment 1 available.
6	Function arguments. Numpy (arrays and vectorization)	Mid-session in your lab class.
7	Simulation.	-
8	File input and file output. Numpy (Boolean selection, broadcast)	Assignment 1 due. Assignment 2 available.
9	Algorithms. numerical precision. main()	-
10	Introduction to Machine Learning. Course Revision.	Assignment 2 due.

** Please note that the above deadlines are subject to change.

Online Modules : There are two online modules and virtual labs that have to be completed. These two online modules are on Spreadsheet and Matlab respectively. The online module for Excel will be available in Week 5 and should be completed in Week 8. The online module for Matlab will be available in Week 7 and should be completed by Week 10. Please note that the weeks for availability and submissions may change.

Resources for Students

Lecture notes, sample programs, lab exercises will be available at the course web-site.

This course is customised for engineering students to learn problem solving and programming in an engineering context. It will also introduce some computing science concept. There is no textbook that covers all these topics. However, here is a selection of some general references:

- John Zelle. Python Programming: An Introduction to Computer Science. 3rd edition.
- An online book on How To Think Like a Computer Scientist (https://runestone.academy/runestone/static/thinkcspy/index.html)

We will provide additional resources on the class web page to cover other many useful topics.

Course Evaluation and Development

This course is evaluated each session using the myExperience system.

In this session, the lecturer will improve the lecture materials and to provide more practice problems for the students to work on.

Resource created 11 months ago (Tuesday 21 May 2019, 08:39:14 PM), last modified 11 months ago (Thursday 30 May 2019, 08:45:28 PM).

Comments

(/ENGG1811/19T2/forums/search?forum_choice=resource/27071) (/ENGG1811/19T2/forums/resource/27071)

There are no comments yet.