ENGG1811 Computing for Engineers Semester 1, 2018

Course Introduction

1. Course Aims

Computing is an integral part of modern engineering. Computing is used in 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 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.

2. Student Learning Outcomes

After completing this course, students should be able to:

- Design algorithms to solve computation problems
- Implement algorithms using the Python programming language
- Able to write Python programs to automate many different tasks
- 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 programming language.

3. Teaching, Learning and Assessment

At university, the focus is on the self-directed search for knowledge. Lectures, textbooks and recommended reading, labs, and assignments are all provided as a service to assist you in this endeavour. You must choose the approach that best suits your learning style and goals in this course. Note that the University expects you to do about 180 hours of work for this course including lecture attendance and review, labs, and time spent on assignments. The best strategy to perform well in this course is to put in consistent and regular effort in **practising**. The word practising is typeset in bold because you can only learn how to solve problems and writing computer programs by actually attempting and doing, and perhaps sometimes agonising over, practice problems. No one has ever said learning is easy, but the reward is there if you are willing to put your effort in.

Lectures

The aim of the lectures is to facilitate learning and understanding of the important concepts within course syllabus. They will provide the main source of course material delivered. There are three hours of lectures per week.

In the lectures, we will

- explain each topic
- demonstrate the principles underpinning each topic with examples
- discuss how related problems arise and can be solved

In the lecture, we will also be asking you to write some short code or solve some small problems. In order to get the most out of this experience, we ask you to install the software package Anaconda for Python 3.6 and bring your laptop with you to the lecture.

You are invited to ask questions at any time if you think the question and answer is relevant to everyone else, or at the end of the class.

Discussion with the person next to you in class is *not acceptable*, as it creates a level of noise that inevitably impacts on the rights of others who are trying to listen and learn.

Lecture Schedule

The proposed lecture schedule is:

Week	Lecture topics:	Remarks:
1	Course introduction. Introduction to Python. Assignment and variables. Operators and Precedence. Writing the first computer program. The math library.	Before the lecture: Install Anaconda for Python 3.6 on your laptop. Bring your laptop to the lectures. No labs this week.
2	Data types. Control structure. Boolean. Software development.	Labs run from Week 2 to Week 13
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. Function arguments. Program testing and debugging.	Assignment 1 available.
6	Numpy	Mid-session in your lab class.
7	Simulation. Mutable and immutatble data types.	
8	Algorithms and numerical precision.	Assignment 1 due. Assignment 2 available.
9	File input and file output. Exception handling.	
10	Matlab	
11	Spreadsheet	
12	Introduction to Machine Learning. Revision.	Assignment 2 due.
13	(No lectures.)	Last Lab

Please note that the above schedule is subject to change.

All lectures will be recorded.

Laboratory Classes

All lab classes start in Week 2. Each weekly lab class is 2 hours long.

Laboratory classes give you a chance to **practise** problem solving and programming skills on small, well defined examples. These examples have been chosen to highlight particular aspects of problem solution, and will give you enough grounding in problem solving to assist you in completing your assignment work. Your tutor will be present in your lab class to answer any questions you may have and to assess your competencies.

The lab exercises will be available during the semester via the course's web page. You must have reviewed the exercises, and preferably attempted at least the first of them, before the class. **Take care not to fall behind in laboratory work, as it is difficult to catch up**.

Surveys consistently rate lab classes as the most valuable learning experience in the course. 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.

Each lab is marked out of 3 marks: 1 mark from a multiple choice question and 2 marks from your tutor's assessment of your lab work.

In each lab, you need to answer online one multiple choice question from a bank of similar questions, and this carries 1 mark of the nominal 3 marks for each lab. These multiple choice questions will be based on the previous week's lecture material and/or the first part of the present lab exercises. Your tutor will assess your other lab work and provide you with a mark out of 2.

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 some else's lab solution is pointless!

If you miss an occasional lab class it will not matter much as the best 10 lab assessments are used for your lab mark. Similarly a day or two of illness should not have a serious impact on assignment work if you've started early (as you should). However, if you are absent for an extended period due to illness, then you may apply to the class administrator (email the class account en1811@cse.unsw.edu.au) for an extension. Written evidence of the problem will be required. A similar procedure applies if you miss the mid-semester test in week 6 due to a documented medical or similar problem.

You can only have your lab work marked during **your** scheduled **lab class**. It is *not* possible to have your work marked in another class.

Consultation

Each week, starting from week 3, the lecturer will be available to discuss with you any aspect of the course that you are having problems with. This is the most personalised form of teaching for this course, and you should take every advantage of it, particularly if you are experiencing difficulties. When assignments are due many additional consultations will be offered. If you need to see the lecturer in the first 2 weeks, please email for an appointment.

Please **check the class web page for current consultation time**s (see "Consultations" in the left panel of the class web page).

Assignments

Assignments give you the chance to practise what you have learned on relatively large problems (compared to the small exercises in the Lab Classes). Assignments are a very important part of this course, therefore it is essential that you attempt them yourself.

There will be two assignments:

- Assignment 1 is due in Week 8 (requirements available in week 5)
- Assignment 2 is due in Week 12 (requirements available in week 8)

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.

The aim for these assignments is to develop skills in all stages of the programming process: start by understanding the requirements; design a method to solve the problem; refine this design; implement and document the design and test extensively to validate that the program meets the requirements.

Examinations

A mid-semester test will be held in Week 6 during your scheduled lab class. It covers the first 4 weeks of lecture material (labs 2 to 5).

The final examination will be held during the examination period. It will examine all material covered in the course. The final exam will take place in computer labs.

Supplementary Exams

The document Essential Advice for CSE Students states the supplementary assessment policy for the School of CSE. The following criteria will be used when considering a student for a supplementary exam:

- has the student submitted a fully documented request for Special Consideration within 3 days of the exam;
- was there medical or other evidence to support the claim that the student's performance was seriously affected? Minor ailments like a cold; a headache unless chronic, or severe enough to require prescribed medication; feeling "unwell", *etc* do not excuse a poor result;
- assignment and laboratory marks may not carry much weight in determining whether further assessment is warranted; and
- the student did not attend the exam, or attended the exam but was forced to leave through illness.

Please take the time to read it carefully. If you are granted a Supplementary examination, then it will be held on the date specified in the above document. If you think that you may be eligible for a supplementary exam, then make sure you are available on that day. It is your responsibility to check at the School Office for details of Supplementary examinations. Please note that there will be NO further supplementary dates for this course. In other words we will NOT be able to offer you supplementary examination at any other time.

Assessment

The assessable components of the course are:

Component	**Mark**
Assignment 1	10%
Assignment 2	10%
Lab exercises, best 10 (of 12)	10%
Mid-Semester Test	10%
Final Exam	60%

Important:

In normal circumstances you receive the sum of the component marks. 50 is a clear pass.

However, if your exam mark is poor (less than one third or 20/60), your final mark is calculated using the formula

Final = 3.75 * **Exam** * **Other** /(**Exam** + **Other**)

where **Exam** is the raw exam mark out of 60 and **Other** is the sum of the other assessments, out of 40. This formula results in a mark that is less than 50.

4. Course Resources: Software Packages, Print and Online Resources

Three software packages will be used in this course: Spyder for Python 3 (10 weeks), Matlab (1 week) and Excel (1 week). All these packages will be available on the CSE lab computers.

Spyder is a software package that we will use to develop Python programs. We recommend that you install Anaconda for Python 3.6 which includes Spyder as a part of the installation. Instructions (both written and video) can be found under Resources in the course website.

Matlab is a commercial software but UNSW students can get it for free during their time at UNSW; licensing conditions apply, see here: https://www.it.unsw.edu.au/students/software/matlab.html

Both Matlab and Excel are also available to UNSW students via myAccess https://www.myaccess.unsw.edu.au

Print and online resources:

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

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

5. Administration

Units of Credits

6 units of credit

Prerequisites

There are no prerequisites for this course.

Assumed knowledge

We will use mathematics frequently, but mostly informal. Fundamental concepts assumed to be familiar to you include trigonometry, calculus (functions and derivatives), complex numbers and matrix arithmetic. The latter is covered in MATH1131.

Course Web Page

http://www.cse.unsw.edu.au/~en1811

Staff

Name	Role	Email	Phone
Dr. Ashesh Mahidadia	Lecturer-in-charge	ashesh@cse.unsw.edu.au (Prefer email contact)	
Mei Cheng Whale	Course Administrator	meicheng@cse.unsw.edu.au	9385 5683 ext 55683

Getting Help

From time to time, various problems may arise in your study of this course. Below is a list of typical problems, with suggestions for where you might seek help with them:

Problem	Solution
Can't understand lecture material	Ask the lecturer after the lecture times, OR ask your tutor during the lab class, OR go to a consultation
Stuck with lab work	Ask your tutor during your lab
Stuck with assignments	Go to a consultation
Want to change lab class	Email class account (en1811@cse.unsw.edu.au)
Problems with lab computers or Problems with your account	CSE Help Desk http://www.cse.unsw.edu.au/~helpdesk/
Problems with Uniwide Access	IT Service Centre, Ground Floor, Library Annexe https://www.it.unsw.edu.au/support/servdesk/servicedesk_home.html

One final piece of advice. If you have a problem, d**o** no**t** wai**t** until late in the semester before you seek help. Problems that are dealt with early are usually easier to resolve than ones that are left to the last minute.

You may find the following links useful:

- UNSW Library: info.library.unsw.edu.au
- UNSW Learning Centre: www.lc.unsw.edu.au
- UNSW IT Services for students: www.it.unsw.edu.au/students
- Occupational Health and Safety policies and expectations: www.ohs.unsw.edu.au/ohs_students/
- CSE student resources: http://www.cse.unsw.edu.au/about-us/helpresources/for-students/
- Equity and Diversity issues: http://www.studentequity.unsw.edu.au

6. Plagiarism

All work submitted for assessment **must be your own work**. Lab exercises and assignments **must be completed individually**. We regard copying of assignments or lab exercises, in whole or part, as a serious offence. **We use plagiarism detection software to search for work derived from other students or from other sources.** Be warned that:

- the submission of work derived from another person, or jointly written with someone else will, at the very least, result in zero marks for the assignment or lab exercise.
- allowing another student to copy from you will, at the very least, result in a reduction in the mark for your own assignment or lab exercises; and
- severe or second offences will result in automatic failure of the course, or even exclusion from the University.

Collaborative learning in the form of "think tanking" is encouraged, but students are not allowed to compose programming solutions together as a group during such discussions. Students are also warned not to send code fragments of the assignments or labs to each other in any form (e.g. as email or listings). In addition, purchasing code from a solution-generating website or service or from a senior student is a very serious kind of academic misconduct.

The Learning Centre website is the main repository for resources for staff and students on plagiarism and academic honesty. These resources can be located via:

http://www.lc.unsw.edu.au/plagiarism

7. Continual course improvement

Student feedback on this course, and on the effectiveness of lectures in this course, is obtained via electronic survey (CATEI) at the end of each semester. Student feedback is taken seriously, and continual improvements are made to the course based in part on this feedback. Students are strongly encouraged to let the lecturer in charge know of any problems as soon as they arise. Suggestions and criticisms will be listened to openly, and every action will be taken to correct any issue or improve the students' learning experience.

As a result of recent student feedback and after consultation with all schools in the Faculty of Engineering we have redesigned the course and now use Python to teach computational problem solving, with introduction to MATLAB and spreadsheets at the end of the session. We believe students will find the revised version of the course more relevant to their studies.
