Course Outline

Contents

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

Course Details

Course Code	COMP1010
Course Title	The Art of Computing
Convenor	Sim Mautner (/users/z3207186)
Admin	Sim Mautner (/users/z3207186)
Classes	**Lectures** : Tuesday 12:00-14:00, Friday 14:00-16:00 Timetable for all classes (/COMP1010/21T3/timetable)
Consultations	TBD
Units of Credit	6
Course Website	http://cse.unsw.edu.au/~cs1010/21T3/ (http://cse.unsw.edu.au/~cs1010/21T3/)
Handbook Entry	http://www.handbook.unsw.edu.au/undergraduate/courses/current/COMP1010.html (http://www.handbook.unsw.edu.au/undergraduate/courses/current/COMP1010.html)
Course Contact	cs1010@cse.unsw.edu.au (mailto: cs1010@cse.unsw.edu.au)

Course Summary

This course aims to provide a grounding in computational thinking for anyone who wants one. It assumes no previous programming background, but does assume that all incoming students have used digital devices, such as tablets and smart phones, for a range of tasks (e.g. social networking, reading, essay writing, etc.). The course will use Python as the programming medium and use real world examples from a variety of domains to motivate understanding.

Assumed Knowledge

Before commencing this course, students should:

- have basic computer skills (using a mouse and keyboard, browsing the web, etc)

Student Learning Outcomes

After completing this course, students will:

1. Be able to use a spreadsheet for simple data management tasks.
2. Be able to write Python programs to solve simple computational problems.
3. Be able to solve problems via computer systems.
4. Be able to build simple human-centered interfaces to computers.

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	Labs & Project
Entrepreneurial leaders capable of initiating and embracing innovation and change, as well as engaging and enabling others to contribute to change	Project
Professionals capable of ethical, self- directed practice and independent lifelong learning	Labs & Project
Global citizens who are culturally adept and capable of respecting diversity and acting in a socially just and responsible way	All work

Teaching Strategies

- Lectures ... introduce concepts, show examples
- Tutorials ... reinforce concepts and provide additional examples
- Lab Work ... provide essential practice in programming
- Project .. allow students to solve a significant problem which matters to them

Teaching Rationale

- Online: Allows for flexibility of student location and timetable.
- Lectures recorded: We endeavour to record all lectures to allow students to view them at their own pace and easily review, and catch up on missed content.
- Learn by example: Following a small amount of explanation/overview, the content will be predominantly taught through the use of examples in which the concepts and skills of the course are applied.
- Learn by doing: Much like swimming, maths, or playing a musical instrument, computer programming cannot be learnt by solely reading books or watching others do it. It requires hands-on practice. Throughout the course we will offer many opportunities to practice the content taught in class including through the tutorial work and lab work.

How to Succeed in this Course

Some students find learning to program challenging, at least at first. Here are some guidelines

Primary approach:

- Watch the lectures.
- Attempt the examples demonstrated in the lectures, on your own.
- Attempt the tutorial questions before your tutorial class for the week.
- Complete the lab work each week.
- (And of course, do the project and final exam.)

Not-So-Secondary requirements:

- Ask questions. During lectures. During tutorials. During labs. On the forum. (https://edstem.org/ (https://edstem.org/))
- Support each other. Answer questions asked by other students on the forum. This will assist them in their learning, as well as increase your understanding of the concept in the process of explaining it.
- Ask questions.
- Find and share resources:
    - Whenever you're stuck, try Google your question first, and also search for your problem on the course forum. If you find your answer, share the problem you had as well as the solution you found, on the course forum. This will both help you to not make the same mistake again as well as provide a resource for other students who may come across the same problem.
    - Everyone learns differently. We encourage you to search out tutorials, articles, youtube videos, books, apps, games, basically anything that supports you in your journey to learning to write computer programs. When you find things which are particularly useful for you, share them on the forum.

  Notes on communicating on the forum:
    - Before asking questions on the forum, do a quick search to see if someone has already asked (and maybe even has an answer to) what you're about to ask.
    - When responding to another's question on the forum, review your response and check the following aspects (*THINK*):
        - Is it *T* rue?
        - Is it *H* elpful?
        - Is it *I* nspiring?
        - Is it *N* ecessary?
        - Is it *K* ind?
- Keep a journal. As time goes on, you'll find more and more of the questions you have, will have the answers in your journal. Include the following things:
    - When you learn something. Write what you learnt.
    - When you see a new keyword. Write the keyword, and maybe an example of its use.
    - When you have a problem. Write out what your problem is.
    - When you solve the problem. Write what the solution was.
- Ask questions.

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	Contributes to
Project - Proposal		Week 7	10%	2,3,4
Project - Implementation	HCI, Python Programming	Week 10	20%	2,3,4
Project - Feedback	HCI, Python Programming	Week 10	5%	2,3,4
Labs	All topics	Weeks 1-5, 7-10	20%	1,2,3,4
Final Exam	All topics	Exam period	45%	1,2,3,4

Lab Exercises

In weeks 1-5 and 7-10 you will need to complete a series of lab exercises. These will typically be programming exercises asking you to solve a particular problem in python. To receive marks for these labs you will need to get them marked by your lab coordinator in your allocated lab time in the week after the lab is released. For example, you will need to get the week 1 lab exercises marked in your week 2 lab. If you finish your exercises early, you can, of course, get them marked in the week they were assigned.

Note: It is likely that the total number of marks for labs for the term will be somewhere between 20 and 30. This will only be confirmed in week 10. You can expect each lab mark to be worth between 0.66% of your course grade and 1% of your course grade.

Project

In the project, you will implement an application in Python solving a problem of your choice. You will be assessed on how well you were able to write the code for this app as well as the design of its interface. While available time and skills are limitations that will need to be taken into account, the problem you choose to solve is up to you. Your project idea will need to be approved by your tutor. You are encouraged to work with a partner on your project, especially if you're new to programming, but you are not required to and can complete the project on your own if you wish.

Exam

The final exam will be open book and online. You will have 3 hours to complete it. It will involve programming exercises, but may also include multiple choice and short answer. You will **NOT** be remotely monitored while taking the exam.

The exam mark is subject to scaling and has a hurdle of 22.5/45 based on the scaled mark. In the event your final mark is greater than 50, but your exam mark is under 22.5, your final grade will be UF.

Course Schedule

This schedule is HIGHLY subject to change based on student feedback throughout the term.

Week	Lectures	Tutes	Labs	Assignments	Notes
1	Course intro, Spreadsheets, Python and Programming Fundamentals	-	-	-	-
2	Python and Programming Fundamentals	-	-	-	-
3	Python and Programming Fundamentals	-	-	-	-
4	Python and Programming Fundamentals, User Interface Design, Web Apps	-	-	-	-
5	Web Apps	-	-	-	-
6	Revision & Practice	-	-	-	Flexibility week
7	Web Apps	-	-	Project proposal due (10%)	-
8	PyGame	-	-	-	-
9	Spreadsheets, Lab Walkthroughs and Revision	-	-	-	-
10	Revision, Review, Exam Preparation, Project Demos	-	-	Project application due (20%) Project feedback due (5%)	-

Topic List:

- Python Programming Fundamentals
- Web Apps:
  - HTML
  - Flask
  - PyHTML
  - CSS
  - Cookies

Resources for Students

There are no formal textbooks for this course, but students may find the following FREE book helpful for learning and practicing python programming

- *Think Python 2nd Edition* by Allen B. Downey, available at https://greenteapress.com/wp/think-python-2e/ (https://greenteapress.com/wp/think-python-2e/)

Other FREE resources which have been discovered and recommended by past students of this course include:

- An app: https://getmimo.com/ (https://getmimo.com/)
- An interactive website: https://www.w3schools.com/python/ (https://www.w3schools.com/python/)
- Tutorial-style articles with accompanying exercises: https://pythonbasics.org/ (https://pythonbasics.org/)

Course Evaluation and Development

This course is evaluated each session using the myExperience system as well as feedback from students and tutors throughout each term.

Students in 2021 T1 indicated they would have liked more challenge exercises, so additional optional challenge exercises will be included in the course, either in the labs each week, or as a separate resource. In 2021 T2 students did not comment on this aspect of the course either positively or negatively so we have continued to include them for future terms.

Students in 2021 T2 indicated that in some areas the pace of the course was too fast. In response to this we have:

- Reduced the number of topics (removing databases, recursion) (in order to spend more time on introducing earlier topics more slowly).
- Created more tutorial and lab questions on certain topics to help students gain more competence in key areas.

In 2021 T3 we are trialing the process of students demonstrating their projects as part of the assessment process. We hope that this will allow:

- Students to demonstrate the full scope of functionality of their projects.
- Students to see what other students have created.
- Students to evaluate their peers work and thereby contribute to their critical evaluation of their own work.

Students are also encouraged to provide informal feedback during the session, and to let course staff know of any problems as soon as they arise. Suggestions will be listened to openly, positively, constructively, and thankfully, and every reasonable effort will be made to address them.

Resource created 3 months ago (Tuesday 31 August 2021, 07:04:11 AM), last modified 2 months ago (Friday 24 September 2021, 04:23:18 PM).

Comments (/COMP1010/21T3/forums/search?forum_choice=resource/65529) (/COMP1010/21T3/forums/resource/65529)

There are no comments yet.