Course Outline

Contents

- Course Details
- Course Summary
- Course Timetable
- Course Aims
- Student Learning Outcomes
- Assumed Knowledge
- Teaching Rationale
- Teaching Strategies
- Assessment
- Academic Honesty and Plagiarism
- Course Schedule
- Resources for Students
- Course Evaluation and Development

Course Details

Course Code	COMP1917
Course Title	Computing 1
Units of Credit	6
Course Website	http://cse.unsw.edu.au/~cs1917 (http://cse.unsw.edu.au/~cs1917)
Handbook Entry	http://www.handbook.unsw.edu.au/undergraduate/courses/current/COMP1917.html (http://www.handbook.unsw.edu.au/undergraduate/courses/current/COMP1917.html)

Course Summary

The objective of this course is for students to develop proficiency in programming using a high level language. Topics covered include: fundamental programming concepts, program testing and debugging, the underlying memory representation of data, programming style. Practical experience of these topics is supplied by laboratory programming exercises and assignments.

Course Timetable

Lectures:

Monday	10:00	12:00	Burrows Theatre
Tuesday	13:00	14:00	Mathews Theatre A

The complete course timetable is available here (/COMP1917/16s2/timetable) .

Course Aims

The objective of this course is for students to develop proficiency in programming in a high level imperative language and to develop a background of relevant knowledge and skills on which to base further study of computing.

Topics covered include:

- fundamental programming concepts
- program testing and debugging
- the underlying memory representation of data
- programming style

Practical experience of these topics is supplied by laboratory programming exercises and assignments.

Student Learning Outcomes

After completing this course, students should:

- be able to design, construct, document and test C programs that accurately comply with a given specification.
- have an understanding of memory allocation and data storage.

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

Graduate Capability	Acquired in
scholarship: understanding of their discipline in its interdisciplinary context	lectures, assignments
scholarship: capable of independent and collaborative enquiry	tutorial work, lab work, assignments
scholarship: rigorous in their analysis, critique, and reflection	assignments
scholarship: able to apply their knowledge and skills to solving problems	tutorial work, lab work, assignments
scholarship: ethical practitioners	all course-work, by doing it yourself
scholarship: capable of effective communication	tutorial work, assignments
scholarship: information literate	everywhere in CSE
scholarship: digitally literate	everywhere in CSE
leadership: enterprising, innovative and creative	tutorial work, assignments
professionalism: capable of independent, self-directed practice	tutorial work, lab work, assignments
professionalism: capable of operating within an agreed Code of Practice	lab work, assignments
global citizens: culturally aware and capable of respecting diversity and acting in socially just/responsible ways	interaction with your fellow students

Assumed Knowledge

There is no assumed knowledge for this course.

Teaching Rationale

As with playing a musical instrument, computer programming cannot be learnt by simply watching and listening. Hands-on experience emphasised through lab work and assignments is essential in mastering the skills taught in this course.

The course follows a problem-based learning approach.

The course is structured so that each week's content builds on the week before. For this reason, students are advised not to fall behind in their lab work.

Teaching Strategies

- Lectures ... introduce concepts, show examples
- Tutorials ... reinforce concepts and provide additional examples
- Lab Work ... introduce technology required for the assignments
- Assignments .. allow students to solve significant problems

Assessment

The assessable components of the course are:

Component	Mark
Assignments	30%
Lab Exercises	9%
Tutorial Presentation	1%

Prac Exam 1	6%
Prac Exam 2	9%
Prac Exam 3	15%
Written Exam	30%
Total	100%

All submitted programs (Lab work and Assignments) must adhere to the Course Style Guide.

To pass the course, you must achieve:

- at least 15/40 for [Assignments + Labs + Presentation]
- at least 12/30 for the Prac Exams [1,2,3]
- at least 12/30 for the Final Exam (Written)
- at least 50/100 overall

Academic Honesty and Plagiarism

Plagiarism is defined as (https://student.unsw.edu.au/what-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:

- Learning Centre: Plagiarism and Academic Integrity (http://www.lc.unsw.edu.au/plagiarism/index.html)
- MyUNSW: Plagiarism (https://student.unsw.edu.au/plagiarism) and Academic Misconduct (https://www.gs.unsw.edu.au/policy/documents/studentmisconductprocedures.pdf)
- CSE: Addendum to UNSW Plagiarism Guidelines (http://www.cse.unsw.edu.au/~chak/plagiarism/plagiarism.html)
- CSE: Yellow Form (http://www.cse.unsw.edu.au/people/studentoffice/policies/yellowform.html) (whose terms you have agreed to)

Make sure that you read and understand these. Ignorance is not accepted as an excuse for plagiarism.

Course Schedule

The following course schedule is a draft only and is subject to change throughout the semester.

Week	Topics
1	Course Introduction Essential Bash Compiling and Running Data Types Math scanf
2	Conditionals Logical Operators Random Number Generation Assignment 1
3	Loops Functions Defining Constants Debugging 1
4	Arrays Strings Pointers
5	Revision / Catch-up
6	Binary and Hexadecimal Git
7	Structs typedef Dynamic Memory Allocation with Arrays Linked Lists
8	Linked Lists (continued) Debugging 2
9	ADTs Makefiles
10	File I/O
11	Recursion
12	Revision
13	No Lectures

Resources for Students

The recommended textbook for this course is:

- Alistair Moffat, *Programming, Problem Solving, and Abstraction with C,* Pearson Educational, Australia, 2003, ISBN 1-74103-080-3

Other resources:

- http://www.tutorialspoint.com/cprogramming (http://www.tutorialspoint.com/cprogramming)
- http://www.cprogramming.com/tutorial/c-tutorial.ht... (http://www.cprogramming.com/tutorial/c-tutorial.html)

Need Help?

1. Check the comments section on the topic related to your question (eg if you have a question about Assignment 1, read through the comments at the bottom of the Assignment 1 page).
2. Ask your question in the comments at the bottom of the related page.
3. Ask one of your COMP1917 peers.
4. Ask your tutor.

Course Evaluation and Development

This course is based on one which was created in response to student feedback from the course COMP1911 in 2006/07. When it ran for the first time in Session 1, 2008, students expressed satisfaction with the course, but felt that the workload was a bit too heavy. We have therefore made changes to the course, and we believe that the workload is now appropriate. In response to feedback from 2008 and Session 1, 2009, the 3-hour written exam has been replaced by a 2-hour practical exam plus a 2-hour written exam.

Student feedback for this course will be obtained via electronic survey at the end of session, and will be used to make continual improvements to the course. Students are also encouraged to provide informal feedback during the session, and to let the lecturer in charge know of any problems, as soon as they arise. Suggestions will be received constructively, and every reasonable effort will be made to address them.

Resource created 5 years ago (Wednesday 13 July 2016, 04:02:32 PM), last modified 5 years ago (Monday 17 October 2016, 10:19:35 AM).

Comments

Error getting response from server. Try refreshing the page.

There are no comments yet.