Computer Programming: Skills & Concepts (CP1)
Course Guide 2010-11

Mary Cryan

September 2010

Contents

NOTE: Large print versions of all course documents are available on request.

1 Course description

This course is intended for students who wish to develop their programming skills. It can be taken by students with little or no prior experience of programming; however, most students will have some previous computing experience.

The material covers an introduction to the basic components of a modern computer system, the working environment provided by such a system including some simple utilities provided by the operating system (*Linux*), the principles of simple program design, construction and testing and their exploitation in the context of the simpler features of the ANSI C programming language. A brief syllabus follows.

- Introduction. The scope of Computer Science. Elements of a modern computer system and computing environment. UNIX, its file system and programming utilities.
- *Imperative* programming - how it effects change-of-state on the computer.
- Program design and development. Specification, problem decomposition. Reasoning about and testing programs.
- Programming in ANSI C. Expressions, types, variables, assignment, conditionals, iteration, arrays, strings, files, functions.
- Pointers and dynamic memory allocation.
- Structured programming. Functional and procedural abstraction, headers and libraries, names and scope.
- Small introduction to Object-oriented Programming (C++)

Details of the course can be found on-line[1].

Note: A large part of the course focuses on the C programming language; almost all the practical work involves programming in C.

2 The Learning Experience

University courses generally provide students with a range of materials and facilities to encourage and enhance learning. Perhaps the most important difference, when compared to your previous educational experiences, is the extent to which you must take responsibility for your own study activities. Here are the main learning activities, ranging from the most structured to the most personal:

- lectures
- tutorials
- practical exercises
- private study and background reading

[1] http://www.inf.ed.ac.uk/teaching/courses/cp1/

As your University career progresses you will acquire a variety of study skills, of which one of the most important is *time management*. Try to divide your time sensibly and productively between the various activities discussed below. Schedule your practical work realistically, allowing time for understanding and thought as well as work at the machines. Get into the habit of referring to textbooks and notes when you are in difficulty.

Although lecturers and their styles vary greatly, it remains true that the *lecture* is, for many students, probably not the most active learning environment. Most lectures in CP1 will be supported either by a compact version of the lecture slides, or sometimes by extra summarised lecture notes. We will also allocate sections of the textbook for personal reading on a lecture-by-lecture basis. This is done so that you do not spend 50 minutes doing thoughtless, mechanical note taking. However, you will probably find it helpful to highlight and annotate the handouts.

Tutorials involve an hour long weekly meeting between a group of around 12 students and a tutor who will be a member of staff or a PhD student. Tutorials provide a more interactive forum for discussion than is possible in a lecture. Tutorial discussion may focus on the current practical exercise, material related to current lectures or indeed any questions you have about programming. In particular, tutorials represent an excellent opportunity to iron out difficulties and misconceptions, so please be prepared to participate and take full advantage.

Attendance at tutorials and lectures is expected (although not compulsory). Your tutor will keep a record of attendance at tutorials, so we are aware of students who are in danger of becoming disengaged from the course.

There will be 4 *practical exercises* assigned during the course. These offer the chance to put the material discussed in lectures and tutorials into practice, and to develop a range of useful skills. Together with examinations, they form an important part of the formal assessment. Typically, a practical exercise will be distributed at a Monday lecture for completion in two or three weeks. You can obtain help and advice regarding your practical work from your tutor, from the `InfBase` helpers, from your lecturers and indeed your colleagues.

Probably the aspect of university study which will be least familiar to you (depending upon your previous experiences) will be the necessity for *private study* and the organisation of your weekly routine to allow for this. For Computer Programming, private study might include reading through and reflecting upon lecture notes, reading material from the recommended texts, experimenting with code given the course lectures, and writing your own small programs.

3 Assessment

The course will be assessed on the basis of the practical work undertaken (25% of the overall mark), and a $1\frac{1}{2}$ hour written examination (75% of the overall mark) in the April/May diet. Those who fail this exam may resit in August.

The rules for passing the course are the following:

- $C \geq 25\%$,
- $E \geq 35\%$, and
- $C * 0.25 + E * 0.75 \geq 40\%$,

where C is your overall percentage mark for coursework and E is your percentage for the exam. Therefore if you do not submit enough of the courseworks, or fail to do sufficiently well in overall coursework, it will be impossible for you to pass the course.

(Note that the CPMT1 students, who belong to our lecture group and do the same courseworks, have a different formula for combining coursework and exam).

Apart from needing the minimal average of 25% in coursework, another serious consideration is that skill in programming is only developed with practice. You should take the opportunity to gain programming experience, and become an independent programmer, by putting your effort into the assigned coursework. As a matter of general policy, if you run into difficulties (of an academic nature) you should alert your tutor as quickly as possible. (In these circumstances, it is often easier for you to approach your tutor than vice versa, as your tutor may be reluctant to raise the matter while other students are present.) If your practical work is seriously affected by illness, you should obtain a medical certificate and ask your *Director of Studies* to apply for either an extension, or consideration of Special Circumstances.

3.1 Plagiarism

You are allowed and even encouraged to discuss practical exercises with your fellow students. Giving hints, explaining ideas, and help debugging is all fine. But you should *never* copy code, neither by copying over a file, nor by downloading something from an internet forum, nor even by manually copying another student's code. See the university Plagarism guidelines[2] online. Plagiarism is taken very seriously and it is often easy to detect.

4 Facilities

4.1 Laboratory

You have access to the labs on Level 5 of Appleton Tower, which are open all hours (you need your student card to get in, for times outside of 9am-5pm Monday-Friday). AT level 5 contains the workstations (DICE machines) and printers which you will use when working on the practical exercises.

There are also various tables and chairs on Level 3 of Appleton Tower. This is space for you to sit and work away from the machines. You can also find spare copies of lecture handouts and practical exercises here.

In recent years the School of Informatics has been running a support system known as `InfBase` at certain times in Appleton Tower 5.07. `InfBase` consists of a group of PhD students who offer help on a variety of Informatics topics. Depending on who exactly is in AT 5.07 when you visit, they may be able to offer help on C programming. `InfBase` operates from 4pm-6pm on Monday, Tuesday and Thursday, and from 10am-1pm on Friday.

4.2 Computing Regulations and Harassment

The University's computing regulations[3] can be read online. A past version of these regulations probably best summarizes the main issue in regard to acceptable use of university systems:

> *criminal law or which is civilly actionable is the holding or distribution of any material which is defamatory, discriminatory, obscene or otherwise illegal or is offensive or calculated to make others fearful, anxious or apprehensive.*

[2] http://www.inf.ed.ac.uk/admin/ITO/DivisionalGuidelinesPlagiarism.html
[3] http://www.ucs.ed.ac.uk/EUCS/regs.html

You should also read the University's Code of Practice on Personal Harassment[4].

If you consider yourself to be a victim of harassment you should raise the matter promptly with your Director of Studies.

5 Communication Mechanisms

5.1 Information

There are a variety of ways in which information and views relating to the course can be communicated between staff and students, some formalised, others less so. Short announcements concerning matters of organisation will sometimes be made at the start of lectures. Crucial announcements will be posted onto the *course webpage* or emailed to the course mailing list.

You also have your own email account for corresponding among yourselves, or with your tutor, lecturers and course organiser. Please be aware of computing regulations for using email appropriately.

5.2 Problems

There are a number of ways to report and seek help with problems relating to the course and/or your work. Equipment which you think is faulty should be reported to support via the online SUPPORT contact form[5].

Conceptual problems regarding the course material and practical exercises should be discussed with your tutor or with the lecturer concerned.

5.3 Changing Things

Obviously no course is perfect. There are two more formal mechanisms through which problems can be raised and suggestions made.

Early in the course you will be asked to volunteer to act as a *class representative*. If more than a small number of students volunteer, then there will be an election to reduce the number appropriately. Class reps. act both as intermediaries in forwarding problems which students may be unwilling to express directly, and as barometers of class opinion. At the end of the semester the class reps will be invited to meet with the course lecturers to discuss the course.

Secondly, at the end of each course, all students have the opportunity to express their opinions directly in a questionnaire.

Note that both these mechanisms help to improve the course for *next year's students*. If you want help for this year, make your feelings known at an earlier date.

6 Staff

CP1 will be lectured by Mary Cryan and Julian Bradfield. They can be contacted in person after the lecture, by email, or (in important cases) by phone in their offices in the Informatics Forum.

[4] http://www.equalops.ed.ac.uk/harass/stud.htm

[5] https://www.inf.ed.ac.uk/systems/support/form/

Name	e-mail	phone	room
Mary Cryan (course organiser)	cp1co@inf.ed.ac.uk	505153	IF 5.16
Julian Bradfield	jcb@inf.ed.ac.uk	505998	IF 4.07

Administration of the course is done via the Informatics Teaching Organisation (ITO), on Level 4 of Appleton Tower. The best way to contact the ITO is using the online ITO contact form[6]

Questions of a general administrative nature (Which is my tutorial group? When do I get my exam back? I cannot attend the exam!) should be addressed to the ITO.

7 Detailed Course Description

7.1 Lecture Times and locations

Lectures are held at the following times/locations each week from weeks 1-10:

- 2 p.m. on Mondays in FH D2 (top floor of Forest Hill building);
- 11:10am on Tuesdays in AT 2.12 (in Appleton Tower);
- 11:10am on Thursdays in AT 2.12 (in Appleton Tower).

8 Electronic Mail and the Student Mail System

Edinburgh University maintains a *Student Mail System* or SMS to which all students of the University are accredited. This is your main email facility within the university. The School of Informatics may set up local mail for you on the Informatics system, but email sent there is automatically forwarded to your SMS account. You may also have other electronic mail accounts within your own department. However, *it is important that you read all your mail regularly, otherwise you risk missing important announcements relating to coursework.*

8.1 Forwarding mail from an SMS account

For convenience you may to want to read all your mail elsewhere than from the Student Mail System, for example on an email account you have set up elsewhere. Advice on how to set up forwarding is available at the following link:

`http://www.ucs.ed.ac.uk/fmd/unix/docs/mail/forwarding.html`

9 Recommended Reading

The recommended textbook for CP1 is *A Book on C,* by Al Kelley and Ira Pohl, 4^{th} edition (£33-34 in various places). This book provides a tutorial reference on ANSI C, and covers all the programming constructs included in the course. The lecture notes issued do *not* cover all the material contained in the textbook. Therefore, if you can afford to buy the book you should do so.

[6] http://www.inf.ed.ac.uk/teaching/contact/

For those of you who cannot or do not want to buy the textbook, copies have been placed on reserve in the main library. Alternatively, if you have access to a different C-programming text which is free/cheaper, that will probably be sufficient - though be careful that you choose one which is accessible enough for a beginner.

The other books on the list are recommended for consultation. Please note that the course uses the ANSI standard version of the C programming language. Be wary of second hand books which cover earlier, *non-ANSI* versions.

Books on C

- A. Kelley & I. Pohl, *A Book on C*, fourth edition, Benjamin Cummings, 1998. (The course text.)
- A. Kelley & I. Pohl, *C by Dissection*, second edition, Benjamin Cummings, 1992. (Fairly large overlap with *A Book on C*. It covers all the programming constructs needed for CP1, and is specifically designed for readers with no programming background.)
- B.W. Kernighan and D.M. Ritchie, *The C Programming Language*, second edition, Prentice-Hall, 1988. (The definitive C textbook, but terse. Beware - the first edition is non-ANSI C.)
  Not really for beginners - maybe good to read *after* you have finished CP1!
- K.A. Barclay, *ANSI C - Problem Solving and Programming*, Prentice-Hall, 1990. (Another good introduction to C.)

General Background on Computer Science

If you want to learn more about the subject of Computer Science you can look into one or more of the following books. They go mostly beyond the material taught in CP1.

- P. Greenfield, *Introduction to Computing*, McGraw-Hill, 1992. (A gentle introduction to the vocabulary and concepts of the subject, especially useful for those with little or no background.)
- A. V. Aho & J. D. Ullman, *Foundations of Computer Science, C Edition*, Freeman and Company, 1995. (Good reference for topics in Computer Science.)
- D. Harel, *Algorithmics - The Spirit of Computing*, Addison-Wesley, 1987. (A readable romp through the fundamental results which underpin the subject and answers to some of the questions about what computers can and can't do.)

Mary Cryan, September 2010.