School of Mathematical & Computer Sciences
Dept of Computer Science

BSc Computer Science
MEng Software Engineering

Programme Handbook
2011 – 2012

Table of Contents

Introduction 4
Programme Structure 4
How To Use This Catalogue 4
Terminology 5
Timetable 5
Session Dates 6
Mentor 6
Enrolment for Courses 7
Course Requirements 7
Plagiarism & Cheating 7
Submission of Coursework 8
Examinations 8
Grades & Assessments 9
Assessment Results and Progress Decisions 9
Final Degree Assessment 9
Graduation 10
Notification of Special Circumstances 10
University Prizes 10
Miscellaneous 11
Departmental Contacts 12
Non Departmental Contacts 13
BSc Computer Science Programme Structure 14
MEng Software Engineering Programme Structure 22
Course Descriptors 31
Year 1, Semester 1 31
Software Development 1 32
Praxis 33
Interactive Systems 34
Logic & Proof 35
Year 1, Semester 2 36
Software Development 2 37
Introduction to Computer Systems 38
Web Design and Databases 39
Software Development 3 40
Year 2, Semester 1 41
Interaction Design 42
Data Structures & Algorithms 43
Internet & Communications 44
Foundation Maths A 45
Interactive Systems 46
Year 2, Semester 2 47
Software Design 48
Database Management Systems 50
Programming Languages 51
Foundation Mathematics B 52
Discrete Mathematics 53

Year 3, Semester 1 54
Software Engineering 55
Artificial Intelligence 56
Computer Graphics 57
Foundations 1 58
Year 3, Semester 2 59
Professional Development 60
Operating Systems & Concurrency 61
Formal Specification 62
Foundations 2 63
Year 4, Semester 1 64
Research Methods & Requirements Engineering 65
Computing in the Classroom 66
Computer Network & Security 68
Data Mining & Machine Learning 70
Distributed Systems Programming 71
Information Systems Methodologies 72
3D Modelling and Animation 73
Mobile Communications & Programming 74
Robotics & Automation 75
Rigorous Methods for Software Engineering 76
Web Intelligence 77
Year 4, Semester 2 78
Project: Design & Implementation 79
Project: Testing & Presentation 79
Advanced Interaction Design 80
Biologically Inspired Computation 81
Distributed & Parallel Technologies 82
E-Commerce Technology 83
Computer Games Programming 85
Internet Engineering 86
Network Applications 87
Software Simulation & Modelling 88
Virtual Environments 89
Software Engineering 90
Year 4, Semester 1 90
New Product & Process Development 91
Year 5, Semester 1 93
Industrial Placement 1 94
Industrial Placement 2 94
Industrial Placement Monthly Reports 95
Industrial Placement Final Report 95
Year 5, Semester 2 96
Design & Code Group Project 97
Software Engineering Master Class 97

Introduction

This programme specific handbook should be read in conjunction with the Undergraduate Course Handbook for the School of Mathematical and Computer Sciences (MACS), which can be found on the School website http://www.macs.hw.ac.uk/home

This handbook contains information on the programme structure, notes, description and the courses offered on the Computer Science and Software Engineering courses. The first three years of both degrees follow the same structure. Progression onto the MEng at the start of year 4 is by invitation.

Further information for current undergraduate students can be found at:
http://www.macs.hw.ac.uk/macshome/csugstudents.htm
http://www.hw.ac.uk/registry/
http://www.hw.ac.uk/new-students.htm

Programme Structure

Our academic year is divided into 2 semesters corresponding to 30 weeks. There will be 12 weeks teaching in each semester. You are expected to study 4 courses each semester, giving a total of 8 courses in a full year. Each course is worth 15 credits. Courses may be mandatory or optional.

Mandatory courses: These courses are compulsory

Optional courses: Students are required to choose from a specified list of courses relevant to the subject area of their degree discipline.

All undergraduate courses are designed to be of equal length in terms of student effort. The average student is expected to put in a total effort of 150 hours per course. These 150 hours includes all lectures, tutorials, computing labs, workshops, background reading, writing up notes, coursework, revision and examinations for the course.

How To Use This Catalogue

The course information, which appears in the format below, is designed to provide you with sufficient details about courses, their content and assessment methods and will help you choose your optional courses.

Course Code:	**Course Title:**	**Course Co-ordinator:**
Pre-requisites:		
Aims:		
Syllabus:		
Learning Outcomes: Subject Mastery	*Understanding, Knowledge and Subject-Specific Skills*	
Learning Outcomes: Personal Abilities	*Cognitive skills, Core Skills and Professional Awareness*	
Assessment Methods:	Assessment:	Re-assessment:

Many of the courses have on-line material available at the University's Virtual Learning Environment (VISION) which can be found at:
http://vision.hw.ac.uk/

Terminology

Course Code	The first character identifies the School (F = MACS) The second digit identifies the discipline area (2=Computer Science). The next digit is the SCQF level of the course: SCQF Level 7 normally studied in Year 1 SCQF Level 8 normally studied in Year 2 SCQF Level 9 normally studied in Year 3 SCQF Level 10 normally studied in Year 4 (A zero in course codes) SCQF Level 11 normally studied in Year 5/Postgraduate (A one in course codes) The next 2 letters identify the topic. Courses coded B are delivered by the School of Engineering & Physical Sciences
Course Co-ordinator:	The name of the member of staff who is responsible for delivery of the course
Pre-requisites:	Students must have gained Grade D or above in the courses listed here in order to gain entry to the course.
Aims:	A brief statement of what the course aims to do
Syllabus:	A brief summary of what is included in the course
Learning Outcomes: Subject Mastery	These will include Understanding, Knowledge and Cognitive Skills; Scholarship, Enquiry and Research (Research-Informed Learning)
Learning Outcomes: Personal Abilities	Industrial, Commercial & Professional Practice; Autonomy, Accountability & Working with Others; Communication, Numeracy & ICT
Assessment Methods:	Details of the weighting and type of assessment(s) and re-assessment (if any) for the course

Timetable

A timetable of classes will be available online at:
http://www.macs.hw.ac.uk/timetable/

Any timetable problems should be notified to Jill Gunn (Room EM1.17)

Session Dates

Activity	Dates
Semester 1 Teaching (12 Weeks)	12 September - 2 December 2011
Semester 1 Assessment (2 weeks)	5 - 16 December 2011
Break 1 (3 weeks)	19 December 2011 - 6 January 2012
Semester Teaching (12 weeks)	9 January - 30 March 2012
Break 2 (3 weeks)	2 - 19 April 2012
Semester 2 Assessment (4 weeks)	23 April - 18 May 2012
Graduations (4 days)	19 - 22 June 2012
Re-Assessments (7 working days)	2- 10 August 2012
Graduations (2 days)	15 & 16 November 2012

Mentor

You will be allocated a mentor when you arrive at the University and, normally, you will retain the same mentor as long as you are registered in the Department of Computer Science. The mentor is your main academic link with the University, and will help you choose courses and register for courses at the beginning of the session. Under certain circumstances, with the permission of the Head of Computer Science, it may be possible to change your mentor. A list of mentors can be found at http://www.macs.hw.ac.uk/macshome/csugstudents.htm

Regular Meetings

It is important that you see your mentor regularly. These meetings are particularly important for monitoring academic progress in the first, second and third years. First, second and third year students must see their mentors at the start of each semester. At the start of semester 2 mentors will release your assessment grades for the previous semester. Marks from semester 1 assessment will remain provisional until ratification by the Progression Board of Examiners following semester 2 assessment in May.

Mentors often arrange meetings via e-mail, or post notices on their office doors. It is your responsibility to find out what arrangements have been made. In particular, **you are expected to check your e-mail regularly** (at least once a week) **and to ensure that your in-box is regularly cleared**.

Help & Advice

Every year a few students run into personal difficulties (e.g. family illness, accommodation, financial, etc.). As well as being generally supportive, mentors can help in a number of practical ways. For example, if illness prevents you from completing project work or sitting examinations, your mentor can sometimes help with re-scheduling or making alternative arrangements for assessment. However, you *must* notify your mentor as soon as possible, or there is very little that can be done. This is particularly important if illness affects your Examinations. Also, it is essential to provide a medical certificate (see **Notification of Special Circumstances**, p 10). With other problems, your mentor can put you in touch with the appropriate University support service (Chaplaincy, Medical Centre, Student Welfare Services or Student Association). ***Mentors are there to help; do not hesitate to contact yours if you need help.***

Staff-Student Committee

The staff-student committee provides an additional channel of communication between staff and students within the Computer Science Department. It consists of members of academic and support staff and student representatives from each of the four undergraduate years and representatives from our MSc courses. Student representatives are elected annually.

The committee meets at least once each semester. One of its major functions is to consider any concerns about current lecture courses, including teaching quality, and to take appropriate action for their resolution. Other matters of interest, such as the provision of computing facilities or the timing of lectures, may be discussed.

Further information can be obtained from: Dr Sandy Louchart (S.Louchart@hw.ac.uk).

Enrolment for Courses

You must be enrolled for the courses which you are studying. This will be done initially during on-line enrolment.

Any subsequent changes to optional choices must be agreed between you and your Director of Studies or mentor, and then recorded on a Change of Course Form available from MACS School Office (EM1.25). The form must then be returned to Room EM1.25 for processing.

All course changes must be made by the end of week 3 of each semester. Any changes submitted after this will incur a charge of £10 per course. No changes can take place after week 5 of each semester.

Course Requirements

Attendance

In order to satisfy the course requirements, a satisfactory record of attendance at lectures and tutorials is required. Coursework must be handed in by the stipulated dates, and students are required to see their personal mentors at agreed times.

All lectures and tutorials are compulsory and registers of attendance will be taken in most cases. Students who, in the opinion of the Head of Computer Science, fail to satisfy the attendance requirements of any of the courses for which they are registered may, after due warning, be disallowed from presenting themselves for examination in those courses. In this case they will be deemed to have failed those examinations. Students who fail to submit compulsory coursework may also be disallowed from presenting themselves for examination in the relevant courses.

If you are absent from class due to illness for four days or less, you should complete a self-certification form, obtainable from the MACS School Office (EM1.25), and return it to the School Office within a week of your return. If you are absent for more than four days, you must supply a medical certificate within a week of your return.

Plagiarism & Cheating

Cheating in examination and plagiarism, which is, the presentation of another person's ideas or work as one's own, are very serious offences and are dealt with severely. They carry a range of penalties up to and including expulsion from the University.

Students are responsible for familiarizing themselves with University policy on these matters. For more detail, see the MACS Student Guide, and the sections on Plagiarism and Regulation 9 on the Registry's website.

Submission of Coursework

All courses will include some coursework which must be done during the semester. Coursework Submission front sheets are available in the Earl Mountbatten Building **outside** the School Office (Room 1.25). The coursework submission front sheets are printed on **lilac** coloured paper. The **CS/IS coursework box** is located beside the coursework submission sheets.

Please ensure that you:

1.	state which **degree programme** you are studying and **year of study**
2.	Complete your **personal details** on the form, i.e., your name, matriculation number.
3.	Write the **course code** and **course title** on the front sheet.
4.	**Sign** and **date** the front sheet to confirm that it is your "sole and original work ……."
5.	**Staple** the front sheet to your coursework **before** you put it in the CS/IT coursework box.

All coursework must be submitted by 3.30pm on the deadline date unless otherwise specified by the lecturer. A list of coursework deadlines will normally appear on the undergraduate student website from week 3 each semester, which will also detail the amount of effort that is expected for each piece of coursework. Penalties may be imposed for late submission of coursework.

Examinations

It is the student's responsibility to check all relevant examination timetables (including resits) on the Registry website. Therefore, **do not book holidays or take on any other commitments during the assessment diet**. Should you be required to resit any exams, you must be available to take them. Note you can apply to take resits at an overseas exam centre.

Only three types of calculator are allowed in examinations: Casio fx-85WA, Casio fx-85MS and Casio fx-85ES. Students must provide their own calculators (available from the campus shop). Students are not allowed to have mobile phones or other communication devices on or about their person during examinations. Phones may be left at the front of the examination room but must be switched off.

Information on examinations, including timetables, re-assessment procedures etc can be found on the Academic Registry website at:
http://www.hw.ac.uk/registry/examinations.htm

Past exam papers for F2 courses can be found at:
http://www.macs.hw.ac.uk/cs/localinfo/pastpapers/index.htm

THESE ARE ONLY ACCESSIBLE ON-CAMPUS OR IF YOU USE THE VPN (http://vpn1.hw.ac.uk)

Grades & Assessments

Grades for each course are awarded as follows:

Grade A	Excellent	Overall mark of approximately 70% or more
Grade B	Very Good	Overall mark of approximately 60% to 69%
Grade C	Good	Overall mark of approximately 50% to 59%
Grade D	Satisfactory	Overall mark of approximately 40% to 49%
Grade E	Adequate	Minimum required for the award of credits but at least a grade D is needed for progression to subsequent courses
Grade F	Inadequate	Fail

Assessment Results and Progress Decisions

Assessment results are issued via your mentor following the relevant Assessment Boards which take place in January, May and August (resit diet).

The Progression Board meets at the end of the academic year to decide which students will be allowed to proceed to the next year of their degree programme. You will receive a letter from the University containing a summary of your results for the year and the Board's progression decision, and whether you must resit any courses.

In years 1, 2 and 3 if you do not pass a course at the first attempt, you have one opportunity to resit the course during the resit diet in early August. In Year 3, re-assessment is for credit only and you cannot improve your overall average (which accounts for 20% of your final degree results) unless you are re-sitting for medical reasons. There are no re-sit opportunities for courses in Years 4 and 5.

If you receive a pass/proceed decision that allows you to progress at the Summer Progression Board you can enrol online from mid August. If you have resits, and are able to progress following the Resit Progression Board you may enrol online once you have received your Assessment Results letter confirming this.

Final Degree Assessment

The Award Board meets in the last week of May to consider the assessment marks and make recommendations on degree classifications.

For the BSc Computer Science honours degree, the Examiners take into account 3rd and 4th year course marks in deciding the class of Honours. The final mark is the average of those marks, weighted as: 20% from 3rd year average, 50% from the 5 taught courses in 4th year and 30% from the individual dissertation in 4th year. In broad terms, an average mark of over 70% for first class honours, 60% - 70% for upper second class honours, 50% - 60% for lower second class honours, and 40% - 50% for third class honours, would be required,

subject to the agreement of the Examiners. (Note that 480 credits are required for the award of an honours degree.).

For the MEng Software Engineering, the Examiners take into account 3rd, 4th & 5th year course marks in deciding the final classification. The final mark is the average of those marks, weighted as: 10% from 3rd year average, 25% from the 5 taught courses in 4th year, 25% from the individual dissertation in 4th year and 40% from the 8 courses in 5th year. A MEng student may select to exit on successful completion of Stage 4 with a BSc in Computer Science (with honours). A student gaining an overall average of 70% or above may be considered for the award of MEng with Distinction by the Award Board. (Note that 600 credits are required for the award of an MEng degree.).

Graduation

When you have completed your degree your award is conferred at a graduation ceremony. Details on graduation, including how to apply, deadlines for applying and the cost, can be found at: http://www.hw.ac.uk/registry/graduation.htm

This website also includes details of gown hire and guest tickets.

Notification of Special Circumstances

It is **very important** that you notify your mentor **as soon as possible** of any special circumstances, such as illness or bereavement, which could adversely affect your assessment performance. In the case of illness, a medical certificate must be supplied to the School Office (EM1.25). The Examiners will always take such circumstances into account where appropriate, but the later the notification, the less scope there is to do so.

In particular, notification should be before the examination diet concerned, and certainly no later than the Examiners Meeting. Late notification will mean that either no account can be taken, or that formal procedures have to be invoked. In the latter case, final year students will not be permitted to graduate until these procedures have been completed. For further details, see the University Regulations and the Undergraduate Course Handbook.

University Prizes

Final Year Awards

Watt Club Medal

Awarded for exceptional merit and distinction in the **final year** of any degree course in the Department of Computer Science. No more than one medal can be awarded in each discipline within a School in any year.

Systems Consultants Ltd Prize (£200)

The best student in the final year of the course for the degree of BSc in Computer Science.

Cooper-Walker Engineering Ltd Prize (£200)

For outstanding project work in a degree course in the Department of Computer Science.

Andrew Stewart Prize 1 (£200)

For the most deserving student in the **fourth year** of a degree course in the Department of Computer Science.

Continuing Years Awards

University Prizes, Years 1, 2 & 3 (£100)
For outstanding merit (In practice an average mark of at least 70% is regarded as the minimum standard). Available to students on any undergraduate course in the Department of Computer Science.

Andrew Stewart Prize 2 (£200)
For the most deserving student in the **second year** of a degree course in the Department of Computer Science.

ICL Prize (£200)
The best student in the **first year** of the course for the degree of BSc in Computer Science.

Atos Origin Prize
For the best Group Project in the **third year** of a degree course. Each member of the winning group will be awarded £80.00.

Scott Logic Prize (£600)
Awarded to the best students in third year of the course for the degree of BSc in Computer Science.
Best Student (£200) and 2nd, 3rd, 4th and 5th Students (£100).

British Computer Society Prize (£150)
Awarded to the best student in the **final year** of the MEng Software Engineering.

Miscellaneous

Lockers
Lockers for use by students are available at a number of sites in the Earl Mountbatten Building. They are allocated for the duration of each academic year on a first-come first-served basis. Keys for lockers in the EM Building are available for a deposit of £10. Please see Alistair Houstin in room EM 1.31.

Mail
Mail (internal and external) to students is delivered to pigeon holes on the first floor of the Earl Mountbatten Building, outside the School Office (EM1.25). Check yours regularly.

Noticeboard
Various notices are posted on the noticeboard in the corridor along from the School Office.

Departmental Contacts

To direct dial a member of staff: (0131) 451 plus extension number

Head of School	Philippe De Wilde	P.De_Wilde@hw.ac.uk	Ext 8306
Head of Computer Science	Nick Taylor	N.K.Taylor@hw.ac.uk	Ext 3436
Director, Undergraduate Study	Andrew Ireland	A.Ireland@hw.ac.uk	Ext 3409
Director of Studies, Year 1	Brian Palmer	B.V.W.Palmer@hw.ac.uk	Ext 3772
Director of Studies, Year 2	Jenny Coady	J.Coady@hw.ac.uk	Ext 4178
Director of Studies, Year 3	Monica Farrow	M.Farrow@hw.ac.uk	Ext 4160
Director of Studies, Year 4	Peter King	P.J.B.King@hw.ac.uk	Ext 3433
Director of Studies, Year 5	Peter King	P.J.B.King@hw.ac.uk	Ext 3433
MEng Industrial Placements	Phil Trinder	P.W.Trinder@hw.ac.uk	Ext 3435
Special Needs Advisor	Pierluigi Frisco	P.Frisco@hw.ac.uk	Ext 8241
Administrator	Lorna Morrow	L.H.Morrow@hw.ac.uk	Ext 3223

Lecturers	E-Mail	Room	Extension
Prof Ruth Aylett	R.Aylett@hw.ac.uk	EM1.37	4189
Dr Albert Burger	A.G.Burger@hw.ac.uk	EMG.36	3428
Prof Mike Chantler	M.J.Chantler@hw.ac.uk	EM1.48	3352
Ms Jenny Coady	J.Coady@hw.ac.uk	EMG.37	4178
Prof David Corne	D.W.Corne@hw.ac.uk	EMG.39	3410
Prof Philippe De Wilde	P.De_Wilde@hw.ac.uk	CMF.07	
Ms Monica Farrow	M.Farrow@hw.ac.uk	EMG.30	4160
Dr Pierluigi Frisco	P.Frisco@hw.ac.uk	EMG.35	8241
Dr Jamie Gabbay	M.Gabbay@hw.ac.uk	EMG.50	3425
Dr Lilia Georgieva	L.Georgieva@hw.ac.uk	EMG.54	8159
Dr Helen Hastie	H.Hastie@hw.ac.uk	EM 1.42	3344
Prof Andrew Ireland	A.Ireland@hw.ac.uk	EMG.57	3409
Prof Fairouz Kamareddine	F.D.Kamareddine@hw.ac.uk	EM1.65	3868
Dr Peter King	P.J.B.King@hw.ac.uk	EMG.51	3433
Prof Oliver Lemon	O.Lemon@hw.ac.uk	EM1.44	3782
Dr Hans-Wolfgang Loidl	H.W.Loidl@hw.ac.uk	EMG.48	3421
Dr Sandy Louchart	S.Louchart@hw.ac.uk	EM1.38	3424
Prof Greg Michaelson	G.J.Michaleson@hw.ac.uk	EMG.56	3422
Mr Brian Palmer	B.V.W.Palmer@hw.ac.uk	EMG.31	3772
Dr Roger Rist	R.J.Rist@hw.ac.uk	CMG.10	3287
Dr Judy Robertson	Judy.Robertson@hw.ac.uk	EM1.59	8223
Dr Hamish Taylor	H.Taylor@hw.ac.uk	EM1.43	3427
Prof Nick Taylor	N.K.Taylor@hw.ac.uk	EM1.62	3436
Prof Phil Trinder	P.W.Trinder@hw.ac.uk	EMG.52	3435
Dr Patricia Vargas	P.A.Vargas@hw.ac.uk	EMG.28	4161
Dr Joe Wells	J.B.Wells@hw.ac.uk	EM1.61	3869

Non Departmental Contacts

Lecturer	Email	Room	Extension	Course
School of Engineering & Physical Sciences (B courses)				
Dr Keith Brown	K.E.Brown@hw.ac.uk	EM2.42	3351	B81NP
Department of Mathematics (F1 courses)				
Dr Mark Lawson	M.V.Lawson@hw.ac.uk	CMS.21	3210	F17LP
Professor Jim Howie	J.Howie@hw.ac.uk	CMT.10	3240	F17SC
Professor Jack Carr	J.Carr@hw.ac.uk	CMG.03	3229	F17SP
Dr Alan Prince	A.R.Price@hw.ac.uk	CMT.14	3232	F17SQ

In the first instance all undergraduate enquiries should be directed to the School Office, room EM 1.25

BSc Computer Science Programme Structure 2011/2012

Programme Code	Programme Title	School	Type	Awards
F291-COS *F2P1-CSA* *F2C1-CGP* *F2J1-CSS*	**Computer Science** *Computer Science (Artificial Intelligence)* *Computer Science (Computer Games Programming)* *Computer Science (Software Engineering)*	Mathematical & Computer Sciences		BSc (Hons), BSc (Ord), Diploma of Higher Education, Certificate of Higher Education

Programme Accredited by	UCAS Code	QAA Subject Benchmarking Group(s)	Date of Production/Revision
BCS & IEE	G400 & G700/ G440/G600	Computing	March 2011 /2011/12

Stage Composition		Arrangement of Courses: (Themes and Subject Streams)						Awards, Credits & Levels
		Mandatory Courses		Optional Courses		Elective Courses		
		Semester 1	Semester 2	Semester 1	Semester 2	Semester 1	Semester 2	
Stage 1	8 courses: All mandatory	F27SA Software Development 1 F27PX Praxis F27IS Interactive Systems F17LP Logic & Proof	F27SB Software Development 2 F27SG Software Development 3 F27CS Introduction to Computer Systems F27WD Web Design & Databases					**Certificate of Higher Education** Requires 120 SCQF credits at level 7
Stage 2	8 courses: All mandatory	F28IN Interaction Design F28DA Data Structures & Algorithms F28IT Internet & Communications	F28SD Software Design F28DM Database Management Systems F28PL Programming Languages	*Choose 1 of:* F17SP Foundation Maths A F27IS Interactive Systems *(direct entrants only)*	*Choose 1 of:* F17SQ Foundation Maths B F17SC Discrete Maths *(direct entrants only)*			**Diploma of Higher Education** Requires 240 SCQF credits incl. 90 at level 8 or higher

BSc Computer Science Course Structure 2011/2012

Programme Code	Programme Title	School	Type	Awards
F291-COS *F2P1-CSA* *F2C1-CGP* *F2J1-CSS*	**Computer Science** *Computer Science (Artificial Intelligence)* *Computer Science (Computer Games Programming)* *Computer Science (Software Engineering)*	Mathematical & Computer Sciences		BSc (Hons), BSc (Ord), Diploma of Higher Education, Certificate of Higher Education

Programme Accredited by	UCAS Code	QAA Subject Benchmarking Group(s)	Date of Production/Revision
BCS & IEE	G400 & G700/ G440/G600	Computing	March 2011 /2011/12

Stage Composition	Arrangement of Courses: (Themes and Subject Streams)			Awards, Credits & Levels
	Mandatory courses	Optional Courses	Elective Courses	

		Semester 1	Semester 2	Semester 1	Semester 2	Semester 1	Semester 2	Ordinary or General Degree
Stage 3	8 courses: All mandatory	F29SO Software Engineering F29AI Artificial Intelligence & Intelligent Agents F29GR Computer Graphics F29FA Foundations 1	F29PD Professional Development F29OC Operating Systems & Concurrency F29FS Formal Specification F29FB Foundations 2					Requires 360 SCQF credits incl 60 at level 9

BSc Computer Science Programme Structure 2011/2012

Programme Code	Programme Title	School	Type	Awards
F291-COS *F2P1-CSA* *F2C1-CGP* *F2J1-CSS*	**Computer Science** *Computer Science (Artificial Intelligence)* *Computer Science (Computer Games Programming)* *Computer Science (Software Engineering)*	Mathematical & Computer Sciences		BSc (Hons), BSc (Ord), Diploma of Higher Education, Certificate of Higher Education

Programme Accredited by	UCAS Code	QAA Subject Benchmarking Group(s)	Date of Production/Revision
BCS & IEE	G400 & G700/ G440/G600	Computing	March 2011 /2011/12

Stage Composition		Arrangement of Courses: (Themes and Subject Streams)						Awards, Credits & Levels
		Mandatory courses		Optional Courses		Elective Courses		
Stage 4	8 courses: 3 mandatory 5 optional	F20PA Project: Research Methods & Requirements Engineering	F20PB Project: Design & Implementation F20PC Project: Testing & Presentation	*Choose 3 of:* F20CL Computing in the Classroom F21CN Computer Network Security F21DL Data Mining & Machine Learning F21DS Distributed Systems Programming F21IF Information System Methodologies F21MA 3D Modelling and Animation F21MC Mobile Communications & Programming F21RO Robotics & Automation F21RS Rigorous Methods for Software Engineering F21WI Web Intelligence	*Choose 2 of:* F21AD Advanced Interaction Design F21BC Biologically Inspired Computation F21DP Distributed and Parallel Technologies F21EC E-Commerce Technology F21GP Computer Games Programming F21IE Internet Engineering F21NA Network Applications F21SI Software Simulation and Modelling F21VE Virtual Environments			**Honours Degree** Requires 480 SCQF credits including a minimum of 180 at Level 9 &10 including 90 at Level 10

BSc Computer Science Programme Notes 2011/2012

Programme Code	Programme Title	School	Type	Awards
F291-COS *F2P1-CSA* *F2C1-CGP* *F2J1-CSS*	**Computer Science** *Computer Science (Artificial Intelligence)* *Computer Science (Computer Games Programming)* *Computer Science (Software Engineering)*	Mathematical & Computer Sciences		BSc (Hons), BSc (Ord), Diploma of Higher Education, Certificate of Higher Education

Programme Accredited by	UCAS Code	QAA Subject Benchmarking Group(s)	Date of Production/Revision
BCS & IEE	G400 & G700/ G440/G600	Computing	March 2011/2011/12

Stage Notes

Stage Two:

- Candidates will normally take the Foundation Mathematics 1 & 2 courses
- Direct entrants to Stage 2 and internal transfers from other degrees will be expected have an appropriate background in programming and database technology. In addition, direct entrants will take either F17SP Foundation Maths A (for HND candidates who may be weaker in Maths) or F27IS Interactive Systems (for students with A'level Maths or equivalent) in Semester 1 and Discrete Maths in Semester 2.

Stage Three:

- Direct entrants to Stage 3 and internal transfers from other degrees will be expected have appropriate programming experience and background knowledge.
- Candidates shall pursue a group project throughout the year, which shall be synoptically assessed in conjunction with material from the associated courses (F29SO1 and F29PD2)

Stage Four:

- In any one year not all optional courses may be offered. Guidance in course choice will be given by academic mentors.
- Students must apply to take the course Computing in the Classroom (F20CL1)prior to the end of Stage 3 to allow time for placements to be organised
- Candidates are required to undertake an individual dissertation project which shall run throughout the year.

Programme Code	Programme Title	School	Type	Awards
F291-COS *F2P1-CSA* *F2C1-CGP* *F2J1-CSS*	**Computer Science** *Computer Science (Artificial Intelligence)* *Computer Science (Computer Games Programming)* *Computer Science (Software Engineering)*	Mathematical & Computer Sciences		BSc (Hons), BSc (Ord), Diploma of Higher Education, Certificate of Higher Education

Programme Accredited by	UCAS Code	QAA Subject Benchmarking Group(s)	Date of Production/Revision
BCS & IEE	G400 & G700/ G440/G600	Computing	March 2011/2011/12

Progression Requirements

Progression through the course requires a *minimum* of number of credit points:

Stage 1 to Stage 2: 90 credits (6 courses)
Stage 2 to Stage 3: 210 credits (14 courses)
Stage 3 to Stage 4: 330 credits (22 courses) and an overall exam average of 50%

Progression through the course requires a *minimum of Grade D* in the following courses:

Stage 1: 7 courses including Software Development 1 (F27SA), Interactive Systems (F27IS), Logic & Proof (F17LP), Web Design and Databases (F27WD), Introduction to Computer Systems (F27CS), Software Development 2 (F27SB), and Software Development 3 (F27SG).
Stage 2: 6 courses including Interaction Design (F28IN), Internet, Communications & Mobility (F28IM), Data Structures & Algorithms (F28DA) Database Management Systems (F28DM), Software Design (F28SD), Programming Languages (F28PL), (PLUS Discrete Maths (F17SC) for direct entrants into Stage 2).
Stage 3: 6 courses including Software Engineering (F29SO) & Professional Development (F29PD). Re-assessment in Stage 3 is available for credit only and not to improve overall average.

Award Requirements

Honours degree classification is determined by performance in:

- Stage 3, averaged over all 8 courses (20%) at the first attempt.
- the 5 taught courses in Stage 4 (50%)
- the individual dissertation project in Stage 4 (30%)

The degree with honours may be awarded in Computer Science, or in one of three specialisms. To graduate with a specialism, candidates must select a minimum of two taught courses and an approved dissertation topic suitable for the chosen specialism. The specialisms are: *Artificial Intelligence* (students must take 2 of the following courses: F21DL (Data Mining & Machine Learning), F21RO (Robotics & Automation), F21WI (Web Intelligence), F21BC (Biologically Inspired Computation); *Computer Games Programming* (students must take the following 2 courses: F21MA (3D Modelling & Animation) & F21GP (Computer Games Programming) and *Software Engineering* (students must take 2 of the following courses: F21RS (Rigorous Methods for Software Engineering), F21DS (Distributed Systems Programming), F21AD (Advanced Interaction Design)).

BSc Computer Science Programme Description 2011/2012

Programme Code	Programme Title	School	Type	Awards
F291-COS *F2P1-CSA* *F2C1-CGP* *F2J1-CSS*	**Computer Science** *Computer Science (Artificial Intelligence)* *Computer Science (Computer Games Programming)* *Computer Science (Software Engineering)*	Mathematical & Computer Sciences		BSc (hons), BSc (ord), Diploma of Higher Education, Certificate of Higher Education

Programme Accredited by:	UCAS Code	QAA Subject Benchmarking Group(s)	Date of Production/Revision
BCS & IEE	G400 & G700/ G440/G600	Computing	13 March 2009/2011/12

Educational Aims of the Course

The educational aim is to provide students with a theoretical foundation and applied skills in Computer Science in addition to other professional skills which will enable graduates to communicate clearly, work independently and co-operate effectively. The balance of skills will enable graduates to work effectively and efficiently in industry and commerce and prepare them for postgraduate study.

The Course provides opportunities for learners to achieve the following outcomes:

Subject Mastery

Understanding, Knowledge and Cognitive Skills

- To develop knowledge and skills in the elicitation and analysis of user requirements, design and evaluation of solutions, and the implementation and quality assurance of the chosen solution.
- To be able to develop well-structured, efficient, usable and well-documented programs.
- To know what general classes of problems are amenable to computer solution and be able to select the appropriate tools required for particular problems.
- To be able to develop an abstract model for a given problem and devise appropriate mechanized techniques to solve the problem.
- To develop the knowledge and skills required to meet the challenges of emerging technologies and methodologies.

Scholarship, Enquiry and Research

- To gain an in depth understanding of the theoretical foundations of computation and its relevance to everyday computing.
- To be able to design, implement, document, verify and validate relatively large heterogeneous software systems.
- To be able to assess the quality of software systems, both in terms of their functional and non-functional properties.

Programme Code	Programme Title	School	Type	Awards
F291-COS *F2P1-CSA* *F2C1-CGP* *F2J1-CSS*	**Computer Science** *Computer Science (Artificial Intelligence)* *Computer Science (Computer Games Programming)* *Computer Science (Software Engineering)*	Mathematical & Computer Sciences		BSc (hons), BSc (ord), Diploma of Higher Education, Certificate of Higher Education

Programme Accredited by:	UCAS Code	QAA Subject Benchmarking Group(s)	Date of Production/Revision
BCS & IEE	G400 & G700/ G440/G600	Computing	13 March 2009/2011/12

Personal Abilities

Industrial, Commercial and Professional Practice

- To maintain and update technical knowledge; to take responsibility for personal and professional development.
- To appraise the impact of computers on society and the influence of society on the development of the technology and use of computers.
- To assess aspects of the law related to computer-based information, or the role of standards in safety, quality and security, of security issues and of the BCS Codes of Practice and Conduct.

Autonomy, Accountability and Working with Others

- To undertake self-directed work; to assimilate information from multiple sources; to examine results and generate conclusions; to impart ideas effectively in visual, verbal or written form.
- To work effectively either individually or as part of a team.
- To apply subject-mastery outcomes to monitor, analyse, model, specify, design, communicate, implement, evaluate, control and plan.
- To be aware of, and be able to respond to, the social and legal implications and consequences of the use of computers.
- To be able to analyse problem spaces; develop and work with abstractions; appraise material and ideas; to apply a methodical and innovative approach to problem solving; to integrate theory and practice

Communication, Numeracy and ICT

- To be able to communicate with peers, more senior colleagues and specialists. In addition, communicate using appropriate methods to a range of audiences, i.e. specialists and non-specialists.

- To be able to undertake critical evaluation/analysis of a wide range of numerical and graphical data.

BSc Computer Science Programme Description 2011/2012

Programme Code	Programme Title	School	Type	Awards
F291-COS	**Computer Science**	Mathematical & Computer Sciences		BSc (hons), BSc (ord), Diploma of Higher Education, Certificate of Higher Education
F2P1-CSA	*Computer Science (Artificial Intelligence)*			
F2C1-CGP	*Computer Science (Computer Games Programming)*			
F2J1-CSS	*Computer Science (Software Engineering)*			

Programme Accredited by:	UCAS Code	QAA Subject Benchmarking Group(s)	Date of Production/Revision
BCS & IEE	G400 & G700/ G440/G600	Computing	13 March 2009/2011/12

Approaches to Teaching and Learning:

Lectures, Tutorials (practicals, laboratories), Coursework, (assignments, individual projects, group projects, essays, reports, presentations, log/journals, dissertation), Self-study are linked to *lecture-based, resource-based and problem-based* teaching styles, to relate with *motivational, assimilative, consolidative and evaluative* phases of learning.

Approaches to teaching and learning are continually reviewed and developed with the aim of matching them to the abilities and experiences of students, with regard also for the subject area. Specific details about teaching and learning methods are provided in the appropriate course descriptors.

Assessment Policies:

The following assessment methods are used:

Understanding, knowledge and subject specific skills are assessed through the range of methods reflected by written examinations, coursework assignments, software artifacts, group and individual projects, written reports and oral presentations. Diagnostic, formative, continuous and summative types of assessment aim to correlate with methods of assessment.

Approaches to assessment are continually reviewed. Specific details about methods of assessment are provided in the appropriate course descriptors.

MEng Software Engineering Programme Structure

2011/2012

Programme Code	**Programme Title**	**School**	**Type**	**Awards**
F2M1-SOE	**Software Engineering**	Mathematical & Computer Sciences		MEng, BSc (hons), BSc (Ord), Diploma of Higher Education, Certificate of Higher Education

Programme Accredited by	**UCAS Code**	**QAA Subject Benchmarking Group(s)**	**Date of Production/Revision**
BCS & IEE	G601 (5 yrs); G602 (4 yrs)	Computing	14 March 2011/2011/12

Stage Composition	**Arrangement of Courses: (Themes and Subject Streams)**			**Awards, Credits & Levels**
	Mandatory Courses	**Optional Courses**	**Elective Courses**	

		Semester 1	**Semester 2**	**Semester 1**	**Semester 2**	**Semester 1**	**Semester 2**	**Certificate of Higher Education**
Stage 1	8 courses: All mandatory	F27SA Software Development 1 F27PX Praxis F27IS Interactive Systems F17LP Logic & Proof	F27SB Software Development 2 F27SG Software Development 3 F27CS Introduction to Computer Systems F27WD Web Design & Databases					Requires 120 SCQF credits at level 7
Stage 2	8 courses: All mandatory	F28IN Interaction Design F28DA Data Structures & Algorithms F28IT Internet & Communications	F28SD Software Design F28DM Database Management Systems F28PL Programming Languages	*Choose 1 of:* F17SP Foundation Maths A F27IS Interactive Systems *(direct entrants only)*	*Choose 1 of:* F17SQ Foundation Maths B F17SC Discrete Maths *(direct entrants only)*			**Diploma of Higher Education** Requires 240 SCQF credits incl. 90 at level 8 or higher

MEng Software Engineering Programme Structure 2011/2012

Programme Code	Programme Title	School	Type	Awards
F2M1-SOE	**Software Engineering**	Mathematical & Computer Sciences		MEng, BSc (hons), BSc (Ord), Diploma of Higher Education, Certificate of Higher Education

Programme Accredited by	UCAS Code	QAA Subject Benchmarking Group(s)	Date of Production/Revision
BCS & IEE	G601 (5 yrs); G602 (4 yrs)	Computing	14 March 2011/2011/12

Stage Composition		Arrangement of Courses: (Themes and Subject Streams)						Awards, Credits & Levels
		Mandatory Courses		Optional Courses		Elective Courses		
Stage 3	8 courses: All mandatory	F29SO Software Engineering F29AI Artificial Intelligence & Intelligent Agents F29GR Computer Graphics F29FA Foundations 1	F29PD Professional Development F29OC Operating Systems & Concurrency F29FS Formal Specification F29FB Foundations 2					**Ordinary or General Degree** Requires 360 SCQF credits incl 60 at level 9

MEng Software Engineering Programme Structure 2011/2012

Programme Code	Programme Title	School	Type	Awards
F2M1-SOE	**Software Engineering**	Mathematical & Computer Sciences		MEng, BSc (hons), BSc (Ord), Diploma of Higher Education, Certificate of Higher Education

Programme Accredited by	UCAS Code	QAA Subject Benchmarking Group(s)	Date of Production/Revision
BCS & IEE	G601 (5 yrs); G602 (4 yrs)	Computing	14 March 2011/2011/12

Stage Composition		Arrangement of Courses: (Themes and Subject Streams)						Awards, Credits & Levels
		Mandatory Courses		Optional Courses		Elective Courses		
Stage 4	8 courses 5 mandatory 3 optional	F20PA Project: Research Methods & Requirements Engineering B81NP New Product & Process Development	F20PB Project: Design & Implementation F20PC Project: Testing & Presentation	*Choose 2 of:* F20CL Computing in the Classroom F21CN Computer Network Security F21DL Data Mining & Machine Learning F21DS Distributed Systems Programming F21IF Information System Methodologies F21MA 3D Modelling and Animation F21MC Mobile Communications & Programming F21RO Robotics & Automation F21RS Rigorous Methods for Software Engineering F21WI Web Intelligence	*Choose 2 of:* F21AD Advanced Interaction Design F21BC Biologically Inspired Computation F21DP Distributed and Parallel Technologies F21EC E-Commerce Technology F21GP Computer Games Programming F21IE Internet Engineering F21NA Network Applications F21SI Software Simulation and Modelling F21VE Virtual Environments			**Honours Degree** Requires 480 SCQF credits including a minimum of 180 at Level 9 & 10 including 90 at Level 10

MEng Software Engineering Programme Structure 2011/2012

Programme Code	Programme Title	School	Type	Awards
F2M1-SOE	**Software Engineering**	Mathematical & Computer Sciences		MEng, BSc (hons), BSc (Ord), Diploma of Higher Education, Certificate of Higher Education

Programme Accredited by	UCAS Code	QAA Subject Benchmarking Group(s)	Date of Production/Revision
BCS & IEE	G601 (5 yrs); G602 (4 yrs)	Computing	14 March 2011/2011/12

Stage Composition	Arrangement of Courses: (Themes and Subject Streams)						Awards, Credits & Levels
	Mandatory Courses		Optional Courses		Elective Courses		
Stage 5	8 courses 6 mandatory 2 optional	F21IA Industrial Placement 1 F21IB Industrial Placement 2 F21IC Industrial Placement Monthly Reports F21ID Industrial Placement Final Report	F21DG Design & Code Group Project F21SM Software Engineering Master Class		Choose 2 of: F21AD Advanced Interaction Design F21BC Biologically Inspired Computation F21DP Distributed and Parallel Technologies F21EC E-Commerce Technology F21GP Computer Games Programming F21IE Internet Engineering F21NA Network Applications F21VE Virtual Environments F21WI Web Intelligence			**Masters Degree** Requires 600 SCQF credits including a minimum of 120 at Level 11

MEng Software Engineering Programme Notes 2011/2012

Programme Code	Programme Title	School	Type	Awards
F2M1-SOE	**Software Engineering**	Mathematical & Computer Sciences		MEng, BSc (Hons), BSc (Ord), Diploma of Higher Education, Certificate of Higher Education

Programme Accredited by	UCAS Code	QAA Subject Benchmarking Group(s)	Date of Production/Revision
BCS & IEE	G601, G602 (year 2 entry)	Computing	14 March 2011/2011/12

Stage Notes

Stage Two:

- Candidates will normally take the Foundation Mathematics 1 & 2 courses
- Direct entrants to Stage 2 and internal transfers from other degrees will be expected have an appropriate background in programming and database technology. In addition, direct entrants will take either F17SP Foundation Maths A (for HND candidates who may be weaker in Maths) or F27IS Interactive Systems (for students with A'level Maths or equivalent) in Semester 1 and Discrete Maths in Semester 2.

Stage Three:

- Direct entrants to Stage 3 and internal transfers from other degrees will be expected have appropriate programming experience and background knowledge.
- Candidates shall pursue a group project throughout the year, which shall be synoptically assessed in conjunction with material from the associated courses (F29SO and F29PD)

Stage Four:

- In any one year not all optional courses may be offered. Guidance in course choice will be given by academic mentors.
- Students must apply to take the course Computing in the Classroom (F20CL)prior to the end of Stage 3 to allow time for placements to be organised
- Candidates are required to undertake an individual dissertation project which shall run throughout the year.

Stage Five:

- Candidates are required to undertake an industrial placement which starts before the summer and continues through to the end of Semester 1 and which shall be synoptically assessed from the associated courses (F21IA/F21IB/F21IC/F21ID).

MEng Software Engineering Programme Notes 2011/2012

Programme Code	Programme Title	School	Type	Awards
F2M1-SOE	**Software Engineering**	Mathematical & Computer Sciences		MEng, BSc (Hons), BSc (Ord), Diploma of Higher Education, Certificate of Higher Education

Programme Accredited by	UCAS Code	QAA Subject Benchmarking Group(s)	Date of Production/Revision
BCS & IEE	G601, G602 (year 2 entry)	Computing	14 March 2011/2011/12

Progression Requirements

(a) Progression through the programme requires a *minimum* of number of credit points:

Stage 1 to Stage 2: 90 credits (6 courses)
Stage 2 to Stage 3: 210 credits (14 courses)

(b) To progress from Stage 3 to Stage 4 students are expected to achieve a 3rd year average of 60% or above at the first attempt and 330 credits.

(c) To progress from Stage 4 to Stage 5 students are expected to achieve a 4th year average of 60% or above at the first attempt and 450 credits.

(d) Progression through the programme requires a *minimum of Grade D* in the following courses:

Stage 1: 7 courses including Software Development 1 (F27SA), Interactive Systems (F27IS), Logic & Proof (F17LP), Web Design and Databases (F27WD), Introduction to Computer Systems (F27CS), Software Development 2 (F27SB), and Software Development 3 (F27SG).

Stage 2: 6 courses including Interaction Design (F28IN), Internet, Communications & Mobility (F28IM), Data Structures & Algorithms (F28DA) Database Management Systems (F28DM), Software Design (F28SD), Programming Languages (F28PL), (PLUS Discrete Maths (F17SC) for direct entrants into Stage 2).

Stage 3: 6 courses including Software Engineering (F29SO) & Professional Development (F29PD). Re-assessment in Stage 3 is available for credit only and not to improve overall average

Award Requirements

The degree of MEng shall be determined by performance in:

- Stage 3, averaged over all 8 courses, at the first attempt (10%)
- Stage 4, averaged over all 5 taught courses (25%)
- the individual dissertation project in Stage 4 (25%)
- Stage 5, averaged over all 8 courses (40%)

A MEng student may select to exit on successful completion of Stage 4 with a BSc in Computer Science (with honours). A student gaining an overall average of 70% or above may be considered for the award of MEng with Distinction by the Award Board.

MEng Software Engineering Programme Description 2011/2012

Programme Code	Programme Title	School	Type	Awards
F2M1-SOE	**Software Engineering**	Mathematical & Computer Sciences		BSc (hons), BSc (ord), Diploma of Higher Education, Certificate of Higher Education

Programme Accredited by:	UCAS Code	QAA Subject Benchmarking Group(s)	Date of Production/Revision
BCS & IEE	G601/G602	Computing	22 November 2007/2011/12

Educational Aims of the Programme

The educational aim is to provide students with a theoretical foundation and applied skills in Computer Science/Software Engineering in addition to other professional skills which will enable graduates to communicate clearly, work independently and co-operate effectively. The balance of skills will enable graduates to work effectively and efficiently in industry and commerce and prepare them for postgraduate study.

The Programme provides opportunities for learners to achieve the following outcomes:

Subject Mastery

Understanding, Knowledge and Cognitive Skills

- To develop knowledge and skills in the elicitation and analysis of user requirements, design and evaluation of solutions, and the implementation and quality assurance of the chosen solution.
- To be able to develop well-structured, efficient, usable and well-documented programs.
- To know what general classes of problems are amenable to computer solution and be able to select the appropriate tools required for particular problems.
- To be able to develop an abstract model for a given problem and devise appropriate mechanized techniques to solve the problem.
- To develop the knowledge and skills required to meet the challenges of emerging technologies and methodologies.
- To be able to analyse problem spaces; develop and work with abstractions; appraise material and ideas; to apply a methodical and innovative approach to problem solving; to integrate theory and practice.

Scholarship, Enquiry and Research

- To gain an in depth understanding of the theoretical foundations of computation and its relevance to everyday computing.
- To be able to design, implement, document, verify and validate relatively large heterogeneous software systems.
- To be able to assess the quality of software systems, both in terms of their functional and non-functional properties.
- To develop knowledge of the aspects of Management required to understand the commercial and business contexts within which IT systems are used.
- To develop the entrepreneurial skills required to identify and exploit opportunities which arise as a result of technological developments and new business paradigms.
- To acquire and disseminate advanced software engineering knowledge

Programme Code	Programme Title	School	Type	Awards
F2M1-SOE	**Software Engineering**	Mathematical & Computer Sciences		BSc (hons), BSc (ord), Diploma of Higher Education, Certificate of Higher Education

Programme Accredited by:	UCAS Code	QAA Subject Benchmarking Group(s)	Date of Production/Revision
BCS & IEE	G601/G602	Computing	22 November 2007/Version 1

Personal Abilities

Industrial, Commercial and Professional Practice

- To maintain and update technical knowledge; to take responsibility for personal and professional development.
- To appraise the impact of computers on society and the influence of society on the development of the technology and use of computers.
- To assess aspects of the law related to computer-based information, or the role of standards in safety, quality and security, of security issues and of the BCS Codes of Practice and Conduct.

Autonomy, Accountability and Working with Others

- To undertake self-directed work; to assimilate information from multiple sources; to examine results and generate conclusions; to impart ideas effectively in visual, verbal or written form.
- To work effectively either individually or as part of a team.
- To apply subject-mastery outcomes to monitor, analyse, model, specify, design, communicate, implement, evaluate, control and plan.
- To be aware of, and be able to respond to, the social and legal implications and consequences of the use of computers.
- To apply theory to practice in the workplace.
- To deliver advanced training material to peers.

Communication, Numeracy and ICT

- To be able to communicate with peers, more senior colleagues and specialists. In addition, communicate using appropriate methods to a range of audiences, i.e. specialists and non-specialists.
- To be able to undertake critical evaluation/analysis of a wide range of numerical and graphical data.

Programme Code	Programme Title	School	Type	Awards
F2M1-SOE	**Software Engineering**	Mathematical & Computer Sciences		BSc (hons), BSc (ord), Diploma of Higher Education, Certificate of Higher Education

Programme Accredited by:	UCAS Code	QAA Subject Benchmarking Group(s)	Date of Production/Revision
BCS & IEE	G601/G602	Computing	22 November 2007/Version 1

Approaches to Teaching and Learning:

Lectures, Tutorials (practicals, laboratories), Coursework, (assignments, individual projects, group projects, essays, reports, presentations, log/journals, dissertation), Self-study are linked to *lecture-based, resource-based and problem-based* teaching styles, to relate with *motivational, assimilative, consolidative and evaluative* phases of learning.

Approaches to teaching and learning are continually reviewed and developed with the aim of matching them to the abilities and experiences of students, with regard also for the subject area. Specific details about teaching and learning methods are provided in the appropriate course descriptors.

Assessment Policies:

The following assessment methods are used:

Understanding, knowledge and subject specific skills are assessed through the range of methods reflected by *written examinations, coursework assignments, software artefacts, group and individual projects, written reports and oral presentations. Diagnostic, formative, continuous and summative* types of assessment aim to correlate with methods of assessment.

Approaches to assessment are continually reviewed. Specific details about methods of assessment are provided in the appropriate course descriptors.

Computer Science & Software Engineering

Course Descriptors

Year 1, Semester 1

The University reserves the right to withdraw or modify the content of any course

Course Code: F27SA	Course Title: Software Development 1	Course Co-ordinator: Brian Palmer, Greg Michaelson
Pre-requisites:		
Aims:	**To introduce the object-oriented paradigm and the use of an object-oriented language**	
Syllabus:	♦ Objects and classes ♦ Class definitions: fields, constructors, methods, parameters ♦ Selection and iteration ♦ Object interaction: abstraction, modularisation, types ♦ Grouping objects: collection classes, iterators, arrays ♦ Library classes, documentation ♦ Testing and debugging ♦ Designing classes: coupling, cohesion, main method	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ Understanding the object-oriented paradigm ♦ Awareness of the contrast with other programming paradigms ♦ Manipulating objects in an IDE ♦ Understanding and using documentation in an API ♦ Reading, understanding, adapting, creating, and documenting object-oriented code	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ Sharing work with random partners in laboratories (pair programming) ♦ Deriving and creating own solutions to problems (PDP) ♦ Competence in the use of a command-line shell (PDP) ♦ Reading and evaluating code, and modifying it	
Assessment Methods:	Assessment: Exam: (weighting – 50%) (Electronic) Coursework: (weighting – 50%)	Re-assessment: Examination 100%

Course Code: F27PX	Course Title: Praxis	Course Co-ordinator: Brian Palmer
Pre-requisites:		
Aims:	♦ **To instruct students in undertaking self-directed study** ♦ **To instruct students in presenting their findings** ♦ **To acquaint students with the work of the department** ♦ **To deepen students' understanding of the degree courses for which they are registered** ♦ **To familiarise students with the computer systems used by the department**	
Syllabus:	♦ Writing reports; sources and referencing; group presentation; the matter of plagiarism ♦ Personal Development Planning (PDP) ♦ History of information and computing ♦ Current departmental research ♦ Exploration of the departmental computer system	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ Knowledge of relevant historical developments ♦ Understanding of the human issues – moral, economic, social, political – arising from the use of computing technology ♦ Acquaintance with new research in computing ♦ Consideration of difficult and even perplexing ideas in their chosen field of study ♦ Knowledge of and ability to use departmental computer systems	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ Personal development planning (PDP) ♦ Undertaking responsibility for self-directed research (PDP) ♦ Assimilating information from multiple sources (PDP) ♦ Analysing results to formulate conclusions (PDP) ♦ Writing reports to professional standards (PDP) ♦ Constructively evaluating the work of peers (PDP) ♦ Reacting sensibly to peer evaluation (PDP) ♦ Re-writing work in response to criticism (PDP) ♦ Co-operating in a group to investigate a complex topic (PDP) ♦ Making spoken and visual presentations (PDP) ♦ Using a chatboard as a means of learning, contributing and discussing (PDP)	
Assessment Methods:	Assessment: Coursework: (weighting – 100%) (Written reports 50% Group presentation 20% Personal Development Plan 10% Contribution to chat board 20%)	Re-assessment: Coursework: 100%

Course Code: **F27IS**	**Course Title:** **Interactive Systems**	**Course Co-ordinator:** **Roger Rist, Judy Robertson**
Pre-requisites:	None	
Aims:	**To give students an opportunity to explore current technological media and creative approaches**	
Syllabus:	♦ Basic comparison and evaluation of designs and prototypes ♦ Reflecting on one's own learning and progress ♦ Development of Interactive Systems, for example - Web site development: page layout, navigation, graphics, animation/interaction - Game development using a current game authoring tool: level design, storyline, game mechanics	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ To give students experience of designing and developing an interactive system. ♦ To give students experience of evaluating and critiquing interactive systems.	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ To gain an awareness of the benefits and pitfalls of different approaches to multimedia project work ♦ To raise awareness of the legal and ethical responsibilities within the discipline ♦ To appreciate and enjoy the challenges of creative work (PDP) ♦ To take responsibility for one's own learning and managing workload (PDP) ♦ For students to appreciate their own strengths and weaknesses, and what is possible within time constraints (PDP) ♦ To develop skills in written, oral and media based communication (PDP) ♦ To present solutions to design challenges in the subject area (PDP) ♦ To develop experience and skills in giving and receiving constructive criticism (PDP)	
Assessment Methods:	Assessment: Coursework: (weighting – 100%)	Re-assessment: Coursework: 100%

Course Code: F17LP	Course Title: Logic & Proof	Course Co-ordinator: Dr Mark Lawson
Pre-requisites:	None	
Aims:	**To give an introduction to and an appreciation of the basic principles and techniques of logic and proof fundamental to Computer Science.**	
Syllabus:	♦ Logic and proof ♦ Propositional calculus ♦ Truth tables, predicate calculus ♦ Inference rules ♦ Soundnes, ♦ completeness ♦ Validity ♦ Satisfiability ♦ Reasoning and calculating with propositions ♦ Practical applications	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ To demonstrate an understanding of the principles of propositional and predicate calculus. ♦ To foresee the role of argument in logical reasoning. ♦ To be able to formulate statements as well formed formulae in propositional and predicate calculus. ♦ To be able to construct formal proofs of logical arguments.	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ To be able to express arguments/problems in propositional and predicate calculus. ♦ To be able to communicate in using formal notations	
Assessment Methods:	Assessment: Exam: (weighting – 70%) Coursework: (weighting – 30%)	Re-assessment: Exam: 100%

Computer Science & Software Engineering

Course Descriptors

Year 1, Semester 2

Course Code: F27SB	Course Title: Software Development 2	Course Co-ordinator: Brian Palmer,Greg Michaelson
Pre-requisites:		
Aims:	♦ **To impart further techniques of object orientation** ♦ **To introduce simple data structures and algorithms**	
Syllabus:	♦ Inheritance: hierarchies, subclasses, polymorphism, static and dynamic type, overriding, dynamic method lookup ♦ Abstract classes, abstract methods, interfaces ♦ GUIs: components, layout, event handling ♦ Error-handling: defensive programming, exceptions, assertions ♦ Collection classes ♦ Stacks, queues, lists, priority queues, binary trees ♦ Basic algorithms such as searching and sorting	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ Knowledge of the object-oriented paradigm ♦ Understanding of inheritance ♦ Knowledge of graphic user interfaces ♦ Experience of an application programming interface ♦ Knowledge of simple data structures: stacks, queues, priority queues, lists, binary trees ♦ Knowledge of basic computing algorithms such as searching and sorting ♦ Grounding in complexity theory (big O notation)	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ Possession of fundamental skills in computer science, applicable throughout the remainder of the degree ♦ Understanding of the importance of regular working habits (PDP) ♦ Understanding of the use of chatboards and other devices to learn from and instruct others in the class (PDP) ♦ Ability to compare and evaluate the applicability of simple data structures to relevant problems (PDP	
Assessment Methods:	Assessment: Exam: (weighting – 50%) (Electronic) Coursework: (weighting – 50%)	Re-assessment: Examination 100%

Course Code: F27CS	Course Title: Introduction to Computer Systems	Course Co-ordinator: Peter King, Pier Frisco
Pre-requisites:		
Aims:	♦ **To introduce students to modern computer systems architecture** ♦ **To give students an appreciation of logical design and data representation**	
Syllabus:	♦ Overview. ♦ Hardware components - peripherals, memory & CPU. ♦ Boolean algebra. ♦ Low-level information representation. ♦ CPU organisation. ♦ Introductory assembly language programming. ♦ Operating system: I/O; interrupts; scheduler; virtual memory; file system. ♦ Concurrency: processes; threads; synchronisation; shared & distributed memory; distributed & parallel architectures. ♦ Language processors: compiler; interpreter; assembler; loader. ♦ Linux shell scripting	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ Overview of hardware/software hierarchy in contemporary computer systems; ♦ Understanding of purpose and function of major system hardware and software components; ♦ Understanding of information representation in computer systems; ♦ Ability to write Linux shell scripting	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ Appreciation of significance of hardware & system software in supporting computer applications ♦ Basic ability to understand hardware and software specifications	
Assessment Methods:	Assessment: Computer Tests: (weighting – 100%) (	Re-assessment: Examination: (weighting – 100%)

Course Code: F27WD	Course Title: Web Design and Databases	Course Co-ordinator: Helen Hastie, TBC
Pre-requisites:		
Aims:	**To develop knowledge and understanding of fundamental web design concepts and combine these with database structuring and querying techniques applying this knowledge by implementing an easy-to-use website.**	
Syllabus:	♦ Introduction to web development. ♦ Information architecture. ♦ Web design and usability. ♦ Fundamentals of Mark-up and CSS. ♦ Introduction to database systems. ♦ Databases and Information Systems. ♦ Modelling of data/entity-relationship modelling. ♦ The relational data model. ♦ The Structured Query Language (SQL). ♦ Web-based database applications including the use of PHP.	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ To explain fundamental web design concepts including usability. ♦ To implement a simple web site which satisfies current standards and uses a database. ♦ To describe the use of CSS and mark-up within a web site and the advantage this gives the developer. ♦ To describe the need for standard XHTML and how this aids cross browser compatibility. ♦ To have knowledge and understanding of data analysis and structuring techniques. ♦ To design database structures as a relational data model. ♦ To implement and query a designed database structure through a web site.	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ To analyse complex information and organise it in a structured way for a web site. ♦ To understand stakeholders' requirements and address them. ♦ To design a web site that is easy and cost efficient to manage. ♦ To analyse data sources and represent them in an efficient structured form. ♦ Problem solving (PDP). ♦ Paired work (PDP). ♦ Time management (PDP). ♦ Reflection, constructive criticism and learning from peers (PDP).	
Assessment Methods:	Assessment: Examination: (weighting – 60%) Coursework: (weighting – 40%)	Re-assessment: Examination: (weighting – 100%)

Course Code: F27SG	Course Title: Software Development 3	Course Co-ordinator: TBC
Pre-requisites:		
Aims:	**To develop further skills and techniques in programming in a high-level language.**	
Syllabus:	♦ static structures – tables ♦ linear techniques e.g. search, delete, update ♦ string & text processing ♦ dynamic structures - stacks & queues ♦ recursive techniques – linear recursion, accumulation recursion ♦ sorting & searching e.g. binary search, quicksort, merge sort, hash tables ♦ linked structures – lists – construction, traversal, delete, update ♦ linked structures – trees – construction, traversal, delete, update, balance ♦ file processing ♦ introductory complexity & "big O" notation ♦	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ To understand properties of and algorithms for fundamental static, dynamic and linked data structures ♦ To know when to deploy fundamental data structures and algorithms in practical problem solving ♦ To gain mastery of fundamental linear and recursive programming techniques ♦ To know when to deploy linear and recursive programming techniques in practical problem solving ♦ To understand fundamental techniques for processing very large data sets from files ♦ To gain skill in elementary analyses of fundamental algorithms and data structures to give insight into their time and space complexity bounds ♦ To understand correspondences between different programming techniques ♦ To understand correspondences between different data structures and algorithms	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ To understand how the choice of algorithms and data structures determines the efficacy of proposed solutions to problems ♦ To be able to explain the implications of choosing particular algorithms and data structures for the time and space behaviour of solutions	
Assessment Methods:	Assessment: Examination: (weighting – 50%) Coursework: (weighting – 50%)	Re-assessment: Examination: (weighting – 100%)

Computer Science & Software Engineering

Course Descriptors

Year 2, Semester 1

Course Code: F28IN	Course Title: Interaction Design	Course Co-ordinator: Sandy Louchart, Helen Hastie
Pre-requisites:	F27IS1 Interaction Systems *or equivalent*	
Aims:	**The course aims to give students the opportunity to develop:** ♦ **A broad knowledge and understanding of requirements gathering, design and evaluation theory and techniques in interaction design.** ♦ **An introduction to commonly used design techniques and pattern for user interfaces.** ♦ **A selection of routine skills and methods involved in working with users.**	
Syllabus:	Current topics in Interaction Design including: interaction design lifecycles, user interface design patterns, basic qualitative and quantitative data gathering and presentation techniques, accessibility.	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ Critically analyse interaction design and interfaces. ♦ Propose solutions in response to interface design problems ♦ Evaluate the effectiveness of user interfaces with respect to user requirements.	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ Use discipline appropriate software for data analysis, ♦ Present, analyse and interpret simple numerical and graphical data gathered as part of evaluation studies. (PDP) ♦ Communicate effectively to knowledgeable audiences by preparing informal presentations and written reports. (PDP) ♦ Exercise autonomy and initiative by planning and managing their own work within a specified project; (PDP) ♦ Take responsibility for their own and other's work by contributing effectively and conscientiously to the work of a group (PDP)	
Assessment Methods:	Assessment: Examination: (weighting – 60%) Coursework: (weighting – 40%)	Re-assessment: Examination: (weighting – 100%)

Course Code: F28DA	Course Title: Data Structures & Algorithms	Course Co-ordinator: Phil Trinder, Lilia Georgieva
Pre-requisites:	F27SB2 Software Development 2 *or equivalent*	
Aims:	♦ **To introduce core algorithms and data structures used in a wide range of applications in Computer Science** ♦ **To further develop skills in algorithm and data structure design, and the development of medium sized programs**	
Syllabus:	♦ Algorithm and data structure topics including: advanced trees, string processing, graphs, hash tables ♦ Algorithm/data structure choice, design and deployment	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ Ability to analyse and hence choose suitable algorithms and data structures for a given problem ♦ To design and implement medium sized programs based on a range of standard algorithms and data structures and making appropriate use of libraries ♦ Understanding the distinction between abstract Algebraic Data Type (ADT) properties and concrete ADT realisations ♦ Appreciation of need for integration of multiple ADTs in substantial programs ♦ Appreciation of efficiencies/reassurances from ADT reuse	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ To be able to critically analyse and hence choose suitable algorithms and data structures for a given problem ♦ To be able to convey the advantages and disadvantages of alternative data structures and algorithms ♦ To develop practical problem-solving skills in the context of programming ♦ To be able to plan & execute a substantial software	
Assessment Methods:	Assessment: Examination: (weighting – 60%) Coursework: (weighting – 40%)	Re-assessment: Examination: (weighting – 100%)

Course Code: F28IT	Course Title: Internet & Communications	Course Co-ordinator: Hamish Taylor, Peter King
Pre-requisites:	F27CS Introduction to Computer Systems, F27SA Software Development 1 *or equivalent*	
Aims:	♦ **Appreciate the structure, organisation and standards of Internet and some key applications such as web and email** ♦ **Impart basic design and application development skills for the web** ♦ **Understand basic data communication protocols and routing techniques in computer networks**	
Syllabus:	**Internet and some Applications** Network types, distributed computing models, Internet - bodies, IP addresses and DNS; socket programming; E-mail – MIME, SMTP; Web – data formats and presentation - HTML, XHTML, DOM, CSS, DHTML, web page design; Simple client and server web programming - JavaScript, CGI, servlets, active web server pages (JSP, PHP). **Data Communications** Introduction, history, switching methods, and architecture Data transparency, error detection, synchronous and asynchronous transmission Data link protocols, sliding windows, flow control; Network layer protocols	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* Knowledge of Internet and some major applications ♦ Understanding of what Internet is and how it is structured and organised ♦ Knowledge of nature of Web, E-mail and DNS as major Internet applications ♦ Know how to design and develop simple client and server web applications Broad and integrated knowledge and understanding of computer networks ♦ Basic understanding of different computer network architectures, Internet, ISO 7 Layer model ♦ Detailed knowledge of circuit switching, message switching and packet switching ♦ Understanding of the need for protocol specification ♦ Basic knowledge of sliding window protocols, routing protocols, and transport protocols	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* Professional Development ♦ Appreciation of role of standards and guidelines in networking and its applications ♦ Appreciation of good practice HCI norms governing accessibility & presentation design in web applications Practical Expertise ♦ Ability to analyse and explain basic issues relating to Internet, web and networking technologies ♦ Practice in ICT, numeracy and report writing, team working	
Assessment Methods:	Assessment: Examination: (weighting – 60%) Coursework: (weighting – 40%)	Re-assessment: Examination: (weighting – 100%)

Course Code: F27SP	Course Title: Foundation Maths A	Course Co-ordinator: Jack Carr
Pre-requisites:	Higher Maths at C (*or equivalent)*	
Aims:	**This course seeks to provide students with a range of techniques in algebra and calculus which equip them to tackle problems in their subject area. The course also provides a foundation in mathematics for more advanced courses in later years.**	
Syllabus:	Algebra revision (symbols, expressions, indices, powers, partial fractions, trig, quadratic equations) Introduction to differentiation Introduction to integration Applications of integration Logarithms, exponentials and hyperbolic functions Further differentiation	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* The students will acquire technical skills in algebra, differentiation and the evaluation of integrals. The course will help to develop the basic mathematical skills required for their main degree subjects.	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* Present mathematical arguments in a clear and precise way. Evaluate information and adopt a suitable strategy to implement a solution.	
Assessment Methods:	Assessment: Examination: (weighting – 70%) Coursework: (weighting – 30%)	Re-assessment: Examination: (weighting – 100%)

Course Code: F27IS	Course Title: Interactive Systems	Course Co-ordinator: Roger Rist, Judy Robertson
Pre-requisites:	None	
Aims:	**To give students an opportunity to explore current technological media and creative approaches**	
Syllabus:	♦ Basic comparison and evaluation of designs and prototypes ♦ Reflecting on one's own learning and progress ♦ Development of Interactive Systems, for example - Web site development: page layout, navigation, graphics, animation/interaction - Game development using a current game authoring tool: level design, storyline, game mechanics	
Learning Outcomes: Subject Mastery	*Understanding, Knowledge and Subject-Specific Skills* ♦ To give students experience of designing and developing an interactive system. ♦ To give students experience of evaluating and critiquing interactive systems.	
Learning Outcomes:: Personal Abilities:	*Cognitive skills, Core skills and Professional Awareness* ♦ To gain an awareness of the benefits and pitfalls of different approaches to multimedia project work ♦ To raise awareness of the legal and ethical responsibilities within the discipline ♦ To appreciate and enjoy the challenges of creative work (PDP) ♦ To take responsibility for one's own learning and managing workload (PDP) ♦ For students to appreciate their own strengths and weaknesses, and what is possible within time constraints (PDP) ♦ To develop skills in written, oral and media based communication (PDP) ♦ To present solutions to design challenges in the subject area (PDP) ♦ To develop experience and skills in giving and receiving constructive criticism (PDP)	
Assessment Methods:	Assessment: Coursework: (weighting – 100%)	Re-assessment: Coursework: 100%

Computer Science & Software Engineering

Course Descriptors

Year 2, Semester 2

Course Code: F27SD	Course Title: Software Design	Course Co-ordinator: Andrew Ireland, Brian Palmer
Pre-requisites:	F27SA – Software Development 1 OR *equivalent*	
Aims:	♦ **To provide an understanding of the challenges associated with the software life-cycle, focusing on the processes and techniques which promote the design of high quality software.** ♦ **To instil an understanding of the concepts and benefits of rigorous design.** ♦ **To equip participants with a broad understanding of software design methods and processes.** ♦ **To equip participants with an in-depth knowledge of UML notation** ♦ **To instil an understanding of design patterns and their value** ♦ **To provide a broad understanding of verification and validation issues associated with software engineering.** ♦ **To equip CS participants with an in-depth understanding of analysis techniques, i.e. static analysis and dynamic analysis.** ♦ **To instil IS participants with a broad understanding of the information systems lifecycle, and an in-depth understanding of feasibility studies.** ♦ **To provide participants with an understanding of software design challenges in the real-world via guest speakers from industry.**	
Syllabus:	♦ Motivations for design and the need for design processes and methodologies. ♦ Software process models, e.g. waterfall model, Boehm's spiral model, evolutionary development, agile development. ♦ Design methods and processes, i.e. architectural design, function-oriented design, object-oriented design, component-based design. ♦ Object-oriented design using UML notation & CRC cards: use case diagrams, noun verb analysis, class diagrams, sequence diagrams, communication diagrams, state machine diagrams, activity diagrams. ♦ An introduction to patterns and anti-patterns ♦ Software testing and analysis: a strategy for dynamic testing; test case design; assertion based testing; functional testing; structural testing; hybrid testing; auto unit testing; reviews & inspections; metrics; flow analysis; formal methods (CS stream) ♦ Open source vs closed source and means of assessing which is appropriate to a project – Boehm's ADA and other methods ♦ Documentation practice and standards in software engineering and information systems ♦ Information systems lifecycle focussing on feasibility studies. Evaluating potential projects by considering operational, technical, schedule and economic feasibility. Cost/benefit analysis. (IS stream).	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ To demonstrate a critical understanding of modern software engineering practice and be able to evaluate the strengths and weaknesses of current software engineering methods and technique ♦ To be able to choose appropriate metrics to measure software quality and quantity in a modern software engineering environment ♦ To understand and demonstrate object oriented design techniques, including the use of UML notation	

	♦ To understand the role of a feasibility study, and to appreciate social and technological aspects of information systems (IS stream)	
Learning Outcomes:: Personal Abilities:	*Cognitive skills, Core skills and Professional Awareness* ♦ Take responsibility for own work and exhibit critical reflection on development process. (PDP) ♦ To be able to use appropriate methods and standards for practice and documentation in software engineering and information systems. ♦ Demonstrate evidence based approaches to problem solving. ♦ Use a range of numerical and graphical skills in evaluating and communicating ideas, as well as measuring progress toward achieving goals.	
Assessment Methods:	Assessment: Examination: (weighting – 60%) Coursework: (weighting – 40%)	Re-assessment: Examination: (weighting – 100%)

Course Code: F28DM	Course Title: Database Management Systems	Course Co-ordinator: Monica Farrow
Pre-requisites:	F27WD Web Design & Databases *or equivalent*	
Aims:	**To familiarise students with the principles of database management systems, to enable them to design and implement databases for specific applications and to integrate databases with application programs.**	
Syllabus:	♦ **Database Design:** data requirements, entity relationship diagrams, relational data model, integrity constraints, key constraints, types, integrity maintenance ♦ **Relational Queries:** SQL, Boolean combinations of queries, aggregation, duplicate elimination, nested queries, negation, views, insertions, deletions, updates, command level interfaces, JDBC ♦ **Query execution and optimisation:** data storage principles, file organisation, indexing, indexes in commercial DBMS's, relational algebra, query execution plans, cost estimation of plans, interpretation of plans, physical database design ♦ **Concurrency:** transactions, schedules, serialisability, concurrency control protocols, locking, two-phase-locking, time stamp based concurrency control. ♦ **Emerging Database Trends:** XML and data warehousing	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ Broad knowledge and understanding of the concepts and formalisms of database design ♦ Detailed knowledge of the building blocks and meaning of relational database queries ♦ Critical understanding of the principles of query evaluation and concurrency control underlying database applications ♦ Practice in the collection of data requirements and the design of conceptual database schemas ♦ Evaluation of emerging database trends and ability to understand their benefits	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ Practice in working on a development project in a small group under the guidance of a tutor (PDP) ♦ Practice in defining the subject and scope of a development project (PDP) ♦ Presenting the project orally, and reporting on it in writing (PDP) ♦ Use of ICT techniques for presenting the project (PDP) ♦ Constructing a website that reports on the progress of the project ♦ Practice in designing tests for quantitative analysis	
Assessment Methods:	Assessment: Examination: (weighting – 70%) Coursework: (weighting – 30%)	Re-assessment: Examination: (weighting – 100%)

Course Code: F28PL	Course Title: Programming Languages	Course Co-ordinator: Greg Michaelson
Pre-requisites:	F27CS2 Introduction to Computer Systems, F27SB Software Development 2 *or equivalent*	
Aims:	♦ **To give an appreciation of the context of programming languages** ♦ **To introduce low level programming techniques** ♦ **To introduce system level programming techniques** ♦ **To introduce simple compiling techniques**	
Syllabus:	♦ Low level programming e.g. assembly language: registers, memory models, arithmetic, comparison, branching, subroutines, recursion ♦ System level programming e.g C: addressing modes, pointers, arrays and structs, control structures, functions and parameter passing, basic libraries ♦ Simple compiling techniques: lexical analysis, recursive descent parsing, AST construction, simple code generation, use of compiler generator tools, ♦ Overview of programming language history/paradigms	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ Appreciation of levels of and correspondences between hardware/software hierarchy ♦ Simple assembly language programming ♦ Simple C programming ♦ Crafting simple language processors ♦ Use of compiler generators	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ Experience of low- and intermediate-level programming ♦ Understanding of role and deployment of low-level and system languages in system development ♦ Appreciation and use of appropriate formalisms for defining languages and communicating language definitions ♦ Ability to apply language processing techniques/tools for wider problem solving	
Assessment Methods:	Assessment: Examination: (weighting – 70%) Coursework: (weighting – 30%)	Re-assessment: Examination: (weighting – 100%)

Course Code: F17SQ	Course Title: Foundation Mathematics B	Course Co-ordinator: Alan Prince
Pre-requisites:	Higher Maths at C (*or equivalent*)	
Aims:	**This course seeks to provide students in their first year in the university with a range of techniques in algebra and calculus which equip them to tackle problems in their subject area. The course also provides a foundation in mathematics for more advanced courses in later years.**	
Syllabus:	♦ Vectors ♦ Matrices ♦ Advanced Integration ♦ First Order Differential Equations ♦ Complex Numbers	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* The students will acquire technical skills in linear algebra, differentiation, the evaluation of integrals and complex numbers. The course will help to develop the basic mathematical skills required for their main degree subjects.	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* Present mathematical arguments in a clear and precise way. Evaluate information and adopt a suitable strategy to implement a solution.	
Assessment Methods:	Assessment: Examination: (weighting – at least 70%) Coursework: (weighting – up to 30%)	Re-assessment: Examination: (weighting – 100%)

Course Code: F17SC	Course Title: Discrete Mathematics	Course Co-ordinator: Jim Howie
Pre-requisites:		
Aims:	**The goal of the course is to explain the basic techniques of discrete mathematics which are used in computer science.**	
Syllabus:	♦ **Set Theory and Combinatorics:** Set algebra, The Inclusion-Exclusion Principle, Binomial Theorem, Elementary counting methods, Mathematical Induction. (6 lectures) ♦ **Graph Theory 1:** Introduction to graphs, Basic graph terminology. Adjacency Matrices, Paths and connectivity, Connected components, Shortest path problems in weighted graphs, Dijkstra's Algorithm. (5 lectures) ♦ **Graph Theory 2:** Trees and spanning trees, Breadth-first search, Kruskal's and Prim's Algorithms for a minimal spanning tree, Euler and Hamilton paths, Fleury's Algorithm for constructing Euler circuits, Prefix Codes and Huffmann Coding. (6 lectures) ♦ **Recurrence Relations:** Solving problems by iteration, First and second order recurrence relations, Reccurrences in Algorithms (3 lectures) ♦ **Matrices and Linear Transformations:** Linear equations and elementary row operations, Elementary matrices and Gaussian elimination, Echelon matrices, Matrices as space transformations, Eigenvectors and eigenvalues, Diagonalization, The rank theorem. (8 lectures) ♦ **Probability Theory:** Probability Space, Conditional Probability, Independence and Bayes' Theorem, Random Variables and Distributions, Expected Value, Variance, Examples of applications to algorithms (5 lectures).	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ Know the basic terminology of set theory, graph theory, liner algebra and probability theory ♦ Understand how formal mathematical objects like sets, graphs, matrices, recurrence relations arise in computer science related problems ♦ Be able to solve elementary counting problems, solve systems of linear equations, apply graph algorithms, solve simple recurrence relations, be able to compute probabilities ♦ Appreciate the power of mathematical formalisation, facilitated by the use of precise definitions and notations, in solving practical problems. ♦ Appreciate the value of careful, quantitative reasoning in analysing problems related to computer science and to recognise that the outcome of such reasoning can defy naïve intuition	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ Understand and use abstract mathematical concepts ♦ Use logical reasoning to prove theorems ♦ Communicate mathematical orally and in writing	
Assessment Methods:	Assessment: Examination: (weighting – 70%) Coursework: (weighting – 30%)	Re-assessment: Examination: (weighting – 100%)

Computer Science & Software Engineering

Course Descriptors

Year 3, Semester 1

Course Code: **F29SO**	**Course Title:** **Software Engineering**	**Course Co-ordinator:** **Monica Farrow**
Pre-requisites:	F28IN Interaction Design, F28IT Internet & Communications, F28DM Database Management Systems, F28SD Software Design, *or equivalent*	
Aims:	♦ **To equip students with knowledge and skills for the effective management of a group project which encompasses the software development lifecycle** ♦ **To enable students to reinforce their knowledge and skills gained in software processes, internet technology, database management and interaction design** ♦ **To build students understanding, knowledge and skills in teamwork, software development in groups, and project planning.** ♦ **To enable students to develop a broader understanding of the interrelationship of development life-cycles and a critical capability in the selection of tools and methods to support project planning, systems analysis, requirements capture, and system specification.**	
Syllabus:	♦ Review and extension of the components studied in earlier years which contribute to the group project ♦ Software project management including working in groups, project planning and costing, risk assessment ♦ Use of Industry-level Standards for software development and documentation, covering aspects such as change control and requirements traceability ♦ Further study of software development tools, especially version control	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ A broad and integrated understanding and knowledge of the various development and programming paradigms, software development life-cycles, teamwork and project planning ♦ Detailed theoretical and practical knowledge of the use of methodologies for requirements capture, iterative design, resource capture and management, deployment and evaluation of systems, at a basic level ♦ Practice in the use of object-oriented programming, databases, scripting and markup languages applied to a substantial project	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ Identification, critical analysis and evaluation of the development of a software system (PDP) ♦ Practice in working in a group, negotiating requirements, reaching a consensus, taking responsibility for own work, taking part in a presentation, and working with others to a deadline (PDP) ♦ Appreciation of the interrelationship of knowledge domains	
Assessment Methods:	Assessment: Group Project: (weighting - 50%) Examination: (weighting – 30%) synoptic with F29PD Coursework: (weighting – 20%) Synoptic with F29PD Professional Development	Re-assessment: Coursework (individual): (weighting – 100%)

Course Code: F29AI	Course Title: Artificial Intelligence	Course Co-ordinator: Ruth Aylett, Lilia Georgieva, Patricia Vargas
Pre-requisites:	Elementary knowledge of logic at the level of undergraduate Computer Science. Knowledge of high-level programming language concepts	
Aims:	♦ **To introduce the fundamental concepts and techniques of AI, including planning, search and knowledge representation** ♦ **To introduce the scope, subfields and applications of AI, topics to be taken from a list including natural language processing, expert systems, robots and autonomous agents, machine learning and neural networks, and vision.** ♦ **To develop skills in AI programming in an appropriate language**	
Syllabus:	♦ Search algorithms (depth first search, breadth first search, uniform cost search, A* search) ♦ constraint satisfaction problems; ♦ games (min-max, alpha-beta pruning); ♦ logic, resolution, introductory logic programming ♦ knowledge representation – logic, rules, frames ♦ goal and data-driven reasoning ♦ practical rule-based programming ♦ Overview of main fields of AI (Vision, Learning, Knowledge Engineering) ♦ In depth view of one field of AI (e.g. Planning, Natural language) ♦ Autonomous agents ♦ Applications of AI ♦ AI programming	
Learning Outcomes: Subject Mastery	*Understanding, Knowledge and Subject-Specific Skills* ♦ Critical understanding of traditional AI problem solving and knowledge representation methods ♦ Use of knowledge representation techniques (such as predicate logic and frames). ♦ Critical understanding of different systematic and heuristic search techniques ♦ Practice in expressing problems in terms of state-space search ♦ Broad knowledge and understanding of the subfields and applications of AI, such as computer vision, machine learning and expert systems. ♦ Detailed knowledge of one subfield of AI (e.g. natural language processing, planning) and ability to apply its formalisms and representations to small problems ♦ Detailed understanding of different approaches to autonomous agent and robot architectures, and the ability to critically evaluate their advantages and disadvantages in different contexts. ♦ Practice in the implementation of simple AI systems using a suitable language	
Learning Outcomes:: Personal Abilities:	*Cognitive skills, Core skills and Professional Awareness* ♦ Identification, representation and solution of problems ♦ Time management and resource organization ♦ Research skills and report writing ♦ Practice in the use of ICT, numeracy and presentation skills.	
Assessment Methods:	Assessment: Examination: (weighting – 70%) Coursework: (weighting – 30%)	Re-assessment: Examination: (weighting – 100%)

Course Code: F29GR	Course Title: Computer Graphics	Course Co-ordinator: Mike Chantler
Pre-requisites:	F28PL Programming Languages *or equivalent*	
Aims:	**To introduce fundamental Computer Graphics theory and programming.**	
Syllabus:	♦ Overview of Computer Graphics & practical introduction to graphics programming ♦ Event driven I/O and callback programming & typical structure of an interactive, real-time computer graphics program ♦ 2&3D transformations, homogeneous co-ordinates, post-multiplication ♦ Modelling and instantiation ♦ Hierarchical modelling and scene graphs ♦ Scene graphs: creating, manipulating, creating a display list ♦ Perspective & orthographic projection ♦ Project specification ♦ Shading models and programming ♦ Texture mapping ♦ Putting it all together: the graphics pipeline ♦ Course summary and review ♦	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ Critical understanding of the theory of 2D and 3D transformations, projection and viewing ♦ Ability to find & combine relevant sources and synthesise designs ♦ Detailed knowledge of the graphics pipeline ♦ Detailed knowledge of shading and texture mapping algorithms ♦ Broad knowledge of 3D modelling and rendering techniques ♦ Ability to understand, design and implement scene graphs ♦ Practical skills in graphics programming including scene graph programming and I/O processing	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ Ability to think and plan critically in three dimensions ♦ General critical analysis, evaluation and synthesis of ideas for the design of their project ♦ Technical report writing and organisation ♦ Team working skills (in pairs) ♦ Representation of, planning for, and solution of problems ♦ Ability to draw upon a range of sources when making decisions in their project work	
Assessment Methods:	Assessment: Examination: (weighting – 65%) Coursework: (weighting – 35%)	Re-assessment: Coursework: (weighting – 100%) (verified by short oral exam)

Course Code: F29FA	Course Title: Foundations 1	Course Co-ordinator: Fairouz Kamareddine, Patricia Vargas
Pre-requisites:		
Aims:	♦ **To give an introduction to and an appreciation of the basic principles and techniques of logic and proof fundamental to Computer Science.** ♦ **Introduce the λ-calculus, how computable functions are represented in the λ-calculus, basic theoretical properties of the λ-calculus, and the relevance of the λ-calculus to computer science.**	
Syllabus:	♦ Logic & proof: propositional calculus – truth tables, predicate calculus, inference rules, soundness, completeness, validity, satisfiability, reasoning and calculating with propositions. ♦ Lambda calculus: syntax, notation, bound & free variables and α-conversion and substitution, reduction and computation, representing computable functions, theoretical properties.	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ To demonstrate an understanding of the principles of propositional and predicate calculus. ♦ To foresee the role of argument in logical reasoning. ♦ Practice in formulating and proving arguments using formal logic ♦ Knowledge of lambda calculus ♦ Understanding of different variable techniques (de Bruijn indices, combinator variables) ♦ Understanding of variable binding and capture-free substitution ♦ Knowledge of how to represent computations in the λ-calculus	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ To be able to formulate statements as well formed formulae in propositional and predicate calculus. ♦ To be able to express arguments/problems in propositional and predicate calculus. ♦ To be able to construct formal proofs of logical arguments. ♦ To be able to think about the meaning of programs mathematically	
Assessment Methods:	Assessment: Examination: (weighting – 70%) Coursework: (weighting – 30%)	Re-assessment: Examination: (weighting – 100%)

Computer Science & Software Engineering

Course Descriptors

Year 3, Semester 2

<table>
<tr><td>Course Code:
F29PD</td><td colspan="2">Course Title:
Professional Development</td><td>Course Co-ordinator:
Sandy Louchart</td></tr>
<tr><td>Pre-requisites:</td><td colspan="3">F28IN Interaction Design, F28IT Internet & Communications, F28DM Database Management Systems, F28SD Software Design, or equivalent</td></tr>
<tr><td>Aims:</td><td colspan="3">♦ To instil a professional and ethical attitude toward the application of computer technology
♦ To introduce methods for the rational resolution of ethical problems
♦ To provide an appreciation of the relevant professional and legal requirements concerning computer-based systems
♦ To ensure an awareness of, and encourage deliberation about, the social implications of information technology</td></tr>
<tr><td>Syllabus:</td><td colspan="3">♦ Professionalism - British Computer Society.
♦ Rules & Regulations - Codes & Standards; Computer Law; Ethical Decision Making.
♦ Risks & Threats - Computer Crime; Viruses.
♦ Privacy & Security – Databases; Biometrics.
♦ Dependence & Change - Safety-Critical Systems; Technology & Society.
♦ Brave New Worlds - Co-operative Computing; eLife.</td></tr>
<tr><td>Learning Outcomes:
Subject Mastery</td><td colspan="3">Understanding, Knowledge and Subject-Specific Skills

♦ British Computer Society Codes - Conduct; Practice
♦ ISO & BSI Standards - Safety; Quality; Security
♦ Statute Law - Contracts, Torts, Restitution; Data Protection; Freedom of Information, Intellectual Property; Computer Misuse
♦ Ethics - Frameworks; Decision Making
♦ Development life-cycle of a software system
♦ Bi-directional influence between technological and societal trends
♦ Current concerns over the application of computer technology
♦ Current and potential remedies to abuse of computer technology</td></tr>
<tr><td>Learning Outcomes::
Personal Abilities:</td><td colspan="3">Cognitive skills, Core skills and Professional Awareness

♦ Practice in personal decision making and introspection
♦ Identification and analysis of justification of personal choices to others
♦ Critical analysis of rational reasoning, consequential reasoning and debate
♦ Practice and reflective analysis of communication skills using a variety of media
♦ Practice in working in a group, negotiating requirements, reaching a consensus, and working with others to a deadline</td></tr>
<tr><td>Assessment Methods:</td><td colspan="2">Assessment:
Group Project: (weighting - 50%)
Examination: (weighting – 30%) synoptic with F29SO
Coursework: (weighting – 20%)
Synoptic with F29SO Software Engineering</td><td>Re-assessment:
Coursework (individual): (weighting – 100%)</td></tr>
</table>

<table>
<tr><td>Course Code:
F29OC</td><td colspan="2">Course Title:
Operating Systems & Concurrency</td><td>Course Co-ordinator:
Pier Frisco</td></tr>
<tr><td>Pre-requisites:</td><td colspan="3">F28DA Data Structures & Algorithms, F28PL Programming Languages or equivalent</td></tr>
<tr><td>Aims:</td><td colspan="3">For the Operating Systems part: To provide an introduction to operating systems, their basic principles and shell programming.

For the Concurrency part: To introduce the theory and practice of concurrent hardware and software systems</td></tr>
<tr><td>Syllabus:</td><td colspan="3">For the Operating system part: overview on operating systems concepts and structures, processes, threads, classical inter-process communication problems, memory management

For the Concurrency part: Process and Threads, Concurrent Execution, Shared Objects and Mutual Exclusion, Monitors and Condition Synchronisation, Deadlock, Safety and Liveness, Model Based Design. Performance, Introductions, Processors, Pipelines.</td></tr>
<tr><td>Learning Outcomes:
Subject Mastery</td><td colspan="3">Understanding, Knowledge and Subject-Specific Skills

For the Operating systems part:
♦ Understanding of the concepts and structures present in modern operating systems.

For the Concurrency part:
♦ Broad and integrated knowledge and understanding of concurrency concepts, techniques and problems
♦ Critical understanding of exclusion, synchronisation and deadlock
♦ Detailed knowledge of abstract modelling and model-based design</td></tr>
<tr><td>Learning Outcomes::
Personal Abilities:</td><td colspan="3">Cognitive skills, Core skills and Professional Awareness

♦ Critically evaluate the problematic and concepts related to operating systems.
♦ Analysis of the different possible solutions to the problematic.</td></tr>
<tr><td>Assessment Methods:</td><td colspan="2">Assessment:
Coursework (Class Tests): (weighting – 100%)</td><td>Re-assessment:
Examination: (weighting – 100%)</td></tr>
</table>

Course Code: F29FS	Course Title: Formal Specification	Course Co-ordinator: Fairouz Kamareddine
Pre-requisites:	F17SC Discrete Maths *or equivalent*	
Aims:	**To introduce students to specification of programs in a formal logical language (Z), and demonstrate the path from this to programs in a programming language (ML).**	
Syllabus:	♦ Reprise of logic and proof. ♦ Basic notions of set theory. ♦ Introduction to Z notation ♦ Relations and functions ♦ Schemas and specification structure ♦ Introduction to SML: atomic types, structured types, pattern matching, recursion, higher order constructs, exceptions ♦ Representing Z constructs in SML: predicates, sets, relations, functions ♦ Converting Z schema to SML: invariants as assertions, preconditions as guards, postconditions as actions ♦ Schema-based prototype testing	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ Accurate understanding of the syntax and informal semantics of Z. ♦ Given a real world example, create a formal specification to model it. ♦ Reason with the specification, once produced. ♦ Given a specification, discuss what it models. ♦ Understand ML: polymorphic types, recursive functions, higher order functions, pattern matching, structured types. ♦ Given a specification, interpret it as operational requirements of a program. ♦ Given a specification, refine it to a program satisfying that specification.	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ Understand the importance of specification and its distinction from implementation ♦ Develop mathematical specification skills ♦ Appreciation of the role of formal techniques in software development. ♦ Ability to deploy declarative abstraction in software development. ♦ Awareness of limitations of formal techniques.	
Assessment Methods:	Assessment: Examination: (weighting – 70%) Coursework: (weighting – 30%)	Re-assessment: Examination: (weighting – 100%)

Course Code: F29FB	Course Title: Foundations 2	Course Co-ordinator: Joe Wells
Pre-requisites:	F17SC Discrete Maths	
Aims:	♦ **To introduce basic notions of computability.** ♦ **To understand two models of computability: the lambda-calculus and Turing machines.** ♦ **To understand which functions can be computed.**	
Syllabus:	Enumerability; countability and non-countability; Goedel numbering; Turing machines; review of the lambda-calculus; computable and non computable functions; Turing computability; Solvability and reduction of decision problems; Church's thesis and effective computability	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* Become competent with enumerability, Turing machines, encoding functions with the lambda-calculus, Goedel numbering, & computability	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ Understand basic mathematical thinking as it applies to computability. ♦ Become aware of limits of computing.	
Assessment Methods:	Assessment: Examination: (weighting – 70%) Coursework: (weighting – 30%)	Re-assessment: Examination: (weighting – 100%)

Computer Science & Software Engineering

Course Descriptors

Year 4, Semester 1

Not all final year optional courses may run in a given year

More information about the Dissertation can be found at:
http://www.macs.hw.ac.uk/~pjbk/project.pdf

<table>
<tr><td>Course Code:
F20PA</td><td colspan="2">Course Title:
Research Methods & Requirements Engineering</td><td>Course Co-ordinator:
Peter King</td></tr>
<tr><td>Pre-requisites:</td><td colspan="3"></td></tr>
<tr><td>Aims:</td><td colspan="3">Development of project research method and requirement analysis skills.</td></tr>
<tr><td>Syllabus:</td><td colspan="3">Requirements analysis of software development project
Researching current state of art in this area
Library resources and their use, Web and online database searching</td></tr>
<tr><td>Learning Outcomes:
Subject Mastery</td><td colspan="3">Understanding, Knowledge and Subject-Specific Skills

♦ Understanding of research or development based problem related to a substantial software development topic
♦ Requirements specification and background research skills for it
♦ Ability to plan a significant project of research, investigation or development</td></tr>
<tr><td>Learning Outcomes::
Personal Abilities:</td><td colspan="3">Cognitive skills, Core skills and Professional Awareness

♦ Ability to research and undertake critical review and evaluation of data and supplied literature
♦ Project planning skills
♦ Written communication skills
♦ Time management</td></tr>
<tr><td>Assessment Methods:</td><td colspan="2">Assessment:
Coursework: (weighting – 100%)
Synoptic with F20PB & F20PC</td><td>Re-assessment:
None</td></tr>
</table>

Course Code: F20CL	Course Title: Computing in the Classroom	Course Co-ordinator: Monica Farrow, Judy Robertson
Pre-requisites:		
Aims:	This course runs under the Undergraduate Ambassadors Scheme (http://www.uas.ac.uk/) and provides an opportunity for students to act as ambassadors for their disciplines. **Aims are:** ♦ To develop a range of skills in the student and to offer an early taste of teaching to those interested in pursuing it as a career. ♦ To help students gain confidence in communicating their subject, develop strong organisational and interpersonal skills, and understand how to address the needs of individuals. ♦ To learn to devise and develop projects and teaching methods appropriate to engage the relevant age group they are working with. ♦ To help inspire a new generation of prospective undergraduates by providing role models for school pupils. ♦ To help teachers convey the excitement of their subject to their pupils by showing them the long-term applications of school studies. ♦ To help teachers by providing an assistant who can work with and support pupils at any point on the ability spectrum.	
Syllabus:	This format is standard within the Undergraduate Ambassadors : ♦ Initial training to provide the student with an introduction to working with children and conduct in the school environment ♦ Competitive interview system to ensure students' suitability for the course. ♦ Student will be matched with an appropriate school and a specific teacher in the local area. ♦ Student will spend the equivalent of half a day a week in the school every week for a semester. ♦ No formal lectures. ♦ A supporting tutorial for one hour once a week for students to share experiences. ♦ Initial contact with the teacher and pupils will be as a classroom assistant - observing the teacher and offering practical support. ♦ The teacher will assign the student teaching tasks which will vary depending on the teacher's needs and the ability of the student. ♦ The student will design and deliver a special project on the basis of discussion with the teacher and their own assessment of what will interest the particular pupils they are working with. ♦ End of unit presentation of special project	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ Communicate and present computing topics to others. ♦ Develop a better understanding of, appreciation of, and confidence in computing through teaching it to others. ♦ Gain a broad understanding of many of the key aspects of teaching computing in schools	

<table>
<tr><td>Learning Outcomes::
Personal Abilities:</td><td colspan="2">Cognitive skills, Core skills and Professional Awareness

To experience working in a challenging and unpredictable environment.

Be able to assess and devise appropriate ways to communicate a difficult principle or concept

To master the following specific and transferable skills:
- Understanding the needs of individuals.
- Employ effective interpersonal skills when dealing with colleagues.
- To understand and support teaching staff responsibilities and conduct oneself accordingly.
- To be able to improvise and adapt to the responses of students in a classroom.
- To give (and take) feedback.
- To make effective use of organisational, prioritisation and negotiating skills.
- To handle difficult and potentially disruptive situations.
- To be able to use public speaking and communication skills, both one-to-one and when speaking to an audience
- Team-working.
- To understand and be able to make use of standard teaching methods.
- To prepare, use and reflect on the effectiveness of lesson plans and teaching materials.

To be able to reflect on the effectiveness of methods of teaching and learning both personally and in review with tutors.</td></tr>
<tr><td>Assessment Methods:</td><td>Assessment:
Coursework (weighting – 100%)
Course will be spread over 2 semesters so students will have to work more on their project in Semester 1 to compensate for spending some time on it in Semester 2 (equivalent to half course)</td><td>Re-assessment:
None</td></tr>
</table>

Course Code: F21CN	Course Title: Computer Network & Security	Course Co-ordinator: Hamish Taylor, Lilia Georgieva &Hans-Wolfgang Loidl
Pre-requisites:	Fundamental knowledge of computer networking, formal methods & Java programming	
Aims:	♦ Impart critical understanding of key concepts, issues, theories and principles of computer network security. ♦ Develop detailed theoretical and practical knowledge of foundational issues in computer network security. ♦ Provide detailed understanding and practical experience with key services and tools used for computer network security purposes. ♦ Give practical experience of analysing requirements, designing, implementing and testing security solutions for computer network applications	
Syllabus:	♦ Introduction – concepts, methods, principles, technologies. Formal methods - logics for cryptography: BAN logic, temporal logic, typed first-order logic. Model checking for security. Introducing SPIN/Alloy. Hash functions for data integrity. Identification and authentication: passwords, challenge response, zero knowledge. Attacks on protocols. Modelling of attacks in SPIN/Alloy. Case study: key establishment protocol: symmetric, asymmetric, public key encryption. ♦ Basics of cryptography: principles & algorithms - concepts (classification, symmetric vs asymmetric encryption etc); public-key encryption: challenges and algorithms. Key Management - key establishment protocols, key management infrastructures. Proof-carrying-code - concepts (role of trust, authentication-based/free certification, logical foundations; case study: PCC for resources. Operating system security - concepts (vulnerabilities in: multi-user, distributed etc OSs), security-enhanced Linux. ♦ X.800 network security model - attacks, mechanisms, services. Digests, symmetric ciphers, JCE. Digital signatures, certificates, signed code. X.509 and PGP certificates. Secure key exchange - SSL/TLS, SSH. Firewalls - gatekeeper proxy, packet filter; Intrusion Detection Systems - signature scanner vs anomaly detector. Hacker software - botnets, port scanners, malware - worms, viruses, trojans, spyware, adware. Malware scanners. Honey pots. Web and e-mail security HTTP authentication, secure HTTP, S/MIME, use of PGP certificates.	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ Extensive, detailed and critical understanding of the concepts, issues, principles and theories of computer network security ♦ Detailed and practical understanding of formalisms for specifying security related properties and validating them using model checking ♦ Critical theoretical and detailed practical knowledge of a range of computer network security technologies as well as network security tools and services ♦ Practical experience of analysing, designing, implementing and validating solutions to computer network security challenges using common network security tools and formal methods.	
Learning Outcomes: **Personal Abilities**	*Cognitive skills, Core skills and Professional Awareness* ♦ Ability to deal with complex issues and make informed judgements about network security in the absence of complete or consistent data. ♦ Exercise substantial autonomy and initiative in addressing computer network security challenges. ♦ Showing creativity, initiative and team working skills in shared computer network security application development. (PDP)	

	♦ Demonstrate critical reflection on network security issues. (PDP)	
Assessment Methods:	Assessment: Exam: (Weighting 40%) Coursework: (weighting – 60%)	Re-assessment: None

Course Code: **F21DL**	**Course Title:** **Data Mining & Machine Learning**	**Course Co-ordinator:** **David Corne, Philippe De Wilde**
Pre-requisites:	F29AI Artificial Intelligence *or equivalent*	
Aims:	♦ To introduce students to the fundamental concepts & techniques used in machine learning. ♦ To develop a critical awareness of the appropriateness of different methods. ♦ To provide familiarity with common applications such as data mining.	
Syllabus:	♦ **Data Mining:** Basic concepts, data warehousing, statistical data mining, clustering methods, soft computing methods. ♦ **Machine Learning:** Concept learning, decision tree learning, introductory artificial neural networks, Bayesian learning, instance-based learning, introductory evolutionary computing.	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ Extensive understanding of the data mining process. ♦ Detailed understanding of the mathematical basis of machine learning. ♦ Critical awareness of the appropriateness and performance of different techniques.	
Learning Outcomes: **Personal Abilities**	*Cognitive skills, Core skills and Professional Awareness* ♦ Rational problem identification and definition. ♦ Critical analysis and solution selection. ♦ Thorough and robust preparation of testing strategies. ♦ Reflection on system development and performance.	
Assessment Methods:	Assessment: Coursework: (weighting – 100%) (verified by short oral exam)	Re-assessment: None

Course Code: F21DS	Course Title: Distributed Systems Programming	Course Co-ordinator: Hamish Taylor, Andrew Ireland
Pre-requisites:	F29OC Operating Systems & Concurrency and knowledge of elementary logic, operating systems, networks *or equivalent*. In addition, software engineering issues and expertise equivalent to an ordinary degree in Computer Science.	
Aims:	♦ Review the principal abstractions and methods used in distributed systems programming. ♦ Give practical experience in modelling, specifying, implementing and verifying distributed systems. ♦ Enable students to appreciate critically a range of distributed computing technologies. ♦ Have students creatively develop rival types of implementations of a distributed application and appraise the differences.	
Syllabus:	♦ Formal methods for distributed systems development; Formal verification covering safety and liveness; design modelling via the Promela modelling language; SPIN design verification tool; Simulation and formal verification within SPIN; theoretical foundations of LTL model checking. ♦ Processes and threads; sockets and services; remote procedure calls (ONC); network security issues, cryptography – symmetric and public key, certificates, digital signatures and SSL, Remote Method Invocation; CORBA; Web services – SOAP; Distributed computing models, SOA.	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ Extensive, detailed and critical understanding of the modelling, specifying and verification of distributed systems including event and state based approaches, simulation vs formal verification, local and global system assertions, deadlock, live-lock and fairness issues, linear time issues ♦ Critical theoretical and detailed practical knowledge of a range of distributed computing technologies including sockets, RPC, RMI, CORBA and SOAP web services and some related security issues ♦ Practical experience of the formal modelling and analysis of distributed communicating systems.	
Learning Outcomes:: **Personal Abilities**	*Cognitive skills, Core skills and Professional Awareness* ♦ Identify, define new and abstract problems and issues. ♦ Deal with complex issues and make informed judgements in situations in the absence of complete or consistent data. ♦ Exercise substantial autonomy and initiative. ♦ Showing initiative, creativity and team working skills in shared distributed application development. (PDP) ♦ Demonstrate critical reflection. (PDP)	
Assessment Methods:	Assessment: Exam: (weighting – 60%) Coursework: (weighting – 40%)	Re-assessment: None

Course Code: F21IF	Course Title: Information Systems Methodologies	Course Co-ordinator: Jenny Coady
Pre-requisites:	None	
Aims:	**This course explores a range of issues concerning advanced contemporary methodological approaches to information systems development. The aim is to enable students to develop critical faculties and techniques in relation to the selection and application of these methodological approaches.**	
Syllabus:	There is a growing requirement in industry for engineers and scientists with good and appropriate analytical skills when considering the development and evolution of systems, in particular information systems. This course develops further the knowledge and skills students should have already gained in the Information Systems and Software Engineering courses in topics such as: ♦ General Systems Principles; ♦ Systems Classification and Taxonomy Models; ♦ Information Systems Life Cycle and Functions; ♦ Paradigmatic Approach to Methodology Classification; ♦ Framework for Analysis and Comparison of Methodologies (NIMSAD & Fitzgerald's); ♦ Process Improvement Models;	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* This course develops further the knowledge and skills students should have already gained in the Information Systems and Software Engineering courses. It will enable students to: ♦ Determine alternative approached to gathering requirements and systems development ♦ Compare methodologies for use in organisations using a standardised Framework ♦ Rationalise systems development to prepare a more relevant system	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ Critical reading and reviewing works in the field ♦ Evaluating Methods under an agreed Framework ♦ Structuring an argument (PDP) ♦ Presentations of mini lectures to show understanding of the topic area (PDP) ♦ Use of VLE as a means of learning, contributing and discussing	
Assessment Methods:	Assessment: Exam: (weighting – 60%) Coursework: (weighting – 40%)	Re-assessment: None

Course Code: F21MA	Course Title: 3D Modelling and Animation	Course Co-ordinator: Sandy Louchart, Mike Chantler
Pre-requisites:	None	
Aims:	To introduce the basic concepts, techniques and skills of 3D modelling and animation	
Syllabus:	♦ 3D modelling ♦ Basic models ♦ Layering ♦ Polygon reduction ♦ Texturing ♦ Animation ♦ Overview of history and types of animation ♦ Tools and working methods ♦ 12 principles of classic animation ♦ Computer-based animation (CGI) ♦ Creating character – believability and naturalism ♦ Procedural animation: inverse and forward kinematics ♦ Speech and expressive behaviour ♦ Motion capture ♦ Behavioural animation ♦ Emotion and story	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ Critical understanding of the history of animation and types of animation ♦ Critical understanding of the advantages and disadvantages of hand-construction, kinematics, and motion capture in animation ♦ Detailed understanding of the principles of animations. ♦ Ability to research and prototype simple animations ♦ Ability to integrate simple animations into Rich Internet Applications ♦ Basic understanding of the theory of 2D and 3D transformations, projection and viewing. ♦ Detailed knowledge of 3D modelling and rendering techniques. ♦ Ability to understand, design and implement 3D models from a 3D graphic package. ♦ Practical skills in developing 3D content for different types of applications and uses.	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ Ability to think and plan in three dimensions ♦ Technical report writing and organisation ♦ Team working skills ♦ Representation of, planning for, and solution of problems	
Assessment Methods:	Assessment: Coursework: (weighting – 100%) (verified by short oral exam)	Re-assessment: None

Course Code: F21MC	Course Title: Mobile Communications & Programming	Course Co-ordinator: Peter King
Pre-requisites:	F28IT Internet & Communications or knowledge of network communications and object oriented programming	
Aims:	♦ To introduce students to the particular problems of building networks which include mobile computing devices and to explain how they may be overcome using current technology ♦ To introduce students to the issues surrounding ad hoc networking and give an understanding of how these can be addressed ♦ To introduce students to programmable mobile and handheld devices ♦ To develop students' skills in developing applications for mobile and handheld devices	
Syllabus:	Fixed node IP routing-routing techniques for conventional wired networks Mobile IP routing - routing for wireless mobiles to IP Ad hoc networks and routing Security protocols-identification and authorisation, infra structure security Small device characteristics - screen size, memory, power consumption, input mechanisms Current devices - tablet PC, mobile phone, PDA Application development environments - Java APIs, C# and .NET	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ To understand and apply the principles of secure, effective communication over networks including mobile elements. ♦ To be able to explain the operation of current and proposed protocols for communication over networks which include mobile elements ♦ To be able to explain the particular problems in security created by wireless connections in networks and their practical solution. ♦ To understand and be able to explain the issues introduced by ad-hoc networking. ♦ To have detailed knowledge of common ad-hoc routing protocols and to compare these critically ♦ To explain and critically evaluate current and proposed mobile devices ♦ To design applications for mobile devices including use of wireless communications where appropriate. ♦ To program such applications using current application development environments	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ To be able to select and apply suitable techniques of analysis in assessing the effectiveness of a technical solution ♦ To be able to critically review the issues of security and privacy relating to networking ♦ To be able to write good technical documents in support of problem solving within the domains of mobile networking and of mobile and handheld device solutions.	
Assessment Methods:	Assessment: Exam: (weighting – 80%) Coursework: (weighting – 20%)	Re-assessment: None

Course Code: **F21RO**	**Course Title:** **Robotics & Automation**	**Course Co-ordinator:** **Nick Taylor, Patricia Vargas**
Pre-requisites:	F29GR Computer Graphics or *equivalent*, F29AI Artificial Intelligence *or equivalent*	
Aims:	To introduce students to concepts and techniques used in Robotics and applications such as Automation.	
Syllabus:	Manufacturing paradigms. Numerical Control - Adaptive Control, Part programming, APT. Industrial manipulators - Robot Control, Kinematics, Programming. Automated Guided Vehicles - Maps, Path Planning, Navigation. Automation - Organisation, Communication, Sensory devices.	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ To appreciate the basic concepts of automation. ♦ Detailed understanding of the geometries of industrial manipulators. ♦ Detailed understanding of the architectures of autonomous guided vehicles (AGVs). ♦ Detailed understanding of the interfacing and control issues of industrial manipulators and AGVs. ♦ To explore the applications and implications of industrial automation. ♦ To gain practical experience in developing a simulation of an industrial manipulator.	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ To critically analyse various paradigms and architectures. ♦ To appreciate the real-world constraints imposed on technical skills. ♦ To offer professional insights into the financial imperatives which apply to the introduction of new technology.	
Assessment Methods:	Assessment: Exam: (weighting – 80%) Coursework: (weighting – 20%)	Re-assessment: None

Course Code: F21RS	Course Title: Rigorous Methods for Software Engineering	Course Co-ordinator: Andrew Ireland, Lilia Georgieva
Pre-requisites:	None.	
Aims:	To provide knowledge and understanding of tools and techniques which support rigorous software engineering.	
Syllabus:	The course addresses the challenges of developing high quality software. We study behavioural interface specification languages, state of the art specification and verification techniques and tools supporting verification methods. We introduce a general verification framework (states, invariants, operations, Hoare triples, weakest preconditions, patterns for analyzing object-oriented programs). We cover reasoning about programs with loops, reasoning about pointer programs and verification condition generation from code. We also study software processes, standards and quality measures that support the development of high integrity software applications. Approaches to coding that promote the development of high quality software, including advanced static analysis techniques - data flow analysis, information flow analysis, formal verification and abstract interpretation. Novel logics and formal verification techniques that support of computer systems. Evidence based software certification.	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ Detailed and integrated knowledge and understanding of the role of software standards within the development of high quality applications ♦ Critical understanding of the relationship between code level annotations and high-level formal specifications ♦ Extensive knowledge of the mechanisms that underlie advanced static analysis techniques. ♦ Critical understanding of the role of evidence based software certification. ♦ Critical understanding of the trade-offs between complementary approaches to developing high quality software.	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ Conceptualize and define new abstract problems within the context of automated software development. ♦ Identify, define new and abstract problems and issues. ♦ Deal with complex issues and make informed judgements in situations in the absence of complete or consistent data. ♦ Exercise substantial autonomy, initiative and creativity in the application of software engineering techniques. ♦ Demonstrate critical reflection. (PDP) ♦ Communicate with peers, more senior colleagues and specialists. (PDP)	
Assessment Methods:	Assessment: Exam: (weighting – 60%) Coursework: (weighting – 40%)	Re-assessment: None

Course Code: F21WI	Course Title: Web Intelligence	Course Co-ordinators: David Corne, Albert Burger, Lilia Georgieva
Pre-requisites:	F29AI Artificial Intelligence *or equivalent*	
Aims:	This course provides a critical and applied understanding of the technologies and standards enabling reasoning over and intelligent access to information on the World Wide Web, and a critical and applied understanding of the structure and properties of the web as a complex system, and how this impacts on the growth and use of the web.	
Syllabus:	Motivation - need to reason about information and data on Web; Semantic Web; Metadata and Ontologies; XML: Goal: Self-Describing data; XML schema; intro to XML query; Introduction to Description Logic (DL); Resource Description Framework: Syntax and Schema; Ontology Languages: OWL; Ontology Tools: Protégé; Reasoning with Ontologies; Grid Intelligence; Agents and Distributed Search; structure and behaviour of complex networks, graph-theoretic properties of the web, anatomy and operation of search engines, reasoning with *unstructured* web information.	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ Understanding of need for languages enabling inference from information on World Wide Web. ♦ Critical understanding of technologies enabling representation of and reasoning using metadata and ontologies. ♦ Ability to apply knowledge of ontologies to practical problems using current tools. ♦ Critical awareness and understanding of how reasoning may be done over the *unstructured* portion of the web. ♦ Awareness of the web as a complex network, and knowledge and understanding of the properties and behaviours of such complex networks (e.g. small-world properties). ♦ Critical awareness of limitations of existing approaches and of current research directions.	
Learning Outcomes: **Personal Abilities**	*Cognitive skills, Core skills and Professional Awareness* ♦ Skills in selecting, applying and evaluating apt technologies and standards in a professional way. ♦ Ability to build on initial skills and knowledge by independent research using online resources, taking into account continuously evolving professional standards (PDP)**.** ♦ Demonstrate critical reflection (a major coursework essay)	
Assessment Methods:	Assessment: Exam: (weighting – 50%) Coursework: (weighting – 50%)	Re-assessment: None

Computer Science & Software Engineering

Course Descriptors

Year 4, Semester 2

Course Code: F20PB	Course Title: Project: Design & Implementation	Course Co-ordinator: Peter King
Pre-requisites:		
Aims:	Development of project design and implementation skills	
Syllabus:	♦ Software and/or experimental design and its documentation ♦ Relevant commercial practice in applied design of software	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ Software design and implementation skills	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ Time management ♦ Project Management	
Assessment Methods:	Assessment: Coursework (weighting – 100%) Synoptic with F20PA & F20PC	Re-assessment: None

Course Code: F20PC	Course Title: Project: Testing & Presentation	Course Co-ordinator: Peter King
Pre-requisites:		
Aims:	Development of knowledge and skills for testing and evaluating a software project	
Syllabus:	♦ Testing of Software ♦ Evaluation of Software ♦ Report Writing	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ Testing and evaluation of software development projects ♦ Documenting Software projects	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ Awareness and experience of methods and tools for validation and verification in professional practice ♦ Practical skills in testing and evaluation ♦ Documentation skills	
Assessment Methods:	Assessment: Coursework (weighting – 100%) Synoptic with F20PA & F20PB	Re-assessment: None

Course Code: **F21AD**	**Course Title:** **Advanced Interaction Design**	**Course Co-ordinators:** **Judy Robertson, Oliver Lemon**
Pre-requisites:	F28IN Interaction Design *or equivalent*	
Aims:	The course aims to give students the opportunity to develop: ♦ An extensive, detailed and critical knowledge of requirements gathering, design and evaluation techniques in interaction design. ♦ An awareness of current research and emerging issues in the field of interaction design. ♦ A range of specialised skills, and research methods involved in working with users.	
Syllabus:	Current and emerging topics in Interaction Design including: user demographics, patterns in technology adoption, interaction design lifecycles, user interface design patterns, prototyping methods, a wide range of qualitative and quantitative data gathering and analysis techniques, accessibility, and a range of research case studies covering cutting edge issues in the field	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* Students will develop skills in the following areas: ♦ Review, critically analyse, evaluate, and synthesise of previous research projects in the field of interaction design ♦ Identify and propose innovative solutions in response to analysis of users' requirements. ♦ Make informed judgements about appropriate methodologies for developing and evaluating technologies suitable for user demographics and background experience.	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* Students will develop skills in the following areas: ♦ Use discipline appropriate software for data analysis, prototyping and learning. ♦ Present, analyse and interpret numerical and graphical data gathered as part of evaluation studies. ♦ Communicate effectively to knowledgeable audiences by preparing formal and informal presentations and written reports. ♦ Exercise autonomy and initiative by planning and managing their own work; develop strategies for independently solving problems and taking the initiative. ♦ Take responsibility for their own and other's work by contributing effectively and conscientiously to the work of a group, actively maintaining good working relationships with group members, and leading the direction of the group where appropriate. ♦ Reflect on roles and responsibilities by critically reflecting on their own and others' roles and responsibilities. ♦ Deal with complex professional and ethical issues including working with human subjects and wider issues relating to technology in society	
Assessment Methods:	Assessment: Exam: (weighting – 60%) Coursework: (weighting – 40%)	Re-assessment: None

Course Code: F21BC	Course Title: Biologically Inspired Computation	Course Co-ordinator: David Corne, Patricia Vargas, Pier Frisco
Pre-requisites:	F29AI Artificial Intelligence and Intelligent Agents *or equivalent* F27SB Software Development 2 *or equivalent involving programming any language*	
Aims:	Traditional computation finds it either difficult or impossible to perform a certain key range of tasks associated with pattern recognition, problem solving and autonomous intelligence. Great progress towards designing software for such tasks has emerged by taking inspiration from a range of natural, mainly biological, systems. The aims of this course are to: ♦ introduce an appreciation of the former ♦ introduce the main biologically-inspired algorithms and techniques which are now commonly researched and applied ♦ establish a practical understanding of the real-world problems to which these techniques may be fruitfully be applied.	
Syllabus:	♦ Classical vs. Biologically-inspired computation, ♦ Evolutionary algorithms (genetic programming, genetic algorithms, evolutionary programming, evolution strategies, memetic algorithms) ♦ Swarm intelligence (ant colony methods, particle swarm optimisation) ♦ Neural computation (perceptrons, multilayer perceptrons, associate networks, self-organising networks, spiking neural networks) ♦ Artificial life, artificial chemistries, cellular automata ♦ Membrane computing, artificial immune systems	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ Understanding of limitations of traditional computation. ♦ A critical understanding of a range of biologically inspired computation methods, their limitations and areas of applicability. ♦ Ability to apply one or more biologically inspired techniques in solving a practical problem.	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ Identify and define approaches that can be used to apply bio-inspired methods to existing problems in optimisation and machine learning. ♦ Exercise substantial autonomy and initiative (both courseworks) (PDP) ♦ Demonstrate critical reflection (both courseworks) (PDP).	
Assessment Methods:	Assessment: Exam: (weighting – 50%) Coursework: (weighting – 50%)	Re-assessment: None

Course Code: **F21DP**	**Course Title:** **Distributed & Parallel Technologies**	**Course Co-ordinator:** **Phil Trinder, Hans Wolfgang Loidl**
Pre-requisites:	F29OC Operating Systems & Concurrency *or equivalent*	
Aims:	To explore technologies and techniques underlying advanced distributed and parallel software development	
Syllabus:	Distributed Technologies: Distribution concepts. low-level, mid-level and high-level distributed technologies. Emerging distribution technologies. Parallel Technologies: Design of parallel systems, parallel performance analysis a) Practical imperative parallel programming b) Practical declarative - parallel programming	
Learning Outcomes:: **Personal Abilities:**	*Understanding, Knowledge and Subject-Specific Skills* ♦ Understanding of foundational concepts of distributed and parallel software ♦ Knowledge and application of contemporary techniques for constructing practical distributed and parallel systems using both declarative and imperative languages ♦ Parallel performance tuning using appropriate tools and methodologies ♦ Understand the role of control and data abstraction in software design and implementation ♦ Appreciation of relationship between imperative and declarative models of parallelism	
Learning Outcomes: **Subject Mastery**	*Cognitive skills, Core skills and Professional Awareness* ♦ Critically analyze parallel and distributed problems. ♦ Generate, interpret and evaluate parallel performance graphs ♦ Develop original and creative parallel problem solutions ♦ Work effectively with peers ♦ Demonstrate critical reflection, e.g. understanding of applicability of, and limitations to, parallel and distributed systems	
Assessment Methods:	Assessment: Exam: (weighting – 80%) Coursework: (weighting – 20%)	Re-assessment: None

Course Code: F21EC	Course Title: E-Commerce Technology	Course Co-ordinator: Roger Rist, Brian Palmer
Pre-requisites:		
Aims:	♦ To review the IT issues raised by electronic business and commerce; ♦ To survey the techniques and technologies available for designing and implementing e-business and e-commerce applications; ♦ To provide first-hand experience of Web-based tools and services to help design e-commerce solutions.	
Syllabus:	♦ *Business Models* - Storefronts and Malls, Auctions, Portal and Community Sites, Dynamic Pricing, Comparison Shopping, Demand Aggregation, Barter, The Virtual Organisation The Click-and-Mortar Model, Application Service Providers, Extranets, B2B Trading Hubs. ♦ *Marketing* - The 4 Ps of Marketing, Email Marketing, Promotions, Banner Adverts, Public Relations, Trust, CRM, Indexes and Portals, Partnerships, Globalisation, Sticky Sites. ♦ *Payment* - Off-line and Online Payment, The Online Credit/Debit Card Process, EDI and EFT, e-Wallets, e-cash, P2P Payments, e-Checks, Smart Cards, B2B transactions, e-Bills, e-Banking. ♦ *Security* - Security per se, e-Security Issues, Encryption, Secret & Public Key Cryptography, Digital Envelopes and Signatures, Hash Functions, Timestamping, Digital Certificates and Certification Authorities, PKI, SSL, SET, Firewalls, Directory Access Control. ♦ *Law & Ethics* - Levels of Service, Privacy, Discrimination, Advertising, Information, Limiting Liability, The Contract, Outsourcing, e-Ethics. ♦ *Architecture* - Network Architectures, Web Site Meta-Architecture, The Web Server, The Proxy Server, TCP/IP, IP Addresses, DNS, Capacity Planning. ♦ *Design* - Structuring the Site, Structuring the Page, Navigation, Error Messages, Trustworthiness, Accessibility, Validation and Testing. ♦ *Languages* - HTML, CSS, PHP, ASP, JavaScript, VBScript, Java Applets & Servlets, JDBC, Cookies, XML ♦ *Methodologies* - The System Development Life Cycle, Rapid Application Development, Alternative Methodologies. ♦ *Web Designing* - Using Open Source Technologies	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ Demonstrate extensive knowledge of various e-Business models. ♦ Understand the significant issues in online marketing. ♦ Demonstrate an awareness of current and emerging alternatives for online payment. ♦ Critically evaluate the concepts important to online security. ♦ Discuss the legal and ethical dimensions of e-Commerce and their implications. ♦ Describe and critically review issues, technologies and concepts in the architecture of e-Commerce solutions. ♦ Explain and critically appraise current approaches to Web site design. ♦ Apply and critically discuss the various programming languages related to the Web. ♦ Describe the evolving methodological issues pertaining to e-Commerce system development. ♦ Good knowledge of appropriate development tools and the capability to use them to create applications	

Learning Outcomes:: Personal Abilities	*Cognitive skills, Core skills and Professional Awareness* ♦ Critically evaluate the search for, and appraisal of, complex, ambiguous and unreliable resources. ♦ Analyse, take responsibility for and reflect on personal and organisational practice. ♦ Develop original and creative solutions to, and judgements on, open-ended problems	
Assessment Methods:	Assessment: Exam: (weighting – 50%) Coursework: (weighting – 50%)	Re-assessment: None

Course Code: F21GP	Course Title: Computer Games Programming	Course Co-ordinator: Peter King, Sandy Louchart
Pre-requisites:	C++ programming skills	
Aims:	To develop programming skills and techniques specific to the area of 2D and 3D computer games	
Syllabus:	♦ History and types of computer games ♦ Elements of game design ♦ Game-state, simulator, renderer, (hierarchical) controllers ♦ Tools and environments – e.g. Flash, games engines ♦ 2D games programming techniques ♦ Physically-based modelling, particle systems, flocking ♦ Use of physics engines ♦ Obstacle avoidance and path planning ♦ Group movement ♦ Learning and adaptation in games ♦ Action and behaviour selection ♦ Game theory and games ♦ Course summary and review	
Learning Outcomes: Subject Mastery	*Understanding, Knowledge and Subject-Specific Skills* ♦ Critical understanding of game theory and computer games history, genres and impact ♦ Critical understanding of available tools and their application ♦ Detailed knowledge of algorithms for particle systems and flocking ♦ Detailed knowledge of algorithms for path planning and navigation ♦ Broad knowledge of physically-based modelling in games and selection of techniques ♦ Broad knowledge of AI techniques in games and selection of techniques ♦ Ability to understand, design and implement a small-scale game using 2D and 3D tools ♦ Practical skills in graphics and AI programming in the computer games context	
Learning Outcomes: Personal Abilities	*Cognitive skills, Core skills and Professional Awareness* ♦ Ability to think and plan in three dimensions ♦ Technical report writing and organisation ♦ Team working skills ♦ Representation of, planning for, and solution of problems	
Assessment Methods:	Assessment: Exam: (weighting – 65%) Coursework: (weighting – 35%)	Re-assessment: None

Course Code: F21IE	Course Title: Internet Engineering	Course Co-ordinator: Peter King
Pre-requisites:	F28IT Internet & Communications, F29OC Operating Systems & Concurrency *or equivalent*	
Aims:	Investigation of advanced topics in communications networks	
Syllabus:	♦ Internet organisation ♦ Routing protocols ♦ Distance Vector (RIP) ♦ Link state (OSPF) ♦ Congestion control (RED), fair queueing ♦ Transport protocols ♦ TCP – slow start, Renoe, Tahoe, etc ♦ Real time Protocols ♦ Security & Management ♦ VPN & SNMP ♦ Firewalls ♦ LANs & Bridges	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ Detailed knowledge of Internet routing and transport protocols ♦ Understanding of security and network management problems and solutions	
Learning Outcomes: **Personal Abilities**	*Cognitive skills, Core skills and Professional Awareness* ♦ Appreciation of modern networking practices ♦ Critically review proposed solutions to network routing and congestion problems ♦ Evaluate network security proposals	
Assessment Methods:	Assessment: Exam: (weighting – 75%) Coursework: (weighting – 25%)	Re-assessment: None

Course Code: F21NA	Course Title: Network Applications	Course Co-ordinator: Hamish Taylor
Pre-requisites:	Either F28IT Internet & Communications and F27SB Software Development 2 or reasonable software development skills in Java and basic knowledge of data communications and the web.	
Aims:	♦ To equip students with knowledge and understanding of the theories, principles and protocols underlying network applications on the Internet ♦ To enable students to appreciate critically the range of network application technologies and standards ♦ To give students significant development skills in a range of the principal network technologies, to grasp the main design and practical issues faced in their application, and confer the ability to select and apply relevant techniques for a given network application development problem. ♦ To have students creatively develop in teams a substantial network application involving web and application server technologies to an original design of their own.	
Syllabus:	Network application fundamentals, IPC via sockets, programming simple services, network information services. Network security issues, cryptography – symmetric and public key, certificates, digital signatures and SSL. Email protocols and formats - SMTP, POP, IMAP, RFC 2822, MIME. Nature of web – URIs and HTTP, web markup languages - (X)HTML, web design issues, CSS, XML, DOM. Client-side web programming - JavaScript, DHTML, AJAX, plugins, applets. Server side web programming – CGI, servlets, active web server pages – SSI, JSP, PHP. Web mediated database access – JDBC, PHP. Web security – HTTP authentication, HTTPS, cookies. Web services - SOAP and REST. Other styles of network applications – textual conferencing. Distributed service models - client server, P2P, publish & subscribe.	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ Extensive, detailed and critical knowledge and understanding of the theories, techniques and principles underlying the design of network applications and the range of their application ♦ Theoretical and practical knowledge of the major network application types including email, web applications and services, IRC, streaming media ♦ Critical awareness of protocols and standards underlying key network applications especially the web and of enabling technologies for network applications such as sockets, DNS, XML ♦ Ability to design and develop useful network applications including WWW applications using apt technologies and languages: HTML, XML, JavaScript, Java applets, CGI, servlets, active web server pages, SOAP services etc. to professional standards	
Learning Outcomes: **Personal Abilities**	*Cognitive skills, Core skills and Professional Awareness* ♦ Skills in selecting, applying and evaluating apt technologies in a professional way given a problem requiring network interaction ♦ Ability to build on initial skills and knowledge by independent research using online resources ♦ Showing initiative, creativity and team working skills in shared network application development	
Assessment Methods:	Assessment: Exam: (weighting – 70%) Coursework: (weighting – 30%)	Re-assessment: None

Course Code: F21SI	Course Title: Software Simulation & Modelling	Course Co-ordinators: Pier Frisco
Pre-requisites:	Knowledge of fundamentals of Web technology, Java programming, elementary logic	
Aims:	♦ Deep understanding of the problem issues arising from software simulations and modelling. ♦ Knowledge of different tools and techniques to tackle the problems present in software simulations and knowledge of different modelling platforms. ♦ Capacity to use different modelling platforms on a broad range of biological phenomena.	
Syllabus:	♦ Advanced Description Logic; Resource Description Framework: Syntax and Schema; Ontology Languages: OWL; Ontology Tools: Protégé; Reasoning with Ontologies. ♦ Concepts in modelling biological systems: use of scientific models, differences between quantitative and qualitative modelling, mistakes computers make, model validation, stochastic models, (advantages and disadvantages in) Forrester diagrams, parameter estimation, classifications of models, meta-models, model analysis, random numbers, sampling strategies, different levels of abstraction, scaling models. ♦ Modelling platforms: Petri nets, Markov chains, (advanced) cellular automata, ODE, P systems, analysis of biological (and not) networks. ♦ Practical examples: sub-cellular level, cellular level (populations of cells), host infection (HIV, etc.), social networks, spatial patterns, etc.	
Learning Outcomes: Subject Mastery	*Understanding, Knowledge and Subject-Specific Skills* ♦ Deep understanding of the problem issues arising from software simulation and modelling; ♦ Deep knowledge of different modelling tools and platforms: description logic, ontology languages, Petri nets, cellular automata, biological (and not) networks, etc. ♦ Capability to identify the essential hypotheses needed to model specific processes, choice of an appropriate software tool/methodology to model specific processes, choice of the proper level of abstraction, validation of the simulations, understanding and analysis of the results.	
Learning Outcomes: Personal Abilities	*Cognitive skills, Core skills and Professional Awareness* ♦ Up to date knowledge of the most currently used tools and methodologies to model processes. ♦ Capability to assess the potential, limits and results of software simulations. ♦ Critically evaluate the search for, and appraisal of, complex processes. ♦ Analyse, take responsibility for and reflect on personal and organisational practice. ♦ Develop original and creative solutions to, and judgements on, open-ended problems.	
Assessment Methods:	Assessment: Coursework: (weighting – 100%) Assessment comprises computer tests and 3 pieces of coursework	Re-Assessment None

Course Code: F21VE	Course Title: Virtual Environments	Course Co-ordinators: Ruth Aylett
Pre-requisites:	Elementary C++ programming equivalent to F29GR Computer Graphics	
Aims:	♦ To enable participants to understand the concepts and benefits of Virtual Environments (VEs) with respect to various applications. ♦ To equip participants with the skills to create a skeleton Virtual Environment using state-of-the-art VE software toolkits	
Syllabus:	♦ Introduction: History of VEs ♦ What a VE is not.; concepts of immersion and presence, RT constraints ♦ Overview of current VE applications ♦ Basic Types and Components of VEs (graphics hardware, displays, interaction devices, software,) ♦ Modelling – low polygon, standards, mechanisms ♦ Construction of models ♦ Physically-based modelling ♦ Web-based 3D ♦ Agents and avatars ♦ Distributed VEs ♦ Construction of VEs and future of VEs ♦ Creation of small VE ♦ Course summary and review	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ Be able to critically evaluate the strengths and weaknesses of current VR technologies ♦ Detailed understanding of the main components of a virtual reality system and the importance and impact of real-time constraints ♦ Detailed understanding of modelling approaches and their uses ♦ Critical understanding of the state-of-the-art in VE application domains ♦ Ability to apply appropriate display and interaction capabilities to specific VR applications and justify choices made ♦ Able to apply basic VE construction skills to the creation of small-scale systems	
Learning Outcomes: **Personal Abilities**	*Cognitive skills, Core skills and Professional Awareness* ♦ Taking responsibility for own work, taking responsibility in the development of resources, critical reflection on development process and work undertaken by self. ♦ Effective communication in electronic and written report form. ♦ Showing initiative, creativity and team working skills in virtual environment development	
Assessment Methods:	Assessment: Exam: (weighting – 70%) Coursework: (individual project) (weighting – 30%)	Re-Assessment None

Software Engineering
Course Descriptors

Year 4, Semester 1

Course Code: B81NP	Course Title: New Product & Process Development	Course Co-ordinator: To be confirmed
Pre-requisites:		
Aims:	♦ To introduce concepts and practices in product development and entrepreneurship, from generation of an idea, to (basic) business planning, through to the infrastructure of support that exists in the UK. ♦ To enhance student understanding of what comprises enterprise activity particularly in an engineering context. ♦ To enhance student understanding of what comprises entrepreneurial behaviour with an emphasis on engineering application. ♦ To enhance students' knowledge, understanding and awareness of the breadth of entrepreneurial activity - including the family business, public sector, funding, interfacing with other organisations and intrapreneurship. ♦ To raise student awareness of enterprise/entrepreneurship, business planning and company organisation in targeted product and process development group projects within engineering disciplines. ♦ To increase student knowledge about enterprise skills application within start-up companies and SMEs. ♦ To examine the impact that enterprise activities have on the community. ♦ To examine the role of innovation in economic and social development. ♦ To increase student knowledge about applications of enterprise skills within company start-ups and other contexts.	
Syllabus:	Ideas and idea generation; 'The Entrepreneur - personality, drive and determination; ;SMEs, innovation and intellectual property; product and process development; evaluating alternatives; business planning processes; start-up funding and company finance; patenting; company structures and law; issues in 'Management and Leadership'; 'supporting enterprise'; alternative strategies in entrepreneurship and understanding of a wide range of engineering materials and components in this topic context.	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ Critically and effectively analyse selected theories of product and process development and entrepreneurship. ♦ Demonstrate an understanding of the theories of entrepreneurship and innovation through the application of principles and procedures appropriate to a range of situations around the creation of a new product and associated business venture. ♦ Demonstrate critical evaluation of a case study scenario, involving analysis, synthesis and reflection of outcomes. ♦ Demonstrate knowledge of the importance of enterprise activity in the modern world and working in teams. ♦ Critically evaluate concepts and principles of entrepreneurship and enterprise application and development ♦ Undertake critical analysis of an advanced topic as part of a working group. ♦ Understanding of concepts from a range of areas in product and process development, including some outside engineering and relating to entrepreneurship and business, and the ability to apply them effectively in engineering projects. ♦ The ability to use fundamental knowledge to investigate new and emerging	

	technologies in the product development/new business environment. ♦ Ability to extract data pertinent to an unfamiliar problem covering a range of issues, ♦ Gain a wide knowledge and comprehensive understanding of design processes and methodologies and the ability to apply and adapt them in an unfamiliar business context. ♦ Generate an innovative design for systems, components or processes to fulfil new needs. ♦ Generate ideas for new products and develop and evaluate a range of new solutions in a financial and business context. ♦ Make general evaluations of commercial risks through some understanding of the basis of such risks with respect to new product development. ♦ Have extensive knowledge and understanding of management and business practices, and their limitations, and how these may be applied appropriately to strategic and tactical issues in new product and process development. ♦ Gain a thorough understanding of current practice and its limitations and some appreciation of likely new developments in this domain. ♦ Acquire extensive knowledge and understanding of a wide range of engineering materials and components in this topic context.	
Learning Outcomes:: Personal Abilities:	*Cognitive skills, Core skills and Professional Awareness* ♦ Work productively in small teams, interacting effectively within the teams while displaying leadership and group skills to appropriate standards. ♦ Critically review, research and develop informed alternatives to given problems. ♦ Demonstrate some originality and creativity in dealing with issues in enterprise, business and associated engineering activities. ♦ Communicate to an audience, findings from research and analysis. ♦ The ability to apply engineering techniques taking account of a range of commercial and industrial constraints with the ability to integrate knowledge and understanding of mathematics, science, ICT, design, the economic, social and environmental context and engineering practice to solve a product development/business centred problem through involvement in group design projects. ♦ The ability to learn new theories, concepts, methods etc in unfamiliar (to them) situations which combine product development, roles in start-up companies, company funding, business planning and entrepreneurship. ♦ The capability to develop, monitor and update a plan, to reflect a changing operating environment ♦ Develop an understanding of different roles within a team and the ability to exercise leadership ♦ The ability to monitor and adjust a personal programme of work on an ongoing basis and learn independently as part of a team with specific responsibilities. ♦ To product formal presentations and reports at a standard appropriate to a Masters' level course.	
Assessment Methods:	Assessment: Coursework: (weighting – 100%)	Re-assessment: None

Software Engineering

Course Descriptors

Year 5, Semester 1

Course Code: F21IA	Course Title: Industrial Placement 1	Course Co-ordinator: Phil Trinder
Pre-requisites:		
Aims:	To prepare for industrial project	
Syllabus:	Industrial project identification (evaluation, critical assessment, scheduling, planning, requirements engineering, specification, risk assessment)	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ To be able to plan a significant project ♦ To understand the time and effort involved in planning of an industrially-based project	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ To be aware of distinctive features of industrial placement. ♦ To be able to identify, define, and analyse alternative project scenarios ♦ Take significant responsibility for their work and for a range of resources	
Assessment Methods:	Assessment: Coursework: (weighting – 100%) Synoptic with F21IB, F21IC, F21ID	Re-assessment: None

Course Code: F21IB	Course Title: Industrial Placement 2	Course Co-ordinator: Phil Trinder
Pre-requisites:		
Aims:	To develop an industrial project	
Syllabus:	Industrial Project Development (design, evaluation, critical assessment, software engineering, refinement)	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ To be able to undertake a significant industrial project ♦ To understand challenges involved in development of an industrially-based project	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ To communicate effectively with professional peers and managers ♦ To work effectively under guidance in a peer relationship with qualified practitioners. ♦ To be able to make judgements using information from a range of sources when evaluating alternative development scenarios ♦ To be aware of distinctive features of industrial projects	
Assessment Methods:	Assessment: Coursework: (weighting – 100%) Synoptic with F21IA, F21IC, F21ID	Re-assessment: None

Course Code: F21IC	Course Title: Industrial Placement Monthly Reports	Course Co-ordinator: Phil Trinder
Pre-requisites:		
Aims:	To record experiences and critical reflections	
Syllabus:	Industrial Project Reporting (Log book, technical authoring, monthly reports)	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ To understand the significance of monitoring and control within an industrially-based project	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ To be able to communicate with professional level peers, senior colleagues and specialists. ♦ To be aware of distinctive features of industrial reporting ♦ To be able to maintain logbook (normally retained on company premises for confidentiality) and submit reports	
Assessment Methods:	Assessment: Coursework: (weighting – 100%) Synoptic with F21IA, F21IB, F21ID	Re-assessment: None

Course Code: F21ID	Course Title: Industrial Placement Final Report	Course Co-ordinator: Phil Trinder
Pre-requisites:		
Aims:	To gain experience in the writing of a technical industrially-based report	
Syllabus:	Technical reporting including technical authoring, final report (dependant on industrial placement)	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ Ability to present technical and project materials clearly in writing, including selecting appropriate document structures and visual aids: tables, diagrams etc.	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ To be able to critically review and consolidate knowledge, skills and practices. ♦ To be aware of distinctive features of industrial reporting ♦ To be able to deliver a final report bringing together all aspects of the industrial placement in a concise manner	
Assessment Methods:	Assessment: Coursework: (weighting – 100%) Synoptic with F21IA, F21IB, F21IC	Re-assessment: None

Software Engineering
Course Descriptors

Year 5, Semester 2

Course Code: F21DG	Course Title: Design & Code Group Project	Course Co-ordinator: Peter King
Pre-requisites:		
Aims:	To develop technical and communication skills	
Syllabus:	Research- associated design and code project (specification, design, development, implementation, evaluation, monitoring and maintenance.	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* To understand the planning, development, control and monitoring associated with university research endeavours.	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ To be aware of technical issues and communication skills in interfacing with (software engineering aspects of) research group activities ♦ To be able to participate in regular progress meetings and results reporting	
Assessment Methods:	Assessment: Coursework (weighting – 100%)	Re-assessment: None

Course Code: F21SM	Course Title: Software Engineering Master Class	Course Co-ordinator: Peter King
Pre-requisites:		
Aims:	To introduce students to the cutting edge of research in their field, using the guidance and expertise of active research groups. To provide students with an opportunity to create and deliver a master-class on a topic to their peers	
Syllabus:	Investigate a topic proposed and supervised by an academic. Develop training/teaching materials (lectures/labs/etc) Self study	
Learning Outcomes: **Subject Mastery**	*Understanding, Knowledge and Subject-Specific Skills* ♦ Demonstrate advanced, critical knowledge of a specialist area of software engineering/computer science. ♦ Apply appropriate technologies to develop and deliver learning materials on this topic. ♦ Demonstrate an awareness of current and emerging applications of, and alternatives to, the chosen topic. ♦ Provide appropriate answers to questions posed by peers on the chosen topic. ♦ Critically reflect on feedback provided by peers on the delivered learning materials.	
Learning Outcomes:: **Personal Abilities:**	*Cognitive skills, Core skills and Professional Awareness* ♦ Critically evaluate, review, analyse and organise complex, ambiguous and unreliable information sources. ♦ Develop original and creative solutions to, and judgements on, open-ended problems. ♦ Make presentations of complex material to professional audiences	
Assessment Methods:	Assessment: Coursework (weighting – 100%)	Re-assessment: None