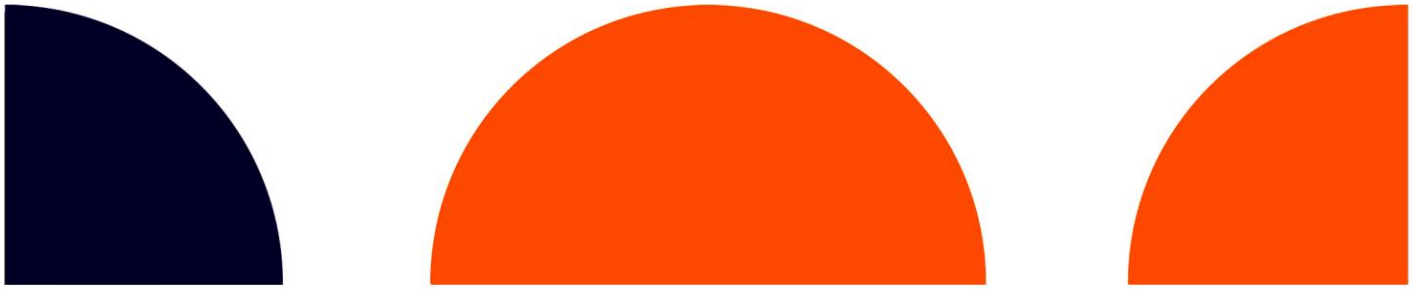
CENTRALSUSSEXCOLLEGE

University of Sussex BSc Degrees in Computing:

Foundation Year

STUDENT HANDBOOK
2012/2013



Contents

Course Introduction

The purpose of the Foundation Year is to enable candidates, with suitable experience or qualifications, to prepare for entry to Year 1 of the BSc Degrees in Computer Science and Artificial Intelligence at the University of Sussex. Requirements for the course are lower than those for direct entry to Year 1, but all candidates are expected to provide some evidence of personal development or academic ability.

Students are enrolled as students at Central Sussex College, Crawley Campus (see Map below), and as undergraduates of Sussex University. There is close liaison with the university, with students attending open day events at Sussex, informal meetings with previous foundation year students, and visits to Central Sussex College by a representative from the School of Informatics. Students receive advice on progression to the university and the selection of degree choices open to them.

The year consists of 2 x 17 week semesters with continuous assessment throughout and some exam based assessment at the end of each semester. Details of assessment are included with unit specifications.

Term dates for 2012/13 are:

Autumn Term:	18/09/2012	-	21/12/2012
Spring Term:	07/01/2013	-	28/03/2013
Summer Term:	15/04/2013	-	28/06/2013
Half-term holidays	29/10/2012	-	02/11/2012
	18/02/2013	-	22/02/2013
	27/05/2013	-	31/05/2013
Resit Week: (provisional dates)	19/08/2013	-	23/08/2013

UNIVERSITY OF SUSSEX DEGREE IN COMPUTING:
FOUNDATION YEAR AT CENTAL SUSSEX COLLEGE

Student Accommodation

Although Central Sussex College does not have Halls of Residence, we can offer rooms in the Crawley area within family homes on either a full-board or self-catering basis, subject to availability.

For further details of this service or for help finding alternative accommodation please contact Student Support Department on 01293 442269 or email rclements@centralsussex.ac.uk

Course Content

The course consists of 10 units of study, **all of which must be passed** to progress to Year 1 of the course at Sussex University. Each unit has equal weighting of 12 points.

Units

1 Personal & Communications Skills
2 Symbolic Reasoning
3 Databases & Application Development
4 Discrete Systems
5 Foundation Programming
6 Systems Analysis
7 Introduction to Artificial Intelligence
8 Introduction to Interactive Web Design
9 Maths for Computing
10 Program Development

Units 1 and 2 will be delivered throughout the year.
Units 3, 4, 5 and 6 will be delivered in Semester 1.
Units 7, 8, 9 and 10 will be delivered in Semester 2.

Unit 10, Program Development, builds on and develops material covered in units 5 and 6. Unit 9, Maths for Computing, is designed to be part of a natural progression from unit 4 and will build on material covered in that unit. Unit 3, Databases & Application Development builds on some of the analysis concepts discussed in unit 6.

Progression

All units on the course are assessed and **all units must be passed** to progress to Year 1 of the degree at Sussex.

Attendance to all lectures is mandatory. If a student's attendance falls below 90% the college's disciplinary procedure will be applied and this could lead to a student being withdrawn from the course.

At the end of Semester 2, students will be awarded final grades in all units. The pass mark for each unit is 40%. If a unit is failed, students will be offered the chance of a resit. The grade for a resit is capped at a bare pass for the unit (i.e. at 40%). All final decisions on grades, and any decisions on mitigating circumstances, will be made at the main Examination Board which is held at the end of Semester 2. This board is Chaired by a representative of the School of Informatics from the University of Sussex, and will record students degree choices for year 1. The board will also make decisions about the timing and conduct of any resits.
Students will receive a transcript of results after the exam board has completed.

Assessment

Units are assessed by a variety of methods and each submission is awarded a percentage mark. Final grades for each unit are presented as percentages calculated according to the stated weighting of each component.

Penalties:

- **Late submission of coursework.** Where a unit assessment is based on assignments, all assignments will be issued with a hand-in time. All work must be submitted by the hand-in time. The following scale applies to all work handed in late:

up to 24 hours late:	loss of 5% of marks
up to 1 week late:	loss of 10% of marks
up to cut off date on assessment front sheet:	mark capped at 40%
after the cut off date:	fail with zero mark.

  If a student is unable to submit an assessed component on time, through no fault of his/her own, they may submit mitigating circumstances, in confidence, to be considered by Sussex University.

- **Plagiarism.** "The issue of plagiarism is a growing academic concern, and one that the University of Sussex takes seriously." See:
  http://www.sussex.ac.uk/academicoffice/1-4-1.html
  Where an assessor suspects that collusion or plagiarism has taken place the matter follows Sussex University procedure.

- **Other Misdemeanours.** In general academic misdemeanours are subject to Sussex University rules and procedures, whereas day-to-day behaviour is subject to Central Sussex College rules and procedures.

UNIT TITLE	**PERSONAL & COMMUNICATION SKILLS**
DURATION	34 weeks
AIMS	To assist the student in developing a range of communication, personal organisation and learning skills to enable them to pursue a Computing degree course at University of Sussex, and a career as a computing professional.
LEARNING OUTCOMES	On completion of this course the student should • be confident in the use of current technology for oral and written presentation of work • be able to present information in a clear and structured fashion • produce reports containing technical & statistical material in a professional manner • be familiar with techniques for conducting, evaluating and reporting on research • have demonstrated successful personal time-management skills in attendance and meeting deadlines • have collaborated effectively with others to plan, implement and evaluate a group project • have an awareness of their own approach to study and how to improve their learning experience
CONTENT	• Study skills & approaches to learning • Learning through activity – structure of the course • Use of Word, Excel, PowerPoint • Giving presentations • Portfolio preparation • Project planning: work breakdown, time management • Writing skills: report structure, presenting technical information • Information sources: books, libraries, the Web • Using the internet for effective research • Referencing and citing sources • Software documentation for developers • Computing industry: legal and ethical issues • Team working: roles and process • Reflection on learning

ASSESSMENT	**Coursework 100%** **Portfolio (weighted 80%):** Students will compile a portfolio of evidence to meet the learning outcomes specified. Much of this evidence can be provided through the tasks required for other modules on the course, students will review their portfolio with their tutor on 3 scheduled dates through the year & will be expected to use this review to discuss their learning needs and progress on the course. **Integrated Group Project (weighted 20%):** This end-of-year project gives students the opportunity to bring together many skills which have been acquired and improved throughout the year.
LEARNING RESOURCES	Students may use a variety of learning resources to support learning on this module. These will include: books, magazines, the internet, lecture notes and the college Virtual Learning Environment.
TEACHING & LEARNING ACTIVITIES	These will include: lectures, tutorials, group discussions, peer tutoring, peer assessment, feedback sessions, simulation, group work, workshops, portfolio reviews.
TUTORS	Ramiz Alihodzic

UNIT TITLE	**SYMBOLIC REASONING**
DURATION	34 weeks
AIMS	• To enable analytical skills to be given expression in other areas of this programme and, more generally, on the Informatics degree-courses at the University of Sussex. • Have the confidence to embrace the mathematical concepts introduced in academic papers. • Have the foundations for further study in areas of interest during the Degree course.
LEARNING OUTCOMES	On completion of this course the student should: • Understand mathematical notation and the functionality of symbolic expressions. • Have a sound foundation in symbolic manipulation and analysis. • Recognise patterns in number sequences and use symbolic reasoning to obtain an explicit formula for the general term. • Be able to construct recurrence relations for a sequence. • Be able to deduce the probabilities of various outcomes. • Produce reports containing symbolic material in a professional manner • Have an understanding of some computational applications of the course material.

CONTENT	Includes: • Symbols as variable names • Symbols as representing relationships • Equations. • Differential calculus. • Binomial theorem. • Expression handling. • Graphs. • Introduction to linear programming. • Indices. • Functions. • Introduction to linear algebra. • Logarithms. • Permutations and combinations. • Series. • Sigma notation.
ASSESSMENT	**Coursework – 50%** **Exams – 50%** Each of the two semesters has an examination at the end during a week that is dedicated to examinations (weighted 50% of the marks for each semester) Two assignments are set at convenient times during each semester (weighted 50% of the marks for each semester).
LEARNING RESOURCES	Students may use a variety of learning resources to support learning on this module. These will include: • Lecture notes. • The college VLE • The Internet. • Open-access computer facilities. • The college library • and most importantly – other students.
TEACHING & LEARNING ACTIVITIES	Two sessions per week. These will usually include lectures, with worked examples, and some exercises. Peer-assisted tutoring will be encouraged in the second weekly session.
TUTORS	Ian Newman

UNIT TITLE	**DATABASES & APPLICATION DEVELOPMENT**
DURATION	17 weeks – Semester 2
AIMS	To provide the student with: (i) an understanding of relational data analysis and database design (ii) skills in manipulating a contemporary database tool to meet user requirements (iii) ability to connect to a database server. (iv) Use a structured query language to extract and manipulate data.
LEARNING OUTCOMES	On completion of this course the student should be able to: • appreciate the need to structure and manipulate large amounts of data • understand relational database concepts • use relational data analysis to design a small database • use an appropriate database tool to build a relational database • use query tools and standard query language to interrogate a database and extract meaningful data • assess user needs and design screen-based forms and reports to meet those needs • construct a small application based on a relational database, and provide basic user documentation • apply problem-solving skills to solving a database requirement • Use SQL and PHP in a web page to extract and update data on a server.

CONTENT	• structuring and storing large amounts of data • data analysis: entities, attributes and relationships • relational data analysis (up to 3NF) • data validation and constraints • building a database • database views and forms • assessing user needs: functionality and usability • using queries to extract and update data • SQL • presenting information to the user: forms and reports • problem-solving and designing applications • testing and verification • documenting the application for the user
ASSESSMENT	**Coursework 100%** Assessment will be by way of : Two phase tests: 40% Assignment : 60%
LEARNING RESOURCES	Students will use a standard small database tool (currently Microsoft Access) to implement database solutions. Students may also find the following texts useful: • The Relational Database, John Carter, Thompson Computing Press • Access 2000 Further Skills, Coles & Rowley, Letts • Database Design & Management using Access, Nick Dowling, Letts • Database Systems – a practical approach to design, implementation and management, Connolly & Begg, Addison-Wesley
TEACHING & LEARNING ACTIVITIES	These will include lectures, tutorials and workshops. All sessions will take place in a computer room. Group discussions, problem-solving sessions and peer tutoring will be used on an informal basis. The group integrated project will provide students with an opportunity to use problem-solving skills to design a database as part of a larger application.
TUTOR	Ramiz Alihodzic

UNIT TITLE	**DISCRETE SYSTEMS**
DURATION	17 weeks – Semester 1
AIMS	To provide the student with the understanding of a range of discrete methods that contribute to (i) computer design, (ii) expressing how a computer stores and uses data, (iii) the reasoning and formal notation that is required for programming.
LEARNING OUTCOMES	On completion of this course the student should be able to: • Convert a number in one base to its equivalent in another. • Perform Arithmetic on numbers in different bases and understand the two's complement notation for negative binary numbers. • Write numbers in Floating Point Notation and understand how numbers are stored in a computer. • Use the language of Sets and construct Venn Diagrams. • Use Karnaugh Maps and the Laws of Sets to simplify Set expressions. • Use the Cartesian product of two sets to define a Relation • Understand the properties of relations. • Understand and determine the properties of functions. • Combine functions and find the inverse of a function. • Recognise a proposition and use connectives to form complex propositions • Take a complex proposition written in English and express it in symbolic logic, and vice-versa. • Construct Truth Tables for Complex Propositions. • Decide if an argument is logically valid. • Use Predicate logic to represent objects and the relationships between them. • Use Propositional functions and Quantifiers • Use a computer to generate a document that contains correct mathematical notation.

CONTENT	• Number Systems: Denary, Binary, Octal Hexadecimal • Set Theory • Propositional Logic • Predicate Logic
ASSESSMENT	**Coursework: 50%** **Exam: 50%** Two assignments (each weighted 30%). These will involve the application of topics taught in the first part of the course. The examination will contain short and long questions on the whole of the syllabus.
LEARNING RESOURCES	Students will be provided with a series of comprehensive information and problem sheets, some of which will be available on a web-site.
TEACHING & LEARNING ACTIVITIES	These will include: lectures, tutorials, use of computer software, group discussions and group work
TUTOR	Ian Newman

UNIT TITLE	**FOUNDATION PROGRAMMING**
DURATION	17 weeks – Semester 1
AIMS	To provide the student with an understanding of how to design algorithms to solve simple problems in a suitable high-level programming language, and how to code, document and test those solutions.
LEARNING OUTCOMES	On completion of this course the student should be able to: • describe a range of programming languages, and how they are suited to different programming needs • transform a program specification into a design using a standard top-down design technique • use a programming environment to edit, debug and compile a simple program • design and use a test plan for verification of a program, and draw conclusions from the outcomes • be familiar with basic programming constructs, and be able to demonstrate several techniques for using them • understand the reasons for, and show ability to document programs to a recognised professional standard • understand the process of developing a complete application, and use appropriate problem-solving techniques to do so
CONTENT	• overview of range and types of programming languages • introduction to a development environment and use of online help facilities • algorithm design: using a top-down technique • simple data types and naming conventions • controlling flow using selection statements • controlling flow using iteration • using functions for re-use • modular programming • complex data types: records and arrays • programming standards: layout and documentation • verification & testing

ASSESSMENT	**Coursework 100%** Students will submit 3 individual assignments: Assignment 1 (weighted 15%) Assignment 2 (weighted 25%) Assignment 3 (weighted 60%)
LEARNING RESOURCES	Students will use a standard programming environment. They will be provided with a range of sample exercises covering all of the necessary material, but will also be expected to use books, web sources and online Help facilities to supplement these. The LRC also has a range of good books available. Specific recommendations may be made by the tutor.
TEACHING & LEARNING ACTIVITIES	These will include lectures, tutorials and workshops. Group discussions, problem-solving sessions and peer tutoring will be used on an informal basis.
TUTOR	Richard A Handy

UNIT TITLE	**SYSTEMS ANALYSIS & SOFTWARE DESIGN**
DURATION	17 weeks – Semester 1
AIMS	To provide the student with: • an understanding of why computer systems often go wrong • an understanding of different life cycle models used for developing computer systems • an understanding why standardised methods and notation are needed in analysis and design • the ability to apply object oriented techniques to the analysis and modelling of a realistic system.
LEARNING OUTCOMES	On completion of this course the student should be able to: • describe the purpose of systems analysis and design • describe and evaluate system life-cycle models • describe some of the main concepts behind object oriented analysis and design • produce class definitions from a system requirements specification • produce Use Case diagrams and scenarios from a system requirements specification • produce Sequence diagrams • produce class and object associations, inheritance and aggregation using standard diagrams.

CONTENT	• Computer systems and what can go wrong with them • The purpose of systems analysis and design • Description and evaluation of system life-cycle methods • The main concepts behind object oriented systems analysis and design • Objects and classes: relationships, attributes and methods • Use Case diagrams and scenarios • Sequence diagrams • Inheritance and aggregation.
ASSESSMENT	**Coursework 100%** Assessment will consist of: Portfolio of individually worked exercises (weighted 40%) Group project (weighted 60%)
LEARNING RESOURCES	Students will be provided with a range of class exercises covering all of the necessary material, but will also be expected to use books and web sources to supplement these. The LRC also has a range of good books available. Specific recommendations may be made by the tutor.
TEACHING & LEARNING ACTIVITIES	These will include lectures, tutorials and exercises. Group discussions, problem-solving sessions and peer tutoring will be used on an informal basis.
TUTOR	Richard A Handy

UNIT TITLE	**INTRODUCTION TO ARTIFICIAL INTELLIGENCE**
DURATION	17 weeks – semester 2
AIMS	To gain an insight into the methodology and scope of the subject. To better equip the student to make informed choices regarding degree courses at Sussex University.
LEARNING OUTCOMES	On completion of this course the student should be able to: • Conduct a project of individual research. • Write a formal report • Give a presentation of a piece of research. • Use suitable programming languages, such as Prolog and Matlab. • Be able to develop methods for problem solving. • Have a basic understanding of Knowledge-based reasoning. • Have a basic understanding of adaptive systems - artificial neural networks. • Have a basic understanding of evolutionary systems – genetic algorithms.
CONTENT	Includes: • Search routines for optimum solutions. • Heuristics • Use of a rule-based programming language. • Use of a programming language, suitable for mathematical applications. • Use of Expert Systems. • Use of Artificial Neural Networks. • Use of Genetic Algorithms

ASSESSMENT	**Assessment: 100% coursework** Based on three assignments and a presentation: • Assignment 1. To produce a written report of an individual research project – (weighted 35%) • Individual Presentation. Of the work carried out for Assignment 1 – (weighted 15%) • Assignments 2 & 3. To produce designs, implementations, and short written reports, of simple evolutionary and adaptive programs – (each weighted 25%**)**
LEARNING RESOURCES	Students may use a variety of learning resources to support learning on this module. These will include: • Computer labs. • Lecture notes. • The college library. • The Internet. • Other students.
TEACHING & LEARNING ACTIVITIES	There will be one theory session and one seminar / workshop session per week.
TUTOR	Ian Newman

UNIT TITLE	**INTODUCTION TO INTERACTIVE WEB DESIGN**
DURATION	17 weeks – Semester 1
AIMS	This module aims to: • Introduce a knowledge of browser and server concepts and features. • Introduce students to web page production and testing. • Provide an awareness of designing usable interfaces using client-side and server-side technologies.
LEARNING OUTCOMES	By the end of this module the students should be able to: • Demonstrate a basic knowledge of Human Computer Interaction issues. • Be able to create an HTML web page • Be able to use simple JavaScript code within a web page. • Be able to use Cascading Sheet Styles • Design, develop and test a group of web pages • Identify business goals • Identify user needs • Incorporate simple multimedia effects

CONTENT	• Overview of the internet • Multi tier systems • General design principles • Basic HTML coding including tables, frames, and forms • Dynamic HTML • Style sheets • Simple scripting for interaction • File formats • Simple animation
ASSESSMENT	**Coursework 100%** Students will submit 2 individually produced assignments: Assignment 1 (30% weighting) Assignment 2: (50% weighting) I**ntegrated Group Project** (weighted 20%)**:** This end-of-year project gives students the opportunity to bring together many skills which have been acquired and improved throughout the year.
LEARNING RESOURCES	Much of the learning support for this module will be available on Blackboard during the delivery of the module. Indicative reading: Deitel H N, Deitel P J and Nieto T R, 2000 Internet and World Wide Web How To Program, Prentice Hall. Various web resources and tutorials.
TEACHING & LEARNING ACTIVITIES	Two contact periods per week, these will include lectures, tutorials and workshops. Group discussions, problem-solving sessions and peer tutoring will be used on an informal basis.
TUTOR	Ramiz Alihodzic

UNIT TITLE	**MATHEMATICS FOR COMPUTING**
DURATION	17 weeks
AIMS	To provide the student with an understanding of (i) a branch of mathematics that is useful in analysing the results of research, (ii) topics that have a wide and growing range of applications in the contemporary world.
LEARNING OUTCOMES	On completion of this course the student should be able to • Choose and use a suitable sampling method when collecting data. • Organise data and use standard statistical charts to illustrate the data. • Obtain the descriptive statistics: mean, median, mode, range, quartiles, variance and standard deviation for a set of data. • Investigate linear correlation and find the equation of a line of regression. • Understand basic probability and the laws of probability. • Calculate conditional probability and use both contingency tables and tree diagrams. • Derive and evaluate formulae in the application of combinations and permutations. • Understand and solve problems involving the Uniform, Binomial, Poisson and Normal probability distributions. • Determine confidence intervals for the mean of a Normal distribution when certain parameters are known. • Perform an investigation and produce a report containing technical and statistical material in a professional manner • Understand directed and undirected graphs and their applications. • Construct a rooted binary tree for both sorting and analysing a mathematical expression. • Use Prim's Algorithm and Dijkstra's algorithm for Networks.

CONTENT	• Statistics. • Probability. • Discrete and Continuous Probability Distributions. • Permutations and Combinations. • Relations and Functions. • Graph Theory.
ASSESSMENT	**Portfolio** (weighted 40%)**:** Students will compile a portfolio of evidence to meet the learning outcomes specified in the first part of the course. Much of this evidence will be the answers to an investigation which will, require the use of Excel or similar software. **Examination** (weighted 40%): This will be an 'open book' test containing long and short questions on the whole syllabus. **Integrated Group Project** (weighted 20%)**:** This end-of-year project gives students the opportunity to bring together many skills which have been acquired and improved throughout the year.
LEARNING RESOURCES	Students may use a variety of learning resources to support learning on this module. These will include: books, magazines, the internet, lecture notes and the college Virtual Learning Environment.
TEACHING & LEARNING ACTIVITIES	These will include: lectures, tutorials, group discussions, group work and computer workshops.
TUTOR	Richard A Handy

UNIT TITLE	**PROGRAM DEVELOPMENT**
DURATION	17 weeks – Semester 2
AIMS	To provide the student with: • an understanding of the differences between procedural, structured and object oriented programming • hands-on experience of designing, writing, testing and documenting object oriented programs.
LEARNING OUTCOMES	On completion of this course the student should be able to: • describe some of the main concepts behind object oriented programming • use object oriented techniques from an OOD method, such as UML, to design a simple program • produce code that implements OOD classes in a suitable programming language • produce and use test plans for classes implemented in a programming language • produce a class library file with appropriate documentation • use a class library file in producing an application program.
CONTENT	• Concepts of object oriented programming • Modelling the real world by working classes and objects • Use language provided classes and objects in writing simple programs • Implementing simple classes, testing and documenting them • Overloading parameters, methods and operators • Inheritance and classification hierarchies • Dynamic structures and advanced algorithmic methods • Class library files • Streams and I/O files • Producing an application program.

ASSESSMENT	**Coursework 100%** Students will submit 2 individually produced assignments and one Integrated Group Project: Assignment 1 (weighted 40%) Assignment 2 (weighted 40%) **Integrated Group Project** (weighted 20%)**:** This end-of-year project gives students the opportunity to bring together many skills which have been acquired and improved throughout the year.
LEARNING RESOURCES	Students will use a standard programming environment. They will be provided with a range of sample exercises covering all of the necessary material, but will also be expected to use books, web sources and online Help facilities to supplement these. The LRC also has a range of good books available. Specific recommendations may be made by the tutor.
TEACHING & LEARNING ACTIVITIES	These will include lectures, tutorials and workshops. Group discussions, problem-solving sessions and peer tutoring will be used on an informal basis.
TUTOR	Richard A Handy