



NUI MAYNOOTH

Ollscoil na hÉireann Má Nuad

National University of Ireland, Maynooth

Computer Sci.& Software Engineering Course Description

Details

Our modern world is run on computing power. PCs, mobile phones, telecoms networks, medical equipment, air traffic control, satellite navigation – hardly any area of 21st century life is untouched by computers. As a student, you will have access to NUI Maynooth's state-of-the-art computer labs as you acquire theoretical knowledge of the fundamental principles of computer science and software engineering, along with the ability to apply these principles in practice using modern tools and methodologies.

You will study all the essentials of computers and software, as well as key areas in mathematics, business, and organisational studies. You will participate in extensive supervised practical work, including a six-month placement in industry. You will acquire hands-on programming skills and advanced problem-solving techniques, and you will learn to think logically and analytically when approaching complex issues. You will also develop your creativity and communication skills.

Course structure















Computer Science and Software Engineering is taken as a Single Honours Bachelor of Science degree. In First Year, students take Computer Science, Mathematics, and two other subjects from Biology, Chemistry, Experimental Physics, or Mathematical Physics. In Second Year, students take Computer Science related topics and two Mathematics modules. In Third Year and Fourth Year, students take Computer Science and Software Engineering topics. In Third Year, students must complete an industrial work placement of at least six months in duration.



[Click here for further details](#)

Year 1

- Select 4 Subjects.

 <u>COMPUTER SCIENCE & SOFTWARE ENGINEERING - SE100</u>	Credits: 15 Compulsory: 
 <u>MATHEMATICS - MT100</u>	Credits: 15 Compulsory: 
 <u>BIOLOGY - BL100</u>	Credits: 15 Not compulsory: 
 <u>CHEMISTRY - CH100</u>	Credits: 15 Not compulsory: 
 <u>ENGINEERING SCIENCE - GE100S</u>	Credits: 15 Not compulsory: 
 <u>EXPERIMENTAL PHYSICS - EP100</u>	Credits: 15 Not compulsory: 
 <u>MATHEMATICAL PHYSICS - MPP100</u>	Credits: 15 Not compulsory: 

Year 2

- Select Subject Below.

 <u>COMPUTER SCIENCE & SOFTWARE ENGINEERING - SE200</u>	Credits: 60 Compulsory: 
------------------------------------------------------------------------------------------------------------------------------------------------	---------------------------------------------------------------------------------------------------------------

Year 3

- Select Subject Below.

 <u>COMPUTER SCIENCE & SOFTWARE ENGINEERING - SE300</u>	Credits: 60 Compulsory: 
------------------------------------------------------------------------------------------------------------------------------------------------	---------------------------------------------------------------------------------------------------------------

Year 4

- Select Subject Below.







 <u>COMPUTER SCIENCE & SOFTWARE ENGINEERING - SE400</u>	Credits: 60 Compulsory: 
------------------------------------------------------------------------------------------------------------------------------------------------	---------------------------------------------------------------------------------------------------------------

Year1

Computer Sci.& Software Engineering

COMPUTER SCIENCE & SOFTWARE ENGINEERING - SE100

Credits: 15 Compulsory: 

Module	Code	Credits	Semester	Compulsory
 INTRODUCTION TO PROGRAMMING	CS141	5	1	
 INTRODUCTION TO COMPUTER SYSTEMS	CS143	5	1 and 2	
 INTRODUCTION TO COMPUTER SCIENCE	CS142	5	2	

INTRODUCTION TO PROGRAMMING

Module code: CS141

Credits: 5

Semester: 1

Department: COMPUTER SCIENCE

International: 

Overview

Programming fundamentals: variables, types, expressions and assignment; simple I/O; Conditional and iterative control structures (if statements and while loops); Strings and string processing; Use of class APIs for creating objects and calling methods; Understanding data abstraction and encapsulation; Problem solving: understanding and developing algorithms; Implementing algorithms as simple programs.

Learning Outcomes

On successful completion of the module, students should be able to:

- On successful completion of this module, students should be able to: understand and evaluate simple algorithms; create simple algorithms; write simple programs; debug runtime errors.

INTRODUCTION TO COMPUTER SYSTEMS

Module code: CS143

Credits: 5

Semester: 1 and 2

Department: COMPUTER SCIENCE

International: 

Overview

This module is composed of three core areas: computer architecture, operating systems and computer networks, and provides foundation material and practical experience of each area.

The computer architecture area covers underlying concepts (Boolean algebra, numbering systems required for computing, gates), the Von Neumann architecture and investigates system technology (processors, storage, communications) in the context of modern hardware systems. The operating systems area examines the basic components of modern operating systems (hardware abstraction, process and application management, memory management, user interfacing, and security). The networking area addresses basic digital communications, both wired and wireless, covering LANs and WANs in the context of modern, internet-accessible computing devices.

Learning Outcomes

On successful completion of the module, students should be able to:

- Describe the evolution of computer systems
- Identify the fundamental types of modern computing devices and peripherals
- Explain the underlying logical concepts that provide the basis for modern computation
- Describe basic computer architecture on a theoretical and practical basis
- Identify the workings of modern operating systems and differentiate between them
- Install, configure, and perform basic management tasks on multiple operating systems
- Explain the fundamentals of computer communications and set-up and configure different network topologies such as a peer-to-peer network, and LAN
- Identify the key components of the Internet and identify security issues relating to it

INTRODUCTION TO COMPUTER SCIENCE

Module code: CS142

Credits: 5

Semester: 2

Department: COMPUTER SCIENCE

International: ✓

Overview

Languages: recursive definitions, regular expressions, context-free grammars; Automata: finite automata (deterministic and non-deterministic), conversion from grammars to automata, transition graphs, push-down automata; Problem solving: understanding and developing algorithms; Implementing algorithms to apply language definitions and to simulate automaton: Array and string processing, iteration (for loops) and recursion, scope of variables, static methods, defining classes, creating objects, defining and calling methods (including constructors), recursion.

Learning Outcomes

On successful completion of the module, students should be able to:

- Acknowledge the links between mathematics, science and computer science
- Appreciate the historical context in which the theory of computer science was developed
- Understand and describe the concept of finite state machines and regular languages
- Write simple programs in Java using conditional statements, loops, arrays and methods
- Apply problem solving techniques to programming problems
- Value the benefits of teamwork and shared solution development

Mathematics

MATHEMATICS - MT100

Credits: 15 Compulsory: ✓

Module	Code	Credits	Semester	Compulsory
DIFFERENTIAL CALCULUS	MT101S	5	1	✓
LINEAR ALGEBRA (S)	MT111S	2.5	1	✓
INTEGRAL CALCULUS	MT102S	5	2	✓
DATA ANALYSIS 1 (S)	MT122S	2.5	2	✓

DIFFERENTIAL CALCULUS

Module code: MT101S

Credits: 5

Semester: 1

Department: MATHEMATICS AND STATISTICS

International: 

Overview

Objectives: To introduce students to Differential Calculus in One Real Variable. Overview: Why study calculus?

Sets and Functions. Review of Logarithms. Inequalities, Limits: Idea of a limit, definition of a limit, continuity. Derivatives: Definition of a derivative, tangents, product rule, chain rule, quotient rule, higher derivatives, implicit differentiation, Newton's method for roots, the law of the mean, exponential and logarithmic functions. Applications of derivatives: Maxima and minima, monotonic functions and the first derivative test. Curvature of plane curves. Taylor's theorem. Concavity and points of inflection. Symmetry and curve sketching. Applied max-min problems and related rates.

Learning Outcomes

On successful completion of the module, students should be able to:

- Evaluate limits of basic functions.
- Obtain derivatives of basic functions and combinations of such functions.
- Solve max-min problems.
- Use Newton's method
- Sketch curves of the basic functions and combinations of such functions.

LINEAR ALGEBRA (S)

Module code: MT111S

Credits: 2.5

Semester: 1

Department: MATHEMATICS AND STATISTICS

International: 

Overview

Module Objective:

To introduce students to linear systems, matrices and vectors.

Linear systems; row reduction; matrices and matrix algebra; determinants and inverses (of 2x2 matrices); vectors; dot and cross products; lines and planes.

Learning Outcomes

On successful completion of the module, students should be able to:

- Use matrix algebra
- Define the determinant and inverse of a matrix and understand their meaning.
- Obtain the determinant and inverse of a 2x2 matrix.
- Use the method of row reduction.
- Find products of vectors and be able to use these in simple geometrical problems.
- Write down the equations for lines and planes in space.

INTEGRAL CALCULUS

Module code: MT102S

Credits: 5

Semester: 2

Department: MATHEMATICS AND STATISTICS

International: 

Overview

Objectives: To introduce students to Integral Calculus.

Anti-derivatives, area, the definite integral, indefinite integrals, the Fundamental Theorem of Calculus, the substitution rule, areas between curves, inverse functions, inverse trigonometric functions.

Techniques of Integration. Tables of integrals, integration by parts, trigonometric integrals, Substitution, integration of rational functions, trapezoidal rule, Simpson's Rule. Applications of Integration: Volumes and surfaces of revolution, length of a curve.

Learning Outcomes

On successful completion of the module, students should be able to:

- Integrate the basic functions and combinations of these functions using standard techniques.
- Use integrals to find areas and volumes.
- Compute an integral numerically to a given degree of accuracy.
- State the Fundamental Theorem of Calculus.

DATA ANALYSIS 1 (S)

Module code: MT122S

Credits: 2.5

Semester: 2

Department: MATHEMATICS AND STATISTICS

International: 

Overview

Module Objective:

To introduce students to Data Analysis.

About data: cases, variables, collecting data. Categorical data: frequency tables, barcharts. Contingency tables, conditional distributions, clustered and segmented barcharts. Quantitative data: stem and leaf diagrams, histograms, boxplots. Numerical summaries: mean, median, quartiles, IQR, standard deviation. Scatterplots, correlation, the regression line.

Learning Outcomes

On successful completion of the module, students should be able to:

- Identify appropriate methods for collecting and collating data.
- Describe various types of data using numerical or graphical techniques.
- Interpret graphical and tabular data presentations.
- Find the regression line for a data set
- Use the Normal Distribution tables

Year 2

Computer Sci.& Software Engineering

Year 2

- Select Subject Below.



COMPUTER SCIENCE & SOFTWARE ENGINEERING - SE200

Credits: 60 Compulsory:



Module	Code	Credits	Semester	Compulsory
DATABASES	CS130	5	1	✓
DISCRETE STRUCTURES 1	CS151	5	1	✓
ALGORITHMS & DATA STRUCTURES 1	CS210	5	1	✓
COMPUTER ARCHITECTURE	CS220	5	1	✓
SOFTWARE TESTING	CS265	5	1	✓
CALCULUS 3 (S)	MT201S	5	1	✓
ALGORITHMS & DATA STRUCTURES 2	CS211	5	2	✓
WEB INFORMATION PROCESSING	CS230	5	2	✓
OPERATING SYSTEMS	CS240	5	2	✓
COMPUTER ARCHITECTURE 2	CS253	5	2	✓
SOFTWARE ENGINEERING & SOFTWARE PROCESS	CS335	5	2	✓
LINEAR ALGEBRA 2 (A)	MT212A	5	2	✓

DATABASES

Module code: CS130

Credits: 5

Semester: 1

Department: COMPUTER SCIENCE

International: 

Overview

Data organisation; file systems; storage and access methods; conceptual modelling; relational database model; data structure; data integrity; data manipulation; Structured Query Language (SQL); relational algebra; alternative database models; basic transaction processing; distributed database technology.

Learning Outcomes

On successful completion of the module, students should be able to:

- Explain the characteristics that distinguish the database approach from the traditional approach of programming with data files
- Identify major DBMS functions and describe their role in a database system
- Prepare a relational schema from a conceptual model developed using the entity- relationship model
- Explain and demonstrate the concepts of entity integrity constraint and referential integrity constraint (including definition of the concept of a foreign key)
- Demonstrate use of the relational algebra operations from mathematical set theory (union, intersection, difference, and Cartesian product) and the relational algebra operations developed specifically for relational databases (select (restrict), project, join, and division)
- Demonstrate queries in the relational algebra
- Use a declarative query language to elicit information from a database

DISCRETE STRUCTURES 1

Module code: CS151

Credits: 5

Semester: 1

Department: COMPUTER SCIENCE

International: 

Overview

First Order Logic and proof strategies for reasoning about programs. Basic set and bag definitions, operations, and properties. Using functions to solve problems: to include definitions of function domain, range, image, composition, identity, inverse, abstraction, application, total & partial functions, injections, bijections & surjections. Reasoning using First Order Logic: truth tables, use of conjunction, disjunction, implication, negations, equivalence rules, assertions, tautologies, contradictions, universal and existential quantifiers. Representing natural language in a formal logic and proving the validity of logical statements using proof by contradiction and natural deduction.

Learning Outcomes

On successful completion of the module, students should be able to:

- Solve problems in computer science applying techniques from discrete mathematics
- Outline the basic structure of standard proof techniques
- Give examples of standard proof techniques
- Use functions to formulate and solve problems
- Relate the ideas of mathematical induction to recursion and recursively defined structures
- Formulate problems using first order logic

ALGORITHMS & DATA STRUCTURES 1

Module code: **CS210**

Credits: **5**

Semester: **1**

Department: **COMPUTER SCIENCE**

International: 

Overview

Introduction to algorithms and data structures. Review of elementary programming concepts suitable for the implementation of abstract data types (operators, types and expressions; control of flow; methods; recursion; input & output); Algorithms for searching: linear, bounded linear and binary searches; Algorithms for sorting: selection, insertion, bubble and quick sorts; Fundamental linear data structures: stacks, queues, linked lists; Object-oriented programming: encapsulation and information hiding, classes, interfaces, class hierarchies, inheritance, polymorphism, basic exception handling; Analysis of basic algorithms.

Learning Outcomes

On successful completion of the module, students should be able to:

- Recognise the importance of program complexity
- Describe a variety of structures for storing data such as arrays, linked lists, stacks and queues
- Explain a range of algorithms involving searching and sorting
- Identify data structuring strategies appropriate to a given context
- Design, develop, test and debug object-oriented programs in Java
- Apply data structuring techniques to the design of computer programs

COMPUTER ARCHITECTURE

Module code: **CS220**

Credits: **5**

Semester: **1**

Department: **COMPUTER SCIENCE**

International: 

Overview

Number Systems, Integer and Floating Point Arithmetic, Boolean Algebra, Function Simplification, Combinational Logic Design Principles and Circuits, Sequential Logic Design Principles and Circuits, Memory Technology and Programmable Logic. Application Circuits: Arithmetic and Logic Operations, Comparators, Multiplexers, Decoders, Counters, Registers and Shift Registers, Sequential Pattern Detectors, Memory Arrays, Parity and Hamming Code Generation and Checking.

Learning Outcomes

On successful completion of the module, students should be able to:

- All students completing this module will have an understanding of the design and operation of the basic logic elements on which processor control and functional circuitry is based and understand how to program a processor at the instruction level.

SOFTWARE TESTING

Module code: CS265

Credits: 5

Semester: 1

Department: COMPUTER SCIENCE

International: 

Overview

Background and reasons for Software Testing. Value and objectives of Software Testing, types of Testing, test methods, testing in the Software Development Process, testability, automation of Software Testing. Topics: Introduction to software testing. Software quality, software faults and failures, phases of testing. Incremental and integration testing, unit testing, whitebox and blackbox testing, system, acceptance, and regression testing. Coverage-based techniques, mutation testing, equivalence testing and boundary value analysis. Cause-effect graphing. Walkthroughs and inspections. Object Oriented software testing.

Learning Outcomes

On successful completion of the module, students should be able to:

- Explain the software testing process from unit testing up to system testing
- Design and apply test cases for both white and black box testing of software
- Describe and evaluate the approaches to integrating software testing into the software development process
- Describe the difference between software verification and software validation

CALCULUS 3 (S)

Module code: MT201S

Credits: 5

Semester: 1

Department: MATHEMATICS AND STATISTICS

International: 

Overview

Module Objective:

To introduce the student to multivariable differential calculus.

Vectors, especially in low dimensions. Vector addition. Dot product. Vector product in \mathbb{R}^3 . Solid analytic geometry. Lines, planes and spheres. Calculus of several variables, especially in two and three dimensions. Functions of two or more variables. Graphs. Limits. Continuity. Partial derivatives. Gradient. Tangents and normals. Max-min. Lagrange multipliers. Computation using mathematical computing software.

Learning Outcomes

On successful completion of the module, students should be able to:

- Find products of vectors and use these to solve simple geometrical problems.
- Draw the level curves for a function of two variables.
- Obtain the partial derivatives and gradient vector of a function.
- Obtain tangents and normals to surfaces.
- Solve max-min problems in several variables.



ALGORITHMS & DATA STRUCTURES 2

Module code: CS211

Credits: 5

Semester: 2

Department: COMPUTER SCIENCE

International: 

Overview

Fundamentals and implementation of Binary Search Trees; Balancing Binary Search Trees; Tree traversals (depth and breadth-first); Graph representations; Hash Tables; Algorithm analysis: upper and average complexity bounds, best, average and worst case algorithm behaviour; Algorithm strategies: brute force, greedy, divide and conquer and backtracking algorithms. Selected advanced topics in Algorithms and Data Structures.

Learning Outcomes

On successful completion of the module, students should be able to:

- Describe a variety of structures for storing data such as binary search trees, balanced binary trees and hash tables
- Outline a range of algorithms in the areas of data compression, cryptography and graph theory
- Explain various object-oriented programming concepts such as encapsulation, inheritance and polymorphism
- Design, develop, test and debug object-oriented programs in Java
- Apply knowledge of algorithm complexity and data structuring techniques to problem solving




WEB INFORMATION PROCESSING

Module code: CS230

Credits: 5

Semester: 2

Department: COMPUTER SCIENCE

International: 

Overview

The course starts with an introduction to Information Processing followed by a look at Data Storage and Data Transfer before moving on to the architecture of the WWW and an in depth look at some of the protocols involved (including SMTP, POP3 and HTTP). The course also deals with client side web programming in detail, covering topics like XHTML, JavaScript, DHTML, CSS, and XML before examining server side web programming and in particular the Common Gateway Interface and server side programming with Perl. The course also deals with Internet security and the operation of SSL as well as web proxies and server farms.

Learning Outcomes

On successful completion of the module, students should be able to:

- Create a software requirements document for a web-based project.
- Describe web protocols, architecture and communication.
- Design and build a dynamic interactive web-based application.
- Describe key principles for design of user interfaces, such as learnability, usability and robustness.
- Describe examples of bad navigation, bad screen layout, and incomprehensible interface design.
- Describe key issues in designing accessible web-sites.



OPERATING SYSTEMS

Module code: CS240

Credits: 5

Semester: 2

Department: COMPUTER SCIENCE

International: ✓

Overview

Functions of operating systems, scheduling theory, concurrency control, deadlock management, memory management, file systems, interprocess communication, protection and security.

Learning Outcomes

On successful completion of the module, students should be able to:

- The student will achieve a broad and thorough understanding of fundamental Operating System Principles, Design and Functionality & a basic understanding of Concurrency Theory and Concurrent Programming problems.



COMPUTER ARCHITECTURE 2

Module code: CS253

Credits: 5

Semester: 2

Department: COMPUTER SCIENCE

International: ✓

Overview

Architecture of a Small Microprocessor based Computer, Assembly language programming, Interrupts and IO, Machine Cycle, Representation of data, Memory Buses.

Learning Outcomes

On successful completion of the module, students should be able to:

- Describe the architecture of a small microprocessor including the function of microprocessor registers and addressing modes
- Implement a simple calculator (operator/operand) and memory using a FPGA
- Implement x86 code to do both binary and floating point calculation and estimate code run time and to modify the function of interrupt routines
- Describe semiconductors and device construction, PN junction and FET
- Describe static and dynamic memory cells, explain cache strategies and estimate effective access time for a cache
- Compare RISC and CISC design philosophies and parallel architectures



SOFTWARE ENGINEERING & SOFTWARE PROCESS

Module code: CS335

Credits: 5

Semester: 2

Department: COMPUTER SCIENCE

International: ✓

Overview

This module provides both understanding of, and practice in, the principles and techniques of software Engineering. It addresses such issues as: understanding and selecting software processes; planning and monitoring software projects; analysing and designing software using current techniques (including UML); and software testing.

Learning Outcomes

On successful completion of the module, students should be able to:

- Explain the concept of a software life cycle and provide an example, illustrating its phases including the deliverables that are produced.
- Select and justify the software development models and process elements most appropriate for the development and maintenance of a diverse range of software products.
- Compare the traditional waterfall model to the incremental model, the unified model, and other appropriate models.
- Distinguish between the different types and levels of testing (unit, integration, systems, and acceptance) for medium-size software products and related materials.
- Analyse and design software using current techniques, for e.g. UML
- Evaluate the outcomes and the process of a software development project.



LINEAR ALGEBRA 2 (A)

Module code: MT212A

Credits: 5

Semester: 2

Department: MATHEMATICS AND STATISTICS

International: ✓

Overview

Module Objective:

To introduce students to inverses and determinants of $n \times n$ – matrices, Vector spaces, Eigenvectors and eigenvalues.

Inverse and determinants of $n \times n$ - matrices; eigenvalues and eigenvectors; vector spaces; basis and dimension; linear maps; rank and nullity. Computation using mathematical computing software.

Learning Outcomes

On successful completion of the module, students should be able to:

- Calculate the inverse and determinant of an $n \times n$ matrix and understand their meaning.
- Obtain eigenvalues and vectors for $n \times n$ matrices.
- Find a basis and the dimension of a vector space.
- Define the terms rank and nullity and understand their meaning.















Year 3

Computer Sci.& Software Engineering

Year 3

- Select Subject Below.

 **COMPUTER SCIENCE & SOFTWARE ENGINEERING - SE300** Credits: 60 Compulsory: 

	Module	Code	Credits	Semester	Compulsory
	EMPIRICAL SOFTWARE ENGINEERING	CS258	5	1	
	SOFTWARE DESIGN	CS264	5	1	
	PROGRAMMING LANGUAGES & COMPILERS	CS310	5	1	
	COMPUTER NETWORKS	CS320	5	1	
	TEAM PROJECT	CS353	5	1	
	SOFTWARE VERIFICATION	CS357	5	1	
	INDUSTRIAL WORK PLACEMENT	CS361	30	2	

EMPIRICAL SOFTWARE ENGINEERING

Module code: CS258

Credits: 5

Semester: 1

Department: COMPUTER SCIENCE

International: 

Overview

Measurement: foundations of measurement; scales; meaningfulness. The Scientific Method: philosophy of science; application to Computer Science and Software Engineering (including the software process). Formulation and testing of hypotheses: expressing hypotheses; selection of a test approach; design and execution of experiments, surveys, and case studies. Data Analysis: basic statistics; distributions; measures of average, variability, error, and confidence; quantiles; measures of correlation; sampling theory; confidence intervals and confidence tests. Results Evaluation: evaluating results with reference to hypotheses. Evaluation of validity: internal and external validity; threats to validity; evaluating the validity of published research results. Indicative reading list: Software Metrics 2nd Ed. by Fenton and Pfleeger, Experimental Methodology by Christensen.

Learning Outcomes

On successful completion of the module, students should be able to:

- Formulate, and select the appropriate methodology to evaluate hypotheses in computer science and software engineering;
- Select meaningful measurements;
- Design and conduct experiments, case-studies, and surveys;
- Use statistical methods to analyse data;
- Interpret and explain the results;
- Evaluate the validity of research results.

SOFTWARE DESIGN

Module code: CS264

Credits: 5

Semester: 1

Department: COMPUTER SCIENCE

International: 

Overview

Detailed software design and construction in depth. In-depth coverage of design principles, design patterns, anti-patterns and refactoring. Introduction to formal approaches to design. Analysis of designs based on internal quality criteria. Performance and maintainability improvement. Reverse engineering. Disciplined approaches to design change.

Learning Outcomes

On successful completion of the module, students should be able to:

- Apply a wide variety of design patterns and frameworks in designing a wide variety of software
- Discuss the implementation of different object oriented programming techniques
- Design and implement software using several different techniques
- Modify designs using sound approaches and principles
- Compare and contrast different software design methodologies
- Describe different software patterns and their consequences

PROGRAMMING LANGUAGES & COMPILERS

Module code: CS310

Credits: 5

Semester: 1

Department: COMPUTER SCIENCE

International: 

Overview

Compilers, interpreters and assemblers; formal grammars; lexical analysis, syntax analysis, predictive parsing; symbol tables and semantic analysis; code generation, optimisation and run-time environments.

Learning Outcomes

On successful completion of the module, students should be able to:

- Describe the phases of program translation from source code to executable code and the files produced by these phases
- Recognise the underlying formal models such as finite state automata, push-down automata and their connection to language definition through regular expressions and grammars
- Explain the benefits of intermediate languages in the compilation process
- Describe the steps and algorithms used in code generation
- Describe how the computer system uses activation records to manage program modules and their data
- Describe approaches to object lifetime management (e.g. garbage collection) and evaluate tradeoffs between different strategies

COMPUTER NETWORKS

Module code: **CS320**

Credits: **5**

Semester: **1**

Department: **COMPUTER SCIENCE**

International: 

Overview

Introduction to networking: history, evolution and architectural elements of networking, network types, layered models, OSI and TCP/IP models. Physical layer: application of information theory to communications, transmission media, the telephone system, modulation, multiplexing. Data-link layer: framing, error detection and correction, flow control, sliding window protocols, case study of real-world protocol. Medium access control sub-layer: the channel allocation problem, ALOHA protocols, CSMA and CSMA/CD, IEEE 802 standards. Network layer: network layer organisation, routing, congestion control, Internet Protocol (IP), Internet control protocols. Transport layer: transport layer services, Transmission Control Protocol (TCP), connection-management, two army problem.

Learning Outcomes

On successful completion of the module, students should be able to:

- For each of the layers in the TCP/IP reference model explain their implementation, functions, common protocols and data structures
- Identify and calculate the different types of delay that can occur during the transmission of a message across a network
- Discuss the techniques that can be used to share network resources
- Show how routing algorithms are used to calculate the shortest path between nodes across a network
- Assess the level of service provided by the various multiple access links and protocols, identifying collision prone scenarios
- List the positives and negatives of implementing reliable data transfer, flow control, and congestion control in packet switched networks
- Explain how IP addresses can be assigned to hosts using sub netting and super netting, and will be able to identify situations in which these are appropriate solutions

TEAM PROJECT

Module code: **CS353**

Credits: **5**

Semester: **1**

Department: **COMPUTER SCIENCE**

International:

Overview

Software requirement elicitation, design, testing and implementation as a team; project management techniques and meeting skills; specific technical content matched to the project topic; product market research; promotion & business strategy.

Learning Outcomes

On successful completion of the module, students should be able to:

- Work effectively as a member of a software development team
- Deliver technical presentations during the software development cycle
- Apply learned skills and experience more effectively in future project work
- Plan and manage team activities
- Communicate effectively with other team members

SOFTWARE VERIFICATION

Module code: CS357

Credits: 5

Semester: 1

Department: COMPUTER SCIENCE

International: 

Overview

Motivation for software verification and formal methods; syntax and semantics and formal proofs in first-order logic; Hoare logic and program verification; temporal logic and model checking; brief case studies using selected formal methods.

Learning Outcomes

On successful completion of the module, students should be able to:

- Explain the limitations of testing as a means to ensure correctness and evaluate the role of verification in software engineering
- Create mathematically precise specifications and designs using logic-based specification languages
- Prove the correctness of programs with respect to a specification using Hoare logic
- Analyse the properties of formal specifications and designs
- Use tools to verify properties of specifications and designs

INDUSTRIAL WORK PLACEMENT

Module code: CS361

Credits: 30

Semester: 2

Department: COMPUTER SCIENCE

International: 

Overview

A minimum of six months work placement from February.

Learning Outcomes

On successful completion of the module, students should be able to:

- Take direction on applying work placements from the work placement officers, meeting all deadlines set by them and by any potential employers
- Find a suitable work placement by participating in suitable application and interview processes
- Participate in and contribute to a place of work in a manner that is agreeable to your employer
- Work effectively with others within the social context of a software development team
- Complete and return your work placement report documenting your main contributions to your place of work































Year 4

Computer Sci.& Software Engineering

Year 4

- Select Subject Below.

 **COMPUTER SCIENCE & SOFTWARE ENGINEERING - SE400** Credits: 60 Compulsory: 

Module	Code	Credits	Semester	Compulsory
 THEORY OF COMPUTATION	CS355	5	1	
 PROGRAMMING LANGUAGE DESIGN & SEMANTICS	CS424	5	1	
 MUSIC PROGRAMMING 2	CS322	5	1	
 SIGNAL, OPTICAL & IMAGE PROCESSING	CS356	5	1	
 MACHINE LEARNING & NEURAL NETWORKS	CS401	5	1	
 COMPUTER VISION	CS410	5	1	
 CRYPTOGRAPHY	CS416	5	1	
 ADVANCED CONCEPTS & ISSUES IN COMP. SCI. 1	CS430	5	1	
 FINAL YEAR PROJECT CSSE	CS440	15	1 and 2	
 COMPUTATION & COMPLEXITY	CS370	5	2	
 PARALLEL & DISTRIBUTED SYSTEMS	CS402	5	2	
 NUMERICAL COMPUTATION	CS417	5	2	
 ROBOTICS & AUTOMATION	CS422	5	2	
 AUDIO & SPEECH PROCESSING	CS425	5	2	
 COMPUTER GRAPHICS	CS426	5	2	
 ADVANCED CONCEPTS & ISSUES IN COMP. SCIENCE 2	CS431	5	2	

THEORY OF COMPUTATION

Module code: CS355

Credits: 5

Semester: 1

Department: COMPUTER SCIENCE

International: 

Overview

Mathematical preliminaries; regular languages, finite automata, and regular expressions; nondeterminism and determinism in finite automata; properties of regular languages; nonregular languages; context-free languages, context-free grammars, and pushdown automata; nondeterminism and determinism in pushdown automata; properties of context-free languages; non-context-free languages; multi-stack machines; recursively-enumerable languages and Turing machines; recursive and non-recursive languages; proofs of undecidability.

Learning Outcomes

On successful completion of the module, students should be able to:

- Design the corresponding machine model to accept a specified language
- Explain how some problems have no algorithmic solution
- Convert among equivalently powerful notations for a language, including among DFAs, NFAs, and regular expressions, and between PDAs and CFGs, explain the Church-Turing thesis and its significance
- Assess and prove the computational power required to decide a language



PROGRAMMING LANGUAGE DESIGN & SEMANTICS

Module code: CS424

Credits: 5

Semester: 1

Department: COMPUTER SCIENCE

International: ✔

Overview

This module is about the design of programming languages. In particular it is concerned with the syntax, the semantics and usability of a programming language. The appearance and structure of a language's sentence will be studied in order to determine which symbol sequences are permitted phrases of the language (syntax), and the assignment of meanings to the sentences of a programming language so that one can explain what the various phrases of a language mean (semantics) and the usability of a language, including the possible areas of application of the language.

Learning Outcomes

On successful completion of the module, students should be able to:

- Explain the importance of formal semantics
- Describe the different approaches to formal semantics
- Summarize the evolution of programming languages from a type theoretic perspective
- Formally describe each of the elementary data types
- Explain the concept of an abstract data type
- Defend the importance of types and type-checking in providing abstraction and safety
- Describe the concepts of encapsulation, abstraction, inheritance, and polymorphism in terms of more elementary data types
- Evaluate languages with regard to typing



FINAL YEAR PROJECT CSSE

Module code: CS440

Credits: 15

Semester: 1 and 2

Department: COMPUTER SCIENCE

International: ✔

Overview

Depends entirely on the individual project, which may be proposed by either the student or a member of staff.

Learning Outcomes

On successful completion of the module, students should be able to:

- Conduct a reasonably thorough investigation into a topic of interest, importance or relevance to computer science or its applications
- Where appropriate, accomplish an acceptable implementation (or other useful model) to fulfil the main goals of the project
- Work largely independently, within agreed project requirements, with minimal supervision
- Deliver technical presentations during the project development cycle
- Complete a detailed project report covering all central aspects of the project
- Work within all project deadlines, interim report deadlines and consultation requirements giving regular and timely feedback to your project supervisor(s)
- Plan and manage all your own project milestones



COMPUTATION & COMPLEXITY

Module code: CS370

Credits: 5

Semester: 2

Department: COMPUTER SCIENCE

International: ✔

Overview

Incompleteness, Turing's computable numbers, algorithmic decidability and uncomputability, models of computation and the reducibility between unrestricted models. Intractability, machine-independent complexity, measures of complexity and asymptotic analysis. Complexity classes, intra-class reductions, complexity analysis (best, worst, average case) of algorithms, optimal algorithms for a given model of computation and given complexity measures. More complexity classes, nondeterminism, NP, NP-completeness, NP-hardness, membership and nonmembership.

Learning Outcomes

On successful completion of the module, students should be able to:

- Define the classes P and NP.
- Explain the significance of NP-completeness.
- Prove that a problem is NP-complete by reducing a classic known NP-complete problem to it.
- Explain how some problems have no algorithmic solution.
- Provide examples that illustrate the concept of incompatibility.

Detailed Information:

http://www.nuim.ie/courses/?TARGET=QS&MODE=VIEW&QUALIFICATION_CODE=CSEN&TARGET_SOURCE=QUALIFICATION