DCS
UNIVERSITY OF JAFFNA

BSc (Computer Science)

(For Direct-intake Computer Science Students)

Level–2S

Effective from the Academic Year: 2017/2018

Department of Computer Science
Faculty of Science
University of Jaffna
Sri Lanka

Developed in November 2018

Course Code:	**CSC201S2**		
Course Title:	**Database Systems Concepts and Design**		
Credit Value:	**02**		
Core/Optional:	**Core**		
Hourly Breakdown:	**Theory**	**Practical**	**Independent Learning**
	30	**--**	**70**

Objectives:

Introduce database system concepts and fundamentals necessary for designing, implementing, and manipulating databases.

Intended Learning Outcomes:

- State key characteristics of a database
- Develop conceptual models for databases
- Create efficient databases
- Apply query language to create and manipulate databases

Course Contents:

- Introduction to database concepts and architecture: File systems, database system concepts, three-schema architecture, classifications of database systems and database users
- Data Modeling: Entity Relationship model, relational model, network model, hierarchical model, object relational model, UML class diagrams.
- Relational database design: Relational model concepts, defining a relational schema from an ER diagram, basics of functional dependencies and normalization (1NF, 2NF, 3NF and BCNF)
- Developing and manipulating databases: Data development and manipulation using SQL, MySQL, PostgreSQL and MongoDB
- Relational algebra and relational calculus: Binary operations, Cartesian product, extended relational operator, tuple relational calculus and domain relational calculus
- File organization for conventional DBMS: Storage devices and their characteristics, file organization, fixed-length records, variable-length records, sequential file organization, indexed sequential access method
- Introduction to transaction management, concurrency control and recovery: Concept of transactions, concurrency in transaction processing, recovering databases from failure

Teaching/Learning Methods:

Lectures, Recitation oral questions, Guided learning, Tutorial discussions

Assessment Strategy:

- In-course Assessments 30%
- End-of-course Examination 70%

References:

- Ramez Elmasri and Shamkant B. Navathe, Fundamentals of Database Systems, 7th Ed, Addison-Wesley, 2015.
- C.J. Date, An Introduction to Database Systems, 8th Ed, Addison-Wesley, 2003
- Ramakrishnan and Gehrke, Database Management Systems, 3rd Ed., McGraw-Hill, 2003.

Course Code:	CSC202S2		
Course Title:	Computer Programming II		
Credit Value:	02		
Core/Optional:	Core		
Hourly Breakdown:	Theory	Practical	Independent Learning
	--	90	110

Objectives:

Develop proficiency in writing programs to solve computational problems using suitable data structures.

Intended Learning Outcomes:

- Implement appropriate data structures to manipulate data for various computational problems
- Devise programs to solve complex computational problems
- Create databases using database management systems
- Develop web based applications that interact with databases

Course Contents:

- Designing and Implementing algorithms: Recursion, backtracking, Divide-and-conquer, and Dynamic programming.
- Fundamental data structures and their applications: Arrays, Lists, Stacks, Queues, Linked lists, Trees, and Graphs
- Database design, modeling and development: SQL (MySQL, MariaDB) and NoSQL (MongoDB, PostgreSQL)
- Develop web based applications: Web development using HTML, CSS and Scripting languages (PHP, JavaScript, JQuery, NodeJS)

Teaching/Learning Methods:

Lectures, Laboratory practical sessions, Guided learning, Assignments, Continuous practical recordings

Assessment Strategy:

- In-course Assessments
  - Assessment on practical records 10%
  - End-of-First Semester Practical Assessment 30%
- End-of-Second Semester Practical examination 60%

References:

- P. Deitel and H. Deitel, Java How to Program (Early Objects), 10th Ed, Prentice Hall, 2014.
- R. Sedgewick and K. Wayne, Algorithms, 4th Ed., Addison Wesley Publishers, 2011.
- N. Karumanchi, Data Structures and Algorithms Made Easy: Data Structures and Algorithmic Puzzles, 5th Ed, 2016
- D. Kalemis, The Fundamental Concepts of Object-Oriented Programming, 2013.
- R. Elmasri and S. B. Navathe, Fundamentals of Database Systems, 7th Ed, Addison-Wesley, 2015.
- D. Bartholomew, Getting Started with MariaDB, 2nd Ed, 2015.

Course Code:	CSC203S2		
Course Title:	Operating Systems		
Credit Value:	02		
Core/Optional:	Core		
Hourly Breakdown:	Theory	Practical	Independent Learning
	30	--	70

Objectives:

Provide fundamental concepts and functionalities of operating systems.

Intended Learning Outcomes:

- Describe the objective and functions of modern operating systems
- Explain how resources (such as CPU, memory, storage, file and devices) are managed by the operating system
- Demonstrate the operations of a prototypical process manager
- Compare various techniques used for concurrency control

Course Contents:

- Introduction to operating system: Architecture of modern operating systems (OS), evolution of OS, OS operations and functionalities, and open source OS
- Processes and Threads: Concept of process, process states, process control block, schedulers, context switch, interprocess communication, process scheduling, overview of threads, multicore programming and multithreading models
- Concurrency: Process synchronisation (race condition, critical-section problem, mutex locks, semaphores, classic problems of synchronization and monitors), deadlock (characterization, prevention, avoidance, detection and recovery)
- Memory management: Swapping, memory allocation, fragmentation, paging, segmentation, virtual memory and address translation
- Storage management: Mass Storage, host attached storage, network attached storage, storage area network and RAID
- File and I/O Device management: File organization and access, file system security, device drivers, direct memory access and interrupt handling

Teaching/Learning Methods:

Lectures, Case studies, Use of chalkboard, Simulation, Recitation oral questions, Guided learning, Tutorial discussions

Assessment Strategy:

- In-course Assessments 30%
- End-of-course Examination 70%

References:

- W. Stalling, Operating systems: Internals and Design Principles, 8th Ed, Pearson, 2014.
- A. Silberschatz, P. B. Galvin, G. Gagne, Operating System Concepts, 9th Ed, 2013.
- S. Tanenbaum and H. Bos, Modern Operating Systems, 4th Ed, 2014.

Course Code:	CSC204S2		
Course Title:	Data Structures and Algorithms		
Credit Value:	02		
Core/Optional:	Core		
Hourly Breakdown:	Theory	Practical	Independent Learning
	30	--	70

Objectives:

Introduce common data structures and standard algorithms for solving various types of problems.

Intended Learning Outcomes:

- Analyse the correctness and the performance of complex algorithms
- Discuss the implementation of standard data structures
- Demonstrate skills in solving complex computational problems using suitable data structures
- Explain the divide-and-conquer paradigm and dynamic programming strategies and their usages

Course Contents:

- Proof of correctness of algorithms: Contrapositive and contradiction, Induction, and Loop invariants
- Recurrence relations: Analysis of iterative and recursive algorithms (Quick sort and merge sort, etc.)
- Fundamental Data Structures: Arrays, Lists, Stacks, Queues, Linked lists, Trees, and Graphs
- Algorithm Design and Implementation Techniques: Divide-and-conquer paradigm, Dynamic programming algorithms, Recursive, and backtracking
- Applications of Trees and Graphs: Binary search, Dijkstra's shortest path, minimum spanning tree

Teaching/Learning Methods:

Lecture, Class discussions, Tutorial discussions, Assignments

Assessment Strategy:

- In-course Assessments 30%
- End-of-course Examination 70%

References:

- T. Cormen, C. Leiserson, R. Rivest, and C. Stein, Introduction to Algorithms, 3rd Ed., MIT Press, 2009.
- R. Sedgewick and K. Wayne, Algorithms, 4th Ed., Addison Wesley Publishers, 2011.
- N. Karumanchi, Data Structures and Algorithms Made Easy: Data Structures and Algorithmic Puzzles, 5th Ed. 2016.
- S. S. Skiena, The Algorithm Design Manual, 2nd Ed., Springer, 2011.

Course Code:	CSC205S2		
Course Title:	Software Engineering		
Credit Value:	02		
Core/Optional:	Core		
Hourly Breakdown:	Theory	Practical	Independent Learning
	30	--	70

Objectives:

Introduce all phases of the life cycle of a software system including requirements analysis and specification, design, construction, testing, deployment, operation, and maintenance.

Intended Learning Outcomes:

- Discuss the software engineering principles and life-cycle
- Identify different roles played by personnel in software development and their responsibilities
- Construct software designs based on user requirements
- Apply appropriate techniques in software development, testing, maintenance, and evolution

Course Contents:

- Introduction to Software Engineering: Software characteristics, impact of software, importance of engineering approaches, challenges and ethics in software development
- Introduction to systems analysis and design: Types of systems (transaction processing, management information, decision support, etc.), need for systems analysis and design, the system development life cycle, roles played by different personnel in system development life cycle including the role of the systems analyst
- Software development process models: Waterfall model, prototyping model, spiral model, evolutionary model, iterative model and agile methodology
- Software requirements and specifications: Types of requirements, requirement gathering processes and techniques, documenting requirements
- Software analysis techniques: Data flow diagrams, data dictionaries, process specifications and structured decisions
- Software design techniques: Object-oriented design using UML, Agile methodologies using SCRUM
- Software testing: Development testing, test-driven development, release testing and user testing
- Software maintenance and evolution: Evolution processes, program evolution dynamics, software maintenance and legacy system management

Teaching/Learning Methods:

Lecture, case studies, Recitation oral questions, Small groups discussions, Guided learning

Assessment Strategy:

- In-course Assessments 30%
- End-of-course Examination 70%

References:

- I. Sommerville, Software Engineering, 10th Ed, 2015.
- K. E. Kendall and J. E. Kendall, System Analysis and Design, 9th Ed, 2013.
- R. E. Beasley, Software Engineering: Principles and Practices, 2nd Ed, 2015.

Course Code:	CSC206S4		
Course Title:	Mathematics for Computing III		
Credit Value:	04		
Core/Optional:	Core		
Hourly Breakdown:	Theory	Practical	Independent Learning
	60	--	140

Objectives:

Encourage computer science students more aware of the importance of linear algebra in various computer science topics.

Intended Learning Outcomes:

- Develop an understanding of the theory of vector spaces.
- Use the theory of linear transformations and their matrix representation
- Solve systems of linear equations and understand the conditions for the existence of solution
- Use determinations and spectral properties.

Course Contents:

Vectors in R^n norms and inner products in R^n Cauchy-Schwartz and triangular inequalities, Gram-Schmidt process. Elementary operations and elementary matrices, echelon and row reduced echelon matrices. Vector spaces, linear dependence, and independence, subspaces, basis and dimension, Steinitz replacement theorem. Linear transformations, matrix representation and change of base, column rank, row rank and nullity of matrix. Determinants and their properties, invertibility of a square matrix, Eigen values and Eigen vectors, characteristic polynomials, Cayley-Hamilton theorem, orthogonal, symmetric and skew symmetric matrices, quadratic forms, diagonalization, System of linear equations.

Teaching/Learning Methods:

Use of chalkboard, Tutorial, Textbook assignments, Guided learning

Assessment Strategy:

- In-course Assessments 30%
- End-of-course Examination 70%

References:

- Devi Prasad, Elementary Linear Algebra, 2nd Ed., Narosa Publishing House, New Delhi, 2012
- David Lay C, Linear Algebra and Its Applications, 4th Ed., Pearson (Addison Wesley) Publication, 2012;
- Seymour Lipschutz, Schaum's Theory and problems of linear algebra, 2011
- Datta K.B, Matrix and Linear Algebra, Prentice hall of India Pvt. Ltd, New Delhi–110001, 2003

Course Code:	**CSC207S3**		
Course Title:	**Computer Architecture**		
Credit Value:	**03**		
Core/Optional:	**Core**		
Hourly Breakdown:	**Theory**	**Practical**	**Independent Learning**
	30	45	75

Objectives:

Understand the design of a digital computer including the structure of a microprocessor, memory organisation and program execution cycle.

Intended Learning Outcomes:

- Explain the conceptual design and the organisation of a computer system
- Describe processor unit design and its operations
- Summarise memory and Input/output organisation
- Build Assembly language programs

Course Contents:

- Introduction to modern computer architecture: Architectural and technological design and development, and performance measures of a processor
- Instruction set architecture models: Instruction set architectures and design, memory locations and operations, addressing modes, instruction types, microprogramming
- Processing unit design: CPU basics, register set, data path, CPU instruction cycle, control unit design, instruction pipelining techniques
- Memory hierarchies and Input/output organisation: Memory structure and hierarchy, cache memory mapping, direct memory access, virtual memory, interrupt-driven I/O, and Input-Output interfaces
- Assembly language programming: Instructions mnemonics and syntax, assembler directives and commands, assembly and execution of programs

Teaching/Learning Methods:

Lecture, Programming practical sessions, Tutorial discussions, Assignments, Guided learning

Assessment Strategy:

- In-course Assessment (Theory) 10%
- In-course Assessment (Practical) 30%
- End-of-course Examination 60%

References:

- D. A. Patterson and J. L. Hennessy, Computer Organization and Design: The Hardware and Software Interface, Morgan Kaufmann Publishers, 5th Ed, 2013.
- M. Abo-El-Barr and H. El-Rewini, Fundamentals of Computer Organization and Architecture, A John Wiley & Sons Publication, 2004.
- W. Stallings, Computer Organization and Architecture, Prentice Hall Publishers, 10th Ed, 2015.

Course Code:	CSC208S3		
Course Title:	Concepts of Programming Languages		
Credit Value:	03		
Core/Optional:	Core		
Hourly Breakdown:	Theory	Practical	Independent Learning
	30	30	90

Objectives:

Provide an overview of the basic concepts that appear in modern programming languages, the principles that underlie the design of programming languages, and their features.

Intended Learning Outcomes:

- Describe the fundamental issues in the design and the use of major programming languages
- Demonstrate the differences of programming paradigms in different programming languages
- Discuss the use of formal methods for program verification
- Build concurrent and functional programs

Course Contents:

- Introduction: Programming domains, evaluation criteria for programming languages, influences on language design, programming language categories
- Introduction to syntactic and semantic description of programming languages
- Programming paradigms in different programming languages: Data types, Abstract data types, Data objects, Control structures, subprograms, lifetime and scope of variables and functions, object-oriented programming, exception handling
- Concurrency: Basics of concurrency, subprogram-level concurrency, monitors, message passing, threads
- Functional programming: Fundamentals and programming with functional programming languages

Teaching/Learning Methods:

Lectures, practical sessions, Tutorial discussions, Assignments

Assessment Strategy:

- In-course Assessment (Theory) 10%
- In-course Assessment (Practical) 30%
- End-of-course Examination 60%

References:

- R. W. Sebesta, Concepts of Programming Languages, Pearson, 2016.
- J. C. Mitchell, Concepts in Programming Languages, Cambridge University Press, 2003.
- C. Ghezzi and M. Jazayeri, Programming language concepts, 3rd Ed, 1997.

Course Code:	CSC209S3		
Course Title:	Bioinformatics		
Credit Value:	03		
Core/Optional:	Core		
Hourly Breakdown:	Theory	Practical	Independent Learning
	30	30	90

Objectives:

Provide theoretical and practical knowledge in Bioinformatics including analysis of protein and genome sequences by various computational tools.

Intended Learning Outcomes:

- Describe computational genomics and phylogenetics concepts
- Demonstrate the use of computational tools for sequence analysis in bioinformatics
- Perform Data analysis and Pattern recognition in biological data
- Formulate a biological problem as a computational problem

Course Contents:

- Introduction to bioinformatics: Aims and tasks of bioinformatics, scope of bioinformatics and its applications, bioinformatics databases.
- Structural bioinformatics: Protein structure and its visualisation, comparison and classification, protein structure prediction, RNA structure prediction, compression of genomic sequences such as Burrows–Wheeler transform, etc.
- Pairwise sequence alignments and database search: Scoring matrix, Needleman-Wunsch algorithm, Smith-Waterman algorithm, Gotoh algorithm, heuristic methods
- Phylogenetic tree and multiple sequence alignment: Neighbour-joining and UPGMA algorithms, phylogenetic tree, Sequence profile & profile based alignments
- Pattern Recognition: Clustering and visualisation, Hidden Markov models and Viterbi algorithm
- Genomics and proteomics: Genome mapping, genome assembly, genome comparison, functional genomics, proteomics and metabolomics

Teaching/Learning Methods:

Lectures, Practical demonstration, recitation oral questions, vocabulary drills, and simulations.

Assessment Strategy:

- In-course Assessment (Theory) 10%
- In-course Assessment (Practical) 30%
- End-of-course Examination 60%

References:

- B. Bergeron, Bioinformatics Computing, Prentice Hall, 2002.
- K. Stephen, Introduction to Bioinformatics: A Theoretical and Practical Approach, 1st Ed, 2003.
- F. Azuaje and J. Dopazo, Data Analysis and Visualization in Genomics and Proteomics, John Wiley, 1st Ed, 2005

Course Code:	**CSC210S3**		
Course Title:	**Web Technologies**		
Credit Value:	**03**		
Core/Optional:	**Core**		
Hourly Breakdown:	**Theory**	**Practical**	**Independent Learning**
	15	60	75

Objectives:

Develop proficiency in designing web applications using different emerging technologies and best practices.

Intended Learning Outcomes:

- Design websites using advanced features of Markup and Client-side scripting languages
- Use XML technologies for web applications
- Employ knowledge on web programming to develop and maintain web applications
- Develop secure web-based systems using server-side scripting languages
- Recommend practices that ensure legal and ethical responsibilities

Course Contents:

- Advanced use of scripting languages: Client-side scripting (HTML, CSS, JavaScript, etc.) and Server-side scripting (PHP, JSP, ASP, etc.)
- XML Technologies: XSL, XSLT, xPath and xQuery
- Secure web programming: Authentication, access control, session management, SQL injections and cross site scripting (XSS)
- Trends in Web development: Web 2.0, AJAX, JSON, Web Services
- Best practices in Web Development: Architectural patterns, search engine optimization (SEO), frameworks, auditing and logging

Teaching/Learning Methods:

Lectures, practical demonstration, assignments, small group discussions, individual mini projects

Assessment Strategy:

- In-course Assessments (Theory) 10%
- In-course Assessments (Practical) 30%
- End-of-course practical Examination 60%

References:

- S. Purewal, Learning Web App development, 1st Ed., 2014.
- D. Stuttard and M. Pinto, The Web Application Hacker's Handbook: Finding and Exploiting Security Flaws, 2nd Ed., 2011.
- J. J. Jackson, Web Technologies: A Computer Science Perspective, 1st Ed., 2006.
- A. Godbole and A. Kahate, Web Technologies, TCP/IP, Web/Java Programming, and Cloud Computing, McGraw Hill Education, 3rd Ed, 2017.

Course Code:	CSC211S2		
Course Title:	Emerging Trends in Computer Science		
Credit Value:	02		
Core/Optional:	Core		
Hourly Breakdown:	Theory	Practical	Independent Learning
	15	30	55

Objectives:

Provide an overview of the emerging trends in computer science.

Intended Learning Outcomes:

- Demonstrate familiarity with latest trends in computer science and their applications
- Describe the key architectures and applications in edge computing
- Summarise standard open-source cloud and edge computing software for data analytics
- Build microcontroller programs for IoT
- Discuss the latest languages and frameworks used in IT industries

Course Contents:

- Edge computing: Introduction to edge computing, cloud computing analytics pipeline, cloud databases
- Data analytics: Introduction to deep learning, data mining, and its applications; introduction to Hadoop, Spark, and MapReduce
- Internet of things (IoT): IoT concepts and technologies, its applications, micro-controller programming using sensors and actuators with arduino, IoT security and privacy issues
- Blockchain: Fundamentals of blockchain, distributed ledger technology, cryptocurrency, and related algorithms
- Introduction to mobile application development: Mobile app development platforms (Android, iOS, etc.), development and deployment of applications

[The course content will come directly from research papers, articles, and documentation of cloud and data center architectures and technologies.]

Teaching/Learning Methods:

Lectures, Guest lectures, TechTalks, workshops, industrial visit

Assessment Strategy:

- Formative Assessment: Industrial Visit*
- Summative Assessment: Individual/Group Assignments†

*Students will be taken to four to six leading software development companies in Sri Lanka in one or two industrial visit(s). Each visit may take up to three days. The type of assignments includes but are not limited to presentations and report writings on the observation of the industrial visit.

†At the end of each of the five chapters, students will be given five assignments (including programming tasks) based on the key areas covered in the five chapters. Of the five assignments, at most two may be done in groups.

Course Code:	**CSC212S2**		
Course Title:	**Professional Practice**		
Credit Value:	**02**		
Core/Optional:	**Core**		
Hourly Breakdown:	**Theory**	**Practical**	**Independent Learning**
	30	--	70

Objectives:

Provide a viewpoint on the commercial realities of software professionals and their required behavioural skills in day to day activities as an Information Technology professional.

Intended Learning Outcomes:

- Discuss the concepts of professional practice in computing
- Explain the context in which computer professionals work
- Apply the key skills, knowledge, attributes and attitudes required to be an IT professional, with particular reference to professional practice, code of ethics and professional standards
- Analyse legal issues in relation to data privacy and software use
- Recognize professional conduct in an ethical manner in day to day activities as an IT professional
- apply the principles of group work and reflect on the nature of working in teams, with the appreciation of the issues, such as ethics, conflict resolution, negotiation in culturally diverse workplace

Course Contents:

- Computer ethics and professional practice: Ethical argumentation and theories, moral assumptions and values, the role of computing professional, and professional communication practices
- Intellectual property: Intellectual property rights, Intangible digital intellectual property, legal foundations for intellectual property protection
- Privacy and data protection: Privacy of computer data, respecting human dignity, protecting data stored on computers, ethical hacking and its implications
- Security policies, laws and computer crimes: Computer crimes and legal redress for computer criminals, Issues surrounding the misuse of access and breaches in security, crime prevention strategies

Teaching/Learning Methods:

Lecture, small group discussions, tutorial classes

Assessment Strategy:

- In-course Assessments 30%
- End-of-course Examination 70%

References:

- G. W. Reynolds, Ethics in Information Technology, 5th Ed, 2014.
- M. F. Bott, Professional Issues in Information Technology, The British Computer Society, 2nd Ed, 2014.
- ACM Code of Ethics, ACM, www.acm.org, 2017.