COMPUTER SCIENCE

M.SC. (CS)

This is a two year master degree programme that aims to prepare professionals for the Computer industry and educational institutes. This post graduate course focuses on theoretical as well as practicals of computer science that enables the students to be knowledgeable in programming, networking, algorithms design, web technology, theory of computer science, software engineering, compiler design, artificial intelligence, computational techniques, analytical ability etc. The programme also covers interdisciplinary courses for improving their communication skills and management skills.

PROGRAMME OUTCOMES (POs)

SCSIT has designed M.Sc.(CS) / M.Sc.(IT) programme to prepare students to attain following abilities:

- To understand both theoretical and practical concepts of computer science.
- To understand various programming languages and apply to solve real world problems from diversified domain.
- To develop better algorithms and analyze them.
- To apply software engineering principles in the development of computer software.

PROGRAMME SPECIFIC OUTCOMES (PSO's)

At the end of this programme, M.Sc.(CS) / M.Sc.(IT) student will be able to:

- Use of numerous software systems in the wide range of areas such as Internet and Web Technology, Cloud computing, Algorithms, Networking, Compiler design, and Web design, Machine learning, Artificial Intelligence, IoT etc.
- To develop better algorithms and solutions for Computing Problems.
- Understanding of latest tools and technology to undertake further research.
- Apply the modern tools and technology to produce cost effective and maintainable software

PROGRAMME SPECIFIC OUTCOMES (PSO's)

At the end of this programme, M.Sc.(CS) student will be able to:

Use of numerous software systems in the wide range of areas such as Internet and Web Technology, Cloud computing, Algorithms, Networking, Compiler design, and Web design, Machine learning, Artificial Intelligence, IoT etc.

To develop better algorithms and solutions for Computing Problems.

Understanding of latest tools and technology to undertake further research.

Apply the modern tools and technology to produce cost effective and maintainable software.

COURSE OUTCOMES

Digital Logic Design and Computer Organization

Subject Learning Outcomes

Upon completing the course:

1. Students will have knowledge of basics of Computer Organization and Architecture.

2. Students will be able to understand various components of computers and their interconnection.

3. Students will have knowledge of various types of memory and their organization

4. Students will be well aware about various external devices and their interconnection through CPU.

5. Students will have knowledge of interrupts and direct memory access technique.

6. Students will be able to understand various characteristics of instruction set.

7. They will be aware with various types of operands and operations used in an instruction set.

8. They will have knowledge of various addressing modes.

9. Students will be aware about various Processor Organization.

10. Students will have knowledge of RISC and CISC architecture.

Fundamentals of Programming and Problem Solving through C-I

Subject Learning Outcomes

Upon completing the course, students will be able to:

1. Define key concepts: Definition of Programming language, History of C language, structured program, modular program, fundamentals of programs, low level language, high level language, assembler, linkers and loaders.

2. Introduce Programming Environment: Flow charts, data types, keywords, character Set

3. Understanding control Structures: if else statements, For loop, Do while, while, break, continue

4. Understanding Basics of 1d-array, 2d-array, multi-dimensional array functions and user defined functions.

Operating Systems

Subject Learning Outcomes

Upon completing the course, students will be able to:

1. Install, configure, and maintain the operating system.

2. Perform basic file management operations.

3. Allocate and organize primary and secondary storage.

4. Manage peripheral devices.

Data Structures using C++

Learning Outcomes

Upon completing the course, students will be able to:

1. Understand well-known generic data structures such as stack, queue, tree, graph and

related algorithms

2. Design and apply appropriate data structures for solving computing problems

3. Develop computer programs to implement different linear data structures and related algorithms

4. Demonstrate the ability to construct and analyse search tree data structure

5. Demonstrate knowledge of searching and sorting algorithms and their run-time complexity

6. Demonstrate knowledge of graph algorithms

7. Recognize the associated algorithm s' operations and complexity

8. Understand the concept of time, space complexity and analyze the time and space complexities of an algorithm.

9. Think critically & Solve problems independently

Database Management System

Subject learning Outcomes

1. Introduction provides the general overview of the nature and purpose of database systems. We explain how the concept of the database systems. We explain how the concept of database system has developed, what the common features of the database system are, what the database system does for the user, and how a database system interfaces with operating systems.

2. Database design provides the overview of the database-design process, with major emphasis on the database design using the entity relationship data model. Entity relationship data model provides a high level view of the issues in database design.

3. Relation database introduces the relational model of data, covering basic concepts as well as the relational algebra. A brief introduction to integrity constraints and focus on

the most influential of the user- oriented relational languages: SQL.

4. SQL provide how to interface between a programming language and the database supporting SQL.

5. Introduction to the theory of relational database design. The theory of functional dependencies and normalization is covered, with emphasis on the motivation and intuitive understanding of each normal form. An overview of relational design and relies on an intuitive understanding of logical implication of functional dependencies. This allows the concept of normalization to be introduced prior to full coverage of functional dependency theory

Computer Architecture

Subject Learning Outcomes Upon completing the course:

1. Students will be familiar with various measuring tools and functional units of CPU.

2. They will be aware about architecture of 8088 microprocessor.

3. They will have knowledge of assembly language programming.

4. They will be aware about computer arithmetic.

5. They will have understanding about various types of instruction formats and addressing modes.

6. They will have knowledge of various types of processor organization and about RISC and CISC features.

7. They will have understanding about the processing unit design

8. Students will be able to explore the pipelining concepts and its implementation.

9. They will have the knowledge of parallel processing and introduction to super scalar processors.

10. They will be aware about instruction level parallelism.

11. They will be well familiar with the concept of cache and their implementation

12. They will have knowledge about interrupts and input output organization.

Discrete Structures

Subject Learning Outcomes

Upon completing the course, students will be able to:

1. Define key terms, sets, set operations, fuzzy sets, and functions for computer science. Student will understand the importance of sequences and summations, fuzzy sets in computer science.

2. Student will understand terms like combinatory, Pigeon hole principle, generalized Pigeon hole principle. Students will be able to solve problems of computing internet addresses, solving recurrence relations of different sorting problems and will be able to compute time complexity of simple algorithms.

3. Students will develop an understanding about relations and their properties, nary relations and their applications, representing relations, closures of relations, equivalence relations, partial ordering and concepts of least upper bond, greatest lower bond, maximal element, minimal element, greatest element, least element of a partially ordered set.

4. Student will develop an understanding of the concepts of graph and tree, graph representation, graph terminology, graph types, graph models, and graph isomorphism, connectivity, Euler and Hamiltonian Paths, shortest path problems, planar graphs, graph colouring, chromatic number, and Euler's formula. Applications of trees, tree traversal, Spanning trees, minimum spanning trees.

5. Student will be acquainted with language theory and will categorize language types. Student will get a clear understanding of derivation trees, parsing concepts.

Object Oriented Programming Using JAVA

Subject Learning Outcomes

Upon completing the course, students will be able to:

1. Understand and evaluate the essential costs and benefits of implementing Java programs, Apply basic concepts for writing, compiling and executing Java programs.

2. Develop Java programs with the standard program design principles of sequence, selection, repetition and data structures using arrays.

3. Create Java applications using basic object-oriented design techniques, Plan and develop robust Java applications using the advanced object-oriented design techniques of inheritance and polymorphism

4. Design standardized Java libraries using packages and interfaces, Design and develop user friendly graphical user interface Java applications using standard visual components, Design, and develop Java applets using effective design principles.

5. Describe and develop fault tolerant applications with basic exception handling using structured programming techniques, describe the advantages, and develop basic multiprocessing applications using multithread techniques.

6. Develop Java applications with simple and advanced input/output streams, Describe the advantages of designing organized classes in collection frameworks, Applying collection framework to various types of application.

7. Master basic JAVA library and tools at a depth that is sufficient to solve real-world programming problems.

Software Engineering

Subject Learning Outcomes

Upon completing the course, students will be able to:

Knowledge and Understanding of

A1) the system development lifecycle;

A2) a wide range of principles and tools available to the software developer, such as software process methodologies, choice of algorithm, language, software libraries and user interface technique; 2

A3) the principles of object-oriented software construction;

A4) the software-development process, including requirements analysis, design, programming, testing and maintenance;

A5) the range of situations in which computer systems are used, the ways in which people interact with them;

A6) professional issues to cover: social, ethical and legal aspects;

A7) communication issues in large, complex software projects;

A8) the principles and techniques of a number of application areas informed by the research directions of the subject, such as software engineering, net-centric, and distributed systems.

B- Intellectual (thinking) skills - able to

B1) model object-oriented software systems;

B2) investigate and improve the specification of a software system;

B3) design and plan software solutions to problems using an object-oriented strategy;

B4) identify a range of solutions and critically evaluate and justify proposed design solutions;

B5) write and test programs using at least one object-oriented programming language;

B6) evaluate systems in terms of general quality attributes and possible trade-offs presented within the given problem;

B7) use and evaluate appropriate tools and techniques

B8) reflect and reason concerning a given information handling problem or opportunity.

C- Practical skills - able to

C1) specify, design and construct CASE tools and application software;

C2) use logic and discrete mathematics to specify software elements;

C3) develop and apply testing strategies for software applications;

C4) develop software applications in a development environment that makes use of commonly supported tools;

C5) identify some of the main risks of software development and use;

C6) use network information services

C7) Prepare and deliver coherent and structured verbal and written technical reports;

C8) use the scientific literature effectively and make discriminating use of Web resources;

C9) analysis of system requirements and the production of system specifications;

C10) use appropriate computer-based design support tools.

D- Transferable skills - able to

D1) effectively participate in team-based activities;

D2) structure and communicate ideas effectively, both orally, in writing, and in cases involving a quantitative dimension;

D3) use IT skills and display mature computer literacy;

D4) work independently and with others;

D5) manage learning and self-development, including time management and the development of organizational skills;

D6) display personal responsibility by working to multiple deadlines in complex activities;

D7) undertake practical training and placements in relevant organizations

D8) appreciate the need for continuing professional development and in recognition of the need for lifelong learning In order to provide students with the “life long learning” attitude, the teaching method is essentially based on self learning (3 hours in class rooms and 6 hours out of class rooms: coursework, practical works, workshops, seminars, etc.)

Database Applications and Tools

Subject Learning Outcomes

Upon completing the course, students will be able to:

1. Define key terms, database, database management system, enterprise data model, data warehouse, user view, constraints etc. Limitations of file system, database approach and its advantages, components of database environment and evolution of database systems

2. Define terms like, information architecture, information engineering, top-down planning, SDLC, CASE, conceptual and physical schema. Understand SDLC, RAD approaches, Different schemas, and scope of database design.

3. Understand key terms: business rule, fact, entity, attribute, Entity Relationship model, degree and cardinality of relationships. Understand importance of data modeling process, Understand unary, binary, ternary relationships, draw ER model for several business situations. Modeling examples from live cases and practice.

4. Understand concepts enhanced ER model, terms like, subtype, supertype, generalization, specialization, disjoint overlap, hierarchy etc. develop a supertype and subtype hierarchy, and solve a problem showing EER model.

5. Understand conceptual terms: key, primary, secondary, composite, foreign, relation, structured relation, anomaly, functional dependency, normalization, and its different forms. Give concise definition of first, second, third normal forms, Understand transforming ER model into Relational model, create relations and normalize the relations.

6. Define and understand the conceptual terms, DDL, DML, DCL, views. Understand and write SQL commands in DDL, DML, DCL, practice queries, Relational algebra, Advanced SQL like, join, and types of joins, triggers and stored procedures. Examples on ACESS and ORACLE environment.

7. Understand the Concept of object oriented modeling, terms like, object, class, and query operations, life cycle of object oriented model development, contrasting with ER model, use

of UML diagram in modeling etc.

8. Develop any multi-phased project as a part of a team

Computer Graphics and Multimedia

Subject Learning Outcomes

Upon completing the course, students will be able to:

1. To learn the basic concepts used in computer graphics.

16

2. To implement various algorithms to scan, convert the basic geometrical primitives,

transformations, area filling and clipping.

3. To describe the importance of viewing and projections.

4. To define the fundamentals of animation and its related technologies.

5. To understand a typical 2 D and 3D graphics pipeline

Linux/Unix Administration

Subject Learning Outcomes

Upon completing the course, students will be able to:

Developing applications in Unix environment and administrating Unix OS.

1. Understanding will be devolved about various OS and usage.

2. Basic commands to use in UNIX.

3. Understanding with file systems.

4. Learn about how to work in editors.

5. Concepts of shell programming and system call will be developed.

6. Understanding of communication facilities used in UNIX.

7. Practicing administrative commands.

Design and Analysis of Algorithms

Subject Learning Outcomes

Students who complete the course will have demonstrated the ability to do the following:

1. Argue the correctness of algorithms using inductive proofs and invariants.

2. Analyze worst-case running times of algorithms using asymptotic analysis.

3. Describe the divide-and-conquer paradigm and explain when an algorithmic design situation calls for it. Recite algorithms that employ this paradigm. Synthesize divideand-conquer algorithms. Derive and solve recurrences describing the performance of divide-and-conquer algorithms.

4. Describe the dynamic-programming paradigm and explain when an algorithmic design situation calls for it. Recite algorithms that employ this paradigm. Synthesize dynamic-programming algorithms, and analyze them.

5. Describe the greedy paradigm and explain when an algorithmic design situation calls for it. Recite algorithms that employ this paradigm. Synthesize greedy algorithms, and analyze them.

6. Explain the major graph algorithms and their analyses. Employ graphs to model engineering problems, when appropriate. Synthesize new graph algorithms and algorithms that employ graph computations as key components, and analyze them.

7. Explain the different ways to analyze randomized algorithms (expected running time, probability of error). Recite algorithms that employ randomization. Explain the difference between a randomized algorithm and an algorithm with probabilistic inputs.

8. Analyze randomized algorithms. Employ indicator random variables and linearity of expectation to perform the analyses. Recite analyses of algorithms that employ this method of analysis.

9. Explain what competitive analysis is and to which situations it applies. Perform competitive analysis.

10. Compare between different data structures. Pick an appropriate data structure for a design situation.

Explain what an approximation algorithm is, and the benefit of using approximation

Compiler Design

Subject Learning Outcomes

Students completing this course should be able to:

1. Understand the structure of compilers.

2. Understand the basic techniques used in compiler construction such as lexical analysis, top-down, bottom-up parsing, context-sensitive analysis, and intermediate code generation.

3. Understand the basic data structures used in compiler construction such as abstract syntax trees, symbol tables, three-address code, and stack machines.

4. Design and implement a compiler using a software engineering approach.

5. Use generators (e.g. Lex and Yacc)

6. Learn the principles of Compiler designing and study how to build a compiler.

7. Understand different functions of Compilers.

Computer Networks

Subject learning outcomes:

Upon completing the course:

1. Familiarity with network terminologies, reference model, applications of network, design issues and how computer network works?

2. Knowledge of Data link layer design issues, Framing, Error correction and Detection techniques.

3. Meaning of flow control and its methods.

4. Problems associated with broadcast network and multiple access control protocols.

5. Knowledge of IEEE 802.3, 802.4 and 802.5, 802.11

6. Latest LAN examples.

7. Design issues related to Network layer like routing, addressing and their protocols.

8. Introductory knowledge of Transport layer protocols like TCP and UDP.

9. Idea about client server architecture and working of DNS, HTTP and E Mail.

10. Security issues in computer network and Introduction to Cryptographic algorithms and Digital Signature.

Internet and Web Technology

Subject Learning Outcomes

Upon completing the course, students will be able to:

1. Identify the need of dynamic website programming.

2. Understand the concept of Web servers and Application servers and Application of configuration files. Gain the idea of various dynamic web programming technology.

3. Understand the servlet model for the dynamic web programming. Developing and deploying the basic servlet application. Understand the lifecycle of the servlet and the various classes from servlet package provided in the API.

4. Understand HTML form handling using servlets. Understand the concept of session handling. To get detail understanding of the various sessions handling techniques. Understand the idea of deployment descriptor and detail about the various elements of deployment descriptor.

5. Understand the need of database programming for dynamic website designing. Develop programmers using various JDBC driver types and the SQL package from the JDBC API. Also develop programmer to access metadata information.

6. Understand various types of Statement classes available in JDBC. Understand the

concept and applicability of connection pooling.

7. Understand the basics of JSP viz. lifecycle of JSP, various scarping elements of JSP. Developing JSP program. Understand better designing concept of Web application using JavaBeans.

8. To get detail understanding of the various sessions handling techniques. Understand and use the standard tag library of JSP. Developing the custom Tag Library.

9. Understand the need and concept of Hibernate. Develop the J2EE application using the Hibernate. Understand the concept of MVC design pattern and develop the web application under the MVC design pattern using Struts.