Paper Code(s): ES-201	L	P	C
Paper: Computational Methods	4	-	4

Marking Scheme:

1. Teachers Continuous Evaluation: 25 marks
2. Term end Theory Examinations: 75 marks

Instructions for paper setter:

1. There should be 9 questions in the term end examinations question paper.
2. The first (1st) question should be compulsory and cover the entire syllabus. This question should be objective, single line answers or short answer type question of total 15 marks.
3. Apart from question 1 which is compulsory, rest of the paper shall consist of 4 units as per the syllabus. Every unit shall have two questions covering the corresponding unit of the syllabus. However, the student shall be asked to attempt only one of the two questions in the unit. Individual questions may contain upto 5 sub-parts / sub-questions. Each Unit shall have a marks weightage of 15.
4. The questions are to be framed keeping in view the learning outcomes of the course / paper. The standard / level of the questions to be asked should be at the level of the prescribed textbook.
5. The requirement of (scientific) calculators / log-tables / data – tables may be specified if required.

Course Objectives :

1.	To understand numerical methods to find roots of functions and first order unconstrained minimization of functions.
2.	To introduce concept of interpolation methods and numerical integration.
3.	To understand numerical methods to solve systems of algebraic equations and curve fitting by splines.
4.	To understand numerical methods for the solution of Ordinary and partial differential equations.

Course Outcomes (CO)

CO 1	Ability to develop mathematical models of low level engineering problems
CO 2	Ability to apply interpolation methods and numerical integration.
CO 3	Ability to solve simultaneous linear equations and curve fitting by splines
CO 4	Ability to numerically solve ordinary differential equations that are initial value or boundary value problems

Course Outcomes (CO) to Programme Outcomes (PO) mapping (scale 1: low, 2: Medium, 3: High)

	PO01	PO02	PO03	PO04	PO05	PO06	PO07	PO08	PO09	PO10	PO11	PO12
CO 1	3	2	2	2	2	-	-	-	2	2	2	3
CO 2	3	2	2	2	2	-	-	-	2	2	2	3
CO 3	3	3	3	3	2	-	-	-	2	2	2	3
CO 4	3	3	3	3	2	-	-	-	2	2	2	3

UNIT-I

Review of Taylor Series, Rolle 's Theorem and Mean Value Theorem, Approximations and Errors in numerical computations, Data representation and computer arithmetic, Loss of significance in computation
Location of roots of equation: Bisection method (convergence analysis and implementation), Newton Method (convergence analysis and implementation), Secant Method (convergence analysis and implementation). Unconstrained one variable function minimization by Fibonacci search, Golden Section Search and Newton's method. Multivariate function minimization by the method of steepest descent, Nelder- Mead Algorithm.

UNIT-II

Interpolation: Assumptions for interpolation, errors in polynomial interpolation, Finite differences, Gregory-Newton's Forward Interpolation, Gregory-Newton's backward Interpolation, Lagrange's Interpolation, Newton's divided difference interpolation
Numerical Integration: Definite Integral, Newton-Cote's Quadrature formula, Trapezoidal Rule, Simpson's one-third rule, simpson's three-eight rule, Errors in quadrature formulae, Romberg's Algorithm, Gaussian Quadrature formula.

UNIT-III

System of Linear Algebraic Equations: Existence of solution, Gauss elimination method and its computational effort, concept of Pivoting, Gauss Jordan method and its computational effort, Triangular Matrix factorization methods: Dolittle algorithm, Crout's Algorithm, Cholesky method, Eigen value problem: Power method Approximation by Spline Function: First-Degree and second degree Splines, Natural Cubic Splines, B Splines, Interpolation and Approximation

UNIT - IV

Numerical solution of ordinary Differential Equations: Picard's method, Taylor series method, Euler's and Runge-Kutta's methods, Predictor-corrector methods: Euler's method, Adams-Bashforth method, Milne's method.
Numerical Solution of Partial Differential equations: Parabolic, Hyperbolic, and elliptic equations
Implementation to be done in C/C++

Textbook(s):
1. E. Ward Cheney & David R. Kincaid , "Numerical Mathematics and Computing" Cengage; 7th ed (2013).

References:
1. R. L. Burden and J. D. Faires, "Numerical Analysis", CENGAGE Learning Custom Publishing; 10th Edition (2015).
2. S. D. Conte and C. de Boor, "Elementary Numerical Analysis: An Algorithmic Approach", McGraw Hill, 3rd ed. (2005).
3. H. M. Antia, "Numerical Methods for Scientists & Engineers", Hindustan Book Agency, (2002).
4. E Balagurusamy "Numerical Methods" McGraw Hill Education (2017).

Paper Code(s): HS-203	L	P	C
Paper: Indian Knowledge System	2	-	2

Marking Scheme:

1. Teachers Continuous Evaluation: 25 marks
2. Term end Theory Examinations: 75 marks
3. This is an NUES paper, hence all examinations to be conducted by the concerned teacher.

Instruction for paper setter:

1. There should be 9 questions in the term end examinations question paper.
2. The first (1st) question should be compulsory and cover the entire syllabus. This question should be objective, single line answers or short answer type question of total 15 marks.
3. Apart from question 1 which is compulsory, rest of the paper shall consist of 4 units as per the syllabus. Every unit shall have two questions covering the corresponding unit of the syllabus. However, the student shall be asked to attempt only one of the two questions in the unit. Individual questions may contain upto 5 sub-parts / sub-questions. Each Unit shall have a marks weightage of 15.
4. The questions are to be framed keeping in view the learning outcomes of the course / paper. The standard / level of the questions to be asked should be at the level of the prescribed textbook.
5. The requirement of (scientific) calculators / log-tables / data – tables may be specified if required.

Course Objectives :

1.	To understand the Indian knowledge System.
2.	To understand the foundational concepts for science and technology.
3.	To understand the ancient Indian mathematics and astronomy.
4.	To understand the ancient Indian engineering and technology.

Course Outcomes (CO)

CO 1	Ability to understand the Indian knowledge System.
CO 2	Ability to understand and apply foundational concepts for science and technology.
CO 3	Ability to understand and apply ancient Indian mathematics and astronomy
CO 4	Ability to understand ancient Indian engineering and technology.

Course Outcomes (CO) to Programme Outcomes (PO) mapping (scale 1: low, 2: Medium, 3: High)

	PO01	PO02	PO03	PO04	PO05	PO06	PO07	PO08	PO09	PO10	PO11	PO12
CO 1	-	-	-	-	-	3	-	-	-	-	-	2
CO 2	-	-	-	-	-	3	-	-	-	2	-	2
CO 3	3	3	-	-	-	-	-	-	-	-	-	2
CO 4	3	3	-	-	-	-	-	-	-	-	-	2

UNIT-I

Indian Knowledge System (IKS) - An Introduction:
Overview of IKS - Importance of Ancient Knowledge; Defining IKS; The IKS Corpus – A Classification Framework; Chaturdaśa-Vidyāsthāna; History of IKS, Some unique aspects of IKS;
The Vedic Corpus – Introduction to Vedas; The Four Vedas and their divisions; Vedāngas; Vedic Life;
Philosophical Systems – Indian Philosophical Systems; Vedic Schools of Philosophy; Non-Vedic Philosophical Systems; Wisdom through the Ages – Purānas, Itihāsa as source of wisdom, Rāmāyana, Mahābhārata, Niti-śāstras, Subhāssitas.

UNIT-II

Foundational Concepts for Science and Technology:
Linguistics - Components of Language; Pānini's work on Sanskrit Grammar; Phonetics in Sanskrit; Patterns in Sanskrit Vocabulary; Computational Concepts in Astādhyāyi, Logic for Sentence Construction; Importance of Verbs; Role of Sanskrit in Natural Language Processing

Number System and Units of Measurement – Number System in India; Salient Features of the Indian Numeral System; Unique approaches to represent numbers; Measurements for Time, Distance and Weight; Pingala and the Binary System
Knowledge: Framework and Classification – The Knowledge Triangle; Prameya; Pramāna; Samśaya; Framework for establishing Valid Knowledge

UNIT-III

Mathematic and Astronomy in IKS:
Mathematics – Unique aspects of Indian Mathematics; Great Mathematicians and their Contributions; Arithmetic; Geometry; Trigonometry; Algebra; Binary Mathematics and Combinatorial Problems in Chandah-śāstra of Pingala, Magic Squares in India
Astronomy - Unique aspects of Indian Astronomy; Historical Development of Astronomy in India; The Celestial Coordinate System; Elements of the Indian Calendar; Āryabhatiya and the Siddhāntic Tradition; Pancānga; Astronomical Instruments; Jantar Mantar of Rājā Jai Singh Sawai

UNIT - IV

Engineering and Technology in IKS:
Engineering and Technology: Metals and Metalworking – The Indian S & T Heritage; Mining and Ore Extraction; Metals and Metalworking Technology; Iron and Steel in India; Lost wax casting of Idols and Artefacts; Apparatuses used for Extraction of Metallic Components
Engineering and Technology: Other Applications – Literary sources for Science and Technology; Physical Structures in India; Irrigation and Water Management; Dyes and Painting Technology; Surgical Techniques; Shipbuilding; Sixty-four Art Forums; Status of Indigenous S & T

Textbook(s):
1. B. Mahadevan, Vinayaka Rajat Bhat & Nagendra Pavana R.N., "Introduction to Knowledge System: Concepts and Applications" PHI (2022).

References:
1. C.M Neelakandhan & K.A. Ravindran, "Vedic Texts and The Knowledge Systems of India", Sri Sankaracharya University of Sanskrit, Kalady (2010).
2. P.P. Divakaran, "The Mathematics of India: Concepts, Methods, Connections", Springer (2018)
3. C.A. Sharma, "Critical Survey of Indian Philosophy", Motilal Banarasidass Publication (1964)
4. G. Huet, A. Kulkarni & P. Scharf, "Sanskrit Computational Linguistics", Springer (2009).
5. A.K. Bag, "History of Technology in India", Indian National Science Academy, Vol 1, (1997)

Paper Code(s): CIC-205	L	P	C
Paper: Discrete Mathematics	4	-	4

Marking Scheme:

1. Teachers Continuous Evaluation: 25 marks
2. Term end Theory Examinations: 75 marks

Instructions for paper setter:

1. There should be 9 questions in the term end examinations question paper.
2. The first (1st) question should be compulsory and cover the entire syllabus. This question should be objective, single line answers or short answer type question of total 15 marks.
3. Apart from question 1 which is compulsory, rest of the paper shall consist of 4 units as per the syllabus. Every unit shall have two questions covering the corresponding unit of the syllabus. However, the student shall be asked to attempt only one of the two questions in the unit. Individual questions may contain upto 5 sub-parts / sub-questions. Each Unit shall have a marks weightage of 15.
4. The questions are to be framed keeping in view the learning outcomes of the course / paper. The standard / level of the questions to be asked should be at the level of the prescribed textbook.
5. The requirement of (scientific) calculators / log-tables / data – tables may be specified if required.

Course Objectives :

1.	To introduce the concept of Mathematical Logic, concepts of sets, relation and functions
2.	To introduce the concept of Algorithm and number theory
3.	To understand Group theory and related examples
4.	To use Graph theory for solving problems

Course Outcomes (CO)

CO1:	Ability for constructing mathematical logic to solve problems
CO2:	Ability to Analyze/ quantify the efficiency of a developed solution (algorithm) of a computational problem
CO3:	Ability to Understand mathematical preliminaries to be used in the subsequent courses of the curriculum. This includes Boolean algebra, number theory, group theory, and combinatorics.
CO4:	Ability to Understand diverse relevant topics in discrete mathematics and computation theory with an emphasis on their applicability as mathematical tools in computer science.

Course Outcomes (CO) to Programme Outcomes (PO) mapping (scale 1: low, 2: Medium, 3: High)

	PO01	PO02	PO03	PO04	PO05	PO06	PO07	PO08	PO09	PO10	PO11	PO12
CO 1	3	3	3	2	2	-	-	-	2	2	3	3
CO 2	3	3	3	2	2	-	-	-	2	2	3	3
CO 3	3	3	3	3	2	-	-	-	2	2	3	3
CO 4	3	3	3	3	2	-	-	-	2	2	3	3

UNIT – I

Sets, Logic, and Relation: Sets, Subsets, powerset, operations on sets, Propositional Logic, Rules of inferences in propositional logic, Quantifiers, Predicates and validity, Predicate Logic, normal forms. Proof Techniques-Direct Proof, Proof by Contraposition, and proof by contradiction. Principle of inclusion and exclusion, pigeonhole principle, permutation and combination. Principle of Well Ordering, principle of mathematical induction, principle of complete induction. Relation, properties of binary relation, equivalence relation and class, closures (symmetric, reflexive, and transitive).

UNIT – II

Functions, Order relations and Boolean Algebra: Functions, Growth of functions, Permutation functions, Partially ordered sets, lattices, Boolean algebra, Minimization of Boolean Expressions. GCD, LCM, prime numbers.

Recurrence relations, solution methods for linear, first-order recurrence relations with constant coefficients, generating functions, Analysis of Algorithms involving recurrence relations, solution method for a divide-and-conquer recurrence relation. Masters theorem (with proof).

UNIT – III

Group theory: Semi-group, Monoid, Groups, Group identity and uniqueness, inverse and its uniqueness, isomorphism and homomorphism, subgroups, Cosets and Lagrange's theorem, Permutation group and Cayley's theorem (without proof), Normal subgroup and quotient groups. Groups and Coding.

UNIT – IV

Graph theory: Graph Terminology, Planar graphs, Euler's formula (proof), Euler and Hamiltonian path/circuit. Chromatic number of a graph, five color theorem (proof), Shortest path and minimal spanning trees and algorithms, Depth-first and breadth first search, trees associated with DFS & BFS, Connected components. Complexity Analysis of the graph MST.

Textbook(s):

1. B. Kolman, R. C. Busby & S.C. Ross "Discrete Mathematical Structures", 6th edition, PHI/Pearson, 2009.
2. R. L. Graham, D. E. Knuth & O. Patashnik, "Concrete Mathematics", Pearson Education, 2000.

References:

1. Neal Koblitz, "A course in number theory and cryptography", Springer – Verlag, 1994.
2. J.P. Tremblay & R. Manohar, "Discrete Mathematical Structure with Application to Computer Science," TMH, New Delhi (2000).
3. Norman L. Biggs, "Discrete Mathematics", Second edition, Oxford University Press, New Delhi (2002).
4. T .H . Cormen, C . E . Leiserson, R .L . Rivest "Introduction to Algorithms", 3rd edition, PHI/Pearson.
5. Anne Benoit, Yves Robert, Frédéric Vivien "A Guide to Algorithm Design: Paradigms, Methods, and Complexity Analysis", CRC Press, 2013.

Paper Code(s): ECC-207	**L**	**P**	**C**
Paper: Digital Logic and Computer Design	**4**	**-**	**4**

Marking Scheme:

1. Teachers Continuous Evaluation: 25 marks
2. Term end Theory Examinations: 75 marks

Instructions for paper setter:

1. There should be 9 questions in the term end examinations question paper.
2. The first (1st) question should be compulsory and cover the entire syllabus. This question should be objective, single line answers or short answer type question of total 15 marks.
3. Apart from question 1 which is compulsory, rest of the paper shall consist of 4 units as per the syllabus. Every unit shall have two questions covering the corresponding unit of the syllabus. However, the student shall be asked to attempt only one of the two questions in the unit. Individual questions may contain upto 5 sub-parts / sub-questions. Each Unit shall have a marks weightage of 15.
4. The questions are to be framed keeping in view the learning outcomes of the course / paper. The standard / level of the questions to be asked should be at the level of the prescribed textbook.
5. The requirement of (scientific) calculators / log-tables / data – tables may be specified if required.

Course Objectives :

1.	To introduce basic concepts of Boolean Algebra and Combinational Logic
2.	To introduce various sequential circuits, designing with examples
3.	To relate combination circuit design and sequential circuit design with respect to the design of a computer system
4.	To introduce machine learning, computer arithmetic, modes of data transfer with respect to I/O and Memory organization of a computer

Course Outcomes (CO) :

CO 1	Ability to understand Boolean Algebra and Design Combinational Circuits .
CO 2	Ability to understand and Design Sequential Circuits.
CO 3	Ability to understand Design of a basic computer.
CO 4	Ability to understand Input-Output and Memory Organization of a Computer.

Course Outcomes (CO) to Programme Outcomes (PO) mapping (scale 1: low, 2: Medium, 3: High)

	PO01	PO02	PO03	PO04	PO05	PO06	PO07	PO08	PO09	PO10	PO11	PO12
CO 1	3	2	3	2	2	-	-	-	3	2	2	3
CO 2	3	2	3	2	2	-	-	-	3	2	2	3
CO 3	3	2	3	3	2	-	-	-	3	2	2	3
CO 4	3	3	3	3	3	-	-	-	3	2	2	3

UNIT – I

Boolean Algebra and Combinational Logic: Review of number systems , signed, unsigned, fixed point, floating point numbers, Binary Codes, Boolean algebra – basic postulates, theorems , Simplification of Boolean function using Karnaugh map and Quine-McCluskey method – Implementations of combinational logic functions using gates, Adders, Subtractors, Magnitude comparator, encoder and decoders, multiplexers, code converters , parity generator/checker, implementation of combinational circuits using multiplexers.

UNIT – II

Sequential Circuits: General model of sequential circuits, Flip-flops, latches , level triggering, edge triggering, master slave configuration , concept of state diagram , state table, state reduction procedures , Design of synchronous sequential circuits , up/down and modulus counters , shift registers, Ring counter , Johnson counter , timing diagram , serial adder , sequence detector, Programmable Logic Array (PLA), Programmable Array Logic (PAL), Memory Unit, Random Access Memory

UNIT – III

Basic Computer organization: Stored Program, Organization, Computer registers, bus system, instruction set completeness, instruction cycle, Register Transfer Language, Arithmetic, Logic and Shift Micro-operations, Instruction Codes, Design of a simple computer, Design of Arithmetic Logic unit, shifter, Design of a simple hardwired control unit, Programming the basic computer, Machine language instructions, assembly language, Microprogrammed control, Horizontal and Vertical Microprogramming, Central Processing Unit, instruction sets and formats, addressing modes, data paths, RISC and CISC characteristics.

UNIT – IV

Computer Arithmetic, addition, subtraction, multiplication and division algorithms, Input-Output Organization, Modes of data transfer, Interrupt cycle, direct memory access, Input-Output processor, Memory Organization, Memory Hierarchy, Associative Memory, Cache Memory, Internal and external Memory, Virtual Memory.

Text Book(s)

1. M. Morris Mano, "Digital Logic and Computer Design", Pearson Education, 2016
2. M. Morris Mano, Rajib Mall "Computer System Architecture", 3rd Edition Pearson Education, 2017

References:

1. Leach, D. P., Albert P. Malvino, "Digital Principles and Applications", McGraw Hill Education, 8th Edition , 2014
2. Jain, R.P. ,"Modern Digital Electronics", McGraw Hill Education, 4th Edition , 2010
3. Floyd, Thomas L. , "Digital Fundamentals" Pearson Education, 11th Edition, 2017
4. M. Rafiquzzaman, "Fundamentals of Digital Logic and Microcomputer Design", Wiley, 5th Ed., 2005.

Paper Code(s): CIC-209	L	P	C
Paper: Data Structures	4	-	4

Marking Scheme:

1. Teachers Continuous Evaluation: 25 marks
2. Term end Theory Examinations: 75 marks

Instructions for paper setter:

1. There should be 9 questions in the term end examinations question paper.
2. The first (1st) question should be compulsory and cover the entire syllabus. This question should be objective, single line answers or short answer type question of total 15 marks.
3. Apart from question 1 which is compulsory, rest of the paper shall consist of 4 units as per the syllabus. Every unit shall have two questions covering the corresponding unit of the syllabus. However, the student shall be asked to attempt only one of the two questions in the unit. Individual questions may contain upto 5 sub-parts / sub-questions. Each Unit shall have a marks weightage of 15.
4. The questions are to be framed keeping in view the learning outcomes of the course / paper. The standard / level of the questions to be asked should be at the level of the prescribed textbook.
5. The requirement of (scientific) calculators / log-tables / data – tables may be specified if required.

Course Objectives :

1.	To introduce basics of Data structures (Arrays, strings, linked list etc.)
2.	To understand the concepts of Stacks, Queues and Trees, related operations and their implementation
3.	To understand sets, heaps and graphs
4.	To introduce various Sorting and searching Algorithms

Course Outcomes (CO)

CO 1	To be able to understand difference between structured data and data structure
CO 2	To be able to create common basic data structures and trees
CO 3	To have a knowledge of sets, heaps and graphs
CO 4	To have basic knowledge of sorting and searching algorithms

Course Outcomes (CO) to Programme Outcomes (PO) mapping (scale 1: low, 2: Medium, 3: High)

	PO01	PO02	PO03	PO04	PO05	PO06	PO07	PO08	PO09	PO10	PO11	PO12
CO 1	3	2	2	2	3	-	-	-	2	2	2	3
CO 2	3	2	2	2	3	-	-	-	2	2	2	3
CO 3	3	2	2	2	3	-	-	-	2	2	2	3
CO 4	3	2	2	2	3	-	-	-	2	2	2	3

UNIT – I

Overview of data structure. Basics of Algorithm Analysis including Running Time Calculations, Abstract Data Types, Arrays, Arrays and Pointers, Multidimensional Array, String processing, General Lists and List ADT, List manipulations, Single, double and circular lists. Stacks and Stack ADT, Stack Manipulation, Prefix, infix and postfix expressions, recursion. Queues and Queue ADT, Queue manipulation.

UNIT – II

Sparse Matrix Representation (Array and Link List representation) and arithmetic (addition, subtraction and multiplication), polynomials and polynomial arithmetic.
Trees, Properties of Trees, Binary trees, Binary Tree traversal, Tree manipulation algorithms, Expression trees and their usage, binary search trees, AVL Trees, Heaps and their implementation, Priority Queues, B-Trees, B* Tree, B+ Tree

UNIT – III

Sorting concept, order, stability, Selection sorts (straight, heap), insertion sort (Straight Insertion, Shell sort), Exchange Sort (Bubble, quicksort), Merge sort (External Sorting) (Natural merge, balanced merge and

polyphase merge). Searching – List search, sequential search, binary search, hashing methods, collision resolution in hashing.

UNIT – IV

Disjoint sets representation, union find algorithm, Graphs, Graph representation, Graph Traversals and their implementations (BFS and DFS). Minimum Spanning Tree algorithms, Shortest Path Algorithms

Textbook(s):

1. Richard Gilberg , Behrouz A. Forouzan, “Data Structures: A Pseudocode Approach with C, 2nd Edition, Cengage Learning, Oct 2004
2. E. Horowitz, S. Sahni, S. Anderson-Freed, "Fundamentals of Data Structures in C", 2nd Edition, Silicon Press (US), 2007.

References:

1. Mark Allen Weiss, “Data Structures and Algorithm Analysis in C”, 2nd Edition, Pearson, September, 1996
2. Robert Kruse, “Data Structures and Program Design in C”, 2nd Edition, Pearson, November, 1990
3. Seymour Lipschutz, “Data Structures with C (Schaum's Outline Series)”, McGrawhill, 2017
4. A. M. Tenenbaum, “Data structures using C”. Pearson Education, India, 1st Edition 2003.
5. Weiss M.A., “Data structures and algorithm analysis in C++”, Pearson Education, 2014.

Paper Code(s): CIC-211	**L**	**P**	**C**
Paper: Object-Oriented Programming Using C++	**4**	**-**	**4**

Marking Scheme:

1. Teachers Continuous Evaluation: 25 marks
2. Term end Theory Examinations: 75 marks

Instructions for paper setter:

1. There should be 9 questions in the term end examinations question paper.
2. The first (1st) question should be compulsory and cover the entire syllabus. This question should be objective, single line answers or short answer type question of total 15 marks.
3. Apart from question 1 which is compulsory, rest of the paper shall consist of 4 units as per the syllabus. Every unit shall have two questions covering the corresponding unit of the syllabus. However, the student shall be asked to attempt only one of the two questions in the unit. Individual questions may contain upto 5 sub-parts / sub-questions. Each Unit shall have a marks weightage of 15.
4. The questions are to be framed keeping in view the learning outcomes of the course / paper. The standard / level of the questions to be asked should be at the level of the prescribed textbook.
5. The requirement of (scientific) calculators / log-tables / data – tables may be specified if required.

Course Objectives :

1.	To introduce the basic Concepts of Object Oriented Programming (data types, operators and functions) using C++
2.	To introduce concepts of Classes and Objects with the examples of C++ programming
3.	To understand object oriented features such as Inheritance and Polymorphism
4.	To use various object oriented concepts (exceptional handling) to solve different problems

Course Outcomes (CO)

CO 1	Ability to have an in-depth knowledge of object oriented programming paradigm
CO 2	To be able to develop basic C++ programming skills
CO 3	To be able to apply various object oriented features using C++
CO 4	Ability to have an understanding of generic programming & standard templates

Course Outcomes (CO) to Programme Outcomes (PO) mapping (scale 1: low, 2: Medium, 3: High)

	PO01	PO02	PO03	PO04	PO05	PO06	PO07	PO08	PO09	PO10	PO11	PO12
CO 1	3	2	2	2	3	-	-	-	3	2	2	3
CO 2	3	2	2	2	3	-	-	-	3	2	2	3
CO 3	3	2	2	2	3	-	-	-	3	2	2	3
CO 4	3	2	2	2	3	-	-	-	3	2	2	3

UNIT – I

Object Oriented Programming Paradigm, Basic Concepts of Object Oriented Programming, Benefits of Object Oriented Programming, Object Oriented Languages, Applications of Object Oriented Programming, C++ Programming Language, Tokens, Keywords, Identifiers and Constants, Data Types, Type Compatibility, Variables, Operators in C++, Implicit Type Conversions, Operator Precedence, The Main Function, Function Prototyping, Call by Reference, Return by Reference, Inline Functions, Function Overloading, Friend Functions, default parameter value.

UNIT – II

Specifying a class, Member Functions, Encapsulation, information hiding, abstract data types, objects & classes, Static Member Functions, Arrays of Objects, Constructors & Destructors, Parameterized Constructors, Copy Constructors, Dynamic Constructors, Destructors, identity and behaviour of an object, C++ garbage collection, dynamic memory allocation, Explicit Type Conversions, Operator Overloading.

UNIT – III

Inheritance, inheritance methods, Class hierarchy, derivation – public, private & protected, aggregation, Inheritance Constructors, composition vs. classification hierarchies, Containership, Initialization List, Polymorphism, categorization of polymorphic techniques, polymorphism by parameter, parametric polymorphism, generic function – template function, function overriding, run time polymorphism, virtual functions.

UNIT – IV

Standard C++ classes, using multiple inheritance, persistant objects, streams and files, namespaces, exception handling, generic classes, standard template library: Library organization and containers, standard containers, algorithm and Function objects, iterators and allocators, strings, streams, manipulators, user defined manipulators, vectors.

Textbook(s):

1. Stanley B. Lippman, Josée Lajoie, Barbara E. Moo, "C++ Primer", Addison-Wesley Professional, 2012.
2. Ivor Horton, "Using the C++ Standard Template Libraries", Apress, 2015.
3. R. Lafore, "Object Oriented Programming using C++", Galgotia.

References:

1. A.R.Venugopal, Rajkumar, T. Ravishanker "Mastering C++", TMH
2. Bjarne Stroustrup, "Programming: principles and practice using C++", Addison-Wesley, 2015.
3. Bjarne Stroustrup, "A Tour of C++", Addison-Wesley Professional, 2018.
4. Bjarne Stroustrup, "The C++ Programming Language", 4th Edition, Addison-Wesley Professional, 2013.
5. Peter Van Weert and Marc Gregoire, "C++17 Standard Library Quick Reference: A Pocket Guide to Data Structures, Algorithms, and Functions", Apress (2019)
6. Rumbaugh et. al. " Object Oriented Modelling & Design", Prentice Hall
7. G . Booch "Object Oriented Design & Applications", Benjamin,Cummings.
8. E.Balaguruswamy, "Objected Oriented Programming with C++", TMH
9. Steven C. Lawlor, "The Art of Programming Computer Science with C++", Vikas Publication.
10. Slobodan Dmitrović, Modern C++ for Absolute Beginners":A Friendly Introduction to C++ Programming Language and C++11 to C++20 Standards", Apress, 2020.

Paper Code(s): ES-251	**L**	**P**	**C**
Paper: Computational Methods Lab	**-**	**2**	**1**

Marking Scheme:
1. Teachers Continuous Evaluation: 40 marks
2. Term end Theory Examinations: 60 marks

Instructions:
1. The course objectives and course outcomes are identical to that of (Computational Methods) as this is the practical component of the corresponding theory paper.
2. The practical list shall be notified by the teacher in the first week of the class commencement under intimation to the office of the Head of Department / Institution in which the paper is being offered from the list of practicals below. Atleast 10 experiments must be performed by the students, they may be asked to do more. Atleast 5 experiments must be from the given list.

Implementation to be done in C/C++

1. Program for finding roots of f(x)=0 Newton Raphson method.
2. Program for finding roots of f(x)=0 by bisection method.
3. Program for finding roots of f(x)=0 by secant method.
4. To implement Langrange's Interpolation formula.
5. To implement Newton's Divided Difference formula.
6. Program for solving numerical integration by Trapezoidal rule
7. Program for solving numerical integration by Simpson's 1/3 rule
8. To implement Numerical Integration Simpson 3/8 rule.
9. Inverse of a system of linear equations using Gauss-Jordan method.
10. Find the Eigen values using Power method.
11. Program for solving ordinary differential equation by Runge-Kutta Method.

Paper Code(s): ECC-253	**L**	**P**	**C**
Paper: Digital Logic and Computer Design Lab	**-**	**2**	**1**

Marking Scheme:
1. Teachers Continuous Evaluation: 40 marks
2. Term end Theory Examinations: 60 marks

Instructions:
1. The course objectives and course outcomes are identical to that of (Digital Logic and Computer Design) as this is the practical component of the corresponding theory paper.
2. The practical list shall be notified by the teacher in the first week of the class commencement under intimation to the office of the Head of Department / Institution in which the paper is being offered from the list of practicals below. Atleast 10 experiments must be performed by the students, they may be asked to do more. Atleast 5 experiments must be from the given list.

1. Design and implementation of adders and subtractors using logic gates.
2. Design and implementation of 4-bit binary adder/subtractor.
3. Design and implementation of multiplexer and demultiplexer.
4. Design and implementation of encoder and decoder.
5. Construction and verification of 4-bit ripple counter and Mod-10/Mod-12 ripple counter.
6. Design and implementation of 3-bit synchronous up/down counter.
7. Design and computer architecture: Design a processor with minimum number of instructions, so that it can do the basic arithmetic and logic operations.
8. Write an assembly language code in GNUsim8085 to implement data transfer instruction.
9. Write an assembly language code in GNUsim8085 to store numbers in reverse order in memory location.
10. Write an assembly language code in GNUsim8085 to implement arithmetic instruction.
11. Write an assembly language code in GNUsim8085 to add two 8 bit numbers.
12. Write an assembly language code in GNUsim8085 to find the factorial of a number.
13. Write an assembly language code in GNUsim8085 to implement logical instructions.
14. Write an assembly language code in GNUsim8085 to implement stack and branch instructions.

Paper Code(s): CIC-255	**L**	**P**	**C**
Paper: Data Structures Lab	**-**	**2**	**1**

Marking Scheme:
1. Teachers Continuous Evaluation: 40 marks
2. Term end Theory Examinations: 60 marks

Instructions:
1. The course objectives and course outcomes are identical to that of (Data Structures) as this is the practical component of the corresponding theory paper.
2. The practical list shall be notified by the teacher in the first week of the class commencement under intimation to the office of the Head of Department / Institution in which the paper is being offered from the list of practicals below. Atleast 10 experiments must be performed by the students, they may be asked to do more. Atleast 5 experiments must be from the given list.

1. Implement sparse matrix using array. Description of program:
   a. Read a 2D array from the user.
   b. Store it in the sparse matrix form, use array of structures.
   c. Print the final array.
2. Create a linked list with nodes having information about a student and perform
   a. Insert a new node at specified position.
   b. Delete of a node with the roll number of student specified.
   c. Reversal of that linked list.
3. Create doubly linked list with nodes having information about an employee and perform Insertion at front of doubly linked list and perform deletion at end of that doubly linked list.
4. Create circular linked list having information about a college and perform Insertion at front perform Deletion at end.
5. Implement two stacks in a using single array.
6. Create a stack and perform Push, Pop, Peek and Traverse operations on the stack using Linked list.
7. Create a Linear Queue using Linked List and implement different operations such as Insert, Delete, and Display the queue elements.
8. Implement Experiment-2 using liked list.
9. Create a Binary Tree and perform Tree traversals (Preorder, Postorder, Inorder) using the concept of recursion.
10. Implement insertion, deletion and traversals (inorder, preorder and postorder) on binary search tree with the information in the tree about the details of an automobile (type, company, year of make).
11. Implement Selection Sort, Bubble Sort, Insertion sort, Merge sort, Quick sort, and Heap Sort using array as a data structure.
12. Perform Linear Search and Binary Search on an array. Description of programs:
    a. Read an array of type integer.
    b. Input element from user for searching.
    c. Search the element by passing the array to a function and then returning the position of the element from the function else return -1 if the element is not found.
    d. Display the position where the element has been found.
13. Implement the searching using hashing method.
14. Create a graph and perform DFS and BFS traversals.

Paper Code(s): CIC-257	**L**	**P**	**C**
Paper: Object-Oriented Programming Using C++ Lab	**-**	**2**	**1**

Marking Scheme:
1. Teachers Continuous Evaluation: 40 marks
2. Term end Theory Examinations: 60 marks

Instructions:
1. The course objectives and course outcomes are identical to that of (Object-Oriented Programming Using C++) as this is the practical component of the corresponding theory paper.
2. The practical list shall be notified by the teacher in the first week of the class commencement under intimation to the office of the Head of Department / Institution in which the paper is being offered from the list of practicals below. Atleast 10 experiments must be performed by the students, they may be asked to do more. Atleast 5 experiments must be from the given list.

1. Write a program for multiplication of two matrices using OOP.
2. Write a program to perform addition of two complex numbers using constructor overloading. The first constructor which takes no argument is used to create objects which are not initialized, second which takes one argument is used to initialize real and imag parts to equal values and third which takes two argument is used to initialized real and imag to two different values.
3. Write a program to find the greatest of two given numbers in two different classes using friend function.
4. Implement a class string containing the following functions:
    a. Overload + operator to carry out the concatenation of strings.
    b. Overload = operator to carry out string copy.
    c. Overload <= operator to carry out the comparison of strings.
    d. Function to display the length of a string.
    e. Function tolower( ) to convert upper case letters to lower case.
    f. Function toupper( ) to convert lower case letters to upper case.
5. Create a class called LIST with two pure virtual function store() and retrieve().To store a value call store and to retrieve call retrieve function. Derive two classes stack and queue from it and override store and retrieve.
6. Write a program to define the function template for calculating the square of given numbers with different data types.
7. Write a program to demonstrate the use of special functions, constructor and destructor in the class template. The program is used to find the bigger of two entered numbers.
8. Write a program to perform the deletion of white spaces such as horizontal tab, vertical tab, space ,line feed ,new line and carriage return from a text file and store the contents of the file without the white spaces on another file.
9. Write a program to read the class object of student info such as name , age ,sex ,height and weight from the keyboard and to store them on a specified file using read() and write() functions. Again the same file is opened for reading and displaying the contents of the file on the screen.
10. Write a program to raise an exception if any attempt is made to refer to an element whose index is beyond the array size.