Second Semester

DISCRETE STRUCTURE

COURSE CODE: BCA 151
YEAR/SEMESTER: I/II
WORKLOAD: 6 HOURS A WEEK

CREDIT HOURS: 3
TEACHING HOURS: 48
LECTURE: 3 Hrs.
PRACTICAL: 3 Hrs.

Course description:

This course is designed to build a strong foundation in discrete mathematics—the kind of math that powers the world of computer science. It sharpens logical thinking, introduces techniques for writing solid mathematical proofs, and strengthens problem-solving skills needed in programming, algorithm design, and system development. Students will dive into key topics like logic, sets, functions, relations, combinatorics, graphs, trees, and Boolean algebra, learning not just the theory but also how these concepts apply in real tech scenarios like databases, circuits, and code structure. Through hands-on practice with tools like Python, Jupyter Notebooks, NetworkX, and Graphviz, students will bring these abstract ideas to life by building models, writing logic-based programs, and exploring how mathematics directly supports computing.

Course objectives:

Upon completion of this course , the students will be able to:

- develop the ability to think logically and construct valid mathematical arguments.
- apply set theory concepts such as set operations, Venn diagrams, Cartesian products, and power sets to solve problems in databases, programming, and decision-making structures.
- analyze relations and functions using matrix and graph representations and understand applications in modeling data and structures.
- use combinatorial techniques to solve real-life counting and arrangement problems.
- explore graph and tree structures, implement traversal algorithms (DFS, BFS, etc.), and apply these concepts in computer science domains such as networking, compiler design, and operating systems.

Course contents

1 Set Theory **6 hrs.**

1.1. Basic Concepts: Sets, elements, roster and set-builder notation, cardinality.

1.2. Set Relationships:

1.2.1. Subsets

1.2.2. Proper subsets

1.2.3. Universal set

1.2.4. Complement

1.2.5. Disjoint sets.

1.3. Set Operations

1.3.1. Union

1.3.2. Intersection

1.3.3. Difference

1.3.4. Complement

1.3.5. Symmetric difference.

1.4. Venn Diagrams: Visual representation of set relationships and operations.

1.5. Set Identities: Proof of identities using algebraic and Venn diagram methods.

1.6. Cartesian Products: Ordered pairs, cross product of two or more sets.

1.7. Power Sets: Definition and computation of power sets.

1.8. Applications: Use of sets in databases, computer programming, and decision structures.

2. Logic and Propositional Calculus **8 hrs.**

2.1. Propositions and Logical Operators: Definition of propositions, types (simple, compound), logical connectives: AND, OR, NOT, IMPLICATION, BICONDITIONAL.

2.2. Truth Tables: Constructing truth tables for expressions involving logical operators.

2.3. Tautologies, Contradictions, and Contingencies: Identifying always true/false/logical expressions.

2.4. Logical Equivalence and Implications: Laws of logic (De Morgan's, distributive, associative, etc.), verifying equivalences.

2.5. Predicate Logic and Quantifiers: Introduction to predicates, universal and existential quantifiers.

2.6. Rules of Inference: Modus ponens, modus tollens, hypothetical syllogism, and others.

2.7. Proof Methods: Direct, indirect, contradiction, contrapositive, and proof by cases.

3. Relations and Functions **8 hrs.**

3.1. Relations: Definition, Binary Relation, Representation, Domain, Range, Universal Relation, Void Relation, Union, Intersection, and Complement Operations on Relations

3.2. Properties of Binary Relations in a Set : Reflexive, Symmetric, Transitive, Anti-symmetric Relations

3.3. Relation Matrix and Graph of a Relation; Partition and Covering of a Set, Equivalence Relation, Equivalence Classes, Compatibility Relation, Maximum Compatibility Block, Composite Relation, Converse of a Relation,

3.4. Transitive Closure of a Relation R in Set X, examples from real-world scenarios.

3.5. Representation of Relations: Using matrices and directed graphs (digraphs).

3.6. Equivalence and Partial Order Relations: Properties and examples, Simple or Linear Ordering, Totally Ordered Set (Chain), Frequently Used Partially Ordered Relations, Representation of Partially Ordered Sets, Hesse Diagrams, Least & Greatest Members, Minimal & Maximal Members, Least Upper Bound (Supremum), Greatest Lower Bound (infimum), Well-ordered Partially Ordered Sets (Posets). Lattice as Posets, complete, distributive modular and complemented lattices Boolean and pseudo Boolean lattices. (Definitions and simple examples only)

3.7. Closures and Composition of Relations: Reflexive, symmetric, transitive closures.

3.8. Functions: Definition, domain, co-domain, range, examples.

3.9. Types of Functions: Injective (one-to-one), surjective (onto), bijective (one-to-one correspondence).

3.10. Inverse and Composition of Functions: Definitions and computations.

3.11. Applications: Use in programming, data mapping, and relational databases.

4. Mathematical Reasoning and Proof Techniques **6 hrs.**

4.1. Mathematical Reasoning: Basic structure of arguments, logical flow.

4.2. Mathematical Induction: Principle of induction, proof by induction, applications in series and recursive definitions.

4.3. Strong Induction: Differences from regular induction, applications.

4.4. Recursive Definitions: Defining sequences and structures recursively.

4.5. Structural Induction: Proofs involving recursively defined structures like trees and lists.

4.6. Applications: Problem-solving and validation of algorithms.

5. Combinatorics and Counting Principles **5 hrs**

5.1. Basic Counting Principles: Introduction to counting, rule of sum and rule of product with real-world examples (e.g., menu combinations, clothing combinations).

5.2. Permutations and Combinations: Concepts of ordered and unordered selections, factorial notation, formulae for permutations (nPr) and combinations (nCr), applications in password generation and team selection.

5.3. Pigeonhole Principle: Understanding the concept, simple and strong pigeonhole principle, applications such as birthday paradox, drawer problems, and error checking.

5.4. Inclusion-Exclusion Principle: Set-based approach to solving overlapping sets, solving problems involving counting elements in unions of sets (up to three sets), and its application in probability and combinatorics.

6. Graph Theory and Trees 12 hrs.

6.1. Graphs: Introduction, definition, examples; Nodes, edges, adjacent nodes, directed and undirected edge, Directed graph, undirected graph, examples; Initiating and terminating nodes, Loop (sling), Distinct edges, Parallel edges, Multi-graph, simple graph, weighted graphs, examples, Isolated nodes, Null graph; Isomorphic graphs, examples; Degree, Indegree, out-degree, total degree of a node, examples

6.2. Subgraphs: definition, examples; Converse (reversal or directional dual) of a digraph, examples;

6.3. Path: Definition, Paths of a given graph, length of path, examples; Simple path (edge simple), elementary path (node simple), examples; Cycle (circuit), elementary cycle, examples;

6.4. Reachability: Definition, geodesic, distance, examples; Properties of reachability, the triangle inequality; Reachable set of a given node, examples, Node base, examples;

6.5. Connectedness: Definition, weakly connected, strongly connected, unilaterally connected, examples; Strong, weak, and unilateral components of a graph, examples, Applications to represent Resource allocation status of an operating system, and detection and correction of deadlocks;

6.6. Matrix representation of graph: Definition, Adjacency matrix, boolean (or bit) matrix, examples; Determine number of paths of length n through Adjacency matrix, examples; Path (Reachability) matrix of a graph, examples; Warshall's algorithm to produce Path matrix, Flowchart

6.7. Types of Graphs: Simple, multigraph, weighted, directed/undirected, complete, bipartite.

6.8. Graph Traversal: Breadth-First Search (BFS), Depth-First Search (DFS).

6.9. Trees: Trees: Definition, branch nodes, leaf (terminal) nodes, root, examples;

6.10. Different representations of a tree, examples; Binary tree, m-ary tree, Full (or complete) binary tree, examples;

6.11. Converting any m-ary tree to a binary tree, examples;

6.12. Representation of a binary tree: Linked-list; algorithms; Applications of List structures and graphs

6.13. Tree Traversals: Inorder, preorder, postorder traversal techniques.

6.14. Applications: Networking, pathfinding algorithms, compiler syntax trees, file systems.

7. **Algebraic Structures** **3 hrs.**

7.1. Binary Operations: Definition and examples of binary operations on sets.

7.2. Algebraic Systems: Semigroups, monoids, and groups - axioms and properties.

7.3. Group Theory Basics: Identity element, inverse, associativity, examples with integers and matrices.

7.4. Boolean Algebra: Basic postulates and theorems, duality, Boolean functions.

7.5. Logic Circuits: Simplification of logic circuits using Boolean expressions.

7.6. Applications: Automata theory, logic design, cryptography

Practical **(48 Hours)**

Instructor should encourage the use of open-source tools like Python, Jupyter Notebooks, NetworkX, and Graphviz for practical sessions.

Practical Report Contents: Theory, Source code, Output, Conclusion

1. Truth tables, logical operators, tautologies using Python or logic tools
2. Set operations implementation and Venn diagram visualization
3. Relation properties and matrix/digraph representation in code
4. Function definitions and mapping types with coding examples
5. Recursion-based programs (factorial, Fibonacci), validating inductive proofs
6. Permutations and combinations via iteration and recursion
7. Graph representation: adjacency list and matrix (using NetworkX or code)
8. Graph traversal: DFS and BFS implementations
9. Tree structures and traversals (inorder, preorder, postorder)
10. Boolean algebra simplification and expression evaluation via code
11. Mini project: Logical puzzle solver or graph simulation

Required readings:

Grimaldi, R. P. (3rd ed.). Discrete and Combinatorial Mathematics: An Applied Introduction. Pearson

Kolman, B., Busby, R. C., & Ross, S. (2000). *Discrete mathematical structures* (3rd ed.). Prentice Hall.

Liu, C. L. (1985). *Elements of discrete mathematics* (2nd ed.). McGraw-Hill.

Rosen, K. H. (2011). *Discrete mathematics and its applications* (8th ed.). McGraw-Hill.

Stein, C., & Drysdale, R. L. (2010). Discrete Mathematics for Computer Scientists. Pearson Education.

Required Journals/ Articles:

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). *Introduction to algorithms* (3rd ed.). MIT Press.

Hopcroft, J. E., & Tarjan, R. E. (1973). Algorithm for finding the strongly connected components of a directed graph. Communications of the ACM, 16(6), 372–378. https://doi.org/10.1145/362248.362272

Knuth, D. E. (1968). The art of computer programming: Volume 1: Fundamental algorithms. Addison-Wesley.

Pippenger, N. (1978). Theories of computation. Journal of Computer and System Sciences, 16(1), 99–115. https://doi.org/10.1016/0022-0000(78)90030-6

MICROPROCESSOR AND COMPUTER ARCHITECTURE

COURSE CODE: BCA 152

NATURE OF COURSE THEORY + LAB **CREDIT HRS: 3**

YEAR/SEMESTER: II /I **WORK LOAD/THEORY/LAB: 3/3 Hrs**

Course description:

This course is designed to familiarized the fundamental knowledge about computer architecture, instruction cycle, components of microprocessor, Intel 8085 and assembly programming.

Course objectives:
Upon completion of this course, students will be able to:

- Understand the basic components of a microprocessor
- Explain the block diagram of Intel 8085
- Demonstrate assembly language programming using Intel 8085
- Interpret timing diagrams, instruction cycles, and machine cycles
- Explain the role of the control unit and central processing unit (CPU)
- Differentiate between RISC and CISC architectures
- Describe the concept of Direct Memory Access (DMA)
- Explain memory organization and operations
- Understand the concept of pipelining in processors
- Describe microprogramming and microinstructions
- Perform computer arithmetic operations such as multiplication and division

Course contents:

Unit 1 Introduction to Microprocessor **3 Hours**

1.1 Definition of Microprocessor Components: Registers, ALU, Control and Timing, System Buses (Address, Data, Control), Microprocessor System with Bus Organization, Application of MP

Unit 2 8085 Microprocessor **12 Hours**

2.1 Functional Block Diagram, Pin Configuration, Description of each Block: Registers, Flag (Description of each Flag), Multiplex Address data bus (Ad0-Ad7), Timing and Control Unit, Interrupts (Introduction Only), Addressing Modes, Instruction cycle, Machine cycle (Opcode, fetch, memory read, memory write) and T states, Timing diagram of MOV,LDA, STA, MVI

2.2 8085 Instruction Set
Data Transfer: - MOV, MVI, STA, LDA, LXI, LDAX, STAX, XCHG
Arithmetic: - ADD, ADI, ADC, SUB, SUI, SBB, INR, DCR, INX, DCX,
Logic: - ANA, ANI, ORA, ORI, XRA, XRI, CMA, CMP
Branching: - JMP, JNZ, JZ, JNC, JC

2.3 Basic Assembly Language Programming using 8085 Instruction Sets Addition (8 and 16 bit), Subtraction (8 and 16 bit), Multiplication (8 bit) and Division (8 bit), Simple Sequence Program, Array Searching using branching and looping

Unit 3 8086 Microprocessor **4 Hours**

3.1 Logical block diagram and components, Bus interface unit and Execution Unit, flag, pipeline concept, Memory Segmentation, Segmentation register

Unit 4 Basic Computer Architecture and Design **6 Hours**

4.1 Stored Program Organization, Computer Registers, Common bus system, Instruction set, Timing and Control-Instruction Cycle

4.2 Micro-Operation, Arithmetic Micro Operations: Addition, Subtraction, Increment, Decrement, Logic Micro Operations: AND, OR, NOT, NAND, NOR, XOR, Shift Micro Operations: Logical, Circular and Arithmetic

Unit 5 Microprogrammed Control Unit **5 hrs.**

5.1 Hardwired vs micro program CU, Control Memory, Address Sequencing, Micro-operation, Micro instruction, Micro Instruction Format, Micro-program: Symbolic and Binary Micro-program (FETCH)

Unit 6: Central Processing Unit **6 Hours**

6.1 Introduction, General Register Organization, Stack Organization, Instruction Formats:3,2,1,0 address Instruction,

6.2 RISC and CISC architecture

Unit 7: Computer Arithmetic **3 hrs.**

7.1 Addition and Subtraction with signed magnitude data, Addition and Subtraction with signed 2's complement data, Booth Multiplication

Unit 8 Input and Output Organization and Memory Organization **5 Hours**

8.1 Introduction to Peripheral Devices, I/O interface-I/O bus and Interface Modules, Isolated versus Memory Mapped I/O, Interrupt

8.2 Direct Memory Access (DMA): Introduction, Basic DMA Procedures (DMA controller only)

8.3 Hierarchy of Memory System

8.4 Primary Memory: RAM and ROM, Memory Address Map with examples of Address Decoding.
Secondary Memory: Structure of Magnetic Disk, Cache Memory

Unit 9 Pipelining **4 Hours**

9.1 Concept of Pipelining and Flynn's Classification, Pipelining Example with Speed Up Ratio

9.2 Arithmetic Pipeline, Pipeline for Floating-point Addition and Subtraction

9.3 Instruction Pipeline: Four Segment Instruction Pipeline

9.4 Data Dependency, Handling of Branch Instruction, pipeline hazard and its solution

Laboratory Works: **48 Hours**

- Write assembly language programs using both the 8085-microprocessor trainer kit and a software-based 8085 simulators.
- Demonstrate the use of all types of instructions and various addressing modes available in the 8085-instruction set.
- Develop programs that include fundamental arithmetic operations (8-bit addition and subtraction,16-bit addition and subtraction,8-bit multiplication and division), logical operations, loops, bitwise manipulation, and branching techniques.
- Implement algorithms for computer arithmetic using high level language

Required readings

Gaonkar, R. S. (1998). *Microprocessor architecture, programming, and applications with the 8085*. Prentice-Hall, Inc.

Hall, D. V. (1986). *Microprocessors and interfacing: programming and hardware*. McGraw-Hill, Inc.

Mano, M. M. (1993). *Computer system architecture*. Prentice-Hall, Inc.

OOP IN JAVA

COURSE CODE: BCA 153
YEAR/SEMESTER: I/II
WORKLOAD: 6 HOURS A WEEK

CREDIT HOURS: 3
LECTURE: 3 Hrs.
PRACTICAL: 3 Hrs.

Course description:

This course provides an in-depth introduction to object-oriented programming (OOP) principles and their implementation using the Java programming language. This course covers the different concepts of object-oriented programming such as classes, objects, inheritance, polymorphism, abstraction, encapsulation, exception handling, multithreading collections, generics, file handling and advanced concept of OOP. The course emphasizes both theoretical understanding and hands-on programming skills.

Course objectives:

The main objective of this course to provide students both the theoretical foundation and practical knowledge of programming using Java. After the completion of the course, students will be able to:

- understand OOP principles and implement them in Java.
- install java compiler and IDE ,compilation and run of java program
- design and develop Java applications using classes, objects, and interfaces.
- use exception handling, collections, and generics and file I/O in real-world programs.
- apply Advanced OOP concept (Design pattern, Lambda, Streams API) for efficient coding.
- build a small-scale Java project using OOP best practices.

Prerequisite:

Conceptual understating on programming logic and technique.

Course Contents

Unit-1 Introduction to Java and OOP concept 4Hrs.

1.1 History, Feature or Buzzwords of Java

1.2 Java Architecture: JVM, JDK and JRE

1.3 Procedural-oriented Vs. Object-oriented Programming

1.4 Setting up java environment and IDE in Local machine

1.5 Sample java programs

1.6 Compiling and running java program

1.7 Command-line arguments

1.8 Scanner class for input

1.9 Handling common errors

Unit-2 Basic of Java Programming **8Hrs.**

2.1 Writing comments and its type

2.2 Java token: keywords, identifier, literal, operators and separators

2.3 Data types: primitive and user-defined data type.

2.4 Variable declaration and assignment, expression

2.5 Control statements: selection statements, looping statement and jump statements.

2.6 Arrays: single dimension array, multi-dimensional array (Rectangular and Jagged)

2.7 Type conversion and casting

2.8 Garbage Collection

2.9 String: creation, concatenation, comparison, modification, changing case and searching

2.10 String Buffer Class

Unit-3 Class and Objects in Java **8 Hrs.**

3.1 Defining class, adding method to class, creating object and calling function/method

3.2 Abstraction and Encapsulation

3.3 Constrictors and its type (Default, Parameterized and copy)

3.4 'this' keyword

3.5 Static fields and methods

3.6 More onmethod: passing by value, by reference

3.7 Recursion

3.8 Nested and inner class

3.9 Variable length arguments

3.10 Package: Defining and importing package

Unit-4 Inheritance and Polymorphism **6 Hrs.**

4.1 Inheritance basics

4.2 Inheritance Type(Single-level, Multi-level, Multiple and Hierarchical)

4.3 'super' keyword

4.4 Polymorphism: Method overloading and method overriding

4.5 Object class

4.6 'final' keyword

4.7 Abstract class and methods

4.8 Access control (private, protected, default and public)

4.9 Interface: Defining, implementing and applying interface

Unit-5 Exception Handling and Multithreading **6 Hrs.**

5.1 Basic exceptions, proper use of exceptions

5.2 Exception hierarchy

5.3 Exception handling keywords: try, catch, throw, throws and finally

5.4 Java's built-in exceptions

5.5 User-defined exceptions

5.6 Multithreading basics

5.7 Thread class and Runnable interface

5.8 Thread priorities

5.9 Thread synchronization and inter-thread communication

Unit-6 File Handling in Java **6 Hrs.**

6.1 Console and File I/O

6.2 Reading and writing file using byte stream

6.3 Reading and writing file using character stream

6.4 Serialization and deserialization

6.5 RandomAccessFile class

Unit-7 Collections and Generics **6 Hrs.**

7.1 Wrapper class and associate methods

7.2 Java collection framework

7.2.1 List, Set, Map interface

7.2.2 ArrayList, LinkedList, HashSet, HashMap and TreeSet Class

7.3 Accessing collections: Iterator/comparator

7.4 Defining generic class and methods

7.5 Using wildcard arguments

7.6 Generic interface and generic hierarchy

7.7 Some generic restrictions

Unit-8 Advanced OOP concepts in Java **4 Hrs.**

8.1 Design pattern: singleton, factory, observer pattern

8.2 Lambda expression

8.3 Stream API: Introduction

8.4 Optional class

8.5 Method references

Laboratory Work **48 Hrs.**

Practical report contains: Theory, source code and output.

- To install Java SE and IDE (Notepad++, NetBeans or Eclispe) on local Machine.
- To run and compile java program and implement concept of command-line argument and scanner class.
- To implement java basic concepts: Token, control statements, arrays, type casting.
- To implement method provided by String and StringBuffer class.
- To define class, object and implement other object-oriented feature like Encapsulation, Inheritance, polymorphism and Abstraction.
- To handle built-in and user-defined exception and implement multithreading concepts.
- To create, write and read file using byte stream, character stream class and use concept of serialization, deserialization and RandomAccessFile.
- To implement java collection Framework and generic concepts.
- To implement advanced OOP concepts in java
- After end of course student must submit programming project on object-oriented concept of Java.

Required readings

Balagurusamy, E. (2024). *Programming with Java* (7th ed.). McGraw Hill.

Deitel, P. J., & Deitel, H. M. (2024). *Java how to program: An objects-natural approach* (12th ed.). Pearson.

Horstmann, C. S. (2024). *Core Java, Volume I: Fundamentals* (13th ed.). Addison-Wesley Professional.

Schildt, H., & Coward, D. (2024). *Java: The complete reference* (13th ed.). McGraw Hill.

MATHEMATICS II

COURSE CODE: BCA 154

NATURE OF COURSE: THEORY/ LAB **CREDIT HOURS 3**

YEAR/SEMESTER: I/I

Course Description:

The course covers the concepts of Limit and Continuity, Derivatives of Algebraic, Trigonometric, Exponential, and Logarithmic functions, and their applications. It also covers Integration and its Applications, Volume and Surface Integral of some selected functions, and Numerical Integration using Trapezoidal and Simpson's Rule. Ordinary and Partial Differential Equations, their examples and uses are also discussed in this course. Other contents of the course include Optimization Problems like Simplex Method for two variables, and Gauss-Seidel, Gauss Elimination, Matrix Inversion, Bisection, and Newton-Raphson Method for solving the linear equations.

Course Objectives:

The objective of this course is to enable students to

- understand the concept of limit and continuity of functions and their connection to the derivative,
- differentiate different types of functions, geometrical meaning of the derivative, and its applications to real-world problems,
- integrate functions, understand their meaning and applications, including the surface and volume integrals,
- solve ordinary and partial differential equation and their connections to real-world problems,
- calculate the optimization problems, describe and interpret the graphical and numerical solutions.

Course contents:

Unit 1: Limit and Continuity **7 Hrs.**

1.1 Definition of limit including epsilon-delta condition, right and left hand limit, and its interpretation

1.2 Algebraic properties of limit

1.3 Definition and conditions of continuity and discontinuity

1.4 Continuity of algebraic, Trigonometric, and exponential functions, examples, and counterexamples.

Unit 2: Derivatives **7 Hrs.**

2.1 Definition and geometrical meaning of derivatives,

2.2 First principle (or by definition) method to differentiate algebraic, trigonometric, exponential, and logarithmic functions,

2.3 Rules of derivatives (sum, product, power, chain, and quotient rule),

2.4 Derivatives of inverse circular, hyperbolic functions and implicit functions,

2.5 Higher order derivatives,

2.6 Relation between derivative and continuity

2.7 Definition and examples of partial derivatives

Unit 3: Applications of Derivatives **8 Hrs.**

3.1 Increasing and decreasing functions,

3.2 Equation of tangents and normals using first derivatives,

3.3 L' Hospital's rule,

3.4 Angle between two lines,

3.5 Maxima and minima, absolute maxima and minima, concavity, stationary points and points of inflection,

3.6 Statement and geometrical interpretation of Rolle's theorem, Cauchy Mean-value theorem and Generalized Mean-value theorem,

3.7 Taylor's theorem, Maclaurin theorem (without proof) and its use in expansion of some simple functions,

3.8 Applications of derivatives in Economics,

3.9 Rate measures

Unit 4: Anti-derivative and its Applications **8 Hrs.**

4.1 Definition and geometrical meaning of integration,

4.2 Basic integration formulas for algebraic, trigonometric, exponential, and logarithmic functions,

4.3 Trigonometric substitution for basic functions, Integration by parts (Product rule for integration),

4.4 Partial fractions,

4.5 Improper integral,

4.6 Definite integral in terms of Riemann sum, and fundamental theorem of integral calculus

4.7 Applications of definite integral (Area under curve, area between curves, Quadrature & rectification)

4.8 Surface and volume integrals,

4.9 Trapezoidal and Simpson's Rule for numerical integration

Unit 5: Differential Equations **8 Hrs.**

5.1 Definition, order, and degree of differential equations,

5.2 Differential equations of first order and first degree,

5.3 Variables separable, homogeneous, exact, and linear differential equations,

5.4 Reducible to linear form,

5.5 Partial differential equations with some basic examples

Unit 6: Computational Methods **10 Hrs.**

6.1 Linear programming problems,

6.2 Linear inequalities in two variables and their graphical solutions,

6.3 Simplex Method (up to 3 variables), Duality problems

6.4 Matrix inversion method,

6.5 Gauss Elimination, Gauss-Seidel method,

6.6 Bisection method and Newton-Raphson Method for non-linear equations

Laboratory Works

Students are expected to use Python, MATLAB, or Mathematica to solve the numerical problems of each units and compare the solution with that of pen and paper.

Examination Scheme				
Internal Assessment		**External Assessment**		
Theory	**Practical**	**Theory**	**Practical**	
20	**20 (3hrs)**	**60 (3 Hrs)**	---	**100**

Required reading:

Bajracharya, B. C. (2025). *Basic Mathematic,* Sukunda Pustak Bhawan, Budnick, F. S. (2017). *Applied Mathematics for Business Economics and Social Sciences.* McGraw-Hill,

Chand, K. B., Sapkota, B. P. (2022). *Principles of Mathematical Analysis*. Pinnacle Publication.

Lay, D. C. (2003). *Linear algebra and its applications*. Pearson Education India.

Stewart, J., Clegg, D., Watson, S. (2020). *Calculus*. Cengage|webassign.

Thomas G. B., Finney, R. L. (1995). *Calculus and Analytical Geometry*. Narosa Publishing House. (Textbook)

UX/UI DESIGN

COURSE CODE: BCA 155
YEAR/SEMESTER: I/II **CREDIT HOURS: 3**
WORKLOAD: 6 Hrs A WEEK (THEORY: 3 Hrs, PRACTICAL: 3 Hrs)

Course description:

This course covers different design principles, concepts and techniques of UX and UI designs focusing on making products with user centric user interface which is easy to use, efficient and user-friendly. Students are introduced with the basic concepts, design techniques and evaluation of UX and UI. It also includes latest concepts used by the designers to design the UX/UI and also trends in the UX/UI design.

Course objectives:

The primary objective of this course is to introduce different principles, techniques and aspects of UX/UI design. Students should be able to:

- Develop a deep understanding of concepts of UX and UI designs
- Conduct user research and analysis for user interaction designs
- Understand the principles and techniques of interaction design
- Implement user friendly interaction styles
- Create UI designs using proper user interface components and controls
- Create wireframes, prototypes, and high-fidelity designs using industry-standard software
- Perform design evaluations on the visual designs and interactions
- Implement advanced techniques for interface design including voice user interface and NLP based interface

Course contents

Unit 1: Introduction **4 Hrs**

1.1 Fundamentals of UX and UI
1.2 UX vs UI
1.3 Tasks of UX designer and UI designer
1.4 UX principles: Usability, Accessibility, Simplicity
1.5 Core discipline of UX: User research, content strategy, Information Architecture, Interaction design, Visual Design, usability evaluation
1.6 User interfaces: CLI, GUI, VUI, Menu-driven, NLP based
1.7 Properties of good UX/UI design
1.8 UX/UI tools: Figma, Adobe XD, Sketch

Unit 2: User interaction design **4 Hrs**

2.1 UX design process and user center design, Mindmap
2.2 UX research: Conducting user research: Interviews, Surveys, Competitive analysis, creating user personas, user journey mapping

2.3 Ideation techniques: using Mood boards, Brainstorming and sketching

Unit 3: User Interface design **6 Hrs**

2.1 Graphical and web user interfaces
2.2 Interaction styles: Command line, Menu selection, Form fill in, Direct manipulation, Anthropomorphic
2.3 Principles of UI design
2.4 Graphical user interface
2.5 UI design process
2.6 Human considerations in interface and screen design
2.7 Technological considerations in interface design

Unit 4: UI components **12 Hrs**

3.1 System menus and functions of menus
3.2 Formatting of menus: Consistency, display, presentation, organization, complexity, item arrangement, ordering, grouping
3.3 Types of menus: Menu bar, pull down menu, cascading menu, popup menu, tear off menu, iconic menu
3.4 Selection of Windows and its components, window presentation styles: Tiled windows, overlapping windows, cascading windows
3.5 Types of windows: Primary, secondary windows, dialog boxes
3.6 Screen based controls: Operable controls (Buttons, toolbars), Text entry/Read-only controls (single and multiple line textboxes,) selection controls (radio buttons, checkboxes, palettes, list boxes, list view controls, drop down/popup list boxes
3.7 Other operable controls: slider, tabs, date picker, tree view, scroll bars
3.8 Selecting the proper controls
3.9 Creating meaningful graphics, icons and images

Unit 5: UI Design considerations **6 Hrs**

5.1 Page layout, Color scheme and font selection, typography, screen size and responsive designs, interactive element
5.2 Visual hierarchy principles: Alignment, Color, Contrast, Proximity, Size, Texture, Time
5.3 Navigation: Global navigation, utility navigation, Associative and Inline Navigation
5.4 Navigational models: Hub and spoke, fully connected, multilevel or tree, stepwise navigation, Pyramid navigation, flat navigation

Unit 6: Wireframing and prototyping **6 Hrs**

6.1 Wireframes and mock-ups
6.2 Prototyping: Low fidelity and high fidelity prototyping, interactive prototyping
6.3 UX storyboarding, mockups
6.4 Software prototyping
6.5 Transition and animation to prototypes
6.6 Creating a simple clickable prototype

Unit 7: Design evaluations **6 Hrs.**

7.1 Formative and summative evaluation

7.2 Usability testing: Moderated vs Unmoderated
7.3 Analyzing test results and gathering insights
7.4 Evaluation through expert analysis and user participation, iterative evaluation and evaluation paradigms
7.5 DECIDE evaluation framework, heuristic evaluation
7.6 Task analysis and performance metrics

Unit 8: Advanced techniques: VUI and NLP based UI **4 Hrs.**

8.1 Command and control vs Conversational UI
8.2 Personas, Avatars, Actors and Video games
8.3 Speech recognition technology and Dialog management
8.4 Designing for Wearable Devices

Laboratory Works: **48 Hrs.**

Laboratory work should be carried out by using some UX/UI tools such as Figma, Adobe XD or Sketch and need to cover the following tasks:

- Use the basic tools and features for visual designs
- Use proper page layout, font selection, screen size and responsive designs
- Implement color theory and typography in interface designs
- Create designs with proper grids and layouts
- Integrating interactive elements to the designs
- Create design with different types of windows and operable controls
- Use different navigational styles in the designs
- Construct wireframe and prototypes for the designs
- Create the storyboards for interaction designs
- Apply different effects, styles and animation for designs

At the end of the semester, students need to submit a project that should include the basic interface and interaction designs for a chosen topic individually and perform the evaluations on the designs.

Required readings:

Galitz, W. O. (2007). *The essential guide to user interface design* (3rd ed.). Wiley Publishing.

Gothelf, J., & Seiden, J. (2013). *Learn UX: Applying lean principles to improve user experience* (1st ed.). O'Reilly Media.

Green, T., & Brandon, K. (2024). *UX design with Figma: User-centered interface design and prototyping with Figma.* Apress.

Tidwell, J., Brewer, C., & Valencia, A. (2020). *Designing interfaces: Patterns for effective interaction design* (3rd ed.). O'Reilly Media.

Wood, D. (2014). *Interface design: An introduction to visual communication in UI design.* Bloomsbury Publishing.

PRINCIPLE OF MANAGEMENT

COURSE CODE: BCA 156
YEAR SEMESTER: I/II
WORKLOAD: 1 Hrs. /WEEK (THEORY: 1 Hrs. **CREDIT HOURS: 1**

Course description:
This course introduces the fundamental principles of management by integrating classical concepts with modern practices and emerging issues in management. It introduces students to the core functions of management - planning, organizing, decision-making and leading - while highlighting social dimensions of organizational behavior. Practical emphasis is placed on understanding the uses of IT in management.

Course objectives:
The main objective of this course is to provide students with a conceptual foundation and practical understanding of management principles, along with their basic applications. After the completion of this course, students will be able to:

- Understand the concepts, evolution, and functions of management,
- Apply management processes including planning, organizing and leading in organizational settings,
- Evaluate managerial decision-making using both traditional approaches and IT-supported methods,
- Examine the ethical and social responsibilities of organizations in modern contexts and
- Appreciate the growing role of IT in enhancing management efficiency.

Course content

Unit – 1 Introduction to Management **5 Hrs.**
1.1 Concept and meaning of management
1.2 Forms of business
1.3 Management process
1.4 Types of managers
1.5 Basic managerial roles
1.6 Managerial skills
1.7 Integrated management framework
1.8 Managing ethics and diversity
1.9 Social responsibilities and organizations
1.10 Role of IT in management

Unit - 2 Planning and Decision making **5 Hrs.**
2.1 Concept of planning
2.2 Planning process
2.3 Types of plan
2.4 Organizational goals
2.5 Organizational planning
2.6 SWOT analysis
2.7 Nature and process of decision-making
2.8 Use of IT in planning and decision-making

Unit – 3 Organizing 3 Hrs.

3.1 Elements of organizing

3.2 Job design, job description and job specification

3.3 Authority distribution

3.4 Forms of organizational design

Unit – 4 Leading 3 Hrs.

4.1 Nature of leadership

4.2 Generic approaches to leadership

4.3 Situational approaches to leadership

4.4 Emerging approaches to leadership

4.5 Managing team in the time of crisis

4.6 Leadership challenges in IT based organization

Required readings

Griffin, R. W. (2024). *Management (12th ed.)*. Cengage Learning.

Jones, R. G., & George, M. J. (2018). *Essentials of Contemporary Management*. McGraw-Hill Higher Education.

Mark, W. H., & Koontz, H. C. (2019). *Essentials of management*. McGraw-Hill Higher Education.

Teaching approach

- The course will be delivered in a seminar format where students take an active role.
- Each student (individually or in groups) will present assigned topics from the syllabus in class.
- Sessions will focus on discussion, critical thinking, and peer learning rather than traditional lectures.
- The instructor will act as a facilitator and moderator, guiding discussion and ensuring key concepts are well covered.

2. Student Responsibilities

- **Presentations:** Students must prepare and present course topics, integrating theory, case studies, and practical applications.
- **Active Participation:** Students are expected to engage in discussions, raise questions, and contribute insights during every class.
- **Term Paper:** Each student will submit a written term paper on a management topic (approved by the instructor) by the end of the semester.
- **Reflection Note:** A short reflective write-up summarizing personal learning and insights from the seminar experience is required.
- **Quizzes/Short Assignments:** Short quizzes may be conducted to reinforce learning.

3. Evaluation Scheme

The course does not include a final examination. Assessment will be continuous and based on:

- **Class Presentation** – 30%
- **Participation & Discussion** – 20%
- **Attendance** – 10%
- **Quizzes/Short Assignments** – 10%
- **Term Paper & Reflection Note** – 30%

4. Seminar Conduct Rules

- Presenters must share presentation slides or handouts in advance.
- All students should complete basic reading before each session to participate effectively.
- Respectful listening, constructive feedback, and academic integrity are expected at all times.
- Reflection notes should highlight key learning, challenges, and takeaways from the course.