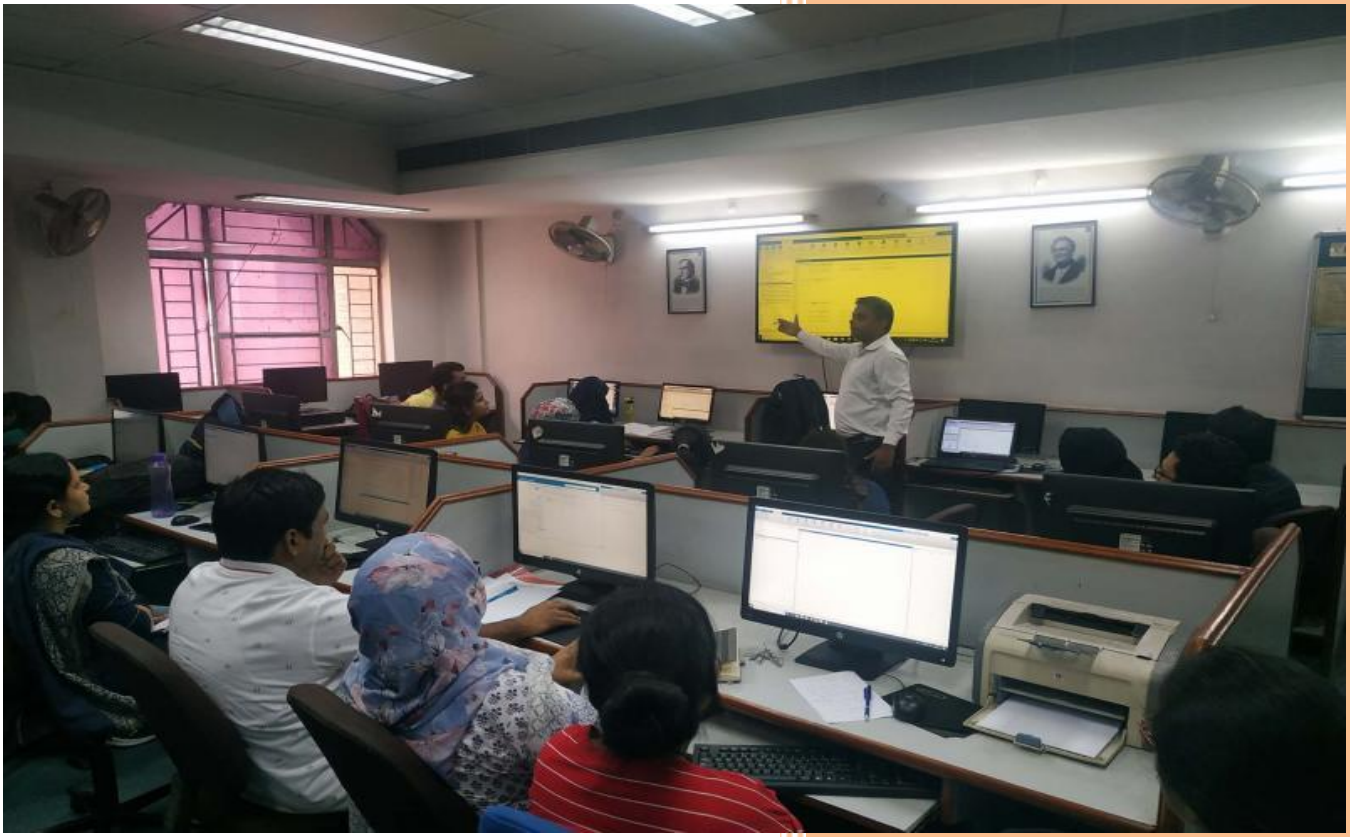
B.Sc./B.A./UG-CBCS
(Revised Curriculum 2023)



Approved in the BoS Meeting held on xx.xx.2023

Department of Computer Science

Jamia Millia Islamia

Detailed Curriculum

S. No.	Programmes	Page No.
1.	B.Sc. (Computer Courses)	2-6
2.	B.A. (Computer Courses)	7-8
3.	UG-Level CBCS	9-12

B.Sc. (Computer Science Courses)

SEM	CODE	COURSE TITLE	L-T-P	CREDIT	MARKS
I	BSCS11	Digital Logic	3-1-0	4	100
II	BSCS21	Algorithm Fundamentals	3-1-0	4	100
III	BSCS31	Programming with C++	3-0-2	4	100 + 50
	BSCS32	Data Structures	3-0-2	4	100 + 50
IV	BSCS41	Database Management System	3-0-2	4	100 + 50
	BSCS42	Applied Operating System	3-0-2	4	100 + 50
V	BSCS51	Computer Networks	4-0-0	4	100
	BSCS52	Discrete Mathematics	4-0-0	4	100
VI	BSCS61	Computer & Information Security	4-0-0	4	100
		Summary: (09 Courses)	30-2-8	**36**	**1100**

Detailed Syllabi

BSCS11: Digital Logic

LEARNING OUTCOMES

Understand basic concepts and logic circuits.
Understanding the simplification of logical statements.
Identification of various addresses.
Application of combinational circuit and understanding counters and registers.

1. Data Representation: Number Systems - Binary, Octal, Decimal, and Hexa-Decimal; Base Conversions; Binary Arithmetic; Complements: (r-1)'s Complement, r's Complement, Subtraction using Complements; Integer Representation, Floating-point Representation; Binary Codes for Decimal Digits: BCD Code, Excess-3 Code, 84-2-1 Code, 2421 Code, Error Detection Code; Character Representation - ASCII, EBCDIC.

2 Boolean Algebra, and Logic Gates: Boolean Algebra, Huntington's Postulate, Switching Algebra, Basic Theorems and Properties of Boolean Algebra; Boolean Functions: Basic Definition, Literals, Minimization of Boolean Functions by Algebraic Manipulation, Complement of a Boolean Function; canon 1 Cal and Standard Forms: Minterms and Maxterms, Conversion Between Canonical and Standard Forms of a Boolean Function; Boolean Function Simplification using k-Map; Digital Logic Gates: Basic Gates - AND, OR, NOT; Universal Gates - NAND, NOR; Other Gates - XOR, XNOR, AND-OR-INVERT, and OR-AND-INVERT.

3. Combinational Logic Circuit: Overview of Combinational Logic Circuit; Design of Some Standard Combinational Circuits: Half Adder, Full Adder, Half Subtractor, Full Subtractor, Code Conversion; Binary Adder, BCD Adder, Decoders, Encoder, Multiplexers, De-multiplexer.

4 Sequential Logic Circuit: Overview of Sequential Logic Circuits, Flip-Flops, Categories of Flip-Flop - RS, JK, T, and D Flip Flops, Registers and Counters.

REFERENCES

M. Morris Mano: Digital Logic and Computer Design, Prentice Hall of India
V. Rajaraman & T. Radhakrishnan: An Introduction to Digital Computer Design, PHI.

BSCS21: Algorithm Fundamentals

LEARNING OUTCOMES

Understanding the fundamental concept of algorithm.
Design of an algorithm for a given problem.
Understanding of algorithm design process for well-known problems.

1. Algorithms Design: Introduction, Problem-Solving Aspect, Top-down Design, Implementation of Algorithms, Program Verification, Efficiency of Algorithms. Introduction to Flowchart, Exchanging the Values of Two Variables, Counting, Summation of a Series of Numbers, Factorial computation, Computation of Sin(X) and Cos(x) values, Reversing the Digits of an Integer.
2. Factoring Methods: Finding Square Root of a Number, Smallest Divisor of an Integer, Greatest Common Divisor of Two Integers, Generating Prime Numbers, raising a Number to a Large Power, Computing the nth Fibonacci Number.
3. Array Techniques: Introduction to array, Array Order Reversal, Array Counting, Finding Max/ Min Number in an Array, Removal of Duplicates from an Ordered Array, Partitioning of Array, Finding Kth Smallest Element.
4. Merging, Sorting and Searching: Two ways merge, Selection Sort, Quick Sort, Insertion Sort, Linear Search, Binary Search.

REFERENCES

R.G. Dromy: How to Solve by Computer. Pearson
Parag Dave and Himanshu Dave, Design and analysis of Algorithms, Pearson Education

BSCS31: Programming with C++

LEARNING OUTCOMES

Understanding the OOP concepts such as classes, objects, inheritance, polymorphism, and encapsulation.
Deploy OOP paradigm in program design and development.
Designing classes and understanding of code reusability using inheritance.

1. Introduction to OOP and C++: Concepts of procedure-oriented and structured programming; OOP paradigm; basic concepts of OOP, its benefits and application; Introduction to C++, applications, simple programs, program structure, IDE of Turbo C++; tokens, expressions, and control structures; dynamic initialization of variables, operators, scope resolution operator, type casting.
2 Classes, Objects, Constructors and Destructors: C structures, specifying a class, defining member functions, making an outside function inline and nesting of member functions, private member functions, arrays within a class, static data members and member functions, arrays of objects, objects as function arguments, returning objects as function arguments, friendly functions; constructors, parameterized constructors, destructors.
3. Operator Overloading and Inheritance: Defining operator overloading, overloading unary and binary operators, rules for overloading operators; defining derived classes, types of inheritance, single, multilevel, multiple, hierarchical and hybrid inheritance, virtual bases classes, this pointer, virtual functions, pure virtual functions.
4. Working with Files: Classes for file stream operators, opening and closing a file, file pointers and their manipulations, sequential input and output operations, and error handling during file operations.

LAB (C++)

1. Implementation of simple classes like Rectangle, Circle, Sphere, Triangle, etc.
2. Implementation of some complex classes like Matrix, Complex Number, Vector] Decimal, etc.
3. Implementation of Matrix, Complex number, Vector, etc. classes with function overloading and constructor functions.
4. Implementation of some classes with operator over loadings.
5. Implementation of some classes like Square Matrix, Box, etc. with the help of inheritance.
6. Implementation of generic classes like Stack, Queue, etc.
7. Implementation of some simple classes with function overriding.
8. Implementation of some classes with «and » operator overloading.
9. Problems based on simple file handling.
10. Creation of student information system or inventory control system (construction of classes, implementing inheritance, overloaded functions, storing records to a file, fetching file records).

REFERENCES

E. Balagrurusamy, Object Oriented Programming with C++, TMH.
Deitel and Deitel, C++ How to program, PHI.

BSCS32: Data Structures

LEARNING OUTCOMES

Design algorithms for various operations of different data structures.
Applications of various data structures to solve different problems.
Device applications based on Graph data structures.

1. Introduction to Data structures: ADT and data structures, Array Data Structure, Multi-Dimensional Array, Matrix Representation using 2D Arrays – Row-major Order, Column-major Order, Special Matrices: Diagonal, Tri-diagonal, Lower Triangular and Upper Triangular Matrices, Sparse Matrices: Representation and Transpose, Addition of Sparse Matrices, Sorting and Searching Techniques.
2. Linked List, Stack, and Queue: Single Linked List, Operations on Single Linked List – Creating, Traversing, Insertion, Deletion, Copy, Merging, Searching, Applications of Linked List: Radix sorts, Sparse Matrix, Polynomial. Stacks and Queues: Introduction to Stacks, Array Representation of Stack, Linked Representation of Stack, Operations on Stacks, Applications of Stack – Infix Expression to Postfix Conversion, Evaluation of Postfix Expression, Recursive functions Implementations,

Introduction to Queues, Array Representation of Queue, Linked Representation of Queue, Applications of Queue.

3. Trees: Basic Concepts of Tree, Binary Trees, Types of Binary Tree, Properties of Binary Trees, Representation of Binary Trees: Array-Based Representation and Linked Representation, Operations on Binary Tree, Binary Search Tree (BST), Operations in BST: Insertion, Deletion, Traversing.

4. Graph: Basic Concepts Related to Graph, Difference between Tree and Graph, Properties of Graph, Graph Representations: Adjacency Matrix, Linked Adjacency Lists, Weighted Graph Representations, Graph Traversing Methods: Breadth-First Traversal and Depth-First Traversal, Applications of Graph.

LAB (DS with C/C++)

1. Implementation of Array, Upper Triangular Matrix, and Diagonal Matrix classes.
2. Implementation of Single Linked list classes.
3. Implementation of radix sort using object of Linked list classes.
4. Implementation of generic Stack class using arrays and LinkedList.
5. Implementation of Queue and Circular queue classes using arrays and pointers.
6. Evaluation of postfix expression using Stack.
7. Conversion of infix to postfix using Stack.
8. Implementation of Binary Tree class using an array.
9. Implementation of Binary Search Tree class using linked.
10. Implementation of Graph classes.

REFERENCES

D. Samanta: Classic Data Structure, PHI
A. S. Tenenbaum, Y. Langsam, Moshe J. Augenstein: Data Structures using C/C++, PHI
Robert Kruse, C.L.Tondo, bruce Leung: Data structures & Program Design in C, Pearson

BSCS41: Database Management System

LEARNING OUTCOMES

Understand the concepts of database and Database models.
Students will be able to design a database for applications.
Awareness about the various types of database users.

1. Databases and Database Users: Introduction, An Example, Characteristics of the Database Approach, Actors on the Scene, Workers behind the Scenes, Advantages of Using the DBMS Approach, A Brief History of Database Applications, When Not to Use a DBMS.

2. Database System Concepts and Architecture: Data Models, Schemas, and Instances, Three-Schema Architecture and Data Independence, Database Languages and Interfaces, The Database System Environment, Centralized and Client/Server Architectures for DBMSs, Classification of Database Management Systems.

3. Data Modeling Using the Entity-Relationship (ER) Model: Using High-Level Conceptual Data Models for Database Design, An Example Database Application, Entity Types, Entity Sets, Attributes, and Keys, Relationship Types, Relationship Sets, Roles, and Structural Constraints, Weak Entity Types, Refining the ER Design for the COMPANY Database, ER Diagrams, Naming Conventions, and Design Issues, Relationship Types of Degree Higher Than Two.

4. The Relational Data Model and Relational Database Constraints: Relational Model Concepts, Relational Model Constraints and Relational Database Schemas, Update Operations, Transactions, and Dealing with Constraint Violations. The Relational Algebra: Unary Relational Operations: SELECT and PROJECT, Relational Algebra Operations from Set Theory Binary Relational Operations: JOIN and DIVISION, Additional Relational Operations, Examples of Queries in Relational Algebra, Functional Dependencies and Normalization for Relational Databases: Informal Design Guidelines for Relation Schemas, Functional Dependencies, Normal Forms Based on Primary Keys, General Definitions of Second and Third Normal Forms, Boyce-Codd Normal Form.

LAB (ORACLE)

1. Implementation of DDL statements to create and manage tables.
2. Implementation of retrieving data using the SQL SELECT statement.
3. Implementation of data manipulation and storage; conversion functions & conditional expressions; aggregated data using group functions.
4. Implementation of display data from multiple tables; usage of sub-queries to solve queries.
5. Implementation of writing executable statements in PLlSQL and control structures.
6. Problems with the creation of forms with different controls, Decisions, conditions & Exception Handling.
7. Implementation of single and multiple validations; menus and submenus for program controls; List boxes, combo boxes and different types of loops; arrays and select case structure for multiple decisions; storing and retrieving data in files and other operations on files.

REFERENCES

Navathe: Fundamentals of Database Systems Concept, Addison-Wesley, 5/E.
Ivan Bayross: *SQL, Pl/SQL The Programming Language of Oracle, BPB Publication.

BSCS42: Applied Operating System

LEARNING OUTCOMES

- To understand the design of an operating system and services provided by the OS.
- To understand what a process is and how processes are synchronized and scheduled.
- To acquire knowledge on different approaches to memory management.
- Be familiar with various types of operating systems including UNIX, Linux, and Windows.

1. Fundamental Concepts: Operating System, OS Generations; Types of Operating Systems; Simple Batch Systems, multiprogrammed batched systems, Time-sharing systems, Parallel, Distributed and real-time Systems; Operating System Operations; Process Management; Memory Management; Storage Management; Protection and Security; Special-Purpose Systems.

2. System Structures: Computer System Operations, I/O Structure, Storage Structure, Storage Hierarchy, Hardware Protection, and General System Architecture. Operating System Components, Operating System Services, Systems calls, System Programs, System structures; System Boot.

3. Managing the System: Process concept, Operations on processes, Process Scheduling, Cooperating Processes, Inter Process Communication; Memory Management: Logical physical address space, Swapping, Paging and Segmentation.

4. Windows and Linux: Basic commands of Windows and Linux operational Environments; Hands-on Exercise and Case studies of Windows and Linux.

Lab (Linux)

1. Some basic commands that are frequently used.
2. Dealing with file operations such as creating files, displaying their contents, deleting files, creating links to files, renaming files, and moving files.
3. Maintaining directories, creating a directory, changing the current directory, and removing a directory
4. Displaying calendars, using basic calculators, displaying information about current systems, deleting symbolic links, and exiting from a Unix system.
5. Advanced commands used in the Unix operating system such as setting access permissions for the existing files and directories, setting default permissions for the newly created files and directories, creating groups, changing ownerships of the files, and sharing files among groups.
6. Sorting content and performing input/output (I/O) redirections.
7. Creating and running simple Bourne shell scripts.
8. Using command line parameters in the shell scripts.
9. Using conditional statements and loops.
10. Reading input, displaying output, testing data, translating content; displaying the exit status of the commands, and applying command substitution.

REFERENCES

Silberschatz, Galvin and Gagne, Operating System Concept, John Willey, 8th Edition.
Guide to Operating System by Michael Palmer, Thomson Learning.

BSCS51: Computer Networks

LEARNING OUTCOMES

1. Introduction to Networks Standards & Model: Introduction to Computer Networks; Communication Media and Nodes; Workstations; Hosts and Servers; Packets, Frames, and Cells; Networking Capabilities; Peer-to-Peer Networking and Workgroups; Networking with Servers; Client-Server Networking; Local Area Network (LAN), Metropolitan Area Network (MAN), Wide Area Network (WAN), Enterprise Network; Networking Standards and their Types; ISO-OSI Model; TCP/IP Model.

2. Topologies, Communication Media and Network Transport Systems: Network Topologies, Communication Media, Communication Media Costs and Considerations; Ethernet and the IEEE 802.3 Standards, Token Ring and the IEEE 802.5 Standards.

3. High-Speed Network Transport and Devices for Network Connectivity: WAN and Enterprise Network Communications; Fast Ethernet; FDDI; X.25, ISDN, Frame Relay; Multistation Access Units (MAU); Multiplexers, Repeaters, Bridges, Routers, Hubs, Gateways; ATM Switches, VLANs, Routing: IP Protocol; IP Addresses; Subnets; Subnet Mask.

4. Introduction to Internet: Domain Name System (DNS); Name Servers, Electronic Mail - Architecture and Services, User Agent, Message Formats, Simple Mail Transfer Protocol (SMTP), POP3; FTP, TELNET, World Wide Web and Hyper Text Transfer Protocol, Network Management.

REFERENCES

Andrew S. Tanenbaum: Computer Networks, 4th Ed., Pearson Education.
William Stallings: Data and Computer Communications, 5th Ed.

BSCS52: Discrete Mathematics
LEARNING OUTCOMES Comprehend the values of basic structures in computer science and their algorithmic utility & significance. Illustrate and apply logic propositional and predicate logic in decision problems. Appreciate the context and utility of relations and posets in computer science. Solve practical problems of complexity analysis by counting and recurrences.
1. Basic Structures: Set, Multi-set and Sequences; Type of sets, Set Operations, Power Set, Cartesian Products, Relation, Representation of relation, composition of relations, Functions, Types of Functions, Inverse of functions, Compositions of functions, function representation, Sequences, Special Integer Sequences, Summations; Mathematical & Structural Induction. **2. Relations and Partial Orders:** Equivalence Relation, Reflexive, Symmetric and Transitive Closure, Transitive Closure and Warshall's Algorithm; Equivalence Classes and Partitions; Partial Ordering, Lexicographic Order, Hasse Diagram, Maximal and Minimal Elements, Lattices. **3. Counting:** Simple and Complex Counting Problems, Inclusion-Exclusion Principle; Tree Diagrams; The Pigeonhole Principle; Permutations, Combinations, Binomial Coefficients, Examples and Applications; Binomial Coefficients, Binomial Theorem, Expression, and other Identities; Permutations and Combinations with Repetition, Permutations with Indistinguishable Objects, Distributing Events into Boxes; Generating Permutations and Combinations. **4. Recurrences:** Recurrences and Recurrence Relations; Homogeneous linear recurrences, non-homogeneous linear recurrences, solving recurrences using induction method, solving recurrences using characteristic equations, Solving recurrences by domain transformation, Solving recurrences by substitution.
REFERENCES K. Rosen: Discrete Mathematics and its Applications with Combinatorics and Graph Theory. McGraw Hill Bernard Kolman, Robert Busby, and Sharon C. Ross: Discrete Mathematical Structures. Prentice Hall J.P. Tremblay, and R. Manohar: Discrete Mathematical Structures with Applications to Computer Science, McGraw Hill

BSCS601: Computer & Information Security
LEARNING OUTCOMES Articulate and perform principled security analysis, design, and implementation. Elaborate and develop security policies and programs for security and BC-DR assurance. Explain, differentiate, and evaluate different security architectures/models in place for security assurance. Illustrate basic cryptographic techniques and their respective significance.
1. Context, CBK, and Principles: IT Security Importance and Opportunities; Multidisciplinary Approach; Contextualizing Information Security; IS Expertise & Business Systems. Security Management Practices: Security Architecture and Models; BCP; Law, Investigations, and Ethics; Physical Security; Operations Security; ACM Systems and Methodology; Cryptography; Network and Internet Security. **2. Security Management:** Security Policies: Programme-Level, Programme-Framework, Issue-Specific and System-Specific Policies; Development and Management of Security Policies: Security Objectives, Operational Security and Policy Implementation; Policy Support Documents Regulations; Standards Taxonomy; Risk Analysis and Management; Responsible for Security? Business Continuity Plan. **3. Security Architecture and Models:** Defining TCB: Rings of Trust; Protection Mechanisms in a TCB: System Security Assurance Concepts, Goals of Security Testing and Formal Security Testing Models; TCSE: Minimal, Discretionary, Mandatory and Verified Protection; Trusted Network Interpretation and TCSEC; Comparing ITSEC and TCSEC & ITSEC; CTCPEC, FCITS; CI Models: Bell-LaPadula Model. **4. Cryptography & Operations Security:** Cryptography Needs and Significance, Terms and Concepts: Cyphertext, Cryptanalysis, Cryptosystem, Message Digest etc; Digital Certificates, Symmetric Cryptography; Operations Security Principles; Operations Security Process Controls; Operations Security Controls in Action.
REFERENCES M. Merkow & J. Breithaupt: **Information Security - Principles and Practices**. Pearson M. E. Whitman & H. J. Mattord: **Principles of Information Security**. CENGAGE M. Palmer: **Guide to Operating Systems Security**. CENGAGE

B.A. (Computer Science Courses)

SEM	CODE	COURSE TITLE	L-T-P	CREDIT	MARKS
III	BACS31	Fundamentals of IT	3-1-0	4	100
IV	BACS41	Business Data Analytics	3-0-2	4	100
V	BACS51	Introduction to DBMS	3-0-2	4	100
VI	BACS61	Website Design & Management	3-0-2	4	100

Detailed Syllabi

BACS31: Fundamentals of IT

LEARNING OUTCOMES

To inculcate basic understanding of computer system organization and internal operations.
To apprise of the information management scenario, scope, and computer utility.
To understand problem-solving strategies and algorithm design.

1. Computing Concepts: Basic Computing Systems, Layers of a Computing System, History of Computing, History of Computing Software, Stored-Program Concept and von Neumann Architecture. Fetch-Execute Cycle, RAM and ROM, Types of RAM and ROM, Secondary and Tertiary Storage Devices, Cache Memory, Memory Hierarchy, Input-Output Devices, Touch Screens.

2. Data Representation and Number Systems: Binary Values and Computers, Data and Computers, Analog and Digital Data; Binary Representation. Number Systems: Binary, Octal, Decimal, and Hexadecimal. Conversions of Data from one Number System to another Number System. Representation of Numeric Data – Negatives and Real Data Representation. Representing Texts - ASCII and Unicode Character Sets. Binary Arithmetic – Addition and Subtraction of Numbers in Different Number Systems. Gates and Circuits: Computers and Electricity; Logic Gates – AND, OR, NOT, XOR, NAND, and NOR Gates. Gate Processing; Gates with More Inputs; Constructing Gates; Transistors; Circuits.

3. Problem-solving and Algorithm Design: Problem-Solving, Problem-Solving strategies, Algorithms, The computer problem-solving process, Pseudocode, Pseudocode Functionality, Top-Down design methodology. Information Systems: Managing information. Spreadsheets, spreadsheet formulae. Circular references, Spreadsheet analysis, Database management systems. The Relational model. Relationships. Structured query language. Database design. Information security. CIA and Cryptography.

REFERENCES

Dale & Lewis: Computer Science Illuminated, Narosa Publishing House
Kedall& Kendall: Systems Analysis and Design, Prentice Hall India
Rajaraman: Fundamentals of Computers, Prentice Hall of India

BACS41: Business Data Analytics

LEARNING OUTCOMES

Fundamental concepts related to business analytics and its utility in decision-making.
Comprehension of data analytics to enrich business processes.
Perform descriptive analytics on business data in the context of practices, with Excel.

1. Business Analytics: Informatics and Business Analytics, Using Business Analytics, Impacts and Challenges, Evolution of Business Analytics, Analytic Foundations, Modern Business Analytics, Software Support, and Spreadsheet Technology, Descriptive, Predictive, and Prescriptive Analytics, Data for Business Analytics; Big Data, Data Reliability and Validity; Models in Business Analytics: Descriptive, Predictive and Prescriptive; Model Assumptions, Uncertainty and Risk, Problem-Solving with Analytics: Interpreting Results and Making a Decision, Implementing the Solution.

2. Data Analytics: Introduction to Excel, Data Sets and Databases, Using Range Names in Databases; Data Queries: Tables, Sorting, and Filtering; Database Functions: Logical Functions, Lookup Functions for Database Queries, Template Design, Data Validation Tools, Form Controls, PivotTables, PivotTable Customization, and Slicers.

3. Data Visualization and Descriptive Analytics: Data Visualization, Value of Data Visualization, Tools and Software for Data Visualization, Creating Charts, Charts from PivotTables, Geographic Data, Data Visualization Tools; Descriptive Statistics: Metrics and Data Classification, Frequency Distributions; Percentiles and Quartiles, Cross-Tabulations, Descriptive Statistical Measures, Business Decisions; Measures of Dispersion; Empirical Rules; Measures of Association; Using Descriptive Statistics to Analyze Survey Data; Statistical Thinking in Business Decisions and Variability in Samples.

REFERENCES

Evans: Business Analytics, Pearson
Business Analysis with Microsoft Excel, 5ed, Pearson
D. Delen & E. Turban: Business Intelligence, Analytics, and Data Science: A Managerial Perspective. Pearson

BACS51: Introduction to DBMS
LEARNING OUTCOMES Learn basic database concepts, types, dimensions, and applications. Apprise of the RDBMS techniques, scope, design, and applications. Equip with necessary MS-ACCESS skills for designing databases and utilities.
1. Data, Database, and Database Management System (DBMS); DBMS vs. Traditional File System; ThreeSchema Architecture of DBMS and Data Independence; Classification of DBMS - Hierarchical, Network and Relational Database Systems; Database Languages and Interfaces; Database Users, Actors, and Workers. **2.** Database Models; Categories of Database Models; Entity Relationship (ER) Model: Basic Concepts and their representations - Entity, Entity Type and Entity Set; Attributes and their types, Keys, Relationships and their Types; Structural Constraints; Weak Entity; Naming Conventions & Design Issues in ER Model; ER Diagrams. **3.** Relational Database Model, Structure of Relational Model; Domains, Attributes, Tuples, and Relations; Characteristics of Relations; Relational Constraints - Domain Constraints, Key Constraints, Entity Integrity, and Referential Integrity Constraints; Relational Database Schema and Views. ***Lab Skills: MS-Access***
REFERENCES Elmasri & Navathe: Fundamentals of Database Systems, Pearson Education Joyce Cox and Joan Lambert: Microsoft Access 2010, Step by Step, Microsoft Press, PHI. Ivan Bayross: SQL, PI/SQL - The Programming Language of Oracle, 3rd Ed., BPB Pub.

BACS61: Website Design and Management
LEARNING OUTCOMES Understand the basic concepts of the Internet, WWW, and the Web Development Process Construct a website that conforms to the web standards of today. Understand fundamental concepts of Website design
1. Web Basics & Overview: Computer Networks; The Internet, The Domain Name System, The Web, Putting Information on the Web, Web Hosting, Domain Registration, Looking Up Host Information, The Web Development Process, Dynamic Generation of Web Pages. **2. Creating Web Pages - HTML:** HTML Basics, Elements and Entities, History of HTML, XHTML Syntax, Core Attributes, Headings and Paragraphs, White Space and Line Wrapping, Inline Elements, Controlling Presentation Styles, Length Units, Colors, Text Fonts, Lists, List Styles, Hyperlinks, Images, Positioning Inline Images, Image Maps, Tables: Cell Content Alignment, Table Width & Height, Grouping Rows & Columns, HTML Forms. **3. Web Design Basics, CSS, & JavaScript:** What is Design, Design and Perception, History of Design on the Web, Elements of Design, Unity and Variety, Emphasis, Focal Points, Hierarchy, Contract, Visual Balance. Introduction to Cascading Style Sheets. JavaScript: Embedding JavaScrpt in a Webpage, JavaScript Objects, Windows, Built-in Functions, A Conversion Calculator, Events, and Event Objects.
REFERENCES Wang & Katila: An Introduction to Web Design+Programming. Indian Edition, Cengage Learning. Joel Sklar: Web Design Principles. 5th Edition, Cengage Learning. Steven Holzner: HTML Black Book. DreamTech Press. Web Resources: http://www.w3schools.com

UG-CBCS (Computer Science Courses)

SEM	CODE	COURSE TITLE	L-T-P	CREDIT	MARKS
I	UGCBCS11	Computer Fundamentals	3-1-0	4	100
II	UGCBCS21	Programming with C	3-0-2	4	100 + 50
III	UGCBCS31	Programming with Python	3-0-2	4	100 + 50
IV	UGCBCS41	Web Programming with PHP	3-0-2	4	100 + 50
V	UGCBCS51	Business Informatics	3-0-2	4	100 + 50
VI	UGCBCS61	Programming with Java	3-0-2	4	100 + 50

Detailed Syllabi

UGCBCS11: Computer Fundamentals

LEARNING OUTCOMES

Inculcate basic understanding of computer organization and internal operations.
Apprise of the information management scenario, scope and computer utility.
Equip with necessary MS Office skills for office management practices.

1. Computing Concepts: Basic Computing Systems, Layers of a Computing System, History of Computing, History of Computing Software, Stored-Program Concept, and von Neumann Architecture. Fetch-Execute Cycle, Input-Output Devices, Mouse, Keyboard, Touch Screens.
2. Data Representation and Logic Gates: Binary Values and Computers, Data and Computers, Analog and Digital Data; Binary Representation. Number Systems: Binary, Octal, Decimal, and Hexadecimal. Conversions of Data from one Number System to another Number System. Representation of Numeric Data – Negatives and Real Data Representation. Representing Texts - ASCII and Unicode Character Sets. Binary Arithmetics – Addition and Subtraction of Numbers in Different Number Systems. Gates and Circuits**:** Computers and Electricity.
3. Logic Gates, Circuits & Memory: AND, OR, NOT, XOR, NAND, and NOR Gates. Gate Processing; Gates with More Inputs; Constructing Gates; Transistors; Circuits – Combinatorial Circuits: Adders and Multiplexers. Circuit as Memory; Integrated Circuits; CPU Chips; Basic Concepts of Memory, Types of Memory, Hierarchy: Registers, Cache, ROM, RAM, ROM BIOS/Firmware, Secondary, Tertiary Storage Devices, and their Relative Characteristics
4. Programming Languages: Computer Operations; Levels of Abstraction; Machine Language; Assembly Language; Pseudo-Operations; Introduction to Interpreter and Compiler, Programming Language Paradigms, Procedural vs. Object-Oriented Paradigms. Boolean Expressions; Strong Typing; Input-Output Structures; Control Structures; Composite Data Types. System Programs: Compilers; Interpreters; Loader, Linkers, and Operating Systems.

REFERENCES

Dale & Lewis: Computer Science Illuminated, Narosa.
Rajaraman: Fundamentals of Computers, PHI.
ITL Esl: Introduction to Computer Science, PE.

UGCBCS21: Programming with C

LEARNING OUTCOMES

Understanding of problem-solving approach & program logic design using flowcharting
Understanding the basic constructs of C language such as data types, expressions, Arrays, and user-defined functions.
Understanding of pointers, data handling with pointers, string manipulation
Understanding of derived data types with struct & union, creating and storing data using file handling functions

1. Problem-Solving aspect: Algorithm Design (Top-down Design); Program Verification, Fundamental of Algorithms & Flowcharting–Exchanging the values of two variables, Counting, Summation of a Set of Numbers, Factorial Computation, Infinite series sum, Sine Function Computation, Generation of the Fibonacci Sequence, Reversing the Digits of an Integer, Base Conversion, etc. **Basics of C:** Character Set; Keywords; Identifier, Constants, and Variables; Constant Types–Numeric and Character Constants; Data Types and Range of Values–Character, Integer and Floating Point; Signed, Unsigned, Short, and Long Integers; Data Declaration and Definition, Various Operators & Expression–Arithmetic. Managing Console I/O – Reading and

Writing Characters, Integers, Floating Point Numbers and Strings; Formatted I/O.

2. Control Structures & User-defined Functions: Decision Making (Branching) Structures–If Statement, If-Else Statement, Nested If-Else Statement, Else-If Ladder, Switch Statement, Goto Statement; Looping Structures – While Statement, Do-While Statement, For Statement, Continue and Break Statements. Functions: Library Functions; User-Defined Functions; Function Declaration (Prototype) and Function Definition; Function Arguments – Dummy, Actual and Formal Arguments; Local and Global Variables; Function Calls – Call by Value and Call by Reference; Recursion and Recursive Functions, Linkage of variables, Storage Class, & Scope of Variables.

3. Arrays and Strings: Single Dimensional Arrays; Accessing Array Elements; Initializing an Array; Multidimensional Arrays; Initializing Multidimensional Arrays; Memory Representation; Accessing Multidimensional Array Elements; Array of Characters; String Manipulation Functions; Introduction to Pointers.

4. Structure and File Handing: Structure Declaration and Initialization; Accessing Structure Members, Structure Assignments; Array of Structures and Arrays within Structures, Nested Structures; Structure as Function Arguments; Structure Pointer; Unions; Opening and Closing Files; I/O Operations on Files, Error Handling During I/O Operations, Command Line Arguments.

LAB (C)

1. Implementation of swapping of 2, 3, and n integer variables.
2. Implementation of simple problems based on simple decisions.
3. Implementation of counting, factorial, sin, square root, Fibonacci series, reversing digits of an integer, sum of digits of the integer.
4. Implementation of base conversions, greatest common divisor, smallest divisor of an integer, prime number generator, and generation of pseudo-random number.
5. Implementation of the array counting, finding the max and min number in a set.
6. Implementations of searching and sorting algorithms.
7. Implementation of the string handling functions using arrays.
8. Implementation of the string handling functions using pointers.
9. Implementation of problems on structure and union.
10. Implementation of file-handling problems

REFERENCES

E. Balagrusamy: Programming in ANSI C, 7th Ed., Tata McGraw Hill

Programming in C – Schaum Series by Gottfried, 3rd edition, TMH publication (2nd ed. downloadable)

R. G. Dromey: How to Solve it by Computer, 2nd Ed., Pearson Education (downloadable)

Deitel & Deitel: C – How to Program, 9th Ed., Pearson Education (6th ed. downloadable)

Mike Banahan, Declan Brady and Doran: The C Book, second edition, Addison Wesley, 1991 (downloadable)

Brian W. Kernighan, Dennis M. Ritchie: The C Programming Language, 2nd Edition, Prentice Hall, 1988

Forouzan and Gilberg: Computer Science: A Structured Programming Approach Using C, Course Technology.

UGCBCS31: Programming with Python

LEARNING OUTCOMES

Understand the basic construct of Python programming language

Apply various constructs and control structures in problem-solving

Understand the object-oriented program design and development in Python

Write clear and effective Python code.

1. Introduction: Getting Started: Setting up Programming Environment, Python on Different Operating Systems, Running Python Programs from a Terminal. Variables & Simple Data Types: Variables, Strings, Numbers, Comments, The Zen of Python. Working with Lists: What is a List, Changing, Adding, Removing Elements, Organizing a List, Avoiding Index Errors, Looping through an Entire List, Avoiding Indentation Errors, Making Numerical Lists, Slicing a List, Working with Tuples and Dictionaries.

2. Basic Constructs: User Inputs: input() and int() Functions, Accepting Input in Python. Conditional Tests: if Statements, Using if Statement with Lists. While Loop: Introducing while Loops, using a flat, break, continue, Using a while Loop with Lists and Dictionaries.

3. Functions, Classes, & Modules: Functions: Defining a Function, Passing Arguments, Return Values, Passing a List, Passing an Arbitrary Number, Storing Your Functions in Modules. Classes: Creating and Using a Class, Working with Classes and Instances, Inheritance, _init_() Method for a Child Class, Overriding Methods, Instances as Attributes, Importing Classes, Modules, Storing Multiple Classes in a Module, Importing Classes from a Module, Importing a Module into a Module.

4. Files & Exceptions: Reading from a File, Reading an Entire File, File Paths, Reading Line-by-line, Making a List of Lines from a File, Working with a File's Contents, Large Files, Writing to a File, Writing to an Empty File, Writing Multiple Lines, Appending to a File. Exceptions: Handling the ZeroDivisionError Exception, Using try-except Blocks, The else Block, Handling the FileNotFoundError Exception, Analyzing Text, Working with Multiple Files,

LAB (Python)

1. To use and understand the basic data types.
2. To apply various constructs and control structures in problem-solving.
3. To use the functions and methods related to Numbers and Strings.
4. To use the functions and methods related to Lists, and Tuples.
5. To use inbuilt modules in the program.
6. To implement the Object-oriented concepts in the program design.
7. To write programs for file handling using single and multiple files.

REFERENCES

Eric Matthes: Python Crash Course: A Hands-On, Project-Based Introduction to Programming. No Starch Press

Mark Lutz: Learning Python. O'Reilly

Zed A. Shaw: Learn Python the Hard Way. Addison-Wesley

UGCBCS41: Web Programming with PHP

LEARNING OUTCOMES

Understand the basic concepts of the World Wide Web and Web-application architecture
Learn HTML 5.0 and apply CSS to Webpages
Learn server-side programming using PHP
Learn to develop dynamic webpages with database connectivity

1. Introduction to Internet and Web Programming: Brief History of the Internet, WWW, Web System Architecture, Protocol used in Internet: TCP/IP, SMTP, Web Servers, Overview of Web Authoring Tools, Design Frameworks.

2. HTML5 & CSS: HTML5, Basic Structures of HTML Documents, HTML5 Semantic Elements: Header, Footer, Article, Section. Ordered & Unordered Lists, Hyperlinks, Working with Table, Working with Forms: Form and Input Tags, Text Box, Radio Button, Checkbox, Select Tag and Pull-Down Lists, Hidden, Submit and Reset, Cascading Style Sheets (CSS) and JavaScript: Benefit of CSS, CSS Properties, CSS Styling, Writing JavaScript into HTML

3. Introduction to PHP: Evaluation of PHP, Basic Syntax, defining variable and constant, Data Types, Operator and Expression, Handling HTML Form With PHP: Capturing Form Data, Dealing with Multi-valued Fields, Conditional Statement, Iterations, Arrays Working with Functions, Working with String.

4. Web Server & Database Connectivity: Introduction to WAMP/XAMPP Server – Configuration and Web Application Deployment, PHP Server Variables, State Management using Session. Database Connectivity with MySql: Connection with MySql Database, Basic Database Operations – Insert, Delete, Update and Select, Setting Query Parameter, Executing Query.

LAB (HTML & PHP)

1. Understanding DNS and exploring the Whois directory for DNS records.
2. Designing simple webpages with Header, Footers, Article, Lists, etc.
3. Working with Hyperlinks, Images, and Tables.
4. Design HTML forms with various form elements.
5. Working with CSS.
6. Writing Simple JavaScript Programs for basic computations, and dynamic content/styling.
7. Writing simple PHP programs using basic constructs.
8. Writing server-side programs using PHP - handling HTML forms.
9. Writing PHP programs using Arrays and user-defined functions.
10. Writing database-driven web applications using PHP & database connectivity.

REFERENCES

Robertw Sebestac: Programming World Wide Web. 8th Edition, Pearson.

Steven Holzner: PHP: The Complete Reference. TMH.

UGCBCS51: Business Informatics

LEARNING OUTCOMES

Fundamental concepts related to business analytics and its utility in decision-making.
Comprehension of data analytics to enrich business processes.
Performing descriptive analytics on business data in the context of practices, with Excel.
Performing typical analytics and business forecasting.

1. Business Analytics: Informatics and Business Analytics, Using Business Analytics, Impacts and Challenges, Evolution of Business Analytics, Analytic Foundations, Modern Business Analytics, Software Support and Spreadsheet Technology, Descriptive, Predictive, and Prescriptive Analytics, Data for Business Analytics; Big Data, Data Reliability and Validity; Models in Business Analytics: Descriptive, Predictive and Prescriptive; Model Assumptions, Uncertainty and Risk, Problem-Solving with Analytics: Interpreting Results and Making a Decision, Implementing the Solution.

2. Data Analytics: Introduction to Excel, Data Sets and Databases, Using Range Names in Databases; Data Queries: Tables, Sorting, and Filtering; Database Functions: Logical Functions, Lookup Functions for Database Queries, Template Design, Data

Validation Tools, Form Controls, PivotTables, PivotTable Customization and Slicers.

3. Data Visualization and Descriptive Analytics: Data Visualization, Value of Data Visualization, Tools and Software for Data Visualization, Creating Charts, Charts from PivotTables, Geographic Data, Data Visualization Tools; Descriptive Statistics: Metrics and Data Classification, Frequency Distributions; Percentiles and Quartiles, Cross-Tabulations, Descriptive Statistical Measures, Business Decisions; Measures of Dispersion; Empirical Rules; Measures of Association; Using Descriptive Statistics to Analyze Survey Data; Statistical Thinking in Business Decisions and Variability in Samples.

4. Business Forecasting: Qualitative and Judgmental Forecasting, Historical Analogy, The Delphi Method, Indicators and Indexes, Statistical Forecasting Models, Forecasting Models for Stationary Time Series, Moving Average Models, Error Metrics and Forecast Accuracy, Exponential Smoothing Models, Forecasting Models for Time Series with a Linear Trend, Double Exponential Smoothing, Regression-Based Forecasting for Time Series with a Linear Trend, Forecasting Time Series with Seasonality and Regression-Based Seasonal Forecasting Models.

REFERENCES

Evans: Business Analytics, Pearson
Business Analysis with Microsoft Excel, 5ed, Pearson
Delen & Turban: Business Intelligence, Analytics, and Data Science: A Managerial Perspective. Pearson

UGCBCS61: Programming with Java

LEARNING OUTCOMES

Understand the basic concepts and fundamentals of the platform-independent object-oriented language.
Demonstrate skills in writing programs using classes, interfaces, inheritance, and exception handling.
Design event-driven GUI applications

1. Introduction, Environment and Programming Structure: Java White Paper Buzzwords, History of Java, Choosing a Development Environment: Command-Line Tools, Running a Graphical Application; A Simple Java Program, Comments, Data Types, Variables, Operators, Input and Output, Control Flow, Arrays.

2. Class, Objects, and Inheritance: Introduction to OOP, Predefined Classes, User Defined Classes, Static Fields and Methods, Method Parameters, Object Construction, Packages, CLASSPATH, Documentation Comments, Inheritance: Super-classes and Subclasses, Types of Inheritance, Polymorphism, Inheritance Guidelines, Abstract class, Interfaces.

3. String Handling, and Exception Handling: String Handling APIs: String, Immutable String, Methods of String Class, StringBuffer, StringBuilder, StringTokenizer. Exceptions: Dealing with Errors, Catching Exceptions, Guidelines for Using Exceptions, Assertions, and Logging.

4. Java GUI Programming: Introduction to Swing, Creating a Frame, Positioning a Frame, Displaying Information in a Component, Event Handling, Basics of Event Handling, Actions, Mouse Events, The AWT Event Hierarchy.

LAB (Java)

1. Basic Data Types, Operators, Input and Output, and Control Flow.
2. Designing Class, Objects.
3. Working with Inheritance.
4. Writing own Java Packages.
5. Dynamic memory allocation using new and delete operators, function and constructor overloading, and operator overloading.
6. String handling - String Comparison, String Concatenation, Substring finding, String tokenization.
7. Exception handling – Exception Catching and Handling, Assertions, Logging, Debugging.
8. Developing GUI applications using Swing components and event handling.

REFERENCES

Balagurusamy: Programming with Java. 6th Edition, TMH
H. Schildt: Java 2: The Complete Reference. 12th edition, TMH