DRAFT SYLLABUS

North Lakhimpur University

CURRICULUM FRAMEWORK FOR ONE YEAR PG DIPLOMA PROGRAMME

Post Graduate Diploma in Computer Application (PGDCA)

Department of Computer Science

(As per NEP 2020 Guidelines)

(As approved by Board of Studies in Computer Science held on 01/06/2025 and passed by Academic Council held on ----------)

Contents

1.0 Introduction:..... 1
1.1 Nature and Extent of the PGDCA Program 1
1.2 Aims of the PGDCA Program..... 2
1.3 Graduate Attributes of the PGDCA Program..... 3
1.4 Qualification Descriptors for the PGDCA Program 3
1.5 Programme Specific Outcomes..... 4
1.6 Programme Learning Outcomes (PLO) for Post Graduate Diploma in Computer Applications (PGDCA) Programme..... 5
1.6 Course Outcomes for Post Graduate Diploma in Computer Applications (PGDCA) program 6
Summary of the PGDCA Course Structure as per NEP Guidelines..... 9
Structure of Syllabus..... 9
SEMESTER I..... 9
SEMESTER II..... 10
PGC-CC-101 Computer Fundamentals and OfficAutomatione 11
PGC-CC-102 Programming Fundamentals using C..... 15
GC-CC-103 Web Development..... 19
PGC-EL-101 Multimedia Systems and Applications..... 24
PGC-EL-102 Operating Systems..... 28
PGC-CC-201Software Engineering..... 31
PGC-CC-202 Database Management System 36
PGC-CC-203 Project Work 40
PGC-EL-201 OOP with JAVA..... 43
PGC-EL-202 Computer Networks..... 47

1.0 Introduction:

The Post Graduate Diploma in Computer Applications (PGDCA) is a dynamic and industry-oriented one-year program designed to provide students with a strong foundation in computer science, software development, and information technology. This program equips graduates with essential technical skills, programming expertise, and computational thinking, preparing them for various roles in the rapidly evolving IT industry.

Blending theoretical concepts with hands-on practical applications, the PGDCA curriculum covers key areas such as software engineering, database management, web technologies, artificial intelligence, cloud computing, and cybersecurity. Students gain valuable experience through coding projects and industry-oriented assignments, enhancing their problem-solving abilities and technological proficiency.

Aligned with the New Education Policy (NEP) 2020, the program follows a multidisciplinary approach, integrating core computing principles with data-driven decision-making, digital innovation, and entrepreneurial skills. Emphasizing ICT applications, leadership, ethics, and sustainability, the PGDCA fosters critical thinking and adaptability in an era of rapid technological advancements.

With the increasing demand for skilled IT professionals, PGDCA graduates can pursue careers in software development, data analytics, IT consulting, cybersecurity, and digital transformation. They can also opt for advanced studies in computer science, artificial intelligence, or related IT fields to specialize in cutting-edge technologies.

Ideal for graduates from diverse academic backgrounds with a keen interest in technology, problem-solving, and software development, the PGDCA program serves as a gateway to a successful and future-ready career in the digital world.

1.1 Nature and Extent of the PGDCA Program

The PGDCA program is designed as an intensive, outcome-based curriculum to establish academic standards and expected learning outcomes for graduates. This framework ensures that students acquiring a qualification in computer science develop the necessary technical knowledge, problem-solving skills, and computational competencies to meet both academic and professional demands.

The curriculum is structured around key components such as Graduate Attributes, Qualification Descriptors, Programme Learning Outcomes, and Course-Level Learning Outcomes. These elements serve as guiding principles for curriculum planning at the postgraduate diploma level.

The overall objectives of the PGDCA curriculum framework are to:

- Define and articulate graduate attributes, qualification descriptors, and learning outcomes at both program and course levels, ensuring clarity in academic expectations.

- Provide students, employers, and stakeholders with a clear understanding of the technical skills, problem-solving abilities, and professional values that a PGDCA graduate is expected to demonstrate upon program completion.
- Maintain national academic standards while ensuring international comparability, enhancing the global competitiveness of graduates and supporting student mobility across institutions and countries.
- Assist higher education institutions in designing effective teaching-learning strategies, student assessment methods, and periodic curriculum reviews to maintain academic excellence and relevance in the rapidly evolving IT sector.

The PGDCA integrates theoretical foundations, hands-on practical skills, and problem-solving expertise, equipping graduates for diverse career opportunities in software development, artificial intelligence, cloud computing, cybersecurity, and emerging digital technologies. By aligning with global standards, the program ensures that graduates remain technologically competent, adaptable, and industry-ready in an ever-evolving digital landscape.

1.2 Aims of the PGDCA Program

The PGDCA program aims to provide students with a strong foundation in computer science, software development, and information technology, equipping them with the necessary skills for professional growth and career advancement. The program is designed to meet national and international standards, ensuring graduates are well-prepared for diverse career opportunities and further studies in emerging technology domains.

The key aims of the program are to:

- Develop a comprehensive understanding of computer science principles, including programming, software development, database management, networking, cybersecurity, and artificial intelligence.
- Foster critical thinking, problem-solving abilities, and analytical reasoning to enable students to address real-world technological challenges using computational tools and methodologies.
- Provide hands-on experience with modern programming languages, development frameworks, and IT infrastructure, enhancing students' ability to design, develop, and implement innovative software solutions.
- Promote the application of computing and IT solutions in various domains such as business, finance, healthcare, e-commerce, education, and smart technologies.
- Equip students with research skills and the ability to analyze, design, and develop software applications, incorporating project-based learning and industry assignments.
- Instill ethical values, professional responsibility, and a commitment to lifelong learning, fostering a responsible approach to cybersecurity, data privacy, and technology ethics.
- Enhance communication, teamwork, leadership, and ICT skills, preparing students for diverse roles in software engineering, IT consulting, system administration, data analytics, cloud computing, and entrepreneurship.

By achieving these aims, the PGDCA program ensures that graduates are well-equipped to contribute to industry, research, and entrepreneurship, fostering innovation, digital transformation, and technology-driven solutions in the modern world.

1.3 Graduate Attributes of the PGDCA Program

The PGDCA program aims to develop well-rounded graduates with a strong foundation in computer science, software development, and emerging technologies. The graduate attributes define the key competencies, skills, and values that students are expected to demonstrate upon completion of the program.

- Disciplinary Knowledge: Graduates will have a comprehensive understanding of computer science principles, programming, algorithms, software engineering, and IT infrastructure.
- Analytical and Problem-Solving Skills: Graduates will be able to analyze complex problems, develop algorithms, and implement efficient solutions in various domains.
- Research and Innovation: Students will develop a research-oriented mindset and the ability to explore, design, and innovate software solutions, AI-driven applications, and emerging technologies.
- Computational and ICT Proficiency: Graduates will gain proficiency in programming languages, databases, cloud computing, artificial intelligence, cybersecurity, and modern development frameworks.
- Communication and Presentation Skills: Graduates will develop the ability to effectively communicate technical concepts and present ideas to technical and non-technical audiences.
- Ethical Awareness and Professional Responsibility: Students will be instilled with ethical principles in data security, privacy laws, and responsible AI, ensuring professional integrity.
- Leadership and Teamwork: Graduates will demonstrate teamwork, collaboration, and leadership abilities in multidisciplinary environments.
- Lifelong Learning and Adaptability: Students will develop a mindset for continuous learning, staying updated with the latest advancements in technology.
- Global Competence and Societal Contribution: Graduates will be equipped to apply computing knowledge for societal development, aligning with Sustainable Development Goals (SDGs).

By acquiring these attributes, PGDCA graduates will be well-prepared for industry roles, research, and entrepreneurial ventures in the expanding field of technology and digital transformation.

1.4 Qualification Descriptors for the PGDCA Program

The Qualification Descriptors for the PGDCA outline the key competencies, skills, and knowledge that graduates are expected to demonstrate upon completion of the program. These descriptors ensure alignment with national and international academic standards, fostering academic excellence, employability, and research potential in the evolving field of technology.

A. Comprehensive Knowledge and Understanding: Graduates will have an in-depth understanding of computer science principles, programming, software engineering, database management, cloud computing, and artificial intelligence. B. Analytical and Problem-Solving Skills: Graduates will develop the ability to analyze computational problems and develop efficient software solutions. C. Technical and Computational Proficiency: Graduates will gain expertise in modern programming languages, full-stack development, and cloud platforms. D.

Employability and Career Readiness: Graduates will be well-prepared for careers in software development, IT consulting, cybersecurity, cloud computing, and AI/ML.

By achieving these qualification descriptors, PGDCA graduates will be equipped with the expertise and skills required for professional success and technological innovation in the digital economy.

1.5 Programme Specific Outcomes

PSO 1: Core Computing Proficiency
Demonstrate foundational knowledge in core computer science areas including computer fundamentals, office automation, programming in C, software engineering, databases, and web development, with the ability to apply these skills to practical computing tasks.

PSO 2: Problem Solving and Programming Logic
Apply analytical and logical thinking to design algorithms and implement solutions using procedural (C) and object-oriented (Java) programming approaches for real-world computing problems.

PSO 3: Web and Application Development Skills
Design and develop dynamic and responsive web applications using modern tools and technologies, demonstrating an understanding of frontend and backend development processes.

PSO 4: Database Management and Data Handling
Create, manage, and query databases efficiently using standard DBMS tools, ensuring data integrity, security, and scalability in application scenarios.

PSO 5: Software Development and Engineering Principles
Understand and apply software development life cycle (SDLC) models and project management techniques to plan, analyze, design, implement, test, and maintain software systems.

PSO 6: Elective-Specific Competence
Gain domain-specific knowledge in either multimedia systems and applications, operating systems, Java programming, or computer networks based on elective choices, and apply them effectively in practical environments.

PSO 7: Project Management and Real-Life Application
Demonstrate the ability to conceptualize, plan, and implement a project through a capstone project or dissertation, integrating knowledge from various courses to address a practical problem or need.

PSO 8: Communication, Ethics, and Professionalism
Communicate effectively in professional settings, uphold ethical standards in computing practices, and demonstrate responsible behavior with regard to data privacy, intellectual property, and societal impacts.

PSO 9: Lifelong Learning and Technology Adaptability
Recognize the need for continual learning and skill development to stay current with emerging tools, languages, frameworks, and practices in the fast-evolving IT industry.

1.6 Programme Learning Outcomes (PLO) for Post Graduate Diploma in Computer Applications (PGDCA) Programme

The Programme Learning Outcomes (PLOs) define the competencies, skills, and knowledge that students will acquire upon completing the PGDCA. These outcomes align with national academic frameworks and industry requirements, preparing graduates for professional careers, research, and lifelong learning in the ever-evolving field of technology.

Core Knowledge and Understanding

- Develop a solid foundation in computer fundamentals, programming, web development, database management, and software engineering.
- Gain expertise in office automation, multimedia applications, operating systems, and emerging technologies to solve industry-specific problems.

Analytical and Problem-Solving Skills

- Apply computational thinking and programming techniques to develop efficient software solutions.
- Enhance logical reasoning, debugging, and optimization skills to tackle real-world IT challenges.

Research and Innovation

- Formulate and analyze research problems in areas such as web development, database management, and multimedia applications.
- Develop innovative software solutions using programming languages, frameworks, and tools relevant to modern IT industries.

Computational and ICT Proficiency

- Gain hands-on experience with programming languages like C and Java, along with web technologies such as HTML, CSS, and JavaScript.
- Develop proficiency in database management systems, software development methodologies, and operating system functionalities.
- Explore cloud computing, networking, and cybersecurity concepts for a comprehensive understanding of IT infrastructure.

Communication and Technical Documentation

- Effectively communicate technical concepts, software designs, and research findings through documentation, reports, and presentations.
- Present complex technical ideas to diverse audiences, including business stakeholders, developers, and end-users.

Ethical and Professional Responsibility

- Understand ethical principles related to data privacy, intellectual property rights, and responsible AI practices.
- Recognize the societal impact of technology and promote ethical computing practices in software development and IT solutions.

Multidisciplinary Approach and Global Perspective

- Apply computing solutions across various domains, including business, healthcare, finance, and education.
- Stay adaptable to global trends in technology, innovation, and digital transformation.

Lifelong Learning and Career Readiness

- Foster a continuous learning mindset to stay updated with advancements in software development, database systems, and web technologies.
- Prepare for diverse career paths in software development, IT consulting, web and database administration, and research.

Leadership, Teamwork, and Entrepreneurial Skills

- Develop the ability to work independently and collaboratively in software development, research, and professional environments.
- Gain leadership and entrepreneurial skills to innovate and contribute to the digital economy through startups, product development, and IT solutions.

The PGDCA program equips graduates with a comprehensive understanding of computing concepts, software development, database management, and ethical responsibility, making them highly competent professionals in an increasingly technology-driven world.

1.6 Course Outcomes for Post Graduate Diploma in Computer Applications (PGDCA) program

The Course Outcomes (COs) define the specific knowledge, skills, and abilities that students will acquire upon completing individual courses in the Post Graduate Diploma in Computer Applications (PGDCA) program. These outcomes ensure alignment with the overall Programme Learning Outcomes (PLOs) and provide students with a structured learning path across different areas of Computer Science and Information Technology.

- Understand the history, characteristics, classifications, and various types of computers.
- Identify and explain the components of a computer system, including the motherboard, CPU, registers, and system bus.
- Describe the function of input and output devices and their applications.
- Differentiate between types of memory and storage devices, including cache, primary, secondary, and SSDs, and understand their structure and function.
- Apply knowledge of number systems, representations, and computer technologies such as Bluetooth, cloud computing, big data, data mining, mobile computing, and embedded systems.
- Understand the history, structure, and compilation process of C programs, and differentiate between C and C++.
- Use variables, constants, operators, and I/O functions to write basic C programs. Apply control flow concepts using conditional and iterative statements to solve problems.
  Develop programs using functions, arrays, strings, pointers, and derived data types like structures and unions.

- Design and develop structured web pages using HTML and CSS.
- Implement dynamic client-side functionalities using JavaScript.
- Create interactive web applications using PHP and connect them to a MySQL database.
- Apply basic web security principles and best practices in web development.
- Understand the basic concepts, components, and applications of multimedia and virtual reality.
- Apply appropriate text styles, fonts, and tools in multimedia design.
- Create and manipulate images using bitmap, vector, and 3D drawing techniques.
- Handle audio and video formats and perform basic editing for multimedia applications.
- Implement animation principles and techniques using suitable tools and formats.
- Explain the fundamental functions, types, and historical evolution of operating systems, including the role of real-time systems.
- Analyze the concepts of processes, threads, scheduling algorithms, and deadlock handling in operating systems.
- Apply memory management techniques, file system operations, and I/O management strategies in various OS environments.
- Implement synchronization mechanisms to manage concurrency and ensure correct inter-process communication.
- Explain the nature of software, define software engineering, and identify key historical developments and challenges.
- Compare software process models such as Waterfall, Spiral, and Incremental, and select appropriate models for a given scenario.
- Perform requirements analysis, differentiate between functional and non-functional requirements, and create a well-structured SRS document.
- Use design methodologies, represent designs with appropriate notations, and apply various testing levels and debugging techniques.
- Describe software quality standards, maintenance types, project estimation methods, and risk management techniques
- Explain the importance of databases, describe DBMS architecture, and construct ER diagrams using entities, attributes, and relationships.
- Identify relational model elements, describe characteristics of relations, and apply integrity constraints like primary and foreign keys.
- Perform relational algebra operations such as SELECT, PROJECT, JOIN, and UNION to process queries.
- Write SQL commands using DDL, DML, aggregate functions, and subqueries to manage and retrieve data.
- Analyze database anomalies and apply normalization up to BCNF using functional
- dependency concepts.
- By the end of the PGDCA Project, students will be able to design, develop, test, and document a real-world software application by applying theoretical and practical knowledge of computer science, demonstrating proficiency in system analysis, software development life cycle (SDLC), user interface design, coding, testing strategies, and deployment, while adhering to professional standards of documentation, teamwork, and project presentation.
- Describe Java features, JVM, bytecode, and use control statements and operators effectively.

- Create and manipulate single/multi-dimensional arrays, perform string operations, and utilize wrapper classes.
- Develop Java programs using classes, constructors, method overloading, and access control features.
- Implement inheritance, method overriding, and apply super, final, and multilevel inheritance in code.
- Define and implement interfaces, extend interfaces, and differentiate between abstract classes and interfaces.
- Describe network hardware/software components, OSI and TCP/IP models, and physical layer concepts such as guided media.
- Explain wireless transmission, telephone system structure, switching techniques, and apply error detection and correction methods.
- Implement and analyze data link layer protocols like sliding window and understand multiple access protocols including Bluetooth.
- Apply routing and congestion control algorithms, explain IP addressing, and understand network control protocols.
- Demonstrate the role of the transport layer in connection management and explain basic cryptographic techniques for secure communication.

These Course Outcomes (COs) ensure that students develop a strong theoretical foundation, practical skills, and research capabilities across various Computer Science and IT domains, preparing them for academia, industry, and research careers.

Summary of the PGDCA Course Structure as per NEP Guidelines

Department of Computer Science

Structure of Syllabus

SEMESTER I

Course	Course Title	Course Code	Credit	Marks	End Sem Marks	Internal Assessment	Remarks
1	Computer Fundamentals and Office Automation	CC-5-PGC-101	4+1=5	100	56+20=76	24	
2	Programming Fundamentals using C	CC-5- PGC-102	3+2=5	100	42+40=82	18	
3	Web Development	CC-5-PGC-103	3+2=5	100	42+40=82	18	
4	Multimedia Systems and Applications	EL-5-PGC-101	4+1=5	100	56+20=76	24	
5	Operating System	EL-5-PGC-102	4+1=5	100	56+20=76	24	
		Total	20				
Note: Students are required to choose **one course** from the two elective courses offered above.							

SEMESTER II

Course	Course Title	Course Code	Credit	Marks	End Sem Marks	Internal Assessment	Remarks
1	Software Engineering	CC-5-PGC-201	5	100	70	30	
2	Programming Fundamentals using C	CC-5- PGC-202	3+2=5	100	42+40=82	18	
3	Project Work / Dissertation	CC-5-PGC-203	5	100	70	30	
4	OOP with JAVA	EL-5-PGC-201	3+2=5	100	42+40=82	18	
5	Introduction to Computer Networks	EL-5-PGC-202	4+1=5	100	56+20=76	24	
		Total	20				
Note: Students are required to choose **one course** from the two elective courses offered above							

Semester-I

Course Code : CC-5-PGC-101
Course Title : Computer Fundamentals and Office Automation
Nature of the Course : CORE
Total Credit assigned : 5
Component of the Course: Lecture: 4 Practical: 1 Tutorial: 0
Distribution of Marks : End Semester: 56 Practical: 20 In Semester: 24

Course Objectives:

CO1. Understand the history, types, and components of computers.
CO2. Identify input/output devices and various memory and storage types.
CO3. Perform conversions and operations using number systems and binary codes.
CO4. Understand the basic concepts and applications of emerging technologies like Bluetooth, cloud computing, big data, data mining, mobile computing, and embedded systems.
CO5. Use word processor, spreadsheet, and presentation software effectively.

Course Content:

Unit	Content	Marks	L	P	T
I	**Introduction to Computer** • Introduction, History of Computers (Generation of Modern Computers), Characteristics of Computer, Classification of Computers, Functional Components of a Computer, Types of Computers, Application of Computers, Motherboard.	10	8	**0**	**0**
II	**Computer peripherals** • Input Devices- Keyboard, Pointing devices, handheld devices, Optical Devices, Output Devices- Soft copy devices, Hard copy Devices, • CPU, Registers, System Bus, Sequential and Random Access, Memory Hierarchy, Cache Memory, Primary memory, Secondary Memory, Magnetic Tapes, Floppy Disks, Hard Disks,External Hard Disks, Optical Drivers, USB Flash Drivers, Memory Card, SSD.	11	**13**	**0**	**0**
III	**Number Systems and Computer Codes** • Introduction to Number Systems, Binary Number System, Octal Number System, Hexadecimal Number System, Working with Fractions, Signed	12	17	**0**	**0**

	number representation in binary form (1's complement, 2's complement), BCD code, Graycode, ASCII code.				
IV	**Overview of Emerging Technologies** • Bluetooth, Cloud computing, Big data, data mining, mobile computing and embedded system.	11	**7**	**0**	**0**
V	**Office Automation Software** (i) **Word Processor**: Introduction, Objective, Creating and Editing a document, Formatting, Advanced Formatting, Mail merge, Macros, Bookmark, Sort text and tables, Auto-text, Autocorrect, Spelling and Grammar tool, Save, Printing, Style, Linking, Using Template. **(ii) Spreadsheet:** Creating and Editing Worksheet, Formatting operations, Formulas and Functions, Inserting charts, Sorting,Data Entry, Text Management and Cell Formatting, Validation. **(iii) Presentation Software:** Creating, Slide Layouts, Adding text & Outline view, Pictures and Graphics, Shapes and Lines, Tables, Transitions, SmartArt, Hyperlinks.	12	**15**	**30**	**0**
Total		**56**	**60**	**30**	**0**

Mode of In-Semester assessment (24 Marks):

A. Sessional Activity

B. Sessional Tests: (two Sessional Tests)

Sessional Activities:

1: Introduction to Computer

Create a timeline of computer generations.

Identify and explain basic computer components.

2: Computer Peripherals

Demonstrate input/output device usage.

Draw and explain the memory hierarchy.

3: Number Systems and Codes

Convert between number systems and perform binary arithmetic.

Create and interpret BCD, Gray, and ASCII codes.

4: Emerging Technologies

Present on emerging technologies (e.g., cloud, IoT).

Watch and summarize a video on embedded systems.

5: Office Automation Software

Create and format a document/spreadsheet using Word or Excel.

Design and present a PowerPoint slide deck.

Lab Work:

1. **Creating a Formal Letter and Resume**
   - Write a job application letter and create a resume using standard formatting.
2. **Designing a Project Report**
   - Format headings, use styles, insert a cover page, and table of contents.
3. **Using Mail Merge for Bulk Letters**
   - Create a letter template and merge it with a recipient list (CSV or table).
4. **Working with Tables and Sorting Text**
   - Create a student marks table and sort it by name or marks.
5. **Recording and Running Macros**
   - Automate repetitive formatting using macros.
6. **Using Bookmarks and Hyperlinks**
   - Insert bookmarks in a long document and link them for quick navigation.
7. **Spell Check, Grammar Tool & Autocorrect**
   - Identify and correct errors using built-in tools. Customize autocorrect entries.
8. **Creating and Using Templates**
   - Design templates for letters, reports, or certificates.

9. **Creating and Formatting a Marksheet**
   - Enter data, use cell formatting, merge cells, and apply borders.
10. **Calculating Student Grades Using Functions**
    - Use IF, AVERAGE, COUNTIF, and nested formulas for grading.
11. **Sales Data Analysis with Charts**
    - Create column, pie, and line charts from sales data.
12. **Creating a Personal Budget Sheet**
    - Include income, expenses, savings, and total calculations.
13. **Data Sorting and Filtering**
    - Sort employee data by name, department, or salary. Apply filters.
14. **Attendance Sheet with Validation**
    - Use drop-downs for Present/Absent and prevent invalid entries.
15. **Working with Date, Time, and Text Functions**
    - Use TODAY(), NOW(), TEXT(), CONCAT(), and LEN().
16. **Linking Multiple Sheets and Referencing**
    - Use cell references across worksheets and summarize data.
17. **Creating a Presentation on a Given Topic**
    - Include title, bullet points, images, and conclusion slides.
18. **Using SmartArt, Shapes, and WordArt**
    - Visually represent processes, hierarchies, or timelines.
19. **Inserting Tables, Charts, and Hyperlinks**
    - Embed Excel data and link to other slides or web pages.
20. **Applying Transitions and Animations**
    - Use subtle transitions and object animations for professionalism.

Course Learning Outcomes:

By the end of the course, students will be able to:

CLO-1. Understand the history, characteristics, classifications, and various types of computers.
CLO-2. Identify and explain the components of a computer system, including the motherboard, CPU, registers, and system bus.
CLO-3. Describe the function of input and output devices and their applications.
CLO-4. Differentiate between types of memory and storage devices, including cache, primary, secondary, and SSDs, and understand their structure and function.
CLO-5. Apply knowledge of number systems, representations, and computer technologies such as Bluetooth, cloud computing, big data, data mining, mobile computing, and embedded systems.

Cognitive Map of Course Outcomes based on Revised Bloom's Taxonomy

Knowledge Dimension	Remember	Understand	Apply	Analyse	Evaluate	Create
Factual Knowledge	CO_1, CO_2	CO_1, CO_2	CO_3			
Conceptual Knowledge		CO_1, CO_4		CO_4		
Procedural Knowledge		CO_5	CO_3, CO_5			CO_5
Metacognitive Knowledge						

Mapping of PSO with CLO

CLO / PSO	PSO1	PSO2	PSO3	PSO4	PSO5	PSO6	PSO7	PSO8	PSO9
CLO1	3	1				2		2	3
CLO2	3	1				2		2	3
CLO3	3	1				2		2	3
CLO4	3	1				2		2	3
CLO5	3	2				2		2	3
Average Contribution	3	1.2				2		2	3

(3=High, 2= Moderate 1= Low)

Suggested readings:

1. Goel, A. (2010). *Computer fundamentals*. Pearson Education India.
2. Mano, M. M. *Digital Design: For Anna University, 4/e*. Pearson Education India.
3. Sinha, P. K., &Sinha, P. (2007). Fundamentals of Computers.
4. Kumar, A. (2010). *Computer basics with office automation*. TechsarPvt. Ltd.

Semester-I

Course Code :CC-5- PGC-102
Course Title : Programming Fundamentals using C
Nature of the Course : CORE
Total Credit assigned : 5
Component of the Course : Lecture: 3 Practical: 2 Tutorial: 0
Distribution of Marks : End Semester: 42 Practical: 40 In Semester: 18

Course Objectives:

CO1. Understand the fundamentals of C, including its structure, compilation process, and key differences from C++.
CO2. Apply data types, variables, constants, operators, and perform input/output operations using C.
CO3. Develop programs using conditional statements, loops, and control structures effectively.
CO4. Implement functions, arrays, pointers, structures, and unions for modular and efficient programming.

Course Content:

Unit	Content	Marks	L	P	T
I	Introduction to C • Overview of programming languages • History of C • Basic structure of a C program • Compilation process • Distinction between C and C++	5	3	10	0
II	**Data Types, Variables, Constants, Operators, and Basic I/O** • Data types and variables • Constants and literals • Operators and expressions • Basic input/output operations • Formatted I/O **Control Flow: Expressions, Conditional Statements, and Iterative Statements** • Conditional statements (if-else, switch) • Iterative statements (while, do-while, for) • Nested loops and conditional statements	13	12	17	0
III	**Functions and Arrays** • Introduction to functions • Function prototypes • Call by value and call by reference • Return values and void functions • Recursive functions	12	15	17	0

	• One-dimensional arrays • Multi-dimensional arrays • Strings and string manipulation				
IV	**Derived Data Types: Structures, Unions, Pointers, and References** • Structures and Unions • Pointers to structures and unions • Arrays of structures and unions • Introduction to pointers • Pointers to basic data types • Pointers to arrays and structures • Pointers vs. references	12	15	16	0
		42	45	60	0

Mode of In-Semester assessment (18 Marks):

A. Sessional Activity

B. Sessional Tests: (two Sessional Tests)

Sessional Activities:

1: Introduction to C

- Write and run a basic "Hello World" C program, explaining each part.
- Demonstrate the compilation process using a C compiler (e.g., GCC).

2: Data Types, Variables, Constants, Operators, and Basic I/O

- Create a program using different data types, variables, constants, and expressions.
- Write a program using scanf() and printf() with format specifiers.

3: Control Flow

- Develop a program using if-else and switch-case for decision-making.
- Implement loops (for, while, do-while) with nested conditions.

4: Functions and Arrays

- Write user-defined functions with call by value and reference, and use recursion.
- Implement one-dimensional and two-dimensional arrays with string manipulation.

5: Derived Data Types

- Create a program using structures, unions, and arrays of structures.
- Implement pointer operations: pointer to variable, array, structure, and pointer arithmetic.

Lab Works (but not Limited to)

1. Write a program to display a simple message ("Hello, World!") using C.
2. Identify and explain the basic structure of a C program with comments.
3. Write a program to demonstrate the use of header files and the compilation process.
4. Compare C and C++ through a sample "Hello World" program.
5. Create and document the flow of the compilation process using a diagram.
6. Declare variables of different data types and display them using printf().
7. Demonstrate use of constants and literals in a simple program.
8. Write programs using arithmetic, relational, logical, and assignment operators.
9. Evaluate expressions and show operator precedence and associativity in output.
10. Create a simple calculator using basic input/output functions.
11. Use scanf() and printf() for formatted input and output.

12. Write a program using if and if-else to check even/odd or grade category.
13. Implement a program using nested if-else to assign letter grades.
14. Create a menu-driven program using switch statement.
15. Demonstrate the use of a while loop to print numbers 1 to 10.
16. Use a do-while loop to create a password retry system.
17. Implement a for loop to display the multiplication table of a number.
18. Write a program that combines loop and condition (e.g., print prime numbers).
19. Use nested loops to print patterns (e.g., triangles, squares).
20. Write a program using a function with no return value and no parameters.
21. Create a function that accepts parameters and returns a value (e.g., sum of two numbers).
22. Demonstrate call by value using a swapping function.
23. Demonstrate call by reference using pointers.
24. Write a recursive function to calculate factorial of a number.
25. Write a recursive function to find Fibonacci series up to N terms.
26. Create and initialize a one-dimensional array and display its elements.
27. Write a program to find the largest and smallest element in an array.
28. Use two-dimensional arrays to perform matrix addition.
29. Use two-dimensional arrays to perform matrix multiplication.
30. Input and output a string using gets() and puts() (or scanf() and printf()).
31. Write a program to reverse a string without using library functions.
32. Use string handling functions like strcpy, strlen, strcat, strcmp.
33. Define and use a structure to store and display student information.
34. Create an array of structures for storing data of multiple students.
35. Demonstrate the use of a union to store different types in the same memory location.
36. Write a program to use pointers with basic data types (int, float, char).
37. Demonstrate pointer arithmetic operations (increment, decrement, comparison).
38. Use pointers to access and manipulate elements of an array.
39. Write a program using pointer to a structure and access structure members.
40. Create a program with an array of structures accessed through pointers.
41. Demonstrate how to pass structure to function by reference using pointers.
42. Compare pointer and reference-like behavior using examples (though C lacks direct references).
43. Combine structures and arrays with functions (e.g., pass array of structures to function).
44. Develop a mini project such as a student management system using structures and file I/O (if introduced).
45. Final review lab with viva, error correction, and consolidation of topics.

Expected Course Learning Outcomes:

By the end of the course, students will be able to:

CLO1: Understand the history, structure, and compilation process of C programs, and differentiate between C and C++.
CLO2: Use variables, constants, operators, and I/O functions to write basic C programs.
CLO3: Apply control flow concepts using conditional and iterative statements to solve problems.
CLO4: Develop programs using functions, arrays, strings, pointers, and derived data types like structures and unions.

Cognitive Map of Course Outcomes based on Revised Bloom's Taxonomy

Knowledge Dimension	Remember	Understand	Apply	Analyse	Evaluate	Create
Factual Knowledge	CO_1, CO_2	CO_1				
Conceptual Knowledge		CO_1, CO_3		CO_3		
Procedural Knowledge		CO_2, CO_4	CO_2,CO_3 CO_4,	CO_4		CO_4
Metacognitive Knowledge						

Mapping of PSO with CLO

CLO / PSO	PSO1	PSO2	PSO3	PSO4	PSO5	PSO6	PSO7	PSO8	PSO9
CLO1	3	2			1	2		2	3
CLO2	3	3			1	2		2	3
CLO3	2	3			1	2		2	3
CLO4	3	3			2	2		2	3
Average Contribution	2.75	2.75			1.25	2		2	3

(3=High, 2= Moderate 1= Low)

Suggested readings:

1. Kanetkar, Y. (2018). *Let us C*. BPB publications.
2. Kernighan, B. W., & Ritchie, D. M. (1988). *The C programming language*. prentice-Hall.
3. Griffiths, D., & Griffiths, D. (2012). *Head First C: A Brain-Friendly Guide*. " O'Reilly Media, Inc.".

Semester-I

Course Code : CC-5-PGC-103
Course Title : Web Development
Nature of the Course : CORE
Total Credit assigned : 5
Component of the Course : Lecture: 3 Practical: 2 Tutorial: 0
Distribution of Marks : End Semester: 42 Practical: 40 In Semester: 18

Course Objectives:

CO-1. Understand the fundamentals of web technologies and develop basic web pages using HTML.
CO-2. Apply CSS to style and layout web pages using selectors, box model, and positioning techniques.
CO-3. Implement client-side interactivity using JavaScript, including functions, events, and object models.
CO-4. Develop dynamic web applications using PHP with database connectivity and form handling. Identify common web security threats and apply basic secure coding practices.

Course Content:

Unit	Content	Marks	L	P	T
I	Introduction to Web Development and HTML • Overview of web technologies • Evolution of Web Development • Client-Side vs. Server-Side Development • Components: Web Server, Web Browser, Web Page, Web Sites. • Introduction to Hypertext Markup Language (HTML) • Document type definition • Creating web pages • Elements: Global Attributes, Headers, Lists, Hyperlinks, Tables, Web forms • Inserting images, Frames, Grouping using Div Span.	10	10	10	
II	CSS • Concept of CSS • Types, Properties • Styling: Background, Text format, controlling fonts • Working with List, Tables, Block elements and Objects	10	10	15	

	• CSS Id and Class • Box Model: Border properties, Padding Properties, Margin properties • CSS Advanced: Grouping, Dimension, Display, Positioning, Floating, Align, Pseudo class, Navigation Bar, Image Sprites, Attribute selector • CSS Color • Creating page Layout and Site Designs.				
III	JavaScript • Introduction of JavaScript • Client-Side JavaScript • Server-Side JavaScript • JavaScript Objects • JavaScript Security • Data types and variables • Operators • Functions • Methods and events • Controlling program flow • JavaScript object model • JavaScript classes.	10	12	17	
IV	Server-Side Scripting with PHP and Database Integration • Introduction to Server-Side Scripting • PHP fundamentals • Handling forms and sessions • Introduction to databases • MySQL database management • Database connectivity with PHP • Web accessibility • Web Security • Common Web Security threats • Secure Coding Practices • Authentication and Authorization mechanisms	12	13	18	
Total		42	45	60	0

Mode of In-Semester assessment (18 Marks):

A. Sessional Activity

B. Sessional Tests: (two Sessional Tests)

Sessional Activities:

1: HTML & Web Basics

Create a basic web page using HTML elements (headings, lists, images, links, tables, forms).
Design a webpage layout using and elements.

2: CSS
Apply internal, external, and inline CSS to style a webpage (text, backgrounds, fonts).
Create a responsive layout using CSS box model, positioning, and navigation bar styling.

3: JavaScript
Write JavaScript programs to validate form inputs and handle user events (e.g., button click).
Use functions and control flow to perform basic calculations and DOM manipulations.
4: PHP & Database Integration
Build a PHP form to collect user input and display the submitted data.
Connect PHP with a MySQL database and perform basic operations (Insert, Display).

5: Web Security
Simulate and discuss common web security issues (e.g., XSS, SQL injection) in a controlled lab.
Implement user authentication using PHP (login form with session handling).

List of Practical (but not limited to)

1. Diagram client-server architecture; identify components like web server, browser, and webpage.
2. Create a basic HTML page structure using <!DOCTYPE>, <html>, <head>, and <body>.
3. Use HTML tags for text formatting: headings, paragraphs, emphasis, breaks, and horizontal lines.
4. Create ordered, unordered, and nested lists along with various hyperlinks.
5. Create a table using thead, tbody, tfoot, and merged cells.
6. Design a basic HTML form with text boxes, radio buttons, checkboxes, and submit/reset buttons.
7. Insert images and use <iframe> to embed content.
8. Use and for grouping and structuring content on a web page.
9. Create a multi-page HTML website with internal navigation links.
10. Take a quiz on HTML and review peer projects for feedback.
11. Apply inline, internal, and external CSS to a simple HTML page.
12. Use CSS to style text: change color, size, font, and alignment.
13. Style HTML lists and tables using CSS: custom bullets, borders, background colors.
14. Demonstrate CSS box model: apply margin, padding, and borders.
15. Use CSS classes and IDs to apply selective styling.
16. Create a responsive two-column layout using CSS Flexbox.
17. Use advanced CSS positioning: absolute, relative, fixed, sticky, and display types.
18. Design a horizontal or vertical navigation bar with hover effects using pseudo-classes.
19. Implement image sprites for navigation or interactive buttons.
20. Apply CSS to create a complete layout and design for a multi-page website.
21. Write a JavaScript program using alert(), document.write(), and console.log().
22. Declare and use JavaScript variables of different data types.
23. Perform operations using JavaScript arithmetic, comparison, and logical operators.
24. Define functions and handle user-triggered events like clicks or form submission.
25. Use if, switch, for, and while control structures to drive logic.

26. Create and manipulate arrays and objects in JavaScript.
27. Use JavaScript to access and modify DOM elements dynamically.
28. Validate form input using JavaScript before submission.
29. Create a real-time clock using setTimeout() or setInterval().
30. Create a class using ES6 syntax with constructor and methods.
31. Use localStorage to save and retrieve persistent user data.
32. Build a mini project such as a to-do list or quiz using HTML, CSS, and JavaScript.
33. Write a PHP script to display output and define variables.
34. Use control structures (if-else, switch, loops) in PHP scripts.
35. Handle form data using $_GET and $_POST in PHP.
36. Use PHP sessions and cookies to manage user data across pages.
37. Create a database and tables using MySQL (phpMyAdmin or command line).
38. Connect a PHP script to a MySQL database using mysqli or PDO.
39. Perform CRUD operations (Create, Read, Update, Delete) in a web form using PHP.
40. Display data from a database in an HTML table using PHP.
41. Build a secure login system using PHP and hashed passwords.
42. Sanitize form inputs to prevent XSS and SQL injection.
43. Demonstrate a SQL injection vulnerability and implement security measures.
44. Implement a role-based access control system (admin/user).
45. Develop a complete dynamic web application integrating HTML, CSS, JavaScript, PHP, and MySQL.

Expected Course Learning Outcomes:

By the end of the course, students will be able to:

CLO1: Design and develop structured web pages using HTML and CSS.
CLO2: Implement dynamic client-side functionalities using JavaScript.
CLO3: Create interactive web applications using PHP and connect them to a MySQL database.
CLO4: Apply basic web security principles and best practices in web development.

Cognitive Map of Course Outcomes based on Revised Bloom's Taxonomy

Knowledge Dimension	Remember	Understand	Apply	Analyse	Evaluate	Create
Factual Knowledge	CO_1, CO_2	CO_1				
Conceptual Knowledge		CO_1,CO_2, CO_3		CO_3, CO_4	CO_4	
Procedural Knowledge		CO_2,CO_3, CO_4	CO_1,CO_2, CO_3, CO_4	CO_4		CO_4
Metacognitive Knowledge						

Mapping of PSO with CLO

CLO / PSO	PSO1	PSO2	PSO3	PSO4	PSO5	PSO6	PSO7	PSO8	PSO9
CLO1	3	2	3			3		2	3
CLO2	2	3	3			3		2	3
CLO3	2	3	3	3		3		2	3
CLO4	2	2	3	2		3		3	3
Average Contribution	2.25	2.25	3	1.25		3		2.25	3

(3=High, 2= Moderate 1= Low)

Suggested readings:

1. Holzner, S. (2007). *PHP: the complete reference*. McGraw-Hill, Inc..
2. DeBolt, V. (2007). *MASTERING INTEGRATED HTML AND CSS (With CD)*. John Wiley & Sons.

Semester-I

Course Code : EL-5-PGC-101
Course Title : Multimedia Systems and Applications
Nature of the Course : Elective
Total Credit assigned : 5
Component of the Course : Lecture: 4 Practical: 1 Tutorial: 0
Distribution of Marks : End Semester: 56 Practical: 20 In Semester: 24

Course Objectives:

CO-1. Understand the fundamentals, components, applications, and scope of multimedia and virtual reality.
CO-2. Use appropriate fonts, faces, and text tools to design and incorporate text effectively in multimedia.
CO-3. Understand and apply image formats, color concepts, and drawing techniques in multimedia design.
CO-4. Work with digital and MIDI audio, video formats, and perform basic editing tasks. Apply animation principles and techniques using suitable tools and file formats.

Course Content:

Unit	Content	Marks	L	P	T
I	**Multimedia Fundamentals** • Introduction to multimedia, components, uses, applications, and virtual reality.	10	10	6	0
II	**Text in Multimedia (10 Hours)** • Fonts & Faces • Using Text in Multimedia • Font Editing & Design Tools • Hypermedia & Hypertext	10	10	4	0
III	**Images in Multimedia (14 hours)** • Still Images: Bitmaps, Vector Drawing, 3D Drawing & Rendering • Natural Light & Colors, Computerized Colors, Color Palettes • Image File Formats	12	14	6	0
IV	**Audio and Video in Multimedia** • Digital Audio, MIDI Audio, MIDI vs Digital Audio • Audio File Formats • How Video Works, Analog and Digital Video • Video File Formats, Video Shooting and Editing	12	14	6	0
V	**Animation in Multimedia** • Principles of Animation • Animation Techniques • Animation File Formats	12	12	8	
Total		56	60	30	0

Mode of In-Semester assessment (24Marks):

A. Sessional Activity

B. Sessional Tests: (two Sessional Tests)

Sessional Activities:

1.Introduction to Multimedia

- Create a presentation on multimedia components and applications.
- Demonstrate a simple virtual reality environment or VR video.

2.Text in Multimedia

- Experiment with different fonts and styles using font editing tools.
- Design a hyperlinked multimedia document using text and images.

3.Images in Multimedia

- Create and edit bitmap and vector images using graphic software.
- Compare different image file formats by saving/exporting and analyzing quality and size.

4.Audio and Video in Multimedia

- Record and edit digital audio clips and compare with MIDI files.
- Import, edit, and export video files using video editing software.

5.Animation in Multimedia

- Create a simple animation applying basic principles (e.g., frame-by-frame).
- Explore different animation techniques and export in various file formats.

Lab Works (But not limited to)

1. Create a presentation/introduction on multimedia components and applications.
2. Explore multimedia usage in various fields (education, entertainment, healthcare).
3. Demonstrate a simple virtual reality simulation or 360-degree media example.
4. Identify and compare different font types and font faces.
5. Use text in a multimedia document with emphasis on style and readability.
6. Create a poster using text creatively in multimedia layout software (e.g., Photoshop, Canva).
7. Explore and compare font editing tools (e.g., FontForge, Glyphr Studio).
8. Create a simple hypertext system (internal linking in HTML).
9. Demonstrate navigation using hyperlinks in a multimedia presentation.
10. Create a concept map or flowchart using hypermedia elements.
11. Use vector drawing tools (e.g., Inkscape or Illustrator) to draw simple graphics.
12. Convert an image to bitmap format and apply basic edits.
13. Render a basic 3D object using 3D modeling software (e.g., Blender, Tinkercad).
14. Demonstrate use of light and color using photo editing software.
15. Create a custom color palette and apply it to a design project.
16. Compare various image file formats (e.g., JPG, PNG, GIF, BMP) with pros/cons.
17. Create an image collage with mixed file formats and export in different formats.
18. Record a short audio clip and save it in different digital audio formats (e.g., WAV, MP3).
19. Use MIDI software to create a simple music sequence.
20. Compare MIDI and digital audio with an example project.

21. Identify common audio file formats and convert between them using tools.
22. Record a short video clip using a phone/camera and explain basic video setup.
23. Convert the recorded video into multiple file formats (MP4, AVI, MOV).
24. Use video editing software (e.g., OpenShot, Shotcut) to trim and edit a video.
25. Create a slideshow-style video combining still images and background audio.
26. Study and demonstrate key principles of animation (timing, squash & stretch, etc.).
27. Create a frame-by-frame animation using free tools (e.g., Pencil2D, Krita).
28. Apply animation techniques like tweening or morphing using animation software.
29. Export animations into multiple file formats (GIF, MP4, SWF) and compare quality.

Expected Course Learning Outcomes:

By the end of the course, students will be able to:

CLO1: Understand the basic concepts, components, and applications of multimedia and virtual reality.
CLO2: Apply appropriate text styles, fonts, and tools in multimedia design.
CLO3: Create and manipulate images using bitmap, vector, and 3D drawing techniques.
CLO4: Handle audio and video formats and perform basic editing for multimedia applications.
CLO5: Implement animation principles and techniques using suitable tools and formats.

Cognitive Map of Course Outcomes based on Revised Bloom's Taxonomy

Knowledge Dimension	Remember	Understand	Apply	Analyse	Evaluate	Create
Factual Knowledge	CO_1, CO_2, CO_3, CO_4	CO_1, CO_2, CO_3, CO_4				
Conceptual Knowledge		CO_1, CO_3, CO_5		CO_3, CO_5	CO_5	
Procedural Knowledge		CO_2, CO_3, CO_4, CO_5	CO_2, CO_3, CO_4, CO_5	CO_4		CO_5
Metacognitive Knowledge						

Mapping of PSO with CLO

CLO / PSO	PSO1	PSO2	PSO3	PSO4	PSO5	PSO6	PSO7	PSO8	PSO9
CLO1	3	1	2			3		2	3
CLO2	2	2	3			3		2	3
CLO3	2	3	3			3		2	3
CLO4	2	2	3			3		2	3
CLO5	2	3	3			3		2	3
Average Contribution	2.2	2.2	2.8			3		2	3

(3=High, 2= Moderate 1= Low)

Suggested readings:

1. Vaughan, T. (2014). *Multimedia: Making it work*. McGraw-Hill Education Group.
2. Steinmetz, R., &Nahrstedt, K. (2012). *Multimedia: computing, communications and applications*. Pearson Education India.

Semester-I

PGC-EL-102 Operating Systems

Course Code : EL-5-PGC-102
Course Title : Operating Systems
Nature of the Course : Elective
Total Credit assigned : 5
Component of the Course : Lecture: 4 Practical: 1 Tutorial: 0
Distribution of Marks : End Semester: 56 Practical: 20 In Semester: 24

Course Objectives:

CO1. Understand basic functions, types, and evolution of operating systems, including real-time systems.
CO2. Explain process and thread management concepts, scheduling, and deadlock handling techniques.
CO3. Apply memory allocation strategies, virtual memory concepts, file systems, and device management.
CO4. Use synchronization mechanisms to solve concurrency issues and prevent deadlocks.

Course Content:

Unit	Content	Marks	L	P	T
I	**Introduction to Operating Systems** • Basic OS functions and types • Process control and real-time systems • Historical evolution of operating systems	12	12	6	0
II	**Process and Thread Management** • System view of processes and resources • Process hierarchy and abstraction • Threads and threading issues • Thread libraries and process scheduling • Deadlocks handling	14	16	8	0
III	**Memory Management, File and I/O Management** • Physical and virtual address space • Memory allocation strategies (fixed and variable partitions, paging, segmentation) • Virtual memory concepts • Directory structure and file operations • File allocation methods • Device management	15	16	8	0
IV	**Synchronization and Concurrency** • Critical sections and synchronization mechanisms • Semaphores and monitors • Inter-process communication methods • Deadlocks handling	15	16	8	0
Total		56	60	30	0

Mode of In-Semester assessment (24Marks):

A. Sessional Activity

B. Sessional Tests: (two Sessional Tests)

Sessional Activities:

1: Introduction to Operating Systems

- Prepare a chart comparing types of operating systems (e.g., Windows, Linux, RTOS).
- Create a poster/timeline showing how operating systems have evolved over time.

2: Process and Thread Management

- Draw and explain process states (Ready, Running, Waiting, etc.) with arrows.
- Write a simple program using threads (e.g., two threads printing messages).

3: Memory, File & I/O Management

- Use diagrams to show how paging or segmentation works.
- Write a basic file handling program to open, read, and write a file.

4: Synchronization and Concurrency

- Simulate the producer-consumer problem using simple code or flowchart.
- Discuss and show a basic example of deadlock using a real-life scenario (e.g., traffic deadlock).

Lab Works (but not limited to)

CO1. Identify and describe the key functions and types of operating systems through a practical presentation.
CO2. Demonstrate installation of different operating systems (Windows, Linux, RTOS example).
CO3. Learn basic linux commands.
CO4. Explore the basic commands of Linux/Unix OS and create a cheat sheet.
CO5. Write a shell script to display system information (OS type, kernel version, uptime).
CO6. Simulate the OS evolution timeline using a diagram or interactive tool.
CO7. Use process-related commands (ps, top, kill, nice, etc.) to monitor and manage processes in Linux.
CO8. Simulate process scheduling using First Come First Serve (FCFS) algorithm.
CO9. Simulate Shortest Job First (SJF) scheduling algorithm.
CO10. Simulate Priority Scheduling with preemptive and non-preemptive variations.
CO11. Implement Round Robin scheduling with configurable time quantum.
CO12. Simulate memory allocation using fixed partitioning.
CO13. Simulate memory allocation using variable partitioning (First Fit, Best Fit, Worst Fit).

Expected Course Learning Outcomes:

By the end of the course, students will be able to:

CLO1: Explain the fundamental functions, types, and historical evolution of operating systems, including the role of real-time systems.
CLO2: Analyze the concepts of processes, threads, scheduling algorithms, and deadlock handling in operating systems.
CLO3: Apply memory management techniques, file system operations, and I/O management strategies in various OS environments.
CLO4: Implement synchronization mechanisms to manage concurrency and ensure correct inter-process communication.

Cognitive Map of Course Outcomes based on Revised Bloom's Taxonomy

Knowledge Dimension	Remember	Understand	Apply	Analyse	Evaluate	Create
Factual Knowledge	CO_1, CO_2	CO_1				
Conceptual Knowledge		CO_1, CO_2, CO_3	CO_2, CO_3	CO_3, CO_4	CO_4	
Procedural Knowledge		CO_2, CO_3, CO_4	CO_3, CO_4	CO_4	CO_4	
Metacognitive Knowledge						

Mapping of PSO with CLO

CLO / PSO	PSO1	PSO2	PSO3	PSO4	PSO5	PSO6	PSO7	PSO8	PSO9
CLO1	3	2			2	3		2	2
CLO2	2	3			2	3		2	3
CLO3	2	2			2	3		2	3
CLO4	2	3			2	3		2	3
Average Contribution	2.25	2.5			2	3		2	2.75

(3=High, 2= Moderate 1= Low)

Suggested readings:

1. Silberschatz, A., Galvin, P. B., & Gagne, G. (2013). *Operating system concepts essentials*. Wiley Publishing.
2. Tanenbaum, A. S., &Bos, H. (2015). *Modern operating systems*. Pearson Education, Inc..
3. Arpaci-Dusseau, R. H., &Arpaci-Dusseau, A. C. (2018). Operating systems: Three easy pieces.
4. Stallings, W. (2009). *Operating systems internals and design principles*. Pearson Prentice Hall.

Semester-II

Course Code : CC-5-PGC-201
Course Title : Software Engineering
Nature of the Course : CORE
Total Credit assigned : 5
Component of the Course : Lecture: 4 Practical:0 Tutorial: 1
Distribution of Marks : End Semester: 70 Practical: 0 In Semester:30

Course Objectives:

CO1. Understand the evolution, characteristics, and challenges of software engineering.
CO2. Describe various software development life cycle models and their applications.
CO3. Analyze and document software requirements using effective specification techniques.
CO4. Apply software design principles and testing techniques to ensure software quality.
CO5. Understand software quality, reliability, maintenance, and project management practices.

Course Content:

Unit	Content	Marks	L	P	T
I	**Introduction to Software Engineering** • Characteristics and attributes of Software • What is Software Engineering? • Emergence of Software Engineering: Early programming to Object-oriented design • Key challenges in software engineering	11	5	0	1
II	**Software Process and Life Cycle Models** • Understanding software processes • Process framework and SDLC models: ▪ Waterfall, Iterative, Prototyping, Incremental, Spiral ▪ Comparison of different life cycle models	11	12	0	2
III	**Software Requirements Analysis and Specification** • Requirements gathering and analysis • Software Requirements Specification (SRS) • Types of requirements: Functional and non-functional • Characteristics and structure of a good SRS document	10	10	0	2
IV	**Software Design and Testing** • Fundamental Design Concepts: Function-oriented and Object-oriented design	12	15	0	5

	• Modularization and decomposition: Cohesion and Coupling • Design notations: Flow charts, Data Flow Diagrams, Structure charts • Coding Style and Documentation guidelines • Code Review: Walk-Throughs, Inspection • Levels of software testing: Unit, Integration, System, Acceptance • Testing techniques and comparison: Black box and White box testing • Debugging approaches and guidelines • Integration and system testing approaches				
V	**Software Reliability, Quality, Maintenance, and Project Management** • What is Software reliability • Software quality management system and ISO 9000 standards • Compatibility maturity model (CMM) • Basics of software maintenance and types • Software project management and responsibilities of a software project manager • Project planning, estimation, and scheduling • Metrics for project size and estimation models • Risk management and Software Configuration Management (SCM)	12	18	0	5
Total		56	60	0	15

Mode of In-Semester assessment (24 Marks):

A. Sessional Activity

B. Sessional Tests: (two Sessional Tests)

Sessional Activities:

1: Introduction to Software Engineering

- Create a chart showing the evolution from early programming to object-oriented design.
- List and explain the key attributes and challenges of software in a group discussion.

2: Software Process and Life Cycle Models

- Draw and compare diagrams of different SDLC models (Waterfall, Spiral, Iterative).
- Group activity: Choose a software idea and decide which life cycle model suits it best, with reasons.

3: Software Requirements Analysis and Specification

- Prepare a sample SRS (Software Requirements Specification) for a simple project (e.g., Library Management System).
- Identify and classify functional vs non-functional requirements from a given case study.

4: Software Design and Testing

- Draw a Data Flow Diagram (DFD) or Flowchart for a small application (e.g., ATM system).
- Perform black-box and white-box testing on a simple program (e.g., calculator app).

5: Software Quality, Maintenance & Project Management

- Create a project plan with estimation (lines of code or function points) for a mini-project.
- Prepare a risk management table and discuss SCM (Software Configuration Management) in pairs

.

Case Study 1: Library Management System

📌 Context:

A college/university library wants to automate its operations such as issuing/returning books, maintaining records of students and books, and tracking defaulters.

📘 **Software Engineering Concepts Applied:**

- **Requirements Analysis:**
  - *Functional Requirements*: Issue and return books, maintain student records, generate reports, search catalog.
  - *Non-Functional Requirements*: Performance, reliability, user-friendly interface.
  - Develop a structured **SRS document**.
- **Software Process Model:**
  - Adopt the **Iterative Model** to gradually refine features and user interface based on librarian feedback.
- **Design:**
  - Use **DFDs and Structure Charts** to represent the flow and hierarchy of system components.
  - Apply **Modular Design** (e.g., modules for book entry, issuing, returning, fine calculation).
- **Testing:**
  - Apply **Black Box Testing** for user input validations and **White Box Testing** for business logic.
  - Conduct **Unit and Integration Testing** of modules.
- **Project Management:**
  - Estimation using **LOC or Function Point Analysis**.
  - Identify risks like data loss or user resistance and create mitigation plans.

Case Study 2: Online Food Delivery App

📌 Context:

A startup plans to launch a food delivery app to connect users with local restaurants. The app must support registration, browsing menus, placing orders, and real-time delivery tracking.

📘 **Software Engineering Concepts Applied:**

- **Requirements Specification:**
  - *Functional*: User registration, order placement, real-time tracking, feedback system.
  - *Non-Functional*: Scalability, security (payment), performance.

    - Draft a well-structured **SRS document** with clear scope and constraints.
- **Software Process Model:**
    - Use the **Spiral Model** to address high risk areas like payment and security early in development.
- **Design:**
    - Apply **Object-Oriented Design Principles** (classes for User, Order, Restaurant, Delivery).
    - Use **UML Diagrams** and **Flowcharts** for visual representation.
- **Testing and Debugging:**
    - Conduct **System Testing** on Android/iOS platforms.
    - Perform **Acceptance Testing** with restaurant partners.
    - Debug using systematic logging and user feedback loops.
- **Quality and Project Management:**
    - Follow ISO standards and focus on **User-Centric Quality**.
    - Manage delivery deadlines using **Gantt Charts** and tools like **Trello/JIRA**.
    - Plan for **Maintenance**, including feature updates and bug fixes.

Expected Course Learning Outcomes:

By the end of the course, students will be able to:

CLO1: Explain the nature of software, define software engineering, and identify key historical developments and challenges.
CLO2: Compare software process models such as Waterfall, Spiral, and Incremental, and select appropriate models for a given scenario.
CLO3: Perform requirements analysis, differentiate between functional and non-functional requirements, and create a well-structured SRS document.
CLO4: Use design methodologies, represent designs with appropriate notations, and apply various testing levels and debugging techniques.
CLO5: Describe software quality standards, maintenance types, project estimation methods, and risk management techniques.

Cognitive Map of Course Outcomes based on Revised Bloom's Taxonomy

Knowledge Dimension	Remember	Understand	Apply	Analyse	Evaluate	Create
Factual Knowledge	CO_1, CO_2	CO_1				
Conceptual Knowledge		CO_1, CO_2, CO_3, CO_5	CO_2, CO_4, CO_5	CO_3, CO_5	CO_4, CO_5	
Procedural Knowledge		CO_3, CO_4, CO_5	CO_3, CO_4	CO_4, CO_5	CO_5	
Metacognitive Knowledge						

Mapping of PSO with CLO

CLO / PSO	PSO1	PSO2	PSO3	PSO4	PSO5	PSO6	PSO7	PSO8	PSO9
CLO1	3	2			3	2	1	2	2
CLO2	2	3			3	2	1	2	2
CLO3	2	3			3	2	2	2	2
CLO4	2	3			3	2	2	2	2
CLO5	2	2			3	2	2	2	2
Average Contribution	2.2	2.6			3	2	1.6	2	2

(3=High, 2= Moderate 1= Low)

Suggested readings:

1. Mall, R. (2018). *Fundamentals of software engineering*. PHI Learning Pvt.Ltd..
2. Pressman, R. S. (2005). *Software engineering: a practitioner's approach*. Palgrave macmillan.
3. Jalote, P. (2012). *An integrated approach to software engineering*. Springer Science & Business Media.
4. Rohit, K. (2010). *Software engineering: Principles and practices*. Vikas Publishing House.

Semester-II

Course Code : CC-5-PGC-202
Course Title : Database Management System
Nature of the Course : CORE
Total Credit assigned : 5
Component of the Course : Lecture: 3 Practical: 2 Tutorial: 0
Distribution of Marks : End Semester: 42 Practical: 40 In Semester: 18

Course Objectives:

CO1. Understand the fundamentals of databases, ER modeling, and the purpose of DBMS architectures.
CO2. Analyze the relational data model and apply relational constraints effectively.
CO3. Apply relational algebra operations for querying and manipulating data.
CO4. Use SQL for defining, manipulating, and retrieving relational data efficiently.
CO5. Understand and apply database design principles and normalization techniques to eliminate anomalies.

Course Content:

Unit	Content	Marks	L	P	T
I	**Introduction to Databases and ER Modeling** • Overview of Databases, Importance, and Evolution • Types of Databases and DBMS Architecture • Entities, Attributes and Relationships • Entity Relationship (ER) Modeling: Concepts and Tools	8	8	0	0
II	**Relational Data Model and SQL** • Relational Model Concepts: domain, attributes, tuples and relations • Characteristics of relations • Constraints: domain constraint, key constraint and referential integrity constraint and foreign key	8	10	12	0
III	**Relational algebra** • Unary relational operations: SELECT , PROJECT • Relational Algebra operations from set theory: UNION, INTERSECTION, MINUS, Cartesian product • Binary relational operations: JOIN operation, DIVISION operation • Different types of JOINs.	8	9	18	0
IV	**Structured Query Language** • Definition of SQL, versions of SQL • Data Definition Language (DDL) : CREATE, ALTER, DROP, TRUNCATE command	10	10	20	0

	• Data Manipulation Language (DML): SELECT, INSERT, UPDATE, DELETE commands • SQL aggregate functions. • Writing SQL Queries and Subqueries				
V	**Database Design and Normalization** • Basics of Database Design • Anomalies in database design • Functional Dependencies and Normalization Techniques (1NF, 2NF, 3NF, BCNF)	8	8	10	0
Total		42	45	60	0

Mode of In-Semester assessment (18 Marks):

A. Sessional Activity

B. Sessional Tests: (two Sessional Tests)

Sessional Activities:

1: Introduction to Databases and ER Modeling

- Create an ER diagram for a basic system (e.g., Library or Student Management System).
- Identify entities, attributes, and relationships from a real-life scenario.

2: Relational Data Model and SQL

- Create relational schemas and apply constraints (Primary Key, Foreign Key).
- Draw tables and fill sample data showing domain, tuples, and relations.

3: Relational Algebra

- Perform relational algebra operations (SELECT, PROJECT, UNION, etc.) using sample datasets.
- Solve basic JOIN and DIVISION operations manually or using diagrams.

4: Structured Query Language (SQL)

- Write SQL queries using DDL and DML commands (CREATE, INSERT, SELECT, etc.).
- Perform aggregate functions and write subqueries on a sample database.

5: Database Design and Normalization

- Identify anomalies in a poorly designed table and normalize it to 1NF, 2NF, 3NF.
- Create dependency diagrams and convert to BCNF.

Lab Work:

Consider the following Schema

Supplier (SID, Sname, branch, city, phone)

Part(PID,Pname, color, price) Supplies(SID, PID, qty, date_supplied)

DDL Commands

1. Create the above tables
2. Add a new attribute state in supplier table

3. Remove attribute city from supplier table
4. Modify the datatype of phone attribute
5. Change the name of attribute city to address
6. Change a table's name, supplier to sup
7. Use truncate to delete the contents of supplies table
8. Remove the part table from database

DML Commands

1. Insert at least 10 records in tables supplier, part and supplies
2. Show the contents in tables supplier, part and supplies
3. Find the name and city of all suppliers
4. Find the name and phone no of all suppliers who stay in 'Delhi'
5. Find all distinct branches of suppliers
6. Delete the record of the supplier whose SID is 204001
7. Delete all records of supplier table
8. Delete all records of suppliers whose city starts with capital A.
9. Find the supplier names which have 'lk' in any position
10. Find the supplier name where 'R' is in the second position
11. Find the name of supplier whose name starts with 'V' and ends with 'A'
12. Change the city of all suppliers to 'BOMBAY'
13. Change the city of supplier 'Vandana' to 'Goa'

Queries with Constraints

1. Create the supplier table with Primary Key Constraint
2. Create supplies table with Foreign key Constraint
3. Create a part table with UNIQUE Constraint
4. Create supplier Table with Check Constraints
5. Create Supplier table with Default

Expected Course Learning Outcomes:

By the end of the course, students will be able to:

CLO1: Explain the importance of databases, describe DBMS architecture, and construct ER diagrams using entities, attributes, and relationships.
CLO2: Identify relational model elements, describe characteristics of relations, and apply integrity constraints like primary and foreign keys.
CLO3: Perform relational algebra operations such as SELECT, PROJECT, JOIN, and UNION to process queries.
CLO4: Write SQL commands using DDL, DML, aggregate functions, and subqueries to manage and retrieve data.
CLO5: Analyze database anomalies and apply normalization up to BCNF using functional dependency concepts.

Cognitive Map of Course Outcomes based on Revised Bloom's Taxonomy

Knowledge Dimension	Remember	Understand	Apply	Analyse	Evaluate	Create
Factual Knowledge	CO_1, CO_2	CO_1				
Conceptual Knowledge		CO_1, CO_2, CO_5	CO_2, CO_4, CO_5	CO_2, CO_5	CO_5	
Procedural Knowledge		CO_3, CO_4, CO_5	CO_3, CO_4, CO_5	CO_3, CO_5	CO_5	
Metacognitive Knowledge						

Mapping of PSO with CLO

CLO / PSO	PSO1	PSO2	PSO3	PSO4	PSO5	PSO6	PSO7	PSO8	PSO9
CLO1	3	2		3	2	2			1
CLO2	2	2		3	2	2			1
CLO3	2	3		3	2	2			2
CLO4	2	3	1	3	2	2			2
CLO5	2	3		3	3	2			2
Average Contribution	2.2	2.6	0.2	3	2.2	2			1.6

(3=High, 2= Moderate 1= Low)

Suggested readings:

1. Elmasri, R., &Navathe, S. B. (2004). *Fundamentals of database systems:[4-th edition*].
2. Ramakrishnan, R., &Gehrke, J. (2003). *Database management systems* third edition.
3. Silberschatz, A., Korth, H. F., & Sudarshan, S. (2011). *Database system concepts*.
4. Elmasri, R., &Navathe, S. B. (2013). *Database systems: models, languages, design, and application programming*. Pearson.

Semester-II

Course Code : CC-5-PGC-203
Course Title : Project Work
Nature of the Course : CORE
Total Credit assigned : 5
Component of the Course : Lecture: 0 Practical: 5 Tutorial: 0
Distribution of Marks : End Semester: 0 Practical: 100 In Semester: 0

Course Objectives:

CO1. To enable students to apply the theoretical and practical knowledge acquired throughout the PGDCA program in developing a comprehensive software solution, simulating real-world problems from industry, academia, or research. The project aims to foster skills in system analysis, design, development, testing, documentation, and deployment through adherence to standard software development practices.

Course Content:

The Postgraduate Diploma in Computer Applications (PGDCA) program aims to prepare students for roles such as programmers, system designers, and software engineers in the IT industry. The curriculum provides comprehensive theoretical and practical knowledge in core computer science areas. A six-month major project in the final semester allows students to apply this knowledge to develop quality software solutions.

Steps to be followed to design and implement the project :

- Apply theoretical and practical techniques to solve real-life problems in industry, academia, or research.
- Understand and describe the System Development Life Cycle (SDLC) phases related to their project.
- Evaluate system requirements and problem definitions.
- Collect and analyse information for requirements gathering.
- Conduct feasibility studies including cost-benefit, technical, time, and operational feasibility.
- Use fact-finding methods for data collection.
- Construct and evaluate Data Flow Diagrams (DFDs), data dictionaries, decision trees, and decision tables.
- Create and analyse system flowcharts, Entity-Relationship (ER) diagrams, and state transition diagrams.
- Prepare Software Requirement Specifications (SRS) and hardware requirements.
- Plan and design functional and non-functional system requirements.
- Design system inputs, outputs, validity checks, and user interfaces.
- Perform coding and documentation of the project.
- Apply testing techniques and strategies.
- Generate project reports.
- Deploy the project in real or lab environments.
- Identify maintenance procedures and plan future enhancements.
- Include appendices as necessary in the project report.

Types of Projects

- Real-life projects in industry, research labs, educational institutions, or software companies are preferred.
- Alternatively, students may propose and develop a project with supervisor approval.
- Projects may be carried out on-campus or at approved organizations.

Mode of In-Semester assessment (30 Marks):

1. Project Proposal Submission
   - Students submit a synopsis including:
     - Project title
     - Objectives
     - Proposed tools and technologies
     - Preliminary analysis diagrams (DFD, ER, etc.)
   - Reviewed and approved by internal supervisor.
2. Phase I Presentation – Project Planning & Analysis
   - Activities:
     - Problem definition and scope
     - Feasibility study (technical, operational, financial)
     - Data collection methods
     - Preliminary system analysis (DFDs, ER diagrams, flowcharts)
   - Evaluation based on clarity, feasibility, and planning.
3. Phase II Presentation – System Design & Development
   - Activities:
     - SRS preparation
     - Database design and data structures
     - UI/UX design
     - Process logic of each module
     - Coding of main modules
   - Evaluation based on design completeness and implementation progress.
4. Phase III Presentation – Testing & Deployment
   - Activities:
     - Implementation of functional modules
     - Testing techniques and error logs
     - Validation checks
     - Report generation
     - Deployment (on lab or real environment)
   - Evaluation includes testing results, project functionality, and readiness.
5. Documentation Submission (Draft)
   - Submission of partial documentation including:
     - Introduction
     - Analysis
     - Design
     - Coding samples
     - Testing strategies
6. Final Presentation & Viva
   - Complete project demonstration in front of evaluation panel.

- Viva based on:
    - Project concept and problem understanding
    - Design and coding process
    - Tools and technologies used
    - Challenges faced and solutions
    - Future enhancements

7. Final Report Submission
    - Two hard copies of the bound report.
    - CD containing:
        - Source code
        - Executables
        - Database
        - Screenshots
        - Documentation (Word + PDF format)

Expected Course Learning Outcomes:

By the end of the course, students will be able to:

CLO-1. By the end of the PGDCA Project, students will be able to design, develop, test, and document a real-world software application by applying theoretical and practical knowledge of computer science, demonstrating proficiency in system analysis, software development life cycle (SDLC), user interface design, coding, testing strategies, and deployment, while adhering to professional standards of documentation, teamwork, and project presentation.

Cognitive Map of Course Outcomes based on Revised Bloom's Taxonomy

Knowledge Dimension	Remember	Understand	Apply	Analyse	Evaluate	Create
Factual Knowledge						
Conceptual Knowledge						
Procedural Knowledge			CO_1	CO_1	CO_1	CO_1
Metacognitive Knowledge						

Mapping of PSO with CLO

CLO / PSO	PSO1	PSO2	PSO3	PSO4	PSO5	PSO6	PSO7	PSO8	PSO9
CLO1	2	3	2	2	3	2	3	3	3
Average Contribution	2	3	2	2	3	2	3	3	3

(3=High, 2= Moderate 1= Low)

Semester-II

Course Code : EL-5-PGC-201
Course Title : OOP with JAVA
Nature of the Course : Elective
Total Credit assigned : 5
Component of the Course : Lecture: 3 Practical: 2 Tutorial: 0
Distribution of Marks : End Semester: 42 Practical: 40 In Semester: 28

Course Objective:

CO1. Understand the core features of Java, the role of JVM, and basic programming constructs.
CO2. Apply arrays, strings, and wrapper classes in Java applications.
CO3. Implement classes, objects, and methods, including object passing and method overloading.
CO4. Explain and use inheritance concepts including method overriding and multilevel inheritance.
CO5. Demonstrate the use of interfaces and abstract classes in building flexible Java programs.

Course Content:

Unit	Content	Marks	L	P	T
I	**Basics of Java** • Introduction to Java Features, Byte Code, and Java Virtual Machine • JDK Overview, Data Types, and Operators • Control Statements: If-else, switch, loops, break, and continue	7	8	12	0
II	**Array, String, and Wrapper Class** • Single and Multidimensional Array Usage • String and StringBuffer Classes, String Operations • Introduction to Wrapper Classes	7	8	12	0
III	**Classes, Objects, and Methods** • Class and Object Fundamentals, Constructor, and Constructor Overloading • Method Overloading, Recursion • Passing and Returning Objects in Methods, 'new' Operator, 'this' and 'static' Keywords • Understanding 'finalize()' Method, Access Control, Modifiers • Abstract Classes	8	10	12	0
IV	**Inheritance** • Inheritance Basics, Inheriting Data Members and Methods, Constructor in Inheritance	10	10	14	0

	• Method Overriding, 'super' Keyword, Multilevel Inheritance • Final Keyword to Stop Inheritance XML parsing and transformation • Combining DHTML and XML to create dynamic web applications				
V	**Interfaces** • Introduction to Interfaces, Interface Implementation, • Extending interfaces • Abstract Class and Interface	10	9	10	
Total		42	45	60	0

Mode of In-Semester assessment (28 Marks):

A. Sessional Activity

B. Sessional Tests: (two Sessional Tests)

Sessional Activities:

1: Basics of Java

- Write a Java program using basic data types, operators, and control statements (if, switch, loops).
- Compile and run a Java program, showing how bytecode and JVM work.

2: Arrays, Strings, and Wrapper Classes

- Implement programs using single and multi-dimensional arrays.
- Write programs using String, StringBuffer, and demonstrate wrapper class usage.

3: Classes, Objects, and Methods

- Create a class with constructors, method overloading, and use of this and static.
- Write a Java program using recursion and object passing between methods.

4: Inheritance

- Demonstrate single and multilevel inheritance with method overriding and use of super.
- Implement a program using abstract class and final keyword.

5: Interfaces

- Create a program that implements and extends interfaces.
- Build a simple app combining XML and Java (or DHTML+XML) to demonstrate structured data display.

Lab Work:

1. Write a program to read two numbers from user and print their product.
2. Write a program to print the square of a number passed through command line arguments.
3. Write a program to send the name and surname of a student through command line arguments and print a welcome message for the student.
4. Write a java program to find the largest number out of n natural numbers.
5. Write a java program to find the Fibonacci series & Factorial of a number using recursive and non recursive functions.
6. Write a java program to multiply two given matrices.
7. Write a Java program for sorting a given list of names in ascending order.

8. Write a Java program that checks whether a given string is a palindrome or not .Ex:MADAM is a palindrome.
9. Write a java program to read n number of values in an array and display it in reverse order.
10. Write a Java program to perform mathematical operations. Create a class called AddSub with methods to add and subtract. Create another class called MulDiv that extends from AddSub class to use the member data of the super class. MulDiv should have methods to multiply and divide A main function should access the methods and perform the mathematical operations.
11. Create a JAVA class called Student with the following details as variables within it.
    a. USN, NAME, BRANCH, PHONE, PERCENTAGE
    b. Write a JAVA program to create n Student objects and print the USN, Name, Branch, Phone, and percentage of these objects with suitable headings.
12. Write a Java program that displays the number of characters, lines and words in a text.
13. Write a Java program to create a class called Shape with methods called getPerimeter() and getArea(). Create a subclass called Circle that overrides the getPerimeter() and getArea() methods to calculate the area and perimeter of a circle.
14. Write a Java program to create a class Employee with a method called calculateSalary(). Create two subclasses Manager and Programmer. In each subclass, override the calculateSalary() method to calculate and return the salary based on their specific roles.
15. Write a Java program using an interface called 'Bank' having function 'rate_of_interest()'. Implement this interface to create two separate bank classes 'SBI' and 'PNB' to print different rates of interest. Include additional member variables, constructors also in classes 'SBI' and 'PNB'.

Expected Course Learning Outcomes:

By the end of the course, students will be able to:

CLO1:	Describe Java features, JVM, bytecode, and use control statements and operators effectively.
CLO2:	Create and manipulate single/multi-dimensional arrays, perform string operations, and utilize wrapper classes.
CLO3:	Develop Java programs using classes, constructors, method overloading, and access control features.
CLO4:	Implement inheritance, method overriding, and apply super, final, and multilevel inheritance in code.
CLO5:	Define and implement interfaces, extend interfaces, and differentiate between abstract classes and interfaces.

Cognitive Map of Course Outcomes based on Revised Bloom's Taxonomy

Knowledge Dimension	Remember	Understand	Apply	Analyse	Evaluate	Create
Factual Knowledge	CO_1	CO_1, CO_4, CO_5				
Conceptual Knowledge		CO_1, CO_3, CO_4, CO_5	CO_3, CO_4, CO_5	CO_4, CO_5	CO_5	
Procedural Knowledge			CO_2, CO_3, CO_4, CO_5	CO_3, CO_4		CO_3, CO_4, CO_5
Metacognitive Knowledge						

Mapping of PSO with CLO

CLO / PSO	PSO1	PSO2	PSO3	PSO4	PSO5	PSO6	PSO7	PSO8	PSO9
CLO1	2	3	1		2	3			2
CLO2	2	3	1		2	3			2
CLO3	2	3	2		3	3	1		2
CLO4	2	3	2		3	3	1		2
CLO5	2	3	2		3	3	1		2
Average Contribution	2	3	1.6		2.6	3	0.6		2

(3=High, 2= Moderate 1= Low)

Suggested readings:

1. Schildt, H., &Skrien, D. (2012). *Java Programming: A Comprehensive Introduction*. McGraw-Hill Higher Education.
2. Balaguruswamy, E. (2014). *Programming with Java-A Primer*. McGraw-Hill Professionals.
3. Horstmann, C. S. (2024). *Core java, volume I: fundamentals*. Pearson Education.

Semester-II

Course Code : EL-5-PGC-202
Course Title : Computer Networks
Nature of the Course : Elective
Total Credit assigned : 5
Component of the Course : Lecture: 4 Practical: 1 Tutorial: 0
Distribution of Marks : End Semester: 56 Practical:20In Semester: 24

Course Objective:

CO1. Understand the structure and functioning of computer networks, including reference models and physical transmission media.
CO2. Explain wireless communication concepts, telephone systems, and error control mechanisms in data transmission.
CO3. Analyze data link layer protocols and medium access control methods in various networking scenarios.
CO4. Understand network layer design, including routing algorithms, congestion control, and IP addressing.
CO5. Demonstrate knowledge of transport layer protocols and basic network security principles.

Course Content:

Unit	Content	Marks	L	P	T
I	**Introduction to Computer Networks** • Overview of network hardware and software, Introduction to reference models: OSI and TCP/IP Models, Examples of different types of networks: Internet, ATM, Ethernet, and Wireless LANs, Understanding the physical layer and theoretical basis for data communication, Study of guided transmission media	10	12	4	0
II	**Wireless Transmission and Communication** • Introduction to wireless transmission and communication satellites, Telephone system structure including local loop, trunks, multiplexing, and switching, Design issues in the data link layer, Error detection and correction techniques	11	12	6	0
III	**Data Link Layer Protocols** • Elementary data link protocols, Sliding window protocols, Data link layer in the Internet, Medium access layer and channel allocation problem, Overview of multiple access protocols, including Bluetooth	12	12	6	0

IV	**Network Layer** • Design issues in the network layer, Routing algorithms and congestion control algorithms, Understanding the IP protocol and IP addresses, Introduction to Internet Control Protocols	11	12	8	0
V	**Transport Layer and Network Security** • Services provided by the transport layer, Connection management and addressing, Introduction to Simple Transport Protocol and Internet Transport Protocols (ITP), Overview of network security, focusing on cryptography techniques	12	12	6	0
Total		56	60	30	0

Mode of In-Semester assessment (24Marks):

A. Sessional Activity

B. Sessional Tests: (two Sessional Tests)

Sessional Activities:

1: Introduction to Computer Networks

1. **Group Discussion**: Compare and contrast OSI and TCP/IP models with diagram-based presentations.
2. **Activity**: Identify different types of networks in real life (Internet, ATM, Ethernet, Wireless LAN) through case studies.
3. **Quiz**: Quick quiz on layers of OSI and TCP/IP models and their functions.
4. **Hands-on Demo**: Display and identify network hardware like hubs, switches, routers, and cables.
5. **Assignment**: Write a brief report on guided media such as twisted pair, coaxial cable, and optical fiber.

2: Wireless Transmission and Communication

1. **Mini Seminar**: Student presentations on the role of satellites in wireless communication.
2. **Diagram Activity**: Draw the structure of a telephone system (local loop, trunk, switching).
3. **Lab Simulation**: Use network simulators (e.g., Cisco Packet Tracer) to visualize multiplexing and switching.
4. **Worksheet**: Solve problems based on error detection techniques (e.g., parity, CRC).
5. **Case Study**: Analyze how mobile phones use wireless transmission for communication.

3: Data Link Layer Protocols

1. **Group Activity**: Create a flowchart for Stop-and-Wait and Sliding Window Protocols.
2. **Simulation**: Implement and test a simple sliding window protocol using simulation tools.

3. **Video Analysis**: Watch and analyze videos explaining MAC protocols and Bluetooth operations.
4. **Role Play**: Demonstrate how medium access control is managed in shared channels.
5. **Assignment**: Write a report on data link layer functionalities in the Internet.

4: Network Layer

1. **Flowchart Exercise**: Visualize shortest path and distance vector routing algorithms.
2. **Class Demo**: Demonstrate IP addressing (IPv4 & IPv6) with subnetting examples.
3. **Worksheet**: Solve practice problems on routing tables and IP address classification.
4. **Quiz**: Quiz on Internet Control Protocols like ICMP and ARP.
5. **Case Study**: Analyze congestion control strategies in real-time network scenarios.

5: Transport Layer and Network Security

1. **Role Play**: Simulate connection establishment using 3-way handshake.
2. **Diagram Activity**: Draw and explain the functioning of Simple Transport Protocol.
3. **Hands-on Lab**: Demonstrate port addressing using network tools (e.g., Wireshark).
4. **Encryption Exercise**: Practice basic cryptography techniques (Caesar cipher, substitution).
5. **Discussion**: Explore common threats in networking and measures to secure data

Lab Work:

1. Configure Basic Network Settings:
   a) IP Address Configuration
   b) Subnet Mask and Gateway Settings
2. Implement Network Protocols:
   a) Write a simple Python script to perform DNS resolution.
   b) Implement a basic HTTP client-server application.
3. Network Simulation:
   a) Use network simulation tools (e.g.,Cisco Packet Tracer) to design and simulate network topologies.
   b) Configure routers and switches in a simulated environment.
4. Performance Measurement:
   a) Measure network performance using tools like`ping`, `traceroute`, and `iperf`.
   b) Analyze network traffic using Wireshark.
5. Implement VLANs:
   a) Configure VLANs on a switch and verify using simulation tools.
6. SetUp a Simple Web Server:
7. Deploy a basic web server and configure HTTP/HTTPS access

Expected Course Learning Outcomes:

By the end of the course, students will be able to:

CLO1: Describe network hardware/software components, OSI and TCP/IP models, and physical layer concepts such as guided media.
CLO2: Explain wireless transmission, telephone system structure, switching techniques, and apply error detection and correction methods.
CLO3: Implement and analyze data link layer protocols like sliding window and understand multiple access protocols including Bluetooth.
CLO4: Apply routing and congestion control algorithms, explain IP addressing, and understand network control protocols.
CLO5: Demonstrate the role of the transport layer in connection management and explain basic cryptographic techniques for secure communication.

Cognitive Map of Course Outcomes based on Revised Bloom's Taxonomy

Knowledge Dimension	Remember	Understand	Apply	Analyse	Evaluate	Create
Factual Knowledge	CO_1, CO_2, CO_5	CO_1, CO_2, CO_4, CO_5				
Conceptual Knowledge		CO_1, CO_2, CO_3, CO_4, CO_5	CO_3, CO_4, CO_5	CO_3, CO_4	CO_5	
Procedural Knowledge			CO_3, CO_4	CO_3, CO_4	CO_5	
Metacognitive Knowledge						

Mapping of PSO with CLO

CLO / PSO	PSO1	PSO2	PSO3	PSO4	PSO5	PSO6	PSO7	PSO8	PSO9
CLO1	3	2		1	1	2			1
CLO2	2	3		1	1	2			1
CLO3	2	3		2	1	3			2
CLO4	2	3		3	2	3			2
CLO5	2	2		2	2	3		3	2
Average Contribution	2.2	2.6		1.8	1.4	2.6		0.6	1.6

(3=High, 2= Moderate 1= Low)

Suggested readings:

1. Tanenbaum, A. S. (1996). Computer Networks, Prentice-Hall International. *Upper Saddle River, NJ*, *2*.
2. Stallings, W. (2007). *Data and computer communications*. Pearson Education India.
3. Forouzan, B. A. (2007). *Data communications and networking*. Huga Media.
4. Halsall, F. (1995). *Data communications, computer networks and open systems*. Addison Wesley Longman Publishing Co., Inc..