Anekant Education Society's

Tuljaram Chaturchand College, Baramati

(Autonomous)

Three Year B.Sc. Degree Program in Computer Science

(Faculty of Science & Technology)

CBCS Syllabus (2022 Pattern)

T.Y. B.Sc. (Computer Science) Sem- V

For Department of Computer Science

Tuljaram Chaturchand College of Arts, Science & Commerce, Baramati

Class: F.Y.B.Sc. (Computer Science)			
Semester I		**Semester II**	
UCSCO111	Basic Programming using C	UCSCO121	Advanced Programming using C
UCSCO112	DBMS-I	UCSCO122	DBMS-II
UCSCO113	Lab. Course I : Basic programming using C	UCSCO123	Lab. Course I : Advanced Programming using C
UCSCO114	Lab. Course II : DBMS I	UCSCO124	Lab. Course II : DBMS II
Physical Education			

Class :S.Y.B.Sc. (Computer Science) W.e.f. 2023-2024			
Semester III		**Semester IV**	
UCSCO231	Data Structure using C	UCSCO241	Object Oriented Concepts using Java
UCSCO232	Introduction to Web Technology	UCSCO242	Software Engineering Principles and Techniques
UCSCO233	Lab. Course I : based on UCSCO231	UCSCO243	Lab Course based on UCSCO241
UCSCO234	Lab. Course II : based on UCSCO232	UCSCO244	Lab Course based on UCSCO242 with Mini Project
Certificate Course I		Certificate Course II	
Environment Science (EVS) An Educational Trip conduct in IV semester			

Class: T.Y.B.Sc. (Computer Science) W.e.f. 2024-2025			
Semester V		**Semester VI**	
UCSCO351	System Programming &Operating System	UCSCO361	Advanced Operating System
UCSCO352	Theoretical Computer Science	UCSCO362	Compiler Construction
UCSCO353	Foundation of Computer Networking	UCSCO363	Computer Network & Network Security
UCSCO354	Basics of Web Development	UCSCO364	Advanced Web Development
UCSCO355	Advanced Programming in Java	UCSCO365	Advanced Java Technologies – Frameworks
UCSCO356	Object Oriented Software Engineering	UCSCO366	Software Metrics & Project Management
UCSCO357	Lab Course I: Based on UCSCO351	UCSCO367	Lab Course I: Based on UCSCO361
UCSCO358	Lab Course II: Based on UCSCO355	UCSCO368	Lab Course II: Based on UCSCO365 & Mini Project using JAVA
UCSCO359	Lab Course III: Based on UCSCO354	UCSCO369	Lab Course III: Based on UCSCO364 & Mini Project using PHP.
Certificate Course III		An Educational Trip conduct in this semester.	

T.Y.B.Sc.(Computer Science)

Semester- V

Credit Structure & Syllabus

(Academic Year 2024-2025, Autonomous)

Course Structure for T. Y. B. Sc. (Computer Science) Sem-V & VI
Subject: Computer Science

Sem	Paper Code	Title of Paper	No. of Credits	Exam	Marks
V	UCSCO351	System Programming &Operating System	3	I / E	60 + 40
	UCSCO352	Theoretical Computer Science	3	I / E	60 + 40
	UCSCO353	Foundation of Computer Networking	3	I / E	60 + 40
	UCSCO354	Basics of Web Development	3	I / E	60 + 40
	UCSCO355	Advanced Programming in Java	3	I / E	60 + 40
	UCSCO356	Object Oriented Software Engineering	3	I / E	60 + 40
	UCSCO357	Lab Course I: Based on UCSCO351	2	I / E	60 + 40
	UCSCO358	Lab Course II: Based on UCSCO355	2	I / E	60 + 40
	UCSCO359	Lab Course III: Based on UCSCO354	2	I / E	60 +40
		Certificate Course - III	2	---	----
VI	UCSCO361	Advanced Operating System	3	I / E	60 + 40
	UCSCO362	Compiler Construction	3	I / E	60 + 40
	UCSCO363	Higher layers of Computer Network & Network Security	3	I / E	60 + 40
	UCSCO364	Advanced Web Development	3	I / E	60 + 40
	UCSCO365	Advanced Java Technologies – Frameworks	3	I / E	60 + 40
	UCSCO366	Software Metrics & Project Management	3	I / E	60 + 40
	UCSCO367	Lab Course I: Based on UCSCO361	2	I / E	60 + 40
	UCSCO368	Lab Course II: Based on UCSCO365 & Mini Project using JAVA	2	I / E	60 + 40
	UCSCO369	Lab Course III: Based on UCSCO364 & Mini Project using PHP.	2	I / E	60 +40
	An Educational Trip conduct in this semester.				

SYLLABUS (CBCS) FOR T.Y.B. Sc. (Computer Science) (Semester- V)
(w.e.f from Academic Year 2024-2025)

Class: T.Y.B.Sc. (Computer Science) (Sem-V)	**Paper Code:** UCSCO351
Title of Paper: System Programming &Operating System	**Paper:** I
Credit: 3 (4 Lectures/Week)	**No. of lectures:** 48

Aim: To understand the design and implementation issues of System programs that play an important role in program development. And also to understand the design and implementation issues of Operating System.

Objectives:

- To understand the design structure of Assembler and macro processor for an hypothetical simulated computer.
- To understand the working of linkers and loaders.
- To understand Complexity of Operating system as a software. .
- To understand design issues related to process management and various related algorithms
- To understand design issues related to memory management and various related algorithms
- To understand design issues related to file management and various related algorithms.

Learning Outcome:

CO1: Learn the basic components of a computer system, including the CPU, memory, I/O devices and storage.
CO2: Introduce assembly language and provide hands-on experience in writing simple programs.
CO3: Able to understand the relationship between high-level languages and assembly language.
CO4: Able to understand the concept of process scheduling.
CO5: Learn system calls and their role in interacting with the operating system.
CO6: Study the role of the operating system in managing memory.
CO7: Learn concept of deadlock management.

Unit No.	Chapter name with Topics	No. of Lectures Required
1.	**Introduction to System Programming** 1.1. Types of program – System program and Application program. 1.2. Difference between system programming and application programming. 1.3. Elements of Programming environment - Editor, Preprocessor, Assembler, Compiler, Interpreter, Linker and Loader, Debugger, Device drivers, Operating System. 1.4. Simulation of simple computer smac0 (hypothetical computer) - Memory, Registers, Condition Codes, Instruction format, Instruction Set, smac0 programs.	**08**
2.	**Operating System as System Software** 2.1 What Operating Systems Do – User View, System View, Defining OS 2.2 Computer System Architecture – Single processor system, Multiprocessor systems, Clustered Systems 2.3 Operating System Operations – Dual mode operation, Timer	**06**

	2.4 Process Management 2.5 Memory Management 2.6 Storage Management – File system management, Mass storage management, Cashing, I/O systems 2.7 Protection and Security 2.8 Distributed Systems 2.9 Special Purpose System – Real time embedded systems, Multimedia systems, Handheld systems, 2.10 Computer Environment – Traditional computing, Client server computing, Peer to peer Computing	
3.	**System Structure** 3.1 Operating System Services 3.2 User Operating-System Interface – Command interpreter, GUI 3.3 System Calls 3.4 Types of System Calls – Process control, File management, Device management, Information maintenance, Communication, Protection	**02**
4.	**Process Management** 4.1 Process Concept – The process, Process states, Process control block. 4.2 Process Scheduling – Scheduling queues, Schedulers, context switch 4.3 Operations on Process – Process creation with program using fork(), Process termination 4.4 Inter-process Communication – Shared memory system, Message passing systems.	**05**
5.	**Process Scheduling** 5.1 Basic Concept – CPU-I/O burst cycle, CPU scheduler, Preemptive scheduling, Dispatcher 5.2 Scheduling Criteria 5.3 Scheduling Algorithms – FCFS, SJF, Priority scheduling, Round-robin scheduling, Multiple queue scheduling, Multilevel feedback queue scheduling. 5.4 Multithreaded Programming 5.5 Multithreading Models 6.6 Thread Scheduling	**10**
6.	**Multithreaded Programming** 6.1 Overview 6.2 Multithreading Model 6.3 Thread Libraries P-Tread, Java Thread 6.4 Thread Life Cycle	**06**
7.	**Process Synchronization** 7.1 Background 7.2 Critical Section Problem 7.3 Semaphores: Usage, Implementation 7.4 Classic Problems of Synchronization – The bounded buffer problem, The reader writer problem, The dining philosopher problem	**04**
8.	**Deadlocks** 8.1 System model 8.2 Deadlock Characterization – Necessary conditions, Resource allocation graph	**08**

		8.3 Deadlock Prevention 8.4 Deadlock Avoidance - Safe state, Resource allocation graph algorithm, Banker’s Algorithm 8.5 Deadlock Detection 8.6 Recovery from Deadlock – Process termination, Resource preemption	

Reference Books:

1. Siberchatz, Galvin, Gagne Operating System Concepts (8th Edition).

2. Pabitra Pal Choudhary Operating Systems: Principles and Design (PHI Learning Private Limited)

Mapping of Course outcomes with Program Outcomes

Course Outcomes	Program Outcomes						
	PO1	PO2	PO3	PO4	PO5	PO6	PO7
CO1	3	3	2	1	1	2	2
CO2	3	2	1	1	1	1	1
CO3	2	2	1	1	1	1	1
CO4	3	2	2	2	2	3	2
CO5	3	3	3	2	3	2	3
CO6	3	3	3	2	1	2	2
CO7	2	1	1	1	1	1	1

PO1 with All COs
CO1: PO1: 3 (Strongly Agree): Understanding computer components is foundational for applying mathematics, statistics, and computer fundamentals in IT applications.
CO2: PO1: 3 (Strongly Agree) : Assembly language knowledge and hands-on experience enhance computer knowledge for effective IT application development.
CO3: PO1: 2 (Moderately Agree): Understanding the relationship is important, but the moderate weightage reflects its significance relative to other objectives.
CO4: PO1: 3 (Strongly Agree): Process scheduling is a critical aspect of computer systems and contributes significantly to IT application performance.
CO5: PO1: 3 (Strongly Agree): Knowledge of system calls is essential for effective interaction with the operating system in IT applications.
CO6: PO1: 3 (Strongly Agree): Understanding memory management is crucial for optimizing system performance in IT applications.
CO7: PO1: 2 (Moderately Agree): Deadlock management is important but has a moderate weightage compared to other objectives.
PO2 with All COs
CO1: PO2: 3 (Strongly Agree): Understanding computer components is fundamental to designing and developing solutions for IT applications using the latest technologies.
CO2: PO2: 2 (Moderately Agree): Assembly language knowledge is relevant but may not be the primary focus in the context of designing solutions with the latest technologies.

CO3: PO2: 2 (Moderately Agree): Understanding language relationships contributes to solution design but may not have the same weight as other objectives.
CO4: PO2: 2 (Moderately Agree): Process scheduling knowledge is important but may have a moderate impact on designing solutions using the latest technologies.
CO5 PO2: 3 (Strongly Agree): System calls are crucial for effective interaction with the operating system, aligning with the design and development of solutions.
CO6: PO2: 3 (Strongly Agree): Understanding memory management is essential for designing efficient solutions using the latest technologies.
CO7: PO2: 1 (Partially Agree): Deadlock management, while important, may have a lower weightage compared to other objectives in the context of designing solutions with the latest technologies.
PO3 with All COs
CO1: PO3: 2 (Moderately Related): Understanding computer components contributes foundational knowledge but may be moderately related to the modern tool usage aspect of applying techniques and resources.
CO2: PO3: 1 (Partially Related) : Assembly language exposure is less directly related to modern tool usage for complex IT applications but may offer some foundational insights.
CO3: PO3: 1 (Partially Related): Understanding language relationships provides a foundational understanding but may have limited direct impact on the application of modern tools.
CO4: PO3: 2 (Moderately Related): Process scheduling knowledge is relevant for understanding system behavior, contributing moderately to the application of modern tools.
CO5: PO3: 3 (Strongly Related): System calls are directly related to the interaction with the operating system, aligning strongly with the application of modern engineering and IT tools.
CO6: PO3: 3 (Strongly Related): Understanding memory management is crucial for optimizing system performance, strongly contributing to the application of modern tools.
CO7: PO3: 1 (Partially Related): Deadlock management, while important, may have limited direct relevance to the broader application of modern tools in complex IT applications.

PO4 with All COs
CO1: PO4: 1 (Partially Related): Understanding computer components provides foundational knowledge but has limited direct relevance to the broader context of environmental sustainability.
CO2: PO4: 1 (Partially Related): Assembly language exposure is less directly related to environmental sustainability considerations.
CO3: PO4: 1 (Partially Related): Understanding language relationships is foundational but may have limited direct impact on environmental and sustainability considerations.
CO4: PO4: 2 (Moderately Related): Process scheduling knowledge may have some moderate relevance to resource utilization and efficiency considerations related to environmental sustainability.
CO5: PO4: 2 (Moderately Related): System calls may be moderately related, as they impact system efficiency and resource utilization, which can have environmental implications.
CO6: PO4: 2 (Moderately Related): Understanding memory management may have some moderate relevance to environmental considerations related to system efficiency.
CO7: PO4: 1 (Partially Related): Deadlock management, while important for system stability, may have limited direct impact on environmental and sustainability aspects.
PO5 with All COs
CO1:PO5: 1 (Partially Related): Understanding computer components provides foundational knowledge, but its direct impact on ethical principles and professional responsibilities may be limited.
CO2: PO5: 1 (Partially Related) : Assembly language exposure is less directly related to ethical principles and professional responsibilities.

CO3: PO5: 1 (Partially Related) : Understanding language relationships is foundational but may have limited direct impact on ethical considerations in engineering practice.
CO4: PO5: 2 (Moderately Related): Process scheduling knowledge may have some moderate relevance to ethical considerations related to system efficiency and fairness.
CO5: PO5: 3 (Strongly Related): System calls are strongly related as they impact the interaction with the operating system, which may have ethical implications.
CO6: PO5: 2 (Moderately Related): Understanding memory management may have moderate relevance to ethical considerations related to resource allocation and efficiency.
CO7: PO5: 1 (Partially Related): Deadlock management, while crucial for system stability, may have limited direct impact on ethical principles and professional responsibilities.

PO6 with All COs
CO1: PO6: 2 (Moderately Related): Understanding computer components contributes foundational knowledge, moderately supporting individual and team effectiveness in diverse settings.
CO2:PO6: 1 (Partially Related): Assembly language exposure is less directly related to individual and team work effectiveness.
CO3:PO6: 1 (Partially Related): Understanding language relationships is foundational but may have limited direct impact on individual and team effectiveness.
CO4: PO6: 3 (Strongly Related): Process scheduling knowledge is strongly related as it impacts system efficiency and fairness, influencing individual and team performance.
CO5: PO6: 2 (Moderately Related): System calls moderately contribute to understanding operating system interactions, influencing teamwork in an IT setting.
CO6: PO6: 2 (Moderately Related): Understanding memory management moderately supports teamwork by contributing to system efficiency and collaborative problem-solving.
CO7: PO6: 1 (Partially Related) : Deadlock management, while important for system stability, may have limited direct impact on individual and team work effectiveness.

PO7 with All COs
CO1 PO7: 2 (Moderately Related): Understanding computer components contributes foundational knowledge, moderately supporting innovation and employability skills in the field.
CO2: PO7: 1 (Partially Related): Assembly language exposure is less directly related to innovation and entrepreneurial skills but may offer foundational insights.
CO3: PO7: 1 (Partially Related): Understanding language relationships is foundational but may have limited direct impact on innovation and entrepreneurial skills.
CO4:PO7: 2 (Moderately Related): Process scheduling knowledge may have moderate relevance to innovation and employability skills, particularly in optimizing system performance.
CO5: PO7: 3 (Strongly Related): System calls are strongly related as they impact operating system interactions, playing a crucial role in innovation and employability skills.
CO6: PO7: 2 (Moderately Related): Understanding memory management moderately contributes to innovation and employability skills, particularly in system efficiency.
CO7: PO7: 1 (Partially Related): Deadlock management, while important for system stability, may have limited direct impact on innovation and entrepreneurial skills.

SYLLABUS (CBCS) FOR T. Y. B. Sc. (Computer Science) Sem-V (w.e.f Academic Year 2024-2025)

Class **:**T.Y. B. Sc.(Computer Science) (Semester- V) **Paper Code** **:** UCSCO352
Subject: Theoretical Computer Science **Paper:** II
Credit **:**3(4 Lectures/week) **No. of lectures :**48

Prerequisite:
• Sets, Operations on sets, Finite & infinite sets Formal Language
• Relation, Equivalence Relation, (reflexive, transitive and symmetric closures)

Learning Objectives: Students successfully completing this course will be able:
• To have an understanding of finite state and pushdown automata.
• To have a knowledge of regular languages and context free languages.
• To know the relation between regular language, context free language and corresponding recognizers.
• To study the Turing machine and classes of problems.

Learning Outcome: Course Outcome:
CO1: Knowledge of automata, formal language theory and computability
CO2: Demonstrate advanced knowledge of formal computation and its relationship to languages.
CO3: Distinguish different computing languages and classify their respective types.
CO4: Recognize and comprehend formal reasoning about languages.
CO5: Show a competent understanding of the basic concepts of complexity theory.
CO6: The students will be able to design.
CO7: To know basic models of information processing.

Units	Topic Contents	No. of Lectures
Unit -I	**Finite Automata** 1.1 Deterministic finite Automaton – Definition, DFA as Language recognizer, DFA as a pattern recognizer. 2.2 Nondeterministic finite automaton – Definition and Ex.. 2.3 NFA TO DFA 2.4 NFA with ε- transitions Definition and Examples. 2.5 NFA with ε-Transitions to DFA & Examples 2.6 Finite automaton with output – Mealy and Moore machine, Definition and Examples. 2.7 Minimization of DFA, Algorithm & Problem using Table Method.	**15**
Unit –II	**Regular Languages** 3.1 Regular language-Definition and Examples. 3.2 Conversion of RE To FA-Examples. 3.3 Pumping lemma for regular languages and applications. 3.4 Closure properties of regular Languages (Union, Concatenation, Complement, Intersection and Kleeneclosure)	**5**
Unit – III	**Context Free Grammar and Languages** 4.1 Grammar - Definition and Examples. 4.2 Derivation-Reduction - Definition and Examples. 4.3 Chomsky Hierarchy. 4.4 CFG: Definition & Examples. LMD, RMD, ,Parse Tree	

	4.5 Ambiguous Grammar: Concept & Examples. 4.6 Simplification of CFG : 4.6.1 Removing Useless Symbols, 4.6.2 Removing unit productions 4.6.3 Removing ϵ productions & Nullable symbols 4.7 Normal Forms : 4.7.1 Chomsky Normal Form (CNF) Method & Problem 4.7.2 Greibach Normal form (GNF) Method & Problem 4.8 Regular Grammar: Definition. 4.8.1 Left linear and Right Linear Grammar-Definition and Example. 4.8.2 Equivalence of FA & Regular Grammar 4.8.2.1 Construction of regular grammar equivalent to a given DFA 4.8.2.2 Construction of a FA from the given right linear grammar 4.9 Closure Properties of CFL's(Union, concatenation and Kleen closure) Method and examples	**12**
Unit- IV	**Push Down Automaton** 5.1 Definition of PDA and examples 5.2 Construction of PDA using empty stack and final State method : Examples using stack method 5.3 Definition DPDA & NPDA, their correlation and Examples of NPDA 5.4 CFG (in GNF) to PDA : Method and examples	**6**
Unit – V	**Turing Machine** 6.1 The Turing Machine Model and Definition of TM 6.2 Design of Turing Machines 6.3 Problems on language recognizers. 6.4 Language accepted by TM 6.5 Recursive Languages 6.5.1. Recursive and Recursively enumerable Languages. 6.5.2. Difference between recursive and recursively enumerable language. 6.6 Turing Machine Limitations 6.7 Decision Problem, Undecidable Problem, Halting Problem of TM	**10**

References:-

1. Introduction to Automata theory, Languages and computation By John E. Hopcroft and Jeffrey Ullman – Narosa Publishing House.
2. Introduction to Automata theory, Languages and computation By John Hopcroft, Rajeev Motwani and Jeffrey Ullman –Third edition Pearson Education
3. Introduction to Computer Theory Daniel I. A. Cohen – 2nd edition – John Wiley & Sons
4. Theory of Computer Science (Automata, Language & Computation) K. L. P. Mishra & N. Chandrasekaran, PHI Second Edition

5. Introduction to Languages and The Theory of Computation John C. Martin TMH, Second Edition

Mapping of this course with Programme Outcomes & it's justification

Course Outcomes	Programme Outcomes (POs)						
	PO1	PO2	PO3	PO4	PO5	PO6	PO7
CO1	3	3	3	2	2	3	3
CO2	3	3	2	3	2	3	3
CO3	3	3	2	2	2	3	3
CO4	3	3	2	2	2	3	3
CO5	3	3	2	2	2	3	3
CO6	3	3	3	3	3	3	3
CO7	3	3	3	2	2	3	3

Weight: 1 - Partially related 2 - Moderately Related 3 - Strongly related

CO1:PO1: Strongly Related (3) - As understanding automata and formal language theory is fundamental to computer science applications.
CO2: PO1: Strongly Related (3) - Strongly Related, as advanced knowledge of formal computation is essential for understanding the relationship between computation and languages.
CO3:PO1: Strongly Related (3) -As the ability to distinguish and classify computing languages is a fundamental skill in computer science applications.
CO4:PO1: Strongly Related (3) - As formal reasoning about languages is crucial for effective problem-solving in computer science.
CO5:PO1: Strongly Related (3) - As a competent understanding of complexity theory is essential for addressing the efficiency of algorithms in various applications.
CO6:PO1: Strongly Related (3) - As the ability to design is a fundamental skill in computer science applications.
CO7:PO1: Strongly Related (3) - As knowledge of basic models of information processing is crucial for understanding and implementing various computer science applications.

CO1:PO2: Strongly Related (3) - As understanding these concepts is essential for designing effective computational solutions.
CO2:PO2: Strongly Related (3) - As advanced knowledge in formal computation enhances the ability to design sophisticated solutions.
CO3:PO2: Strongly Related (3) - As the ability to distinguish and classify languages aids in making informed decisions during the design process.
CO4:PO2: Strongly Related (3) - As formal reasoning skills are crucial for creating well-structured and logically sound computational solutions.
CO5:PO2: Strongly Related (3) - As a competent understanding of complexity theory is necessary for designing efficient solutions to computational problems.
CO6:PO2: Strongly Related (3) - As designing solutions is a central focus of this program objective.
CO7:PO2: Strongly Related (3) - As understanding basic models of information processing is foundational to designing effective computational solutions.

CO1:PO3: Strongly Related (3) - As these concepts are fundamental basics in the discipline of computer science.
CO2:PO3: Moderately Related (2) - As advanced knowledge may be more specialized but still contributes to the understanding of the discipline.

CO3:PO3: Moderately Related (2) - As understanding language types contributes to the basics of the discipline.
CO4:PO3: Moderately Related (2) - As formal reasoning contributes to a deeper understanding of the discipline.
CO5:PO3: Moderately Related (2) - As understanding complexity theory is a more specialized aspect but still contributes to the basics of the discipline.
CO6:PO3: Strongly Related (3) - As the ability to design is a basic and essential skill in the discipline.
CO7:PO3: Strongly Related (3) - As understanding basic models of information processing is fundamental to the basics of the discipline.

CO1:PO4: Moderately Related (2) - As a foundational understanding of automata and formal language theory contributes to professional development.
CO2:PO4: Strongly Related (3) - As advanced knowledge in formal computation is crucial for staying updated and advancing in the field.
CO3:PO4: Moderately Related (2) - As knowledge of different languages contributes to versatility in professional development.
CO4:PO4: Moderately Related (2) - As formal reasoning skills enhance problem-solving abilities in professional development.
CO5:PO4: Moderately Related (2) - As a competent understanding of complexity theory is valuable in approaching complex problems in professional development.
CO6:PO4: Strongly Related (3) - As the ability to design solutions is a key skill for continued professional development.
CO7:PO4: Moderately Related (2) - As understanding basic models of information processing provides a foundational knowledge for professional development in various computing domains.

CO1:PO5: Moderately Related (2) - As understanding foundational concepts contributes to the ability to analyze and address societal and environmental impacts.
CO2:PO5: Moderately Related (2) - As advanced knowledge in formal computation provides a basis for understanding the societal and environmental implications of IT solutions.
CO3:PO5: Moderately Related (2) - As knowledge of computing languages contributes to the broader understanding of IT solutions in different contexts.
CO4:PO5: Moderately Related (2) - As formal reasoning skills aid in understanding and addressing the societal and environmental aspects of IT solutions.
CO5:PO5: Moderately Related (2) - As a competent understanding of complexity theory contributes to the ability to assess the impact of IT solutions in various contexts.
CO6:PO5: Strongly Related (3) - As the design process is crucial for considering and implementing sustainable practices in IT solutions.
CO7:PO5: Moderately Related (2) - As knowledge of basic models of information processing contributes to understanding the societal and environmental implications of IT solutions.

CO1:PO6: Strongly Related (3) - As proficiency in computing requires a strong foundation in automata and formal language theory.
CO2:PO6: Strongly Related (3) - As advanced knowledge in formal computation is crucial for achieving proficiency in computing.
CO3:PO6: Strongly Related (3) - As proficiency in computing involves a thorough understanding of various computing languages.
CO4:PO6: Strongly Related (3) - As formal reasoning skills are integral to developing proficiency in computing.

CO5:PO6: Strongly Related (3) - As a competent understanding of complexity theory is essential for proficient problem-solving in computing.
CO6:PO6: Strongly Related (3) - As the ability to design is a key component of proficiency in computing.
CO7:PO6: Strongly Related (3) - As knowledge of basic models of information processing is foundational to developing proficiency in computing.

CO1:PO7: Strongly Related (3) - As independent study and research in these areas are crucial for success in hardware/software companies.
CO2:PO7: Strongly Related (3) - As advanced knowledge in formal computation enhances the capacity for independent study and research, preparing for employment in the industry.
CO3:PO7: Strongly Related (3) - As proficiency in various computing languages is essential for transitioning to employment in hardware/software companies.
CO4:PO7: Strongly Related (3) - As formal reasoning skills contribute to the ability to independently analyze and solve problems encountered in the industry.
CO5:PO7: Strongly Related (3) - As a competent understanding of complexity theory is beneficial for addressing complex challenges in hardware/software development independently.
CO6:PO7: Strongly Related (3) - As the ability to design is a key skill required for success in hardware/software companies, and developing this skill independently is crucial.
CO7:PO7: Strongly Related (3) - As understanding basic models of information processing is foundational for independent research and study in the field, aiding in the transition to employment.

SYLLABUS (CBCS) FOR T.Y.B.Sc. (Computer Science) (SEM-V)
(w.e.f. A.Y.- 2024-2025)

Class: T.Y.B.Sc. (Computer Science)(Semester-V) **Paper Code:**UCSCO353
Title of paper: Foundations of Computer Networking **Paper :** III
Credit -3(4 Lect./Week) **No. of Lectures:** 48

Pre-requisites: Basics knowledge of computer

Objectives: This course will prepare students in Basic networking concepts.
1. Understand different types of networks, various topologies and application of networks.
2. Understand types of addresses, data communication.
3. Understand the concept of networking models, protocols, functionality of each layer.
4. Learn basic networking hardware and tools.
5. Understand wired and wireless networks, its types, functionality of layer.

Learning Outcomes:
CO1: Understanding Networking Concepts - Define and explain fundamental networking concepts, including protocols, data communication and network architectures.
CO2: Network Models - Understand and apply knowledge of OSI, TCP/IP Models.
CO3: Network Protocols – Describe and analyze various networking protocols and their Functionalities.
C04: Network Design and implementation - Design and implement LAN, WAN & MAN.
CO5: Networking devices and Technologies - Evaluate and select appropriate networking devices and technologies for specific scenarios.
CO6: Wired & Wireless Networking: Understand principles, understand design and implementationof wired & wireless communication.
CO7: Internet Technologies: Understand the functioning of the internet and related technologies.

Units No.	Title & Contents	No. of Lectures
I	**Introduction to Computer Network** Computer Networks- Goals, applications Network Hardware's – Broadcast and point to point. Topology – Star, Bus, Mesh, Ring etc. Network Types : LAN, MAN, WAN, Wireless Network, internetwork Data Communication – Definition, Components, data representation, Data flow. , Protocols and Standards Defacto, Dejure standard Network Software- Protocol Hierarchies, Design issues of the layer, Connection and connectionless services,	08
II	**Network Models** Reference Model – OSI Reference Model, TCP/IP Reference Model, Comparison of OSI & TCP/IP Model, Addressing – Physical, Logical and Port addresses	04
III	**Transmission Media**	04

	Guided Media – Twisted pair cable, Coaxial Cable, Fiber optic cable Unguided Media – Radio Waves, Micro wave Transmission, Infrared, Light wave Transmission	
IV	**Lower layers : Physical and Data link layers** Communication at the physical layer, Data and signals. Transmission Impairment, Data rate limits, Performance Transmission Modes. Switching – Circuit, Message and Packet Switching. Design issues of Data Link Layer, Services – Framing, Error control, Flow Control, Congestion Control, Link layer addressing. Data link Protocols – simplex, stop and wait and stop and wait Automatic Repeat Request (ARQ). Sliding Window Protocols – One-bit sliding window protocol, Pipeline technique, Go back N and Selective Repeat Automatic Repeat Request with comparison. DLL Protocols – HDLC, PPP Physical and Data link layer devices – Repeater, Hubs, Bridge	18
V	**The Medium Access Sub layer** Introduction., Random Access Protocols – ALOHA – Pure & Slotted CSMA – 1 Persistent, P-persistent and non-persistent CSMA/CD, CSMA/CA. ,Controlled Access – Reservation, Polling and Token Passing, Channelization – FDMA, TDMA, CDMA	07
VI	**Wired and Wireless LAN** IEEE Standards, changes in the standard – bridged Ethernet, switched Ethernet, full duplex Ethernet. Fast Ethernet, Gigabit Ethernet, Ten-Gigabit Ethernet: Goals, MAC Sublayer, Topology and Implementation. Backbone Network – Bus backbone, Star backbone, Remote LANs Virtual LANs: Membership, configuration, communication, Advantages. Wireless LAN - IEEE 802.11 Architecture – BSS, ESS, Station Types, Bluetooth Architecture – Piconet, Scatternet	07

Reference Books:

1) Computer Networks by Andrew Tanenbaum, Pearson Education.[4th Edition]

2) Data Communication and Networking by Behrouz Forouzan,

TATA McGraw Hill.[4th/5th Ed.]

3) Networking All In One Dummies Wiley Publication.[5th Edition]

Mapping of this course with Programme Outcomes

Course Outcomes	Programme Outcomes (POs)						
	PO1	PO2	PO3	PO4	PO5	PO6	PO7
CO1	2	2	2	1	2	2	1
CO2	2	3	2	1	2	2	1
CO3	2	3	2	1	2	2	1
CO4	1	3	3	2	2	3	2
CO5	2	3	3	2	2	3	2
CO6	2	3	3	2	2	3	2
CO7	2	3	3	2	2	3	2

Weight: 1 - Partially related 2 - Moderately Related 3 - Strongly related

Justification of above mapping:

Mapping C01 with all POs:
C01- PO1: Understanding networking concepts requires the application of mathematical and statistical knowledge along with computer fundamentals.
C01 - PO2: Designing solutions for IT application often involves incorporating networking concepts, protocols and architectures.
C01 - PO3: Networking concepts involve the use of modern engineering and IT tools for analysis and design.
CO1 – PO4: Considering the impact of networking solutions in societal and environmental contexts align with understanding networking concepts.
CO1 – PO5: Applying ethical principles in the development and implementation of networking solutions is crucial.
CO1 – PO6: Understanding networking concepts contributes to both individual work and collaborative efforts within a team.
CO1 – PO7: Identifying networking opportunities, pursuing them, and creating value align with entrepreneurial skills.

Mapping C02 with all POs:
C02- PO1: Understanding network models requires a foundation in mathematics, statistics, and computer fundamentals.
C02 - PO2: Applying knowledge of OSI and TCP/IP models is essential in designing and developing IT solutions.
C02 - PO3: Understanding network models involves using modern engineering and IT tools for analysis and design.
CO2 – PO4: Understanding network models contributes to considering the impact of IT solutions in societal and environmental contexts.
CO2 – PO5: Applying knowledge of network models aligns with ethical principles in IT solution development.
CO2 – PO6: Understanding network models contributes to both individual work and collaborative efforts within a team.
CO2 – PO7: Applying network models is relevant to identifying opportunities and pursuing them for innovation and employability.

Mapping C03 with all POs:
C03- PO1: Describing and analysing networking protocols requires a foundation in mathematics, statistics, and computer fundamentals.
C03 - PO2: Understanding and analysing networking protocols is crucial in designing and developing IT solutions.
C03 - PO3: Describing and analysing networking protocols involves using modern engineering and IT tools for analysis and design.
CO3 – PO4:Understanding network protocols contributes to considering the impact of IT solutions in societal and environmental contexts.
CO3 – PO5:Describing and analysing networking protocols aligns with ethical principles in IT solution development.
CO3 – PO6:Describing and analysing networking protocols contributes to both individual work and collaborative efforts within a team.
CO3 – PO7:Understanding network protocols is relevant to identifying opportunities and pursuing them for innovation and employability.

Mapping C04 with all POs:
C04- PO1: Designing and implementing network solutions requires the application of mathematics, statistics, and computer fundamentals.
C04 - PO2: Designing and implementing LAN, WAN, and MAN involve the design and development of IT solutions.
C04 - PO3: Designing and implementing networks involves using modern engineering and IT tools for analysis and design.
CO4 – PO4:Considering the impact of network solutions in societal and environmental contexts aligns with designing and implementing LAN, WAN, and MAN.
CO4 – PO5:Designing and implementing network solutions aligns with ethical principles in IT solution development.
CO4 – PO6:Designing and implementing network solutions contributes to both individual work and collaborative efforts within a team.
CO4 – PO7:Identifying opportunities and implementing network solutions for innovation and employability aligns with network design and implementation.

Mapping C05 with all POs:
C05- PO1: Evaluating and selecting networking devices and technologies requires knowledge in mathematics, statistics, and computer fundamentals.
C05 - PO2: Designing solutions for IT applications involves the evaluation and selection of appropriate networking devices and technologies.
C05 - PO3: Evaluating and selecting networking devices and technologies involves using modern engineering and IT tools for analysis and design.
CO5 – PO4: Considering the impact of selected networking devices and technologies in societal and environmental contexts aligns with sustainable development.
CO5 – PO5: Evaluating and selecting networking devices and technologies aligns with ethical principles in IT solution development.
CO5 – PO6: Evaluating and selecting networking devices and technologies contributes to both individual work and collaborative efforts within a team.
CO5 – PO7: Identifying opportunities and selecting appropriate networking devices and technologies aligns with innovation and employability.

Mapping C06 with all POs:
C06- PO1: Understanding principles, design, and implementation of wired and wireless communication requires knowledge in mathematics, statistics, and computer fundamentals.
C06 - PO2: Designing solutions for IT applications involves understanding the principles and implementation of both wired and wireless communication.
C06 - PO3: Understanding and implementing wired and wireless communication involves using modern engineering and IT tools for analysis and design.
CO6 – PO4: Considering the impact of wired and wireless communication solutions in societal and environmental contexts aligns with sustainable development.
CO6 – PO5: Understanding and implementing wired and wireless communication aligns with ethical principles in IT solution development.
CO6 – PO6: Understanding and implementing wired and wireless communication contributes to both individual work and collaborative efforts within a team.
CO6 – PO7: Identifying opportunities and implementing wired and wireless communication aligns with innovation and employability.

Mapping C07 with all POs:
C07- PO1: Understanding internet technologies requires the application of mathematics, statistics, and computer fundamentals.
C07 - PO2: Designing solutions for IT applications involves understanding and utilizing internet technologies.
C07 - PO3: Understanding the functioning of the internet and related technologies involves using modern engineering and IT tools for analysis and design.
CO7 – PO4: Considering the impact of internet technologies in societal and environmental contexts aligns with sustainable development.
CO7 – PO5: Understanding internet technologies aligns with ethical principles in IT solution development and usage.
CO7 – PO6: Understanding internet technologies contributes to both individual work and collaborative efforts within a team.
CO7 – PO7: Identifying opportunities and utilizing internet technologies aligns with innovation and employability.

SYLLABUS (CBCS) FOR T.Y.B.Sc. (Computer Science) (Semester-V)
(w.e.f. from Academic Year 2024-2025)

Class: T.Y.B.Sc. (Computer Science) (Sem-V) **Paper Code:** UCSCO354
Title of Paper: Basics of Web Development **Paper:** IV
Credits: 03 (4 Lectures/Week) **No. of lectures:** 48

Prerequisite: Know HTML Programming

Objectives:

- To design dynamic, interactive web pages.
- To learn the server side scripting language.
- To learn database connectivity with PHP.

Course Outcomes:

CO1.On completion of the course, student will be able to understand how to develop dynamic and interactive web pages.
CO2.Evaluate common errors in the web languages and repair them to meet standards.
CO3.Distinguish between personalized and dynamic web pages and how servers and web languages can be used for different website needs.
CO4.Distinguish between objective and subjective analysis of a website and conduct both analyses for website designs.
CO5.Distinguish between personalized and dynamic web pages and how servers and web languages can be used for different website needs.
CO6.Distinguish between objective and subjective analysis.
CO7.Design and produce a completed website for a specified client.

Chapter No.	**Chapter name with Topics**	**No. of Lectures Required**
1.	**Introduction to PHP** 1.1 HTTP basics, Web Server, Web Browser 1.2 Introduction to PHP(Why PHP?) 1.3 What does PHP do? 1.4 Lexical structure 1.5 Language basics	**04**
2.	**Function and String** 2.1 Defining and calling a function 2.2 Default parameters 2.3 Variable parameters, Missing parameters 2.4 Variable function, Anonymous function 2.5 Types of strings in PHP 2.6 Printing functions 2.7 Encoding and escaping 2.8 Comparing strings 2.9 Manipulating and searching strings 2.10 Regular Expressions	**08**

3.	**Arrays** 3.1 Indexed Vs Associative arrays 3.2 Identifying elements of an array 3.3 Storing data in arrays 3.4 Multidimensional arrays 3.5 Extracting multiple values 3.6 Converting between arrays and variables 3.7 Traversing arrays 3.8 Sorting 3.9 Action on entire arrays 3.10 Using arrays	**06**
4.	**Introduction to Object Oriented Programming** 4.1 Classes and Objects 4.2 Inheritance 4.3 Interfaces 4.4 Encapsulation 4.5 Traits 4.6 Autoloading classes 4.7 Exception handling 4.8 Predefined exceptions 4.9 Namespaces in OOP in PHP 4.10 Predefined PHP classes and interfaces	**16**
5.	**Databases (PHP-PostgreSQL)** 5.1 Introduction to PDO 5.2 Installing PDO 5.3 Predefined constants 5.4 Supported databases 5.5 The PDO class 5.6 PDO class methods 5.7 Security using PDO 5.8 PDOStatement class 5.9 Create, Read, Update and Delete (CRUD) operations	**14**

References:

1. Kevin Tatroe, Peter MacIntyre (2020), Programming PHP : Creating Dynamic Web Pages(4th ed.). O’Reilly.

Web References :

1. https://www.php.net/manual/en/manual.php
2. https://www.php-fig.org/
3. https://phptherightway.com
4. https://w3schools.com

Mapping of this course with Program Outcomes

	PO1	PO2	PO3	PO4	PO5	PO6	PO7
CO1	3	3	3	2	1	3	2
CO2	3	3	3	2	1	3	2
CO3	3	2	3	1	1	3	2
CO4	3	2	3	1	1	2	2
CO5	3	2	3	1	1	3	2
CO6	3	1	3	1	1	2	2
CO7	3	3	3	3	1	3	3

Weight: 1 - Partially related 2 - Moderately Related 3 - Strongly related

Justification of Mapping

Mapping of PO1 to all Cos

PO1: CO1: Developing dynamic and interactive web pages involves the application of fundamental principles and methods of Computer Science. This directly aligns with the overarching goal of applying computer science principles to various applications.
CO2: Identifying and rectifying errors in web languages require a deep understanding of fundamental principles in Computer Science. This aligns with the broader objective of applying computer science principles to address challenges in web development.
CO3: Understanding the distinction between personalized and dynamic web pages and utilizing servers and web languages for diverse website requirements directly involves the application of fundamental principles of Computer Science in web development.
CO4: Conducting objective and subjective analyses of a website requires a comprehensive understanding of fundamental principles in Computer Science, involving both technical and user experience considerations in web design.
CO5: Reiteration of CO3, emphasizing the continued application of fundamental principles in Computer Science to design and implement personalized and dynamic web pages based on server-side interactions.
CO6: The ability to distinguish between objective and subjective analysis is an essential skill in Computer Science, ensuring that evaluations are both technically sound and aligned with user experiences.
CO7: Designing and producing a completed website for a client requires the practical application of fundamental principles and methods of Computer Science, encompassing the entire web development process.

Mapping of PO2 to all Cos

PO2:CO1: Developing dynamic and interactive web pages involves solving computational problems related to user interface design, functionality, and data manipulation, aligning strongly with the ability to design and implement solutions to computational challenges.
CO2: Identifying and correcting errors in web languages is a practical application of problem-solving skills, a key aspect of designing and implementing effective solutions to computational problems.
CO3: While understanding the distinction between personalized and dynamic web pages is crucial for effective web development, the direct link to solving significant computational problems is slightly less explicit.
CO4: Conducting analyses for website designs contributes to the broader understanding of web development but is not directly tied to solving significant computational problems.
CO5: Similar to CO3, the understanding of personalized and dynamic web pages is relevant to web development but is not explicitly tied to solving significant computational problems.
CO6: Distinguishing between objective and subjective analysis is valuable but is not directly linked to solving significant computational problems, making the relationship partial.
CO7: Designing and producing a website for a specified client involves solving a multitude of computational problems related to functionality, user experience, and data management, aligning strongly with the ability to design, implement, and document solutions to computational challenges.

Mapping of PO3 to all COs

PO3: CO1: Understanding how to develop dynamic and interactive web pages is a fundamental aspect of the discipline, and this CO directly contributes to imparting knowledge of the basics of web development.
CO2: Evaluating and rectifying common errors in web languages is foundational knowledge in the discipline of web development, aligning with the basics of ensuring code quality and adherence to standards.
CO3: The ability to distinguish between personalized and dynamic web pages and understanding the utilization of servers and web languages for diverse needs is fundamental to the basics of web development.
CO4: Distinguishing between objective and subjective analysis and applying both to website designs reflects a foundational understanding of the basics of evaluating websites from both technical and user-oriented perspectives.
CO5: Reiteration of CO3, emphasizing the foundational knowledge required to understand the distinctions between personalized and dynamic web pages and their application to meet different website needs.
CO6: The ability to distinguish between objective and subjective analysis is a basic skill essential for a well-rounded understanding of evaluating websites, contributing to the basics of the discipline.
CO7: Designing and producing a completed website for a specified client involves applying fundamental knowledge of web development, contributing to the basics of the discipline.

Mapping of PO4 to all Cos

PO4: CO1: Understanding how to develop dynamic and interactive web pages contributes to foundational knowledge that can be applied in a professional context, but the direct link to preparing for continued professional development is moderate.
CO2: Evaluating and correcting errors in web languages is a practical skill that can contribute to professional development but is not explicitly focused on preparing for continued professional development.

CO3: While understanding the distinctions in web pages and server usage is relevant to web development, the direct link to preparing for continued professional development is not as strong.
CO4: Distinguishing between objective and subjective analysis is a valuable skill, but the direct connection to preparing for continued professional development is partial.
CO5: Similar to CO3, the understanding of personalized and dynamic web pages is relevant to web development but has a partial connection to preparing for continued professional development.
CO6: Distinguishing between objective and subjective analysis is a valuable skill, but the direct connection to preparing for continued professional development is partial.
CO7: Designing and producing a completed website for a specified client involves practical skills and experiences that strongly contribute to preparing for continued professional development in the field of web development.

Mapping of PO5 to all Cos

PO5: CO1: While web development skills are foundational, the direct connection to understanding the broader impact of IT analyst solutions in societal and environmental contexts is limited.
CO2: The skill of evaluating and correcting errors in web languages is essential for quality solutions, but its direct link to understanding societal and environmental impacts is limited.
CO3:: Understanding the distinctions in web pages and server usage is relevant to web development but has a limited direct connection to understanding the broader societal and environmental impacts.
CO4: While website analysis skills are valuable, their direct connection to understanding the societal and environmental impacts of IT analyst solutions is limited.
CO5: Similar to CO3, the understanding of personalized and dynamic web pages is relevant to web development but has a limited direct connection to societal and environmental impacts.
CO6: The skill of distinguishing between objective and subjective analysis is valuable but has a limited direct connection to understanding the broader societal and environmental impacts of IT analyst solutions.
CO7:While web design skills are crucial, the direct connection to understanding the societal and environmental impacts of IT analyst solutions is limited.

Mapping of PO6 to all Cos

PO6: CO1:: Developing dynamic and interactive web pages is a core component of computing practice, and proficiency in this skill contributes directly to the overall goal of developing computing proficiency.
CO2: Proficiency in identifying and correcting errors in web languages is a fundamental aspect of computing practice, aligning strongly with the goal of developing proficiency in computing.
CO3: Distinguishing between personalized and dynamic web pages and utilizing servers and web languages for diverse website needs directly contributes to developing proficiency in web development within the computing domain.
CO4: While conducting analyses is valuable, the direct link to computing practice proficiency is moderate, as other aspects such as programming skills and problem-solving may be more directly tied to proficiency.
CO5: Reiteration of CO3, emphasizing the strong connection between understanding different web pages and server usage and the goal of developing proficiency in web development.
CO6: While distinguishing between objective and subjective analysis is a valuable skill, the direct link to computing practice proficiency is moderate compared to other computing skills.
CO7: Designing and producing a completed website for a client is a hands-on application of computing skills, contributing strongly to the development of proficiency in computing practice.

Mapping of PO7 to all Cos

PO7: CO1: Understanding how to develop dynamic and interactive web pages contributes to foundational knowledge that can support independent study, but the direct link to research and transitioning to employment in hardware/software companies is moderate.
CO2: The skill of evaluating and correcting errors in web languages is relevant to practical problem-solving but has a moderate connection to independent research and transitioning to employment.
CO3: Understanding the distinctions in web pages and server usage is relevant to practical skills but has a moderate connection to independent research and transitioning to employment.
CO4: Conducting analyses for website designs is a valuable skill but has a moderate connection to independent research and transitioning to employment compared to other computing skills.
CO5: Similar to CO3, the understanding of personalized and dynamic web pages is relevant to practical skills but has a moderate connection to independent research and transitioning to employment.
CO6: Distinguishing between objective and subjective analysis is a valuable skill but has a moderate connection to independent research and transitioning to employment compared to other computing skills.
CO7: Designing and producing a completed website involves practical skills and experiences that strongly contribute to the capacity for independent study, research, and transitioning to employment in hardware/software companies.

SYLLABUS (CBCS) FOR T.Y.B. Sc. (Computer Science) (Semester- V)
(w.e.f from Academic Year 2024-2025)

Class: T.Y.B.Sc. (Computer Science) (Sem-V) **Paper Code:** UCSCO355
Title of Paper: Advanced Java Programming **Paper:** V
Credit: 3 (4 Lectures/Week) **No. of lectures:** 48

Aim: Advanced Java is everything that goes beyond Core Java – most importantly the APIs defined in Java Enterprise Edition, includes Swing, Database Servlet programming, Web Services, the Persistence API, etc. It is a Web & Enterprise application development platform which basically follows client & server architecture.

Objectives:
- To learn Swing and Database programming using Java
- To study web development concept using Servlet and JSP
- To learn socket programming concept

Learning Outcome:
CO1. Advanced GUI Development
CO2. Database Integration and Programming
CO3. Servlet Development and Deployment
CO4. JSP for Dynamic Web Content
CO5. Networking in Java
CO6. Security Measures in Java Web Applications
CO7. Integration of Components for Comprehensive Applications

Unit No.	Chapter name with Topics	No. of Lectures Required
1.	**User Interface Components with AWT and Swing** 1.1 What is AWT ? What is Swing? Difference between AWT and Swing. 1.2 The MVC Architecture and Swing 1.3 Layout Manager and Layouts, The JComponent class 1.4 Components – J Button, JLabel, JText, JTextArea, JCheckBox and JRadioButton, JList, JComboBox, JMenu and JPopupMenu Class, JMenuItem and JCheckBoxMenuItem, JRadioButtonMenuItem ,JScrollBar 1.5 Dialogs (Message, confirmation, input), JFileChooser, JColorChooser 1.6 Event Handling: Event sources, Listeners 1.7 Mouse and Keyboard Event Handling 1.8 Adapters 1.9 Anonymous inner class	**10**
2.	**Database Programming** 2.1The design of jdbc, jdbc configuration 2.2 Types of drivers 2.3 Executing sql statements, query execution 2.4 Scrollable and updatable result sets	**10**

	2.5 Metadata – Database Metadata, Result SetMetadata 2.6 Transactions – commit(), rollback(), SavePoint	
3.	**Servlet** 3.1 Introduction to Servlet and Hierarchy of Servlet 3.2 Life cycle of servlet 3.3 Tomcat configuration (Note: Only for Lab Demonstration) 3.4 Handing get and post request (HTTP) 3.5 Handling a data from HTML to servlet 3.6 Retriving a data from database to servlet 3.7 Session tracking – User Authorization, URL rewriting, Hidden form fields, Cookies and HttpSession	**12**
4.	**JSP** 4.1 Simple first JSP program 4.2 Life cycle of **JSP** 4.2 Implicit Objects 4.3 Scripting elements – Declarations, Expressions, Scriptlets, Comments 4.4 JSP Directives – Page Directive, include directive 4.5 Mixing Scriptlets and HTML 4.6 Example of forwarding contents from database to servlet, servlet to JSP and displaying it using JSP scriptlet tag	**10**
5.	**Networking** 5.1 Networking basics – Protocol, Addressing, DNS, URL, Socket, Port 5.2 The java.net package – InetAddress, URL, URLConnection class 5.3 SocketServer and Socket class 5.4 Creating a Socket to a remote host on a port (creating TCP client and server) 5.5 Simple Socket Program Example	**6**

Reference Books:

1. Complete reference Java by Herbert Schildt
2. Java 2 programming black books, Steven Horlzner
3. Programming with Java , A primer ,Forth edition , By E. Balagurusamy
4. Core Java Volume-I-Fundamentals, Eighth Edition, Cay S. Horstmann, Gary Cornell, Prentice Hall, Sun Microsystems Press
5. Core Java Volume-II-Advanced Features, Eighth Edition, Cay S. Horstmann, Gary Cornell, Prentice Hall, Sun Microsystems Press

CO-PO Mapping:

Course Outcomes	Program Outcomes						
	PO1	PO2	PO3	PO4	PO5	PO6	PO7
CO1	3	3	3	1	1	2	2
CO2	3	3	3	1	1	2	2
CO3	3	3	3	1	1	2	2

CO4	3	3	3	1	1	2	2
CO5	3	3	3	1	1	2	2
CO6	3	3	3	1	2	2	2
CO7	3	3	3	1	2	3	3

1. **PO1 with All COs**

CO1: PO1:3 (Strongly related): as computer system simulation involves applying mathematical and computational fundamentals to understand and model IT systems.
CO2: PO1: 3 (Strongly related): as implementing assembly programs requires a deep understanding of computer fundamentals and mathematics.
CO3: PO1: 3 (Strongly related): as working with shell commands and system calls requires knowledge of computer fundamentals and IT applications.
CO4:PO1: 3(Strongly related): as CPU scheduling algorithms involve applying mathematical and computational principles to optimize system performance.
CO5: PO1: 3 (Strongly related): as shell scripting and command-line interfaces are integral parts of IT applications, requiring computer knowledge.
CO6: PO1: 3 (Strongly related): as implementing the Banker's algorithm involves applying mathematical and computational concepts to address system deadlock issues.
CO7: PO1: 3 (Strongly related): as troubleshooting in these areas requires applying computer knowledge and problem-solving skills.

2. **PO2 with All COs**

CO1: PO2: 3 (Strongly related): as understanding computer system simulation is foundational to designing and developing solutions in various IT contexts.
CO2: PO2: 3 (Strongly related): as the ability to implement assembly programs is a key aspect of designing and developing solutions at a low level.
CO3: PO2: 3 (Strongly related): as familiarity with shell commands and system calls is essential for designing and developing solutions involving command-line interfaces.
CO4: PO2: 3 (Strongly related): as knowledge of CPU scheduling algorithms is crucial for designing efficient and optimized system solutions.
CO5: PO2: 3(Strongly related): as proficiency in shell scripting and command-line interfaces is valuable in designing and developing practical solutions.
CO6: PO2: 3 (Strongly related): as implementing the Banker's algorithm is a specific design solution for deadlock avoidance in system development.
CO7: PO2: 3 (Strongly related): as troubleshooting skills are essential for the ongoing development and maintenance of solutions.

3. **PO3 with All COs**

CO1: PO3: 3 (Strongly related): as simulation tools are modern tools extensively used for understanding and modeling complex computer systems.
CO2: PO3: 3(Strongly related) as modern tools for assembly programming are essential for efficient and error-free implementation.
CO3: PO3: 3 (Strongly related): as proficiency in using modern command-line tools is crucial for effective utilization of shell commands and system calls.
CO4: PO3: 3(Strongly related): as modern tools are employed to simulate and analyze the performance of various CPU scheduling algorithms.

CO5: PO3: 3 (Strongly related): as modern tools and editors are commonly used for efficient development and execution of shell scripts.
CO6: PO3: 3 (Strongly related): as the implementation and simulation of algorithms often involve the use of modern programming tools and environments.
CO7: PO3: 3 (Strongly related): as modern debugging and profiling tools are essential for effective troubleshooting in various areas of system development.

4. **PO4 with All COs**

CO1: PO4: 1(Partially related): as computer system simulation, is more aligned with technical aspects than direct environmental and sustainability considerations.
CO2: PO4: 1(Partially related): as assembly programming focuses on technical skills rather than direct implications for environmental and sustainability concerns.
CO3: PO4: 1(Partially related): as shell commands and system calls are more technical in nature and have limited direct connection to environmental and sustainability aspects.
CO4: PO4: 1 (Partially related): as CPU scheduling algorithms are primarily technical and do not have a strong direct link to environmental and sustainability considerations.
CO5: PO4: 1 (Partially related): as shell scripting and command-line interfaces are technical skills with limited direct impact on environmental and sustainability aspects.
CO6:PO4: 1(Partially related): as the Banker's algorithm focuses on technical aspects of deadlock avoidance rather than environmental or sustainability implications.
CO7: PO4: 1 (Partially related): as troubleshooting skills are more aligned with technical problem-solving and have limited direct connection to environmental and sustainability concerns.

5. **PO5 with All COs**

CO1: PO5: 1 (Partially related): as computer system simulation is more aligned with technical aspects, and its connection to ethical principles is indirect.
CO2PO5: 1(Partially related): as assembly programming primarily focuses on technical skills rather than direct ethical considerations.
CO3: PO5: 1 (Partially related): as shell commands and system calls are technical in nature and have limited direct connection to ethical principles.
CO4: PO5: 1 (Partially related): as CPU scheduling algorithms are primarily technical and do not have a strong direct link to ethical considerations.
CO5: PO5: 1(Partially related): as shell scripting and command-line interfaces are technical skills with limited direct impact on ethical principles.
CO6: PO5: 2 (Moderately related): as the implementation of the Banker's algorithm may involve considerations related to ethical and responsible programming practices.
CO7: PO5: 2 (Moderately related) as troubleshooting involves ethical considerations, such as maintaining the integrity and security of systems.

6. **PO6 with All COs**

CO1: PO6: 2(Moderately related): as collaborative efforts may be involved in designing and interpreting simulations.
CO2: PO6: 2 (Moderately related): as teamwork may be required for collaborative coding, code reviews, or troubleshooting.
CO3: PO6: 2 (Moderately related): as working on command-line interfaces and scripting may involve collaboration and knowledge sharing within a team.
CO4: PO6: 2 (Moderately related): as understanding and implementing scheduling algorithms may require teamwork for analysis and optimization.

CO5: PO6: 2 (Moderately related): as collaboration and sharing of scripts within a team may be necessary for effective system management.
CO6: PO6: 2(Moderately related): as collaborative efforts may be needed to implement and test the Banker's algorithm in a simulated environment.
CO7: PO6: 3(Strongly related): as troubleshooting often involves collaboration and collective problem-solving within a team.

7. **PO7 with All COs**

CO1: PO7: 2 (Moderately related): as simulation skills may contribute to innovative problem-solving and employability in technical roles.
CO2: PO7: 2 (Moderately related): as assembly programming skills may enhance employability in technical fields and contribute to innovative solutions.
CO3: PO7: 2 (Moderately related): as proficiency in shell commands and system calls is valuable for employability and innovation in system administration and development.
CO4: PO7: 2 (Moderately related):as knowledge of CPU scheduling algorithms can contribute to innovative solutions and employability in system optimization roles.
CO5:PO7: 2 (Moderately related): as scripting skills are often sought after in IT roles, contributing to employability and potential innovation in automation.
CO6: PO7: 2 (Moderately related): as implementation of algorithms demonstrates technical competence relevant to employability and innovation.
CO7: PO7: 3 (Strongly related): as troubleshooting skills are critical for employability and can contribute to innovation by solving complex technical challenges.

SYLLABUS (CBCS) FOR T.Y.B. Sc. (Computer Science) (Semester- V)
(w.e.f from Academic Year 2024-2025)

Class: T.Y.B.Sc. (Computer Science) (Sem-V) **Paper Code:** UCSCO356
Title of Paper: Object Oriented Software Engineering **Paper:** VI
Credit: 3 (4 Lectures/Week) **No. of lectures:** 48
Prerequisites: Knowledge of Classical Software Engineering

Aim: To Understand Object Oriented Modeling techniques and their applicability.

Objectives:

- Understanding Object Orientation in Software engineering concepts and importance
- Understand the Unified Modeling Language concepts, importance and its components
- Understand Structural, Behavioral, Dynamic modeling techniques and diagrams.
- Understand Object Oriented analysis, design, testing concepts and its techniques

OUTCOMES:

CO1: Develop models using the UML notation.
CO2: Apply an iterative, agile process.
CO3: Analyze requirements with use cases.
CO4: Create domain models
CO5: Relate analysis and design artifacts.
CO6: Design object solutions with patterns and architectural layers.
CO7: Apply concepts to a semester-long software engineering project.

Title and Contents		**No. of Lectures**
Unit 1	**Object Oriented Concepts and Principles** 1.1 Introduction, Object, Classes and Instance,Polymorphism, Inheritance 1. 2 Object Oriented System Development- Introduction, Function/Data Methods (With Visibility), Object Oriented Analysis, Object Oriented Construction 1.2 Identifying the Elements of an Object ModelAggregations, 1.3 Identifying Classes and Objects, Identity, Dynamicbinding, Persistence, Meta classes 1.5 Specifying the Attributes (With Visibility) 1.6 Defining Operations 1.7 Finalizing the Object Definition	**04**
Unit 2	**Introduction to UML and Object Oriented Methodology** 2.1 Concept of UML 2.2 Advantages of UML 2.3 Object oriented Methods (The Booch Method, The Coad and Yourdon Method, Jacobson Method and Raumbaugh Method)	**06**
Unit 3	**Basic Structural Modeling** 3.1 Classes 3.2 Relationship	**05**

	3.3 Common Mechanism 3.4 Class Diagram (Minimum three examples should be covered)	
Unit 4	**Advanced Structural Modeling** 4.1 Advanced Classes 4.2 Advanced Relationship 4.3 Interface 4.4 Types and Roles 4.5 Packages 4.6 Object Diagram (Minimum three examples should be covered)	**05**
Unit 5	**Basic Behavioral Modeling** 5.1 Interactions 5.2 Use Cases and Use Case Diagram with stereo types (Minimum three examples should be covered) 5.3 Interaction Diagram (Minimum two examples should be covered) 5.4 Sequence Diagram (Minimum two examples should be covered) 5.6 Activity Diagram (Minimum two examples should be covered) 5.6 State Chart Diagram (Minimum two examples should be covered)	**06**
Unit 6	**Object Oriented Analysis** 6.1 Iterative Development and the Rational Unified Process 6.2 Inception 6.3 Understanding Requirements 6.4 Use Case Model from Inception to Elaboration 6.5 Elaboration	**06**
Unit 7	**Object Oriented Design** 7.1 The Generic Components of the OO Design Model 7.2 The System Design Process - Partitioning the Analysis Model, Concurrency and Sub System Allocation, Task Management Component, The Data Management Component, The Resource Management Component, Inter Sub System Communication 7.3 Design process and benchmarking, Designing classes, Messages, Information hiding , Class hierarchy , Relationships , Databases , Object relational systems ,Designing interface objects 7.4 Object Design Process, Object oriented system development life cycle.	**05**
Unit 8	**Architectural modeling** 8.1 Component 8.2 Components Diagram (Minimum two examples should be covered) 8.3 Deployment Diagram (Minimum two examples should be covered) 8.4 Collaboration Diagram (Minimum two examples should be covered)	**06**
Unit 9	**Object Oriented Testing** 9.1 Object Oriented Testing Strategies 9.2 Test Case Design for Object Oriented Software 9.3 Inter Class Test Case Design(Use of any freeware designing tool)	**05**

OUTCOMES:

- Understand the activities during the software application development by using Object oriented Design.
- Learn the preparing of documentation allocation for the projects.
- Design and develop the software project development using Object oriented modeling techniques

References

1. Ivar Jacobson, "Object Oriented Software Engineering", Pearson Education INC

2. Craig Larman, "Applying UML and Patterns" Pearson Education INC
3. Bennett, Simon, "Object Oriented Analysis and Design" McGraw Hill
4. Ali Bahrami, "Object Oriented System Development", McGraw Hill International Edition, 2008
5. Brahma Dathan, Sarnath Ramnath, "Object-Oriented Analysis, Design and Implementation", Universities Press, 2010
6. Bernd Bruegge, Allen H. Dutoit, Object Oriented Software Engineering using UML,Patterns and Java, Pearson 2004
7. Craig Larman, Applying UML and Patterns – An Introduction to Object-Oriented Analysis and Design and Iterative Development" , 3rd Edition, Pearson Education, 2005
8. Grady Booch, James Rumbaugh, Ivar Jacobson, "The Unified Modeling Language User Guide", Addison Wesley Long man, 1999
9. Martin Fowler, "UML Distilled A Brief Guide to Standard Object Modeling Language", 3rd Edition, Addison Wesley, 2003

10.Russ Miles, Kim Hamilton, "Learning UML 2.0", O'Reilly, 2008

Mapping of this course with Programme Outcomes

Course Outcomes	**Programme Outcomes (POs)**						
	PO1	PO2	PO3	PO4	PO5	PO6	PO7
CO1	2	2	2	2	2	2	2
CO2	2	3	2	3	2	3	2
CO3	3	3	3	2	3	2	2
CO4	2	2	2	1	2	2	2
CO5	1	1	1	2	2	2	1
CO6	2	3	2	3	3	3	3
CO7	3	3	3	1	2	3	3

Weight: 1 - Partially related 2 - Moderately Related 3 - Strongly related

Here is a mapping of Course Outcomes (CO) to Program Outcomes (PO1) based on the provided weightage:

CO1: Develop models using the UML notation.
PO1: Apply fundamental principles and methods of Computer Science to a wide range of applications. **(Weightage: 2)**
CO2: Apply an iterative, agile process.
PO1: Apply fundamental principles and methods of Computer Science to a wide range of applications. **(Weightage: 2)**
CO3: Analyze requirements with use cases.
PO1: Apply fundamental principles and methods of Computer Science to a wide range of applications. **(Weightage: 3)**
CO4: Create domain models.
PO1: Apply fundamental principles and methods of Computer Science to a wide range of applications**. (Weightage: 2)**

CO5: Relate analysis and design artifacts.
PO1: Apply fundamental principles and methods of Computer Science to a wide range of applications. **(Weightage: 1)**
CO6: Design object solutions with patterns and architectural layers.
PO1: Apply fundamental principles and methods of Computer Science to a wide range of applications. **(Weightage: 2)**
CO7: Apply concepts to a semester-long software engineering project.
PO1: Apply fundamental principles and methods of Computer Science to a wide range of applications**. (Weightage: 3)**

This mapping is based on the given weightage, where a higher weightage indicates a stronger relationship between the course outcome and program outcome. Keep in mind that the interpretation of the weightage may vary based on the specific context and goals of the educational program.

Let's map the Course Outcomes (CO) to Program Outcome (PO2) based on the provided weightage:

CO1: Develop models using the UML notation.
PO2: Design, correctly implement, and document solutions to significant computational problems. **(Weightage: 2)**
CO2: Apply an iterative, agile process.
PO2: Design, correctly implement, and document solutions to significant computational problems. **(Weightage: 3)**

CO3: Analyze requirements with use cases.
PO2: Design, correctly implement, and document solutions to significant computational problems. **(Weightage: 3)**
CO4: Create domain models.
PO2: Design, correctly implement, and document solutions to significant computational problems. **(Weightage: 2)**
CO5: Relate analysis and design artifacts.
PO2: Design, correctly implement, and document solutions to significant computational problems. **(Weightage: 1)**
CO6: Design object solutions with patterns and architectural layers.
PO2: Design, correctly implement, and document solutions to significant computational problems. **(Weightage: 3)**
CO7: Apply concepts to a semester-long software engineering project.
PO2: Design, correctly implement, and document solutions to significant computational problems. **(Weightage: 3)**

This mapping is based on the given weightage, where a higher weightage indicates a stronger relationship between the course outcome and program outcome. Keep in mind that the interpretation of the weightage may vary based on the specific context and goals of the educational program.

Let's map the Course Outcomes (CO) to Program Outcome (PO3) based on the provided weightage:

CO1: Develop models using the UML notation.
PO3: Impart an understanding of the basics of our discipline. **(Weightage: 2)**
CO2: Apply an iterative, agile process.
PO3: Impart an understanding of the basics of our discipline. **(Weightage: 2)**

CO3: Analyze requirements with use cases.
PO3: Impart an understanding of the basics of our discipline. **(Weightage: 3)**
CO4: Create domain models.
PO3: Impart an understanding of the basics of our discipline. **(Weightage: 2)**
CO5: Relate analysis and design artifacts.
PO3: Impart an understanding of the basics of our discipline. **(Weightage: 1)**
CO6: Design object solutions with patterns and architectural layers.
PO3: Impart an understanding of the basics of our discipline. **(Weightage: 3)**
CO7: Apply concepts to a semester-long software engineering project.
PO3: Impart an understanding of the basics of our discipline. **(Weightage: 2)**
This mapping is based on the given weightage, where a higher weightage indicates a stronger relationship between the course outcome and program outcome. Keep in mind that the interpretation of the weightage may vary based on the specific context and goals of the educational program.
Map the Course Outcomes (CO) to Program Outcome (PO4) based on the provided weightage:
CO1: Develop models using the UML notation.
PO4: Prepare for continued professional development. **(Weightage: 2)**
CO2: Apply an iterative, agile process.
PO4: Prepare for continued professional development. **(Weightage: 3)**
CO3: Analyze requirements with use cases.
PO4: Prepare for continued professional development. **(Weightage: 2)**
CO4: Create domain models.
PO4: Prepare for continued professional development. (**Weightage: 1)**
CO5: Relate analysis and design artifacts.
PO4: Prepare for continued professional development. **(Weightage: 2)**
CO6: Design object solutions with patterns and architectural layers.
PO4: Prepare for continued professional development. **(Weightage: 3)**
CO7: Apply concepts to a semester-long software engineering project.
PO4: Prepare for continued professional development. **(Weightage: 2)**
This mapping is based on the given weightage, where a higher weightage indicates a stronger relationship between the course outcome and program outcome. Keep in mind that the interpretation of the weightage may vary based on the specific context and goals of the educational program
Let's map the Course Outcomes (CO) to Program Outcome (PO5) based on the provided weightage:
CO1: Develop models using the UML notation.
PO5: Understand the impact of the IT analyst solutions in societal and environmental contexts, and demonstrate the knowledge and need for sustainable development.
(Weightage: 2)
CO2: Apply an iterative, agile process.
PO5: Understand the impact of the IT analyst solutions in societal and environmental contexts, and demonstrate the knowledge and need for sustainable development. (**Weightage: 2)**
CO3: Analyze requirements with use cases.
PO5: Understand the impact of the IT analyst solutions in societal and environmental contexts, and demonstrate the knowledge and need for sustainable development. **(Weightage: 3)**
CO4: Create domain models.

PO5: Understand the impact of the IT analyst solutions in societal and environmental contexts, and demonstrate the knowledge and need for sustainable development. **(Weightage: 1)**
CO5: Relate analysis and design artifacts.
PO5: Understand the impact of the IT analyst solutions in societal and environmental contexts, and demonstrate the knowledge and need for sustainable development. **(Weightage: 2)**
CO6: Design object solutions with patterns and architectural layers.
PO5: Understand the impact of the IT analyst solutions in societal and environmental contexts, and demonstrate the knowledge and need for sustainable development. **(Weightage: 3)**
CO7: Apply concepts to a semester-long software engineering project.
PO5: Understand the impact of the IT analyst solutions in societal and environmental contexts, and demonstrate the knowledge and need for sustainable development. **(Weightage: 2)**

This mapping is based on the given weightage, where a higher weightage indicates a stronger relationship between the course outcome and program outcome. Keep in mind that the interpretation of the weightage may vary based on the specific context and goals of the educational program.

Let's map the Course Outcomes (CO) to Program Outcome (PO6) based on the provided weightage:
CO1: Develop models using the UML notation.
PO6: Develop proficiency in the practice of computing. **(Weightage: 2)**
CO2: Apply an iterative, agile process.
PO6: Develop proficiency in the practice of computing. **(Weightage: 3)**
CO3: Analyze requirements with use cases.
PO6: Develop proficiency in the practice of computing. **(Weightage: 2)**
CO4: Create domain models.
PO6: Develop proficiency in the practice of computing. **(Weightage: 2)**
CO5: Relate analysis and design artifacts.
PO6: Develop proficiency in the practice of computing. **(Weightage: 2)**
CO6: Design object solutions with patterns and architectural layers.
PO6: Develop proficiency in the practice of computing. **(Weightage: 3)**
CO7: Apply concepts to a semester-long software engineering project.
PO6: Develop proficiency in the practice of computing. **(Weightage: 3)**

This mapping is based on the given weightage, where a higher weightage indicates a stronger relationship between the course outcome and program outcome. Keep in mind that the interpretation of the weightage may vary based on the specific context and goals of the educational program.

Let's map the Course Outcomes (CO) to Program Outcome (PO7) based on the provided weightage:
CO1: Develop models using the UML notation.
PO7: Develop the capacity to study and research independently that will help to develop skills for transition to employment in hardware/software companies. **(Weightage: 2)**
CO2: Apply an iterative, agile process.
PO7: Develop the capacity to study and research independently that will help to develop skills for transition to employment in hardware/software companies. **(Weightage: 2)**
CO3: Analyze requirements with use cases.
PO7: Develop the capacity to study and research independently that will help to develop skills for transition to employment in hardware/software companies. **(Weightage: 2)**

CO4: Create domain models.

PO7: Develop the capacity to study and research independently that will help to develop skills for transition to employment in hardware/software companies. **(Weightage: 2)**

CO5: Relate analysis and design artifacts.

PO7: Develop the capacity to study and research independently that will help to develop skills for transition to employment in hardware/software companies. **(Weightage: 1)**

CO6: Design object solutions with patterns and architectural layers.

PO7: Develop the capacity to study and research independently that will help to develop skills for transition to employment in hardware/software companies. **(Weightage: 3)**

CO7: Apply concepts to a semester-long software engineering project.

PO7: Develop the capacity to study and research independently that will help to develop skills for transition to employment in hardware/software companies. **(Weightage: 3)**

This mapping is based on the given weightage, where a higher weightage indicates a stronger relationship between the course outcome and program outcome. Keep in mind that the interpretation of the weightage may vary based on the specific context and goals of the educational program.

SYLLABUS (CBCS) FOR T.Y.B. Sc. (Computer Science) (Semester- V)
(w.e.f from Academic Year 2024-2025)

Class: T.Y.B.Sc. (Computer Science) (Sem-V) **Paper Code:** UCSCO357

Title of Paper: Lab Course-I on UCSCO351 **Paper:** VII **(**Lab Course-I)

Credit: 2 (3Hr practical/week/batch) **No. of Practical: 14**

Course Outcomes:
CO1: Develop a practical understanding of computer system simulation.
CO2: Develop implementation skills in processing assembly program.
CO3: Develop a practical understanding of various shell commands and system calls
CO4: Develop a practical understanding of CPU scheduling algorithms.
CO5: Develop a practical understanding shell scripting and command-line interfaces.
CO6: Implement the Banker's algorithm for deadlock avoidance in a simulated environment.
CO7: Encourage troubleshooting abilities to address issues related to simulation, assembly, shell scripting, and algorithm implementation.

Assignment No.	**Name of Assignment**	**No of Practical Sessions Required**
1.	Simulator	02
2.	Assembler	02
3.	Shell Program to implement System Calls	02
4.	Process Scheduling	04
5.	Memory Management	04

Course Outcomes	Program Outcomes						
	PO1	PO2	PO3	PO4	PO5	PO6	PO7
CO1	3	3	3	1	1	2	2
CO2	3	3	3	1	1	2	2
CO3	3	3	3	1	1	2	2
CO4	3	3	3	1	1	2	2
CO5	3	3	3	1	1	2	2
CO6	3	3	3	1	2	2	2
CO7	3	3	3	1	2	3	3

PO1 with All COs
CO1: PO1:3 (Strongly related): as computer system simulation involves applying mathematical and computational fundamentals to understand and model IT systems.

CO2: PO1: 3 (Strongly related): as implementing assembly programs requires a deep understanding of computer fundamentals and mathematics.
CO3: PO1: 3 (Strongly related): as working with shell commands and system calls requires knowledge of computer fundamentals and IT applications.
CO4:PO1: 3(Strongly related): as CPU scheduling algorithms involve applying mathematical and computational principles to optimize system performance.
CO5: PO1: 3 (Strongly related): as shell scripting and command-line interfaces are integral parts of IT applications, requiring computer knowledge.
CO6: PO1: 3 (Strongly related): as implementing the Banker's algorithm involves applying mathematical and computational concepts to address system deadlock issues.
CO7: PO1: 3 (Strongly related): as troubleshooting in these areas requires applying computer knowledge and problem-solving skills.

PO2 with All COs
CO1: PO2: 3 (Strongly related): as understanding computer system simulation is foundational to designing and developing solutions in various IT contexts.
CO2: PO2: 3 (Strongly related): as the ability to implement assembly programs is a key aspect of designing and developing solutions at a low level.
CO3: PO2: 3 (Strongly related): as familiarity with shell commands and system calls is essential for designing and developing solutions involving command-line interfaces.
CO4: PO2: 3 (Strongly related): as knowledge of CPU scheduling algorithms is crucial for designing efficient and optimized system solutions.
CO5: PO2: 3(Strongly related): as proficiency in shell scripting and command-line interfaces is valuable in designing and developing practical solutions.
CO6: PO2: 3 (Strongly related): as implementing the Banker's algorithm is a specific design solution for deadlock avoidance in system development.
CO7: PO2: 3 (Strongly related): as troubleshooting skills are essential for the ongoing development and maintenance of solutions.

PO3 with All COs
CO1: PO3: 3 (Strongly related): as simulation tools are modern tools extensively used for understanding and modeling complex computer systems.
CO2: PO3: 3(Strongly related) as modern tools for assembly programming are essential for efficient and error-free implementation.
CO3: PO3: 3 (Strongly related): as proficiency in using modern command-line tools is crucial for effective utilization of shell commands and system calls.
CO4: PO3: 3(Strongly related): as modern tools are employed to simulate and analyze the performance of various CPU scheduling algorithms.
CO5: PO3: 3 (Strongly related): as modern tools and editors are commonly used for efficient development and execution of shell scripts.
CO6: PO3: 3 (Strongly related): as the implementation and simulation of algorithms often involve the use of modern programming tools and environments.
CO7: PO3: 3 (Strongly related): as modern debugging and profiling tools are essential for effective troubleshooting in various areas of system development.

PO4 with All COs
CO1: PO4: 1(Partially related): as computer system simulation, is more aligned with technical aspects than direct environmental and sustainability considerations.
CO2: PO4: 1(Partially related): as assembly programming focuses on technical skills rather than direct implications for environmental and sustainability concerns.

CO3: PO4: 1(Partially related): as shell commands and system calls are more technical in nature and have limited direct connection to environmental and sustainability aspects.
CO4: PO4: 1 (Partially related): as CPU scheduling algorithms are primarily technical and do not have a strong direct link to environmental and sustainability considerations.
CO5: PO4: 1 (Partially related): as shell scripting and command-line interfaces are technical skills with limited direct impact on environmental and sustainability aspects.
CO6:PO4: 1(Partially related): as the Banker's algorithm focuses on technical aspects of deadlock avoidance rather than environmental or sustainability implications.
CO7: PO4: 1 (Partially related): as troubleshooting skills are more aligned with technical problem-solving and have limited direct connection to environmental and sustainability concerns.

PO5 with All COs
CO1: PO5: 1 (Partially related): as computer system simulation is more aligned with technical aspects, and its connection to ethical principles is indirect.
CO2PO5: 1(Partially related): as assembly programming primarily focuses on technical skills rather than direct ethical considerations.
CO3: PO5: 1 (Partially related): as shell commands and system calls are technical in nature and have limited direct connection to ethical principles.
CO4: PO5: 1 (Partially related): as CPU scheduling algorithms are primarily technical and do not have a strong direct link to ethical considerations.
CO5: PO5: 1(Partially related): as shell scripting and command-line interfaces are technical skills with limited direct impact on ethical principles.
CO6: PO5: 2 (Moderately related): as the implementation of the Banker's algorithm may involve considerations related to ethical and responsible programming practices.
CO7: PO5: 2 (Moderately related) as troubleshooting involves ethical considerations, such as maintaining the integrity and security of systems.

PO6 with All COs
CO1: PO6: 2(Moderately related): as collaborative efforts may be involved in designing and interpreting simulations.
CO2: PO6: 2 (Moderately related): as teamwork may be required for collaborative coding, code reviews, or troubleshooting.
CO3: PO6: 2 (Moderately related): as working on command-line interfaces and scripting may involve collaboration and knowledge sharing within a team.
CO4: PO6: 2 (Moderately related): as understanding and implementing scheduling algorithms may require teamwork for analysis and optimization.
CO5: PO6: 2 (Moderately related): as collaboration and sharing of scripts within a team may be necessary for effective system management.
CO6: PO6: 2(Moderately related): as collaborative efforts may be needed to implement and test the Banker's algorithm in a simulated environment.
CO7: PO6: 3(Strongly related): as troubleshooting often involves collaboration and collective problem-solving within a team.

PO7 with All COs
CO1: PO7: 2 (Moderately related): as simulation skills may contribute to innovative problem-solving and employability in technical roles.
CO2: PO7: 2 (Moderately related): as assembly programming skills may enhance employability in technical fields and contribute to innovative solutions.
CO3: PO7: 2 (Moderately related): as proficiency in shell commands and system calls is valuable for employability and innovation in system administration and development.

CO4: PO7: 2 (Moderately related):as knowledge of CPU scheduling algorithms can contribute to innovative solutions and employability in system optimization roles.
CO5:PO7: 2 (Moderately related): as scripting skills are often sought after in IT roles, contributing to employability and potential innovation in automation.
CO6: PO7: 2 (Moderately related): as implementation of algorithms demonstrates technical competence relevant to employability and innovation.
CO7: PO7: 3 (Strongly related): as troubleshooting skills are critical for employability and can contribute to innovation by solving complex technical challenges.

SYLLABUS (CBCS) FOR T.Y.B. Sc. (Computer Science) (Semester- V)

(w.e.f from Academic Year 2024-2025)

Class: T.Y.B.Sc. (Computer Science) (Sem-V) **Paper Code:** UCSCO358
Title of Paper: Lab. Course – II: Advanced Java Prog. **Paper:** VIII Lab Course - II
Credit: 2 (3 Hr. Practical/Week/batch) **No. of Practical: 13**

Course Outcomes:

1. Advanced GUI Development
2. Database Integration and Programming
3. Servlet Development and Deployment
4. JSP for Dynamic Web Content
5. Networking in Java
6. Security Measures in Java Web Applications
7. Integration of Components for Comprehensive Applications

LAB WORKBOOK (Proposed)

Chapter No.	**Chapter name with Topics**
1.	**User Interface Components with AWT and Swing** Set A - Assignment 1 Assignment 2 Set B – Assignment 1 Assignment 2
2.	**Database Programming** Set A - Assignment 1 Assignment 2 Set B – Assignment 1 Assignment 2
3.	**Servlet** Set A - Assignment 1 Assignment 2 Set B – Assignment 1 Assignment 2
4.	**JSP** Set A - Assignment 1 Assignment 2 Set B – Assignment 1 Assignment 2
5.	**Networking** Set A - Assignment 1 Assignment 2 Set B – Assignment 1 Assignment 2

CO-PO Mapping:

Course Outcomes	Program Outcomes						
	PO1	PO2	PO3	PO4	PO5	PO6	PO7
CO1	3	3	3	1	1	2	2
CO2	3	3	3	1	1	2	2
CO3	3	3	3	1	1	2	2
CO4	3	3	3	1	1	2	2
CO5	3	3	3	1	1	2	2
CO6	3	3	3	1	2	2	2
CO7	3	3	3	1	2	3	3

1. **PO1 with All COs**

CO1: PO1:3 (Strongly related): as computer system simulation involves applying mathematical and computational fundamentals to understand and model IT systems.
CO2: PO1: 3 (Strongly related): as implementing assembly programs requires a deep understanding of computer fundamentals and mathematics.
CO3: PO1: 3 (Strongly related): as working with shell commands and system calls requires knowledge of computer fundamentals and IT applications.
CO4:PO1: 3(Strongly related): as CPU scheduling algorithms involve applying mathematical and computational principles to optimize system performance.
CO5: PO1: 3 (Strongly related): as shell scripting and command-line interfaces are integral parts of IT applications, requiring computer knowledge.
CO6: PO1: 3 (Strongly related): as implementing the Banker's algorithm involves applying mathematical and computational concepts to address system deadlock issues.
CO7: PO1: 3 (Strongly related): as troubleshooting in these areas requires applying computer knowledge and problem-solving skills.

2. **PO2 with All COs**

CO1: PO2: 3 (Strongly related): as understanding computer system simulation is foundational to designing and developing solutions in various IT contexts.
CO2: PO2: 3 (Strongly related): as the ability to implement assembly programs is a key aspect of designing and developing solutions at a low level.
CO3: PO2: 3 (Strongly related): as familiarity with shell commands and system calls is essential for designing and developing solutions involving command-line interfaces.
CO4: PO2: 3 (Strongly related): as knowledge of CPU scheduling algorithms is crucial for designing efficient and optimized system solutions.
CO5: PO2: 3(Strongly related): as proficiency in shell scripting and command-line interfaces is valuable in designing and developing practical solutions.
CO6: PO2: 3 (Strongly related): as implementing the Banker's algorithm is a specific design solution for deadlock avoidance in system development.
CO7: PO2: 3 (Strongly related): as troubleshooting skills are essential for the ongoing development and maintenance of solutions.

3. **PO3 with All COs**

CO1: PO3: 3 (Strongly related): as simulation tools are modern tools extensively used for understanding and modeling complex computer systems.
CO2: PO3: 3(Strongly related) as modern tools for assembly programming are essential for efficient and error-free implementation.
CO3: PO3: 3 (Strongly related): as proficiency in using modern command-line tools is crucial for effective utilization of shell commands and system calls.
CO4: PO3: 3(Strongly related): as modern tools are employed to simulate and analyze the performance of various CPU scheduling algorithms.
CO5: PO3: 3 (Strongly related): as modern tools and editors are commonly used for efficient development and execution of shell scripts.
CO6: PO3: 3 (Strongly related): as the implementation and simulation of algorithms often involve the use of modern programming tools and environments.
CO7: PO3: 3 (Strongly related): as modern debugging and profiling tools are essential for effective troubleshooting in various areas of system development.

4. **PO4 with All COs**

CO1: PO4: 1(Partially related): as computer system simulation, is more aligned with technical aspects than direct environmental and sustainability considerations.
CO2: PO4: 1(Partially related): as assembly programming focuses on technical skills rather than direct implications for environmental and sustainability concerns.
CO3: PO4: 1(Partially related): as shell commands and system calls are more technical in nature and have limited direct connection to environmental and sustainability aspects.
CO4: PO4: 1 (Partially related): as CPU scheduling algorithms are primarily technical and do not have a strong direct link to environmental and sustainability considerations.
CO5: PO4: 1 (Partially related): as shell scripting and command-line interfaces are technical skills with limited direct impact on environmental and sustainability aspects.
CO6:PO4: 1(Partially related): as the Banker's algorithm focuses on technical aspects of deadlock avoidance rather than environmental or sustainability implications.
CO7: PO4: 1 (Partially related): as troubleshooting skills are more aligned with technical problem-solving and have limited direct connection to environmental and sustainability concerns.

5. **PO5 with All COs**

CO1: PO5: 1 (Partially related): as computer system simulation is more aligned with technical aspects, and its connection to ethical principles is indirect.
CO2PO5: 1(Partially related): as assembly programming primarily focuses on technical skills rather than direct ethical considerations.
CO3: PO5: 1 (Partially related): as shell commands and system calls are technical in nature and have limited direct connection to ethical principles.
CO4: PO5: 1 (Partially related): as CPU scheduling algorithms are primarily technical and do not have a strong direct link to ethical considerations.
CO5: PO5: 1(Partially related): as shell scripting and command-line interfaces are technical skills with limited direct impact on ethical principles.
CO6: PO5: 2 (Moderately related): as the implementation of the Banker's algorithm may involve considerations related to ethical and responsible programming practices.
CO7: PO5: 2 (Moderately related) as troubleshooting involves ethical considerations, such as maintaining the integrity and security of systems.

6. **PO6 with All COs**

CO1: PO6: 2(Moderately related): as collaborative efforts may be involved in designing and interpreting simulations.
CO2: PO6: 2 (Moderately related): as teamwork may be required for collaborative coding, code reviews, or troubleshooting.
CO3: PO6: 2 (Moderately related): as working on command-line interfaces and scripting may involve collaboration and knowledge sharing within a team.
CO4: PO6: 2 (Moderately related): as understanding and implementing scheduling algorithms may require teamwork for analysis and optimization.
CO5: PO6: 2 (Moderately related): as collaboration and sharing of scripts within a team may be necessary for effective system management.
CO6: PO6: 2(Moderately related): as collaborative efforts may be needed to implement and test the Banker's algorithm in a simulated environment.
CO7: PO6: 3(Strongly related): as troubleshooting often involves collaboration and collective problem-solving within a team.

7. **PO7 with All COs**

CO1: PO7: 2 (Moderately related): as simulation skills may contribute to innovative problem-solving and employability in technical roles.
CO2: PO7: 2 (Moderately related): as assembly programming skills may enhance employability in technical fields and contribute to innovative solutions.
CO3: PO7: 2 (Moderately related): as proficiency in shell commands and system calls is valuable for employability and innovation in system administration and development.
CO4: PO7: 2 (Moderately related):as knowledge of CPU scheduling algorithms can contribute to innovative solutions and employability in system optimization roles.
CO5:PO7: 2 (Moderately related): as scripting skills are often sought after in IT roles, contributing to employability and potential innovation in automation.
CO6: PO7: 2 (Moderately related): as implementation of algorithms demonstrates technical competence relevant to employability and innovation.
CO7: PO7: 3 (Strongly related): as troubleshooting skills are critical for employability and can contribute to innovation by solving complex technical challenges.

SYLLABUS (CBCS) FOR T.Y.B.Sc. (Computer Science) (Semester-V)
(w.e.f. from Academic Year 2024-2025)

Class : T.Y.B.Sc. (Computer Science) (Sem-V) **Paper Code :** UCSCO359
Title of Paper : Lab. Course – III : Based on UCSCO354 **Paper :** IX (Lab. Course – III)
Credits :02 (3 Hr. Practical/Week/batch) **No. of Practicals :** 14
Prerequisite: HTML

Objectives:
- To design dynamic, interactive web pages.
- To learn the server side scripting language.
- To learn database connectivity with PHP

Outcome: CO1. Learn the environment of Server Side Scripting.

CO2. Learn the use of control structures and numerous native data types

CO3. Design web pages with the ability to retrieve and present data from a database.

CO4. Learn the basic building blocks of PHP like strings, functions, arrays, objects.

CO5.Compare and contrast between Client Side Script & Server Side Script.

CO6.Build PHP script to create dynamic web content.

CO7. Compare between PHP and other server side scripting languages.

Assignment No.	**Title**
1.	Assignment on basic programs using control structures
2.	Assignment on functions
3.	Assignment on functions
4.	Assignment on strings
5.	Assignment on strings
6.	Assignment on arrays
7.	Assignment on arrays
8.	Assignment on arrays
9.	Assignment on Object Oriented Programming
10.	Assignment on Object Oriented Programming
11.	Assignment on Object Oriented Programming
12.	Assignment on Databases
13.	Assignment on Databases
14.	Assignment on Databases

Mapping of this course with Programme Outcomes

	PO1	PO2	PO3	PO4	PO5	PO6	PO7
CO1	3	3	3	3	2	3	2
CO2	3	3	3	3	2	3	2
CO3	3	3	3	3	3	3	2
CO4	3	3	3	3	2	3	2
CO5	3	2	2	2	2	2	2
CO6	3	3	3	3	3	3	2
CO7	3	2	2	2	2	2	2

Weight: 1 - Partially related 2 - Moderately Related 3 - Strongly related

Justification of mapping of PO1 with all COs

PO1: Apply fundamental principles and methods of Computer Science to a wide range of applications.
CO1. Learn the environment of Server Side Scripting.
Mapping: 3 - Strongly related
Justification: Understanding the environment of server-side scripting is fundamental to applying computer science principles in various applications, aligning strongly with the overarching goal of PO1.
CO2. Learn the use of control structures and numerous native data types.
Mapping: 3 - Strongly related
Justification: Learning control structures and native data types is a core aspect of applying fundamental principles in computer science, providing the foundation for writing efficient server-side scripts, supporting the goals of PO1.
CO3. Design web pages with the ability to retrieve and present data from a database.
Mapping: 3 - Strongly related
Justification: Designing web pages that retrieve and present data from a database requires the application of computer science principles, directly linking to the objective of PO1.
CO4. Learn the basic building blocks of PHP like strings, functions, arrays, objects.
Mapping: 3 - Strongly related
Justification: Learning the basic building blocks of PHP involves understanding the core elements of server-side scripting, aligning directly with the goal of applying fundamental principles in computer science as stated in PO1.
CO5. Compare and contrast between Client Side Script & Server Side Script.
Mapping: 3 - Strongly related
Justification: Comparing client-side and server-side scripting involves understanding the principles behind both, contributing to the application of computer science principles in diverse scenarios, in line with the objectives of PO1.
CO6. Build PHP script to create dynamic web content.
Mapping: 3 - Strongly related
Justification: Building PHP scripts to create dynamic web content directly applies computer science principles to practical web development, supporting the overarching goal of PO1.
CO7. Compare between PHP and other server-side scripting languages.
Mapping: 3 - Strongly related
Justification: Comparing PHP with other server-side scripting languages requires a deep understanding of their principles and functionalities, contributing directly to the application of computer science principles as outlined in PO1.

Justification of mapping of PO2 with all COs
PO2: Design, correctly implement and document solutions to significant computational problems.
CO1. Learn the environment of Server Side Scripting.
Mapping: 3 - Strongly related

Justification: Understanding the environment of server-side scripting is fundamental to designing and implementing solutions to computational problems, aligning directly with the goal of PO2.
CO2. Learn the use of control structures and numerous native data types.
Mapping: 3 - Strongly related
Justification: Learning control structures and native data types is essential for designing and implementing efficient solutions to computational problems, supporting the objectives of PO2.
CO3. Design web pages with the ability to retrieve and present data from a database.
Mapping: 3 - Strongly related
Justification: Designing web pages with database interaction involves solving computational problems related to data retrieval and presentation, contributing directly to the goal of PO2.
CO4. Learn the basic building blocks of PHP like strings, functions, arrays, objects.
Mapping: 3 - Strongly related
Justification: Learning the basic building blocks of PHP is essential for designing and implementing solutions using the language, aligning directly with the objectives of PO2.
CO5. Compare and contrast between Client Side Script & Server Side Script.
Mapping: 2 - Moderately related
Justification: While understanding the differences between client-side and server-side scripting is relevant, it is moderately related to the direct implementation of solutions to computational problems as emphasized in PO2.
CO6. Build PHP script to create dynamic web content.
Mapping: 3 - Strongly related
Justification: Building PHP scripts for dynamic web content involves implementing solutions to computational problems related to web development, directly supporting the goal of PO2.
CO7. Compare between PHP and other server-side scripting languages.
Mapping: 2 - Moderately related
Justification: Comparing PHP with other server-side scripting languages is informative but is moderately related to the direct implementation of solutions to computational problems as outlined in PO2.

Justification of mapping of PO3 with all COs
PO3: Impart an understanding of the basics of our discipline.
CO1. Learn the environment of Server Side Scripting.
Mapping: 3 - Strongly related
Justification: Understanding the environment of server-side scripting is foundational to acquiring a basic understanding of web development, aligning closely with the goal of imparting basics as specified in PO3.
CO2. Learn the use of control structures and numerous native data types.
Mapping: 3 - Strongly related
Justification: Learning control structures and native data types forms the basics of programming, contributing directly to the understanding of foundational concepts within our discipline as emphasized in PO3.
CO3. Design web pages with the ability to retrieve and present data from a database.
Mapping: 3 - Strongly related
Justification: Designing web pages with database interaction imparts a practical understanding of applying programming concepts to real-world scenarios, aligning with the goal of imparting basics in our discipline as per PO3.
CO4. Learn the basic building blocks of PHP like strings, functions, arrays, objects.
Mapping: 3 - Strongly related
Justification: Learning the basic building blocks of PHP is integral to gaining a foundational understanding of server-side scripting, contributing directly to the basics of our discipline as outlined in PO3.
CO5. Compare and contrast between Client Side Script & Server Side Script.
Mapping: 2 - Moderately related
Justification: While understanding the differences between client-side and server-side scripting contributes to a holistic understanding, it is moderately related to the foundational aspects emphasized in PO3.
CO6. Build PHP script to create dynamic web content.
Mapping: 3 - Strongly related

Justification: Building PHP scripts for dynamic web content is a practical application of server-side scripting concepts, aligning directly with the goal of imparting a foundational understanding of our discipline as stated in PO3.
CO7. Compare between PHP and other server-side scripting languages.
Mapping: 2 - Moderately related
Justification: Comparing PHP with other server-side scripting languages provides additional context but is moderately related to the core understanding of basics in our discipline, as specified in PO3.

Justification of mapping of PO4 with all COs

PO4: Prepare for continued professional development.
CO1. Learn the environment of Server Side Scripting.
Mapping: 3 - Strongly related
Justification: Learning the environment of server-side scripting is fundamental to professional development in web development, aligning directly with the goal of preparing for continued professional development as outlined in PO4.
CO2. Learn the use of control structures and numerous native data types.
Mapping: 3 - Strongly related
Justification: Mastering control structures and native data types is crucial for a strong foundation in programming, directly supporting the goal of continued professional development as specified in PO4.
CO3. Design web pages with the ability to retrieve and present data from a database.
Mapping: 3 - Strongly related
Justification: Designing web pages with database interaction is a practical skill that contributes to professional development in web development, aligning with the goal of PO4.
CO4. Learn the basic building blocks of PHP like strings, functions, arrays, objects.
Mapping: 3 - Strongly related
Justification: Learning the basic building blocks of PHP is essential for professional development in server-side scripting, directly supporting the objectives of PO4.
CO5. Compare and contrast between Client Side Script & Server Side Script.
Mapping: 2 - Moderately related
Justification: While understanding the difference between client-side and server-side scripting is valuable, it is moderately related to the broader goal of professional development outlined in PO4.
CO6. Build PHP script to create dynamic web content.
Mapping: 3 - Strongly related
Justification: Building PHP scripts for dynamic web content is a practical skill that directly contributes to professional development in web development, aligning with the objectives of PO4.
CO7. Compare between PHP and other server-side scripting languages.
Mapping: 2 - Moderately related
Justification: Comparing PHP with other server-side scripting languages provides additional context, moderately supporting the goal of professional development as specified in PO4.

Justification of mapping of PO5 with all COs

PO5: Understand the impact of the IT analyst solutions in societal and environmental contexts, and demonstrate the knowledge and need for sustainable development.
CO1. Learn the environment of Server Side Scripting.
Mapping: 2 - Moderately related
Justification: Learning the environment of server-side scripting provides foundational knowledge but is moderately related to understanding the broader societal and environmental impact as emphasized in PO5.
CO2. Learn the use of control structures and numerous native data types.
Mapping: 2 - Moderately related
Justification: Understanding control structures and data types is important for programming skills but is moderately related to the societal and environmental impact aspect emphasized in PO5.
CO3. Design web pages with the ability to retrieve and present data from a database.
Mapping: 3 - Strongly related

Justification: Designing web pages with database interaction has a direct connection to the impact on societal and environmental contexts, aligning strongly with the goals of PO5.
CO4. Learn the basic building blocks of PHP like strings, functions, arrays, objects.
Mapping: 2 - Moderately related
Justification: Learning the basic building blocks of PHP contributes to programming skills but is moderately related to understanding the broader societal and environmental impact as emphasized in PO5.
CO5. Compare and contrast between Client Side Script & Server Side Script.
Mapping: 2 - Moderately related
Justification: Comparing client-side and server-side scripting is informative but is moderately related to the societal and environmental impact aspect as outlined in PO5.
CO6. Build PHP script to create dynamic web content.
Mapping: 3 - Strongly related
Justification: Building PHP scripts for dynamic web content have a direct impact on societal and environmental contexts, aligning strongly with the goals of PO5.
CO7. Compare between PHP and other server-side scripting languages.
Mapping: 2 - Moderately related
Justification: Comparing PHP with other scripting languages provides additional knowledge but is moderately related to the societal and environmental impact aspect emphasized in PO5.

Justification of mapping of PO6 with all COs

PO6: Develop proficiency in the practice of computing.
CO1. Learn the environment of Server Side Scripting.
Mapping: 3 - Strongly related
Justification: Learning the environment of server-side scripting is fundamental to developing proficiency in computing, aligning directly with the goal of PO6.
CO2. Learn the use of control structures and numerous native data types.
Mapping: 3 - Strongly related
Justification: Mastering control structures and native data types is crucial for proficiency in programming, supporting the overarching goal of developing computing proficiency as specified in PO6.
CO3. Design web pages with the ability to retrieve and present data from a database.
Mapping: 3 - Strongly related
Justification: Designing web pages with database interaction contributes to practical skills in web development, aligning directly with the goal of developing proficiency in computing as outlined in PO6.
CO4. Learn the basic building blocks of PHP like strings, functions, arrays, objects.
Mapping: 3 - Strongly related
Justification: Learning the basic building blocks of PHP is essential for proficiency in server-side scripting and computing, supporting the objectives of PO6.
CO5. Compare and contrast between Client Side Script & Server Side Script.
Mapping: 2 - Moderately related
Justification: While understanding the differences between client-side and server-side scripting is valuable, it is moderately related to the broader goal of developing proficiency in computing as specified in PO6.
CO6. Build PHP script to create dynamic web content.
Mapping: 3 - Strongly related
Justification: Building PHP scripts for dynamic web content is a practical application that directly contributes to proficiency in computing, aligning with the objectives of PO6.
CO7. Compare between PHP and other server-side scripting languages.
Mapping: 2 - Moderately related
Justification: Comparing PHP with other server-side scripting languages provides additional context, moderately supporting the goal of developing proficiency in computing as emphasized in PO6.

Justification of mapping of PO7 with all COs

PO7: Develop the capacity to study and research independently that will help to develop skills for transition to employment in hardware/software companies.
CO1. Learn the environment of Server Side Scripting.
Mapping: 2 - Moderately related
Justification: Learning the environment of server-side scripting contributes partially to developing the capacity for independent study and research, which aids in the transition to employment, as stated in PO7.
CO2. Learn the use of control structures and numerous native data types.
Mapping: 2 - Moderately related
Justification: Mastering control structures and data types is relevant for independent study and research but is moderately related to the overall goal of transitioning to employment, as specified in PO7.
CO3. Design web pages with the ability to retrieve and present data from a database.
Mapping: 2 - Moderately related
Justification: Designing web pages with database interaction contributes partially to developing skills for independent study and research, supporting the transitional goal of employment as mentioned in PO7.
CO4. Learn the basic building blocks of PHP like strings, functions, arrays, objects.
Mapping: 2 - Moderately related
Justification: Learning the basic building blocks of PHP is a partial contribution to developing skills for independent study and research, aligning with the transitional goal of employment stated in PO7.
CO5. Compare and contrast between Client Side Script & Server Side Script.
Mapping: 2 - Moderately related
Justification: Understanding the differences between client-side and server-side scripting is moderately related to developing independent study skills for transitioning to employment, as emphasized in PO7.
CO6. Build PHP script to create dynamic web content.
Mapping: 2 - Moderately related
Justification: Building PHP scripts for dynamic web content contributes partially to developing skills for independent study and research, aligning with the transitional goal of employment as outlined in PO7.
CO7. Compare between PHP and other server-side scripting languages.
Mapping: 2 - Moderately related
Justification: Comparing PHP with other server-side scripting languages is moderately related to developing skills for independent study and research for transitioning to employment, as mentioned in PO7.