PALLAVI ENGINEERING COLLEGE

UGC-AUTONOMOUS
Affiliated to
JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY HYDERABAD
B.Tech. in COMPUTER SCIENCE AND ENGINEERING- (CYBER SECURITY)
COURSE STRUCTURE & SYLLABUS
(PR25 Regulations)

Applicable from AY: 2025-26 Batch

I Year I Semester

S. No.	Course Code	Course	L	T	P	Credits
1.	PMA101BS	Matrices and Calculus	3	1	0	4
2.	PCH102BS	Engineering Chemistry	3	0	0	3
3.	PEN103BS	English for Skill Enhancement	3	0	0	3
4.	PEC104ES	Electronic Devices and Circuits	3	0	0	3
5.	PCS105ES	Programming for Problem Solving	3	0	0	3
6.	PCH106BS	Engineering Chemistry Lab	0	0	2	1
7.	PCS107ES	Programming for Problem Solving Lab	0	0	2	1
8.	PEN108BS	English Language and Communication Skills Lab	0	0	2	1
9.	PCS110ES	IT Workshop	0	0	2	1
		Induction Program				
		Total Credits	15	1	8	20

I Year II Semester

S. No.	Course Code	Course	L	T	P	Credits
1.	PMA201BS	Ordinary Differential Equations and Vector Calculus	3	0	0	3
2.	PPH202BS	Advanced Engineering Physics	3	0	0	3
3.	PME203ES	Computer Aided Engineering Graphics	2	0	2	3
4.	PEE204ES	Basic Electrical Engineering	3	0	0	3
5.	PCS205ES	Data Structures	3	0	0	3
6.	PPH206BS	Advanced Engineering Physics Lab	0	0	2	1
7.	PCS207ES	Data Structures Lab	0	0	2	1
8.	PCS208ES	Python Programming Lab	0	0	2	1
9.	PEE209ES	Basic Electrical Engineering Lab	0	0	2	1
10.	PME210ES	Engineering Workshop	0	0	2	1
		Total Credits	14	0	12	20

[illegible] [illegible] [illegible] [illegible] [illegible] [illegible] [illegible]

II Year I Semester

S. No.	Course Code	Course	L	T	P	Credits
1.	PMA301BS	Mathematical and Statistical Foundations	3	0	0	3
2.	PCY302PC	Computer Organization	3	0	0	3
3.	PCY303PC	Java Programming	3	0	0	3
4.	PCS304PC	Software Engineering	3	0	0	3
5.	PCS305PC	Data Base Management Systems	3	0	0	3
6.	PMA306BS	Computational Mathematics Lab	0	0	2	1
7.	PCY307PC	Java Programming Lab	0	0	2	1
8.	PCS308PC	Software Engineering Lab	0	0	2	1
9.	PCS309PC	Data Base Management Systems Lab	0	0	2	1
10.	PCY310PC	Data Visualization-R programming/Power BI/Tableau/Google Chart	0	0	2	1
		Total Credits	15	0	10	20

II Year II Semester

S. No.	Course Code	Course	L	T	P	Credits
1.	PCS401PC	Discrete Mathematics	3	0	0	3
2.	PCS402PC	Operating Systems	3	0	0	3
3.	PCY403PC	Formal languages and Automata Theory	3	0	0	3
4.	PCS404PC	Computer Networks	3	0	0	3
5.	PCY405PC	Mathematical Foundations of Cryptography	3	0	0	3
6.	PSM406BS	Innovation and Entrepreneurship	2	0	0	2
7.	PCS407PC	Operating Systems Lab	0	0	2	1
8.	PCS408PC	Computer Networks Lab	0	0	2	1
9.	PCY409PC	Mathematical Foundations of Cryptography Lab	0	0	2	1
10.	PCY410PC	Node JS/React JS/ Django, UI Design-Flutter	0	0	2	1
11.	PVA411HS	Indian Knowledge System	1	0	0	1
		Total Credits	18	0	08	22

*Note: Students who wish to exit after II Year II Semester has to register for this optional course and acquire the credits allotted by doing 6 weeks Work-based Vocational Course/ Internship or Apprenticeship. Please refer PR25 Academic Regulations for more information.

[illegible] [illegible] [illegible] [illegible] [illegible] [illegible] [illegible] [illegible]

PCY302PC: COMPUTER ORGANIZATION

B.Tech. II Year I Sem.

L	T	P	C
3	0	0	3

Prerequisites: No prerequisites.
Co-requisite: A Course on "Digital Electronics".

Course Objectives:

1. The purpose of the course is to introduce principles of computer organization and the basic architectural concepts.
2. It begins with basic organization, design, and programming of a simple digital computer and introduces simple register transfer language to specify various computer operations.
3. Topics include computer arithmetic, instruction set design, microprogrammed control unit, pipelining and vector processing, memory organization and I/O systems, and multiprocessors

Course Outcomes:

1. Demonstrate the ability to represent and manipulate data using binary number systems and apply Boolean algebra to analyze and design digital logic circuits.
2. Simplify Boolean functions using Karnaugh Maps and design minimal logic circuits using universal logic gates.
3. Classify data representation formats and implement arithmetic, logic, and shift microoperations using register transfer language.
4. Illustrate the basic organization of a computer and differentiate between hardwired and microprogrammed control unit designs.
5. Analyze CPU structure, addressing modes, and execute arithmetic operations including fixed point, floating-point, and BCD formats.

UNIT - I

Binary Systems: Digital Systems, Binary Numbers, Number Base Conversions, Octal and Hexadecimal Numbers, Complements, Signed Binary Numbers, Binary Codes, Binary Storage and Registers, Binary Logic. Boolean Algebra **And Logic Gates:** Basic Definitions, Axiomatic definition of Boolean Algebra, Basic theorems and properties of Boolean Algebra, Boolean Functions, Canonical and Standard Forms, Other Logic Operations, Digital Logic Gates, Integrated Circuits.

UNIT - II

Gate-Level Minimization: The Map method, Four-variable map, Five-variable map, Product of Sum's simplifications, Don't care conditions, NAND and NOR implementation, other two level implementations, Exclusive-OR Function.

[illegible] [illegible] [illegible] [illegible]

[illegible] [illegible] [illegible]

UNIT – III

Basic Structure of Computers: Computer Types, Functional unit, Data Representation, Fixed Point Representation, Floating Point Representation, Error Detection codes.

Register Transfer Language and Micro operations: Register Transfer language, Register Transfer, Bus and memory transfers, Arithmetic Micro operations, Logic micro-operations, Shift micro operations, Arithmetic logic shift unit.

UNIT - IV

Basic Computer Organization and Design: Instruction codes, Computer Registers, Computer instructions, Timing and Control, Instruction cycle, Memory Reference Instructions, Input – Output and Interrupt, Complete Computer Description.

Micro Programmed Control: Control memory, Address sequencing, micro program example, design of control unit, Micro program Sequencer, Hard wired control Vs Micro programmed control

UNIT - V

Central Processing Unit Organization: General Register Organization, STACK organization. Instruction formats, Addressing modes. DATA Transfer and manipulation, Program control. Reduced Instruction set computer.

Computer Arithmetic: Addition and subtraction, multiplication Algorithms, Floating – point Arithmetic, BCD Adder.

TEXT BOOKS:

1. Digital Design: With an Introduction to the Verilog HDL – Fifth Edition, M. Morris Mano, Pearson Education.
2. Fundamentals of Logic Design – Roth, 7th Edition,
3. Thomson. Computer Systems Architecture – M. Moris Mano, 3rd Edition, Pearson/PHI
4. Computer Organization – Carl Hamacher, Zvonks Vranesic, SafeaZaky, 5th Edition, McGraw Hill.

REFERENCE BOOKS:

1. Digital Principles and Design – Donald D. Givone, Tata Mc Graw Hill.
2. Fundamentals of Digital Logic and Micro Computer Design, 5th Edition, M. Rafiquzzaman, Wiley-Interscience.
3. Computer Organization and Architecture – William Stallings 7th Edition, Pearson/ PHI.
4. Structured Computer Organization – Andrew S. Tanenbaum, 6th Edition PHI/Pearson.

[illegible] [illegible]

[illegible] [illegible] [illegible]

[illegible] [illegible] [illegible]

PCY303PC: JAVA PROGRAMMING

B.Tech. II Year I Sem.

L	T	P	C
3	0	0	3

Course Objectives:

1. To Understand the basic object-oriented programming concepts and apply them in problem solving.
2. To Illustrate inheritance concepts for reusing the program.
3. To Demonstrate multitasking by using multiple threads and event handling
4. To Develop data-centric applications using JDBC.
5. To Understand the basics of java console and GUI based programming

Course Outcomes:

1. Identify the model of Object-Oriented Programming: Abstract data types, Encapsulation, Inheritance and Polymorphism.
2. Summarize the fundamental features like Interfaces, Exceptions and Collections.
3. Correlate the advantages of Multi-threading.
4. Design interactive programs using Applets, AWT and Swings.
5. Develop real time applications using the features of Java.

UNIT - I

Object oriented thinking and Java Basics- Need for oop paradigm, summary of oop concepts, coping with complexity, abstraction mechanisms. History of Java, Java buzzwords, data types, variables, scope and lifetime of variables, arrays, operators, expressions, control statements, type conversion and casting, simple java program, concepts of classes, objects, constructors, methods, access control, this keyword, garbage collection, overloading methods and constructors, parameter passing, recursion, nested and inner classes, exploring String class.

UNIT - II

Inheritance, Packages and Interfaces – Hierarchical abstractions, Base class object, subclass, subtype, substitutability, forms of inheritance specialization, specification, construction, extension, limitation, combination, benefits of inheritance, costs of inheritance. Member access rules, super keyword uses, using final keyword with inheritance, polymorphism- method overriding, abstract classes, the Object class. Defining, Creating and Accessing a Package, Understanding CLASSPATH, importing packages, differences between classes and interfaces, defining an interface, implementing interface, applying interfaces, variables in interface and extending interfaces.

[illegible] [illegible] [illegible] [illegible] [illegible]

[illegible] [illegible]

UNIT - III

Exception handling and Multithreading-- Concepts of exception handling, benefits of exception handling, Termination or resumptive models, exception hierarchy, usage of try, catch, throw, throws and finally, built in exceptions, creating own exception subclasses. Differences between multithreading and multitasking, thread life cycle, creating threads, thread priorities, synchronizing threads, inter thread communication, thread groups, daemon threads.

UNIT - IV

Exploring String class, Object class, Exploring java.util package, Exploring java.io package

Event Handling: Events, Event sources, Event classes, Event Listeners, Delegation event model, handling mouse and keyboard events, Adapter classes. graphics, layout manager – layout manager types – border, grid, flow, card and grid bag.

UNIT - V

Swing – Introduction, limitations of AWT, MVC architecture, components, containers, exploring swing- JFrame and JComponent, JLabel, ImageIcon, JTextField, JButton, JCheckBox, JRadioButton, JList, JComboBox, Tabbed Panes, Scroll Panes, Trees, and Tables. Menu Basics, Menu related classes

JMenuBar, JMenu, JMenuItem, JCheck Box MenuItem, JRadio Button MenuItem, JSeperator. creating a popup menu

TEXT BOOKS:

1. Java the complete reference, 13th edition, Herbert schildt, Dr. Denny Coward, Mc Graw Hill.
2. Understanding OOP with Java, updated edition, T. Budd, Pearson education.

REFERENCE BOOKS:

1. An Introduction to programming and OO design using Java, J.Nino and F.A. Hosch, John Wiley & sons.
2. An Introduction to OOP, third edition, T. Budd, Pearson education.
3. Introduction to Java programming, Y. Daniel Liang, Pearson education.
4. An introduction to Java programming and object-oriented application development, R.A. Johnson- Thomson.
5. Core Java 2, Vol 1, Fundamentals, Cay.S. Horstmann and Gary Cornell, eighth Edition, Pearson Education.
6. Core Java 2, Vol 2, Advanced Features, Cay.S. Horstmann and Gary Cornell, eighth Edition, Pearson Education
7. Object Oriented Programming with Java, R.Buyya, S.T.Selvi, X.Chu, TMH.
8. Java and Object Orientation, an introduction, John Hunt, second edition, Springer.
9. Maurach's Beginning Java2 JDK 5, SPD.

[illegible] [illegible] [illegible] [illegible] D. Saikrup [illegible]

PCY307PC: JAVA PROGRAMMING LAB

B.Tech. II Year I Sem.

L	T	P	C
0	0	2	1

Course Objectives:

1. To write programs using abstract classes.
2. To write programs for solving real world problems using the java collection framework.
3. To write multithreaded programs.
4. To write GUI programs using swing controls in Java.
5. To introduce java compiler and eclipse platform.
6. To impart hands-on experience with java programming.

Course Outcomes:

1. Analyze a problem, identify and define the computing requirements appropriate to its solution using object-oriented programming concepts.
2. Design the applications using Inheritance, Polymorphism and Synchronization concepts.
3. Handle exceptions at Compile time and Run time.
4. Solve the real-world problems using Java Collection framework.
5. Develop GUI applications using Applets, AWT and Swings.

Note:

1. Use LINUX and MySQL for the Lab Experiments. Though not mandatory, encourage the use of the Eclipse platform.

2. The list suggests the minimum program set. Hence, the concerned staff is requested to add more problems to the list as needed.

TASK 1

Write java programs that implement the following

a. Class and object
b. Constructor
c. Parameterized constructor
d. Method overloading
e. Constructor overloading.

TASK 2

a. Write a Java program that checks whether a given string is a palindrome or not. Ex: MADAM is a palindrome.

b. Write a Java program for sorting a given list of names in ascending order.

c. Write a Java Program that reads a line of integers, and then displays each integer and the sum of all the integers (Use StringTokenizer class of java.util)

TASK 3

Write java programs that uses the following keywords

a) this b) super c) static d) final

[illegible] [illegible] [illegible] [illegible] D. Sai [illegible] [illegible]

TASK 4

a. Write a java program to implement method overriding

b. Write a java program to implement dynamic method dispatch.

c. Write a Java program to implement multiple inheritance.

d. Write a java program that uses access specifiers.

TASK 5

a. Write a Java program that reads a file name from the user, then displays information about whether the file exists, whether the file is readable, whether the file is writable, the type of file and the length of the file in bytes.

b. Write a Java program that reads a file and displays the file on the screen, with a line number before each line.

c. Write a Java program that displays the number of characters, lines and words in a text file.

TASK 6

a. Write a Java program for handling Checked Exceptions.

b. Write a Java program for handling Unchecked Exceptions.

TASK 7

a. Write a Java program that creates three threads. First thread displays "Good Morning" every one second, the second thread displays "Hello" every two seconds and the third thread displays "Welcome" every three seconds.

b. Write a Java program that correctly implements producer consumer problem using the concept of interthread communication.

TASK 8

Write a program illustrating following collections framework

a. ArrayList

b. Vector

c. Hash Table

d. Stack

TASK 9

a. Develop an applet that displays a simple message.

b. Develop an applet that receives an integer in one text field and compute its factorial value and return it in another text field, when the button named "Compute" is clicked.

c. Write a Java program that works as a simple calculator. Use a grid layout to arrange button for the digitsand for the +, -,*, % operations. Add a text field to display the result.

[illegible] [illegible] [illegible] [illegible] [illegible] [illegible] [illegible]

TASK 10

a. Write a Java program for handling mouse events.

b. Write a Java program for handling key events.

TASK 11

a. Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields Num1 and Num 2.

b. The division of Num1 and Num2 is displayed in the Result field when the Divide button is clicked. If Num1or Num2 were not an integer, the program would throw Number Format Exception. If Num2 were Zero, the program would throw an Arithmetic Exception and display the exception in a message dialog box.

TASK 12

a. Write a java program that simulates traffic light. The program lets the user select one of three lights: red, yellow or green. When a radio button is selected, the light is turned on, and only one light can be on at a time. No light is on when the program starts.

b. Write a Java program that allows the user to draw lines, rectangles and ovals.

c. Create a table in Table.txt file such that the first line in the file is the header, and the remaining lines correspond to rows in the table. The elements are separated by commas. Write a java program to display the table using JTable component.

TEXT BOOKS:

1. Java: The Complete Reference, 10th edition, Herbert Schildt, Mcgraw Hill.
2. Java Fundamentals - A Comprehensive introduction, Herbert Schildt and Dale skrien, TMH.
3. Java for programming, P.J. Dietel Pearson education (OR) Java: How to Program P.J. Dietel and H.M. Dietel, PHI

REFERENCE BOOKS:

1. Object Oriented Programming through java, P. Radha Krishna, Universities Press.
2. Thinking in Java, Bruce Eckel, Pearson Education
3. Programming in Java, S. Malhotra and S. Choudhary, Oxford University Press.

[illegible]

[illegible] [illegible] [illegible] [illegible] [illegible]

[illegible]

PCY310SD: DATA VISUALIZATION - R PROGRAMMING/ POWER BI/TABLEAU/GOOGLE CHART

B.Tech. II Year I Sem.

L	T	P	C
0	0	2	1

Course Objectives:

1. Effective use of Business Intelligence (BI) technology (Tableau) to apply data visualization
2. To discern patterns and relationships in the data.
3. To build Dashboard applications.
4. To communicate the results clearly and concisely.
5. To be able to work with different formats of data sets.

Course Outcomes:

1. Understand How to import data into Tableau.
2. Understand Tableau concepts of Dimensions and Measures.
3. Develop Programs and understand how to map Visual Layouts and Graphical Properties.
4. Create a Dashboard that links multiple visualizations.
5. Use graphical user interfaces to create Frames for providing solutions to real world problems.

Lab Problems:

1. Understanding Data, What is data, where to find data, Foundations for building Data Visualizations, Creating Your First visualization?
2. Getting started with Tableau Software using Data file formats, connecting your Data to Tableau, creating basic charts(line, bar charts, Tree maps),Using the Show me panel.
3. Tableau Calculations, Overview of SUM, AVR, and Aggregate features, Creating custom calculations and fields.
4. Applying new data calculations to your visualizations, Formatting Visualizations, Formatting Tools and Menus, Formatting specific parts of the view.
5. Editing and Formatting Axes, Manipulating Data in Tableau data, Pivoting Tableau data.
6. Structuring your data, Sorting and filtering Tableau data, Pivoting Tableau data.
7. Advanced Visualization Tools: Using Filters, Using the Detail panel, using the Size panels, customizing filters, Using and Customizing tooltips, Formatting your data with colors.
8. Creating Dashboards &amp; Storytelling, creating your first dashboard and Story, Design for different displays, adding interactivity to your Dashboard, Distributing &amp; Publishing your Visualization.
9. Tableau file types, publishing to Tableau Online, Sharing your visualizations, printing, and Exporting.
10. Creating custom charts, cyclical data and circular area charts, Dual Axis charts.
11. Visualize various data patterns taking any dataset from Kaggle.
12. Visualize data patterns using Google Charts by creating interactive Line, Bar, and Pie charts along with Dashboards using HTML and JavaScript.

[illegible] [illegible] [illegible] [illegible] [illegible] [illegible] [illegible] [illegible]

REFERENCE BOOKS:

1. Brett Powell, Microsoft Power BI cookbook, 2nd edition.
2. Roger D. Peng, R Programming for Data Science
3. Norman Matloff Cengage Learning India, The Art of R Programming.

WEB RESOURCE:

1. https://developers.google.com/chart/interactive/docs.

[illegible]

[illegible] [illegible] [illegible] D. Sai [illegible] [illegible]

[illegible]

PCY403PC: FORMAL LANGUAGES AND AUTOMATA THEORY

B.Tech. II Year II Sem.

L	T	P	C
3	0	0	3

Prerequisites: Mathematical Foundations

Course Objectives:

1. To provide introduction to some of the central ideas of theoretical computer science from the perspective of formal languages.
2. To introduce the fundamental concepts of formal languages, grammars and automata theory.
3. Classify machines by their power to recognize languages.
4. Employ finite state machines to solve problems in computing.
5. To understand deterministic and non-deterministic machines.
6. To understand the differences between decidability and undecidability.

Course Outcomes:

1. Understand the concept of abstract machines and their power to recognize the languages.
2. Demonstrate on regular expressions.
3. Employ finite state machines for modeling and solving computing problems.
4. Design context free grammars for formal languages.
5. Distinguish between decidability and undecidability.

UNIT – I:

Introduction to Finite Automata: Structural Representations, Automata and Complexity, the Central Concepts of Automata Theory – Alphabets, Strings, Languages, Problems.
Nondeterministic Finite Automata: Formal Definition, an application, Text Search, Finite Automata with Epsilon-Transitions.
Deterministic Finite Automata: Definition of DFA, How A DFA Process Strings, The language of DFA, Conversion of NFA with €-transitions to NFA without €-transitions. Conversion of NFA to DFA, Moore and Melay machines

UNIT – II:

Regular Expressions: Finite Automata and Regular Expressions, Applications of Regular Expressions, Algebraic Laws for Regular Expressions, Conversion of Finite Automata to Regular Expressions.
Pumping Lemma for Regular Languages: Statement of the pumping lemma, Applications of the Pumping Lemma. **Closure Properties of Regular Languages:** Closure properties of Regular languages, Decision Properties of Regular Languages, Equivalence and Minimization of Automata.

[illegible] [illegible] [illegible] [illegible] D. Sai Krup [illegible]

UNIT – III:

Context-Free Grammars: Definition of Context-Free Grammars, Derivations Using a Grammar, Leftmost and Rightmost Derivations, the Language of a Grammar, Sentential Forms, Parse Tress, Applications of Context-Free Grammars, Ambiguity in Grammars and Languages.

Push Down Automata: Definition of the Pushdown Automaton, the Languages of a PDA, Equivalence of PDA's and CFG's, Acceptance by final state, Acceptance by empty stack, Deterministic Pushdown Automata. From CFG to PDA, From PDA to CFG.

UNIT – IV:

Normal Forms for Context- Free Grammars: Eliminating useless symbols, Eliminating €-Productions. Chomsky Normal form Greibach Normal form.
Pumping Lemma for Context-Free Languages: Statement of pumping lemma, Applications.
Closure Properties of Context-Free Languages: Closure properties of CFL's, Decision Properties of CFL's Turing Machines: Introduction to Turing Machine, Formal Description, Instantaneous description, The language of a Turing machine.

UNIT – V:

Types of Turing machine: Turing machines and halting
Undecidability: Undecidability, A Language that is Not Recursively Enumerable, An Undecidable Problem that is RE, Undecidable Problems about Turing Machines, Recursive languages, Properties of recursive languages, Post's Correspondence Problem, Modified Post Correspondence problem, Other Undecidable Problems, Counter machines.

TEXT BOOKS:

1. Introduction to Automata Theory, Languages, and Computation, 3nd Edition, John E. Hopcroft, Rajeev Motwani, Jeffrey D. Ullman, Pearson Education.
2. Theory of Computer Science – Automata languages and computation, Mishra and Chandrasekaran, 2nd edition, PH

REFERENCE BOOKS:

1. Introduction to Languages and The Theory of Computation, John C Martin, TMH.
2. Introduction to Computer Theory, Daniel I.A. Cohen, John Wiley.
3. A Text book on Automata Theory, P. K. Srimani, Nasir S. F. B, Cambridge University Press.
4. Introduction to the Theory of Computation, Michael Sipser, 3rd edition, Cengage Learning.
5. Introduction to Formal languages Automata Theory and Computation Kamala Krithivasan, Rama R, Pearson.

[illegible] [illegible]

[illegible] [illegible] D. Saikeuf [illegible]

[illegible]

PCY405PC: MATHEMATICAL FOUNDATIONS OF CRYPTOGRAPHY

B.Tech. II Year II Sem.

L	T	P	C
3	0	0	3

Course Objectives:

1. Build a solid mathematical basis to understand foundations of cryptography
2. Formally understand the notions related to security authentication and privacy.
3. Provide a rigorous treatment of the emerging and key subject subarea of CSE - security.

Course Outcomes:

1. Discuss the basic functions of cryptography.
2. Demonstrate the different types of computational difficulty.
3. Apply the suitable tool for solving different zero knowledge protocols systems.
4. Use different encryption algorithms for encrypting confidential text.
5. Explain the digital signatures and message authentication.

UNIT - I

Basic functions of cryptography - Encryption Schemes, Digital Signatures, Fault Tolerant Protocols and Zero-Knowledge Proofs. The Computational Model: P, NP, and NP- Completeness, Probabilistic Polynomial Time, Non-Uniform Polynomial Time

UNIT – II

Computational Difficulty:

One-Way Functions Definitions, Strong One- Way Functions, Weak One-Way Functions, Universal One-Way Function, Trapdoor One-Way Permutations Computational Indistinguishability: Definition, Relation to Statistical Closeness, Indistinguishability by Repeated Experiments, Indistinguishability by Circuits

UNIT - III

Zero-Knowledge Proof Systems Zero:

Knowledge Proofs, Perfect and Computational Zero-Knowledge, An Example (Graph Isomorphism in PZK) Zero-Knowledge with Respect to Auxiliary Inputs

UNIT - IV

Encryption Schemes:

Private-Key versus Public-Key Schemes, The Syntax of Encryption Schemes, Semantic Security, Indistinguishability of Encryptions, Stream--Ciphers, Preliminaries: Block—Ciphers

UNIT – V

Digital Signatures and Message Authentication: Attacks and security, Variants Constructions of Message Authentication Schemes: Applying a pseudorandom function to the document

[illegible] [illegible] [illegible]

[illegible] [illegible] [illegible] [illegible]

[illegible]

TEXTBOOK:

1. Foundations of Cryptography (two volumes), Oded Goldreich, Cambridge university Press, 2004. (Indian print available).

REFERENCES:

1. Introduction to Modern Cryptography, J. Katz, Y. Lindell, Chapman Hall, USA 2007.
2. Modern cryptography - Theory and practice, Wen Bo Mao, Prentice Hall, USA, 2003 (Indian edition available)

[illegible] [illegible]

[illegible] [illegible] [illegible] [illegible]

[illegible]

PCY409PC: MATHEMATICAL FOUNDATIONS OF CRYPTOGRAPHY LAB

B.Tech. II Year II Sem.

L	T	P	C
0	0	2	1

Course Objectives:

1. Understand the basic concepts of cryptographic systems
2. Understand the various algorithms for how they are working in the systems.

Course Outcomes:

1. Implement basic encryption and decryption algorithms.
2. Demonstrate Digital Signature Generation and Verification and Block Cipher Modes.
3. Implement Zero-Knowledge Proof (ZKP) Concept, Discrete Logarithms and Simple One-Way Hash Function.
4. Compare Private-Key vs Public-Key Encryption algorithms in cryptography.
5. Implement a Digital Signature Scheme Using RSA and MAC.

EXPERIMENT-1: Implement Basic Encryption/Decryption Schemes

a. Implement Caesar cipher, Vigenère cipher, and XOR cipher.
b. Test encryption and decryption on text input.

EXPERIMENT-2: Demonstrate Digital Signature Generation and Verification

a. Use Python libraries (e.g., cryptography or pycryptodome) to sign a message with a private key and verify it using a public key.

EXPERIMENT-3: Simulate a Fault-Tolerant Protocol

a. Simulate a basic consensus protocol (like 2-phase commit) handling failure scenarios.

EXPERIMENT-4: Implement Zero-Knowledge Proof (ZKP) Concept

a. Simulate a ZKP for Graph Isomorphism using a challenge-response protocol.

EXPERIMENT-5: Implement a Simple One-Way Hash Function

a. Build a toy hash function and demonstrate its one-way property. Compare it to SHA-256.

EXPERIMENT-6: Test for Strong and Weak One-Way Functions

a. Write a program to simulate strong vs weak one-way functions. Use time complexity analysis to show difficulty of inversion.

EXPERIMENT-7: Demonstrate Trapdoor One-Way Permutations

a. Use RSA key generation to demonstrate how trapdoor functions work in public-key encryption.

EXPERIMENT-8: Demonstrate Computational Indistinguishability

a. Create two similar distributions and test whether a computational process can distinguish between them (simulate using coin tosses).

[illegible] [illegible] [illegible] [illegible] [illegible] [illegible]

EXPERIMENT-9: Implement a Zero-Knowledge Proof for Discrete Logarithms

a. Given a number $y=g^x \bmod p$, simulate ZKP to prove knowledge of xxx without revealing it.

EXPERIMENT-10: Implement Perfect and Computational ZKP

a. Demonstrate perfect ZKP using small problems like Sudoku validation without revealing the solution.

EXPERIMENT-11: Auxiliary Inputs in ZKP

a. Simulate a scenario where an adversary has auxiliary information and test how ZKP maintains secrecy.

EXPERIMENT-12: Compare Private-Key vs Public-Key Encryption

a. Implement AES (symmetric) and RSA (asymmetric) and compare their usage on messages.

EXPERIMENT-13: Demonstrate Block Cipher Modes

a. Implement ECB, CBC, and CFB modes for AES and show their impact on ciphertext for patterned plaintext.

EXPERIMENT-14: Implement a Digital Signature Scheme Using RSA

a. Generate keys, sign a message, and verify using the RSA algorithm.

EXPERIMENT-15: Construct a Message Authentication Code (MAC)

a. Use a pseudorandom function like HMAC-SHA256 to generate and verify MACs for file integrity.

TEXT BOOK:

1. Foundations of Cryptography (two volumes), Oded Goldreich, Cambridge university Press, 2004. (Indian print available).

REFERENCE BOOKS:

1. Introduction to Modern Cryptography, J. Katz, Y. Lindell, Chapman Hall, USA 2007.
2. Modern cryptography - Theory and practice, Wen Bo Mao, Prentice Hall, USA, 2003 (Indian edition available).

[illegible] [illegible]

[illegible] [illegible] [illegible]

[illegible]

PCY410PC: NODE JS/ REACT JS/ DJANGO, UI DESIGN - FLUTTER

B.Tech. II Year II Sem.

L	T	P	C
0	0	2	1

Prerequisites: Object Oriented Programming through Java, HTML Basics.

Course Objectives:

1. To implement responsive web pages using HTML, CSS3, Bootstrap, and JavaScript validation.
2. To develop full-stack applications with Node.js, Express, React.js, and secure them using JWT.
3. To build and integrate RESTful APIs using Node.js and Django with frontend CRUD operations.
4. To create Flutter apps with responsive layouts, state management, and custom widgets.
5. To enhance Flutter apps by integrating REST APIs and adding animations for better user experience.

Course Outcomes:

1. Design responsive web pages using HTML, CSS, Bootstrap, and JavaScript.
2. Develop backend applications using Java and Node.js.
3. Build single-page applications with React.js.
4. Create Flutter apps with custom widgets, layouts, and forms.
5. Integrate APIs, add animations, and perform UI testing in Flutter apps.

List of Experiments: Students need to implement the following experiments

1. Build a responsive website with registration, login, catalog, and cart pages using HTML, CSS3, Bootstrap, and JavaScript validation.
2. Create a react application for the student management system having registration, login, contact, about pages and implement routing to navigate through these pages.
3. Create a service in react that fetches the weather information from openweathermap.org and the display the current and historical weather information using graphical representation using chart.js.
4. Develop an express web application that can interact with REST API to perform CRUD operations on student data. (Use Postman).
5. Implement JWT authentication in Node.js to create secure endpoints.
6. Create a TODO application in react with necessary components and deploy it into github.
7. Integrate a React frontend with Node.js backend to perform CRUD operations on a shared dataset (e.g., MongoDB, MySQL.).
8. Build a Django REST API for managing student data, tested using Postman.

[illegible] [illegible] [illegible] [illegible] [illegible] [illegible] [illegible]

9. Install Flutter and Dart SDK; Write Dart programs on data types, control flow, Functions, Class & Objects and collections to understand syntax and features.
10. Create a Flutter app showcasing common widgets (Text, Image, Container, Card, ListView).
11. Design a responsive UI using Row, Column, Stack, media queries, and breakpoints.
12. Implement screen navigation using Navigator and named routes.
13. Use stateful & stateless widgets; manage state using Provider or setState.
14. Design a form in Flutter with input fields, validation, and error handling (e.g., student registration form).
15. Fetch and display data from a REST API (e.g., weather or student info) in Flutter.
16. Add basic animations (fade or slide) to a Flutter app for enhancing UI interactions

TEXT BOOK:

1. Marco L. Napoli, Beginning Flutter: A Hands-on Guide to App Development.

REFERENCE BOOKS:

1. Jon Duckett, Beginning HTML, XHTML, CSS, and JavaScript, Wrox Publications, 2010
2. Bryan Basham, Kathy Sierra and Bert Bates, Head First Servlets and JSP, O'Reilly Media, 2nd Edition, 2008.
3. Vasan Subramanian, Pro MERN Stack, Full Stack Web App Development with Mongo, Express, React, and Node, 2nd Edition, A Press.

[illegible] [illegible]

[illegible] [illegible] [illegible]

[illegible] [illegible]