Minor Courses offered by the Department of Computer Science

Year / Sem	Type of Course	Course Code	Title of the Course	Credits	Teaching Hours
1st Year (Semester-I & II)	MID-1	**CSCS102**	**Microprocessor and Assembly Language Programming**	4	5
	MID-2	**CSCS106**	**Microcontrollers Programming**	4	5
2nd Year (Semester-III & IV)	MID-3	**CSCS203**	**System Software**	4	5
	MID-4	**CSC209**	**Embedded Application Development**	4	5
3rd Year (Semester-V & VI)	MID-5	**CSCS304**	**Theory of Computation**	4	5
	MID-6	**CSCS310/ CSCS311**	**UNIX System Programming / Network Programming**	4	5
4th Year (Semester-VII)	MID-7	**CSCS404/ CSCS405**	**Artificial Intelligence / Compiler Design**	4	5
	MID-8	**CSCS406/ CSCS407**	**Cyber Security / Internet of Things**	4	5

Year	I	Course Code: CSCS102	Credits	4
Sem.	I	Course Title: Microprocessor & Assembly Language Programming	Hours	75
			Category	C
Course Prerequisites, if any	• Number Systems (binary, octal, hexadecimal) and their conversions • Boolean Algebra, logic gates, flip-flops and registers • Concepts in Combinational and Sequential logic			
Internal Assessment Marks: 25	End Semester Marks: 75	Duration of ESA (Theory): 03 hrs. Duration of ESA (Practical): 03 hrs.		
Course Outcomes	• Learn the architecture & organization of 8085 Microprocessor • Understand and classify the instruction set of the 8085Microprocessor • Apply the memory & I/O Interfacing with 8085 Microprocessor • Analyze the architecture and operation of Programmable Interface • Create applications to interface various peripheral IC's with Intel 8085 microprocessor			

Unit No.	Course Content	Hours
	Theory Component	
Unit I	**Introduction to Microprocessors & 8085 Assembly Language Programming** Microprocessors – Instruction set and computer languages – 8085 programming model – Instruction classification – Instruction – Data format and storage – Execute a simple program – 8085 Instruction Set	9
Unit II	**8085 Microprocessor architecture** Microprocessor Architecture and its operations – Memory – I/O Devices, 8085 MPU – 8085 based microcomputer – memory interfacing – 8155 memory segment Interfacing – Interfacing I/O devices: Basics – Interfacing input and output devices – memory mapped I/O	9
Unit III	**Programming 8085** Instruction Set of 8085 – Data Transfer – arithmetic – Logic – Branch – Writing ALP and Debugging programs – Looping – Counting and Indexing – 16-bit Arithmetic instructions – Logic operations – Counters and Time Delay	9
Unit IV	**Interfacing I/O Devices** Stack and subroutines – Restart – Conditional call and Return instruction – Advanced subroutine concepts – Code conversion – BCD Arithmetic and 16-bit operations – BCD-Binary conversion – Binary to BCD conversion – BCD to seven segment LED code conversion – Binary to ASCII and ASCII to binary conversion – BCD addition and subtraction	9
Unit V	**Interfacing Peripheral (I/O) and Applications** Interrupts: 8085 Interrupt – RST instructions – Software and Hardware interrupt – multiple Interrupts and Priorities – 8085 Vectored Interrupts – Restart as Software Instructions – 8155 – Multipurpose programmable Device – 8279 Programmable Keyboard/Display Interface – 8255 Programmable peripheral Interface	9

Practical Component		
Exercises	1. Assembly Language Programming for Arithmetic Operations like Addition, Subtraction, Multiplication and Division on 8, 16-bit data 2. Assembly Language Programming for different logical operations 3. Assembly Language Programming for code conversions 4. Assembly Language Programming for sorting 5. Assembly Language Programming for Searching 6. Assembly Language Programming for memory block transfer 7. Assembly Language Programming using subroutines 8. Assembly Language Programming using counters and time delay	30
Recommended Learning Resources		
Print Resources	1. Ramesh S. Gaonkar, "Microprocessor – Architecture, Programming and Applications with the 8085", Penram International Publisher, Sixth Edition, 2013. 2. Douglas V. Hall, "Microprocessors and Interfacing", Tata McGraw Hill publications, Third Edition, 2017.	
Syllabus Design: Dr. M. Sathya, Assistant Professor, PUDoCS		

<table>
<tr><td>Year</td><td>I</td><td colspan="2" rowspan="2">Course Code: CSCS106
Course Title: Microcontrollers Programming</td><td>Credits</td><td>4</td></tr>
<tr><td>Sem.</td><td>II</td><td>Hours</td><td>75</td></tr>
<tr><td colspan="4"></td><td>Category</td><td>C</td></tr>
<tr><td>Course Prerequisites, if any</td><td colspan="5">• Digital Logic Fundamentals
• Microprocessors
• Assembly Language Programming</td></tr>
<tr><td>Internal Assessment Marks: 25</td><td colspan="2">End Semester Marks: 75</td><td colspan="3">Duration of ESA (Theory): 03 hrs.
Duration of ESA (Practical): 03 hrs.</td></tr>
<tr><td>Course Outcomes</td><td colspan="5">• Learn the fundamentals of Microcontrollers
• Understand the internal design of 8051 microcontroller along with the features and their programming
• Analyze the on-chip peripherals of microcontrollers
• Design different interfacing applications using microcontrollers and peripherals
• Build systems using microcontrollers for real time applications</td></tr>
<tr><td>Unit No.</td><td colspan="3">Course Content</td><td colspan="2">Hours</td></tr>
<tr><td colspan="6">Theory Component</td></tr>
<tr><td>Unit I</td><td colspan="3">Microprocessors and Microcontrollers
Microprocessors vs Microcontrollers – 8051 Architecture – Input/Output Pins – Ports – External Memory – Counter and Timers – Serial Data I/O – Interrupts</td><td colspan="2">9</td></tr>
<tr><td>Unit II</td><td colspan="3">Programming 8051
Addressing Modes – External Data Moves – Code Memory Read-Only Data Moves – PUSH and POP Opcodes – Data Exchanges – Logical Operations – Arithmetic Operations – Jump and Call Opcodes</td><td colspan="2">9</td></tr>
<tr><td>Unit III</td><td colspan="3">8051 Microcontroller Design
Microcontroller Specification – Design – Testing – Timing Subroutines – Lookup Tables for 8051 – Serial Data Transmission</td><td colspan="2">9</td></tr>
<tr><td>Unit IV</td><td colspan="3">Applications
Keyboards – Displays – Pulse Measurement – D/A and A/D Conversions – Multiple Interrupts</td><td colspan="2">9</td></tr>
<tr><td>Unit V</td><td colspan="3">Serial Data Communication
Network Configurations – 8051 Data Communication Modes</td><td colspan="2">9</td></tr>
<tr><td colspan="6">Practical Component</td></tr>
<tr><td>Exercises</td><td colspan="3">1. Blinking LED
2. Digital Counter with Seven-Segment Display
3. Analog-to-Digital Conversion (ADC)
4. UART Communication
5. Timer Interrupt - Using a timer interrupt to perform a task at regular intervals
6. External Interrupt
7. Temperature Sensor (DS18B20) Interface
8. Matrix Keypad Interface
9. LCD Display Interface
10. Traffic Light Controller</td><td colspan="2">30</td></tr>
<tr><td colspan="6">Recommended Learning Resources</td></tr>
</table>

Print Resources	1. Kenneth J. Ayala, “The 8051 Microcontroller Architecture, Programming, and Applications”, Delmar Cengage Learning, Third Edition, 2004. 2. Martin Bates, “PIC Microcontrollers - An Introduction to Microelectronics”, Third Edition, Newnes, Elsevier, 2011. 3. Hubert Henry Ward, “C Programming for the PIC Microcontroller-Demystify Coding with Embedded Programming”, Apress, UK, 2020. https://doi.org/10.1007/978-1-4842-5525-4
Syllabus Design: Dr. M. Sathya, Assistant Professor, PUDoCS	

Year	II	Course Code: CSCS203 Course Title: System Software	Credits	4
Sem.	III		Hours	75
			Category	C
Course Prerequisites, if any	• Knowledge of digital logic design • Introductory knowledge in problem Solving			
Internal Assessment Marks: 25	End Semester Marks: 75	Duration of ESA (Theory): 03 hrs. Duration of ESA (Practical): 03 hrs.		
Course Outcomes	• Understand basic computer architecture via Simplified Instructional Computer (SIC) • Analyze differences in assemblers and machine features • Apply dynamic linking and bootstrap loaders in program preparation • Design macros demonstrating machine features • Apply machine-independent compiler features in design			
Unit No.	**Course Content**		**Hours**	
Theory Component				
Unit I	**Introduction** System Software and Machine Architecture – Simplified Instructional Computer (SIC) – Traditional (CISC) Machines – RISC Machines		9	
Unit II	**Assemblers** Basic Assembler Functions – Machine Dependent and Machine Independent Assembler Features – One-Pass Assemblers – Multi Pass Assemblers – MASM assembler – SPARC assembler		9	
Unit III	**Loaders and Linkers** Basic Loader Functions – Machine Dependent and Machine Independent Loader Features – Linkage Editors – Dynamic Linking - Bootstrap Loaders		9	
Unit IV	**Macro Processors** Basic Macro Processor Functions – Machine Dependent and Machine Independent Macro Processor Features – Macro Processor Design Options		9	
Unit V	**Compilers** Basic Compiler Functions – Machine-Dependent Compiler Features – Machine Independent Compiler Features – Compiler Design Options – YACC		9	
Practical Component				
Exercises	1. Simulate a simple arithmetic operation (e.g., addition, subtraction) in both a CISC-like and RISC-like manner. The CISC simulation should perform the operation in a single step, while the RISC simulation should break it down into simpler steps 2. Design a program that translates a small set of assembly-like instructions (define your simple instruction set) into a simulated machine code. Your program should handle basic operations like load, store, add, and subtract 3. Design a program that simulates the basic functions of a linker and loader for a simplified computational system 4. Implement a simple macro processor that allows for the definition and expansion of macros within a text file. The macros should perform simple text replacement or predefined operations (like incrementing a number)		30	

	5. Design and implement a simple arithmetic expression evaluator using YACC. The evaluator should be capable of handling basic arithmetic operations (+, -, *, /) and correctly respects the standard mathematical precedence of operations and handles parentheses to alter the precedence order	
Recommended Learning Resources		
Print Resources	1. Leland L. Beck, D. Manjula "System Software – An Introduction to Systems Programming", Third Edition, Pearson India, 2007. 2. Alfred V. Aho, Monica S. Lam, Ravi Sethi, Jeffrey D. Ullman, "Compilers: Principles, Techniques, and Tools", Second Edition, Pearson Addison Wesley, 2023.	
Syllabus Design: Dr. M. Sathya, Assistant Professor, PUDoCS		

Year	II	Course Code: CSCS209 Course Title: Embedded Application Development	Credits	4
Sem.	IV		Hours	75
			Category	C
Course Prerequisites, if any	• Microprocessor & Microcontrollers introduction • Assembly Language Programming • Operating System and Computer Organization Concepts			
Internal Assessment Marks: 25	End Semester Marks: 75	Duration of ESA (Theory): 03 hrs. Duration of ESA (Practical): 03 hrs.		
Course Outcomes	• Understand the basics of Embedded Systems • Appreciate the application domains of Embedded Systems • Gain proficiency in programming embedded systems • Explore interfacing techniques for sensors, actuators, and other peripheral devices commonly used in embedded applications • Develop skills in designing, implementing, and debugging embedded software			

Unit No.	Course Content	Hours
	Theory Component	
Unit I	**Introduction** Embedded Systems and General-purpose Computer Systems – History – Classifications – Applications – Purpose of Embedded Systems – Characteristics and Quality Attributes	9
Unit II	**Embedded Systems** Application specific – washing machine – domain specific – automotive Embedded Hardware: Memory – I/O – Interrupt – Processors – External peripherals Peripherals: Control and Status Registers – Device Driver – Timer Driver – Watchdog Timers	9
Unit III	**Microcontrollers** Microcontrollers and Embedded processors – Overview of 8051 family. 8051 hardware – I/O pins – Ports – Circuits – External Memory Programming: Data Types – I/O Programming – Logic operations – Data conversion Programs	9
Unit IV	**Designing Embedded System with 8051 Microcontroller** Factors to be considered in selecting a controller – 8051 Microcontroller – Designing with 8051 Programming: Structure of embedded program – infinite loop – compiling, linking & debugging	9
Unit V	**Real Time Operating System (RTOS)** Operating system basics – Types of OS – Real-Time Characteristics – Selection Process of an RTOS Design and Development: Embedded system development Environment – IDE – types of file generated, disassembler – de-compiler – simulator – emulator and debugging, embedded product development life-cycle, trends in embedded industry	9
	Practical Component	
Exercises	1. Configure timer control registers of 8051 and develop a program to generate given time delay 2. Port I/O: Use one of the four ports of 8051 for O/P 3. interfaced to eight LED's. Simulate binary counter (8 bit) on LED's 4. Serial I/O: Configure 8051 serial port for asynchronous serial communication with serial port of PC exchange text messages	30

	to PC and display on PC screen. Signify end of message by carriage return 5. Interface 8051 with D/A converter and generate square wave of given frequency on oscilloscope 6. Interface the microcontroller with external devices (e.g., sensors, displays, or other microcontrollers) using serial communication. Implement simple data exchange protocols and verify communication 7. Generate PWM signals to control the brightness of LEDs or the speed of a motor. Experiment with different duty cycles and frequencies 8. Write programs to store and retrieve data from non-volatile memory (e.g., EEPROM or Flash). Implement dynamic memory allocation techniques using RAM	
Recommended Learning Resources		
Print Resources	1. Shibu K V, "Introduction to Embedded Systems" Second Edition, Tata McGraw Hill, 2017. 2. Rajkamal, “Embedded Systems - Architecture, Programming and Design", Third Edition, McGraw Hill Education, 2008.	
Syllabus Design: Dr. S.K.V. Jayakumar, Professor, PUDoCS		

Year	III	Course Code: CSCS304 Course Title: Theory of Computation	Credits	4
Sem.	V		Hours	75
			Category	A
Course Prerequisites, if any	• Knowledge in Mathematics for Computer Science			
Internal Assessment Marks: 25	End Semester Marks: 75	Duration of ESA (Theory): 03 hrs.		
Course Outcomes	• Understand foundational concepts of formal languages • Apply regular expressions to create DFA for lexical analyzers • Analyze equivalence and transformations between NFA, DFA, and TG • Evaluate context-free grammars and limitations of regular grammars • Design models using PDA			
Unit No.	**Course Content**		**Hours**	
Theory Component				
Unit I	**Languages** Alphabets – String – Language – Basic Operations on Language – Concatenation – Union – Kleene Star		15	
Unit II	**Regular Expressions and Finite Automata** Regular expressions – Deterministic finite automata (DFA)		15	
Unit III	**Regular Languages** Non-Deterministic Finite Automata (NFA) – Relationship Between NFA and DFA – Transition Graphs (TG) – Properties of Regular Languages – The Relationship Between Regular Languages and Finite Automata – Kleene's Theorem		15	
Unit IV	**Non-Regular Languages and Context Free Grammars** Pumping Lemma for Regular Grammars – Context-Free Grammars (CFG)		15	
Unit V	**PDA and Context-Free Languages (CFL)** Deterministic And Non-Deterministic Pushdown Automata (PDA) – Parse Trees – Leftmost Derivation – Pumping Lemma for CFL – Properties Of CFL		15	
Practical Component				
-	-		-	
Recommended Learning Resources				
Print Resources	1. Cohen, D. I. A, "Introduction to Computer Theory", Second Edition, Wiley India, 2011. 2. Lewis, H.R. & Papadimitriou, H. R., "Elements of the Theory of Computation", Second Edition, Prentice Hall of India (PHI), 2015.			
Syllabus Design: Dr. M. Sathya, Assistant Professor, PUDoCS				

Year	III	Course Code: CSCS310 Course Title: Unix System Programming	Credits	4
Sem.	VI		Hours	75
			Category	C
Course Prerequisites, if any	• Computer Organization and Architecture • Operating System			
Internal Assessment Marks: 25	End Semester Marks: 75	Duration of ESA (Theory): 03 hrs. Duration of ESA (Practical): 03 hrs.		
Course Outcomes	• Understand Unix history, features, and system architecture • Manage files, directories, processes, and memory • Implement IPC with shared memory and semaphores • Develop network applications using socket programming • Write and execute shell scripts for text and pattern manipulation			
Unit No.	**Course Content**		**Hours**	
Theory Component				
Unit I	**Introduction** Introduction to Unix – History – Salient features of Unix – Unix System Architecture – Unix Programming Environment – Unix Process		9	
Unit II	**Standard I/O, Process and Memory Management** File Management: File input/output – Directory related System Calls – Process Management Processes: Creation – Execution – Termination – Process States – Process Control – Process groups – Thread – Memory Management		9	
Unit III	**Inter-Process Communication** Introduction to IPC – Shared Memory: Creating Shared Memory – Controlling Shared memory Segment – Process Synchronization: Semaphore		9	
Unit IV	**Socket Programming** Socket – Types of Sockets – Socket Data Structure – System Calls – I/O Models – Name and Address Conversion – Resource records		9	
Unit V	**Tools and Programming** Shell Scripting – Shell Scripting Operations – Text Manipulation – Pattern Matching – Text Transformation		9	
Practical Component				
Exercises	1. Perform operations like file creation, deletion, copying, moving, listing directory contents 2. Write a shell script that takes a directory name as an argument and lists all files and directories inside it 3. Write programs using 'fork', 'exec', and 'Wait' system calls to create processes 4. Create a program that uses unnamed pipes for communication between a parent and its child process 5. Write a simple client-server application using TCP sockets where the client sends a message to the server, and the server echoes it back 6. Implement a program that manipulates file permissions, accesses file metadata (like inode information), and performs file locking 7. Develop a simple shell that can interpret commands, launch programs, and support basic piping and redirection		30	

Recommended Learning Resources	
Print Resources	1. Vineeta khemchandani, Dappan Anand, Mishra, Sandeep Harit, "Unix Programming", BPB Online, 2022.
Syllabus Design: Dr. S. K. V. Jayakumar, Professor, PUDoCS	

Year	III	Course Code: CSCS311 Course Title: Network Programming	Credits	4
Sem.	VI		Hours	75
			Category	C

Course Prerequisites, if any	• Computer Networking Fundamentals • Programming Languages	
Internal Assessment Marks: 25	End Semester Marks: 75	Duration of ESA (Theory): 03 hrs. Duration of ESA (Practical): 03 hrs.
Course Outcomes	• Understand client-server networking and socket API • Learn server architectures: single-threaded, multithreaded, and async servers • Implement message queues, caching, and HTTP handling • Understand various networking protocols (TCP, UDP, POP, IMAP, etc.) • Explore case studies using Cisco Packet Tracer, Network Simulator 2, and GNS3	

Unit No.	Course Content	Hours
	Theory Component	
Unit I	**Introduction** Client Server Networking – UDP – TCP – DNS – Client-server Model – Socket API – Socket Addresses	9
Unit II	**Architecture of Servers** Data and Errors on Internet: Strings and bytes – SSL/TLS – Architecture of Server – Single Threaded Server – Multithreaded Servers – Async Servers	9
Unit III	**Message Queues and Caches** Memory Caching - Hashing and Sharding - Message Queues – HTTP Client – Server Handling HTTP – World Wide Web – SMTP	9
Unit IV	**Protocols** TCP – UDP – POP – IMAP – IPV4 – IpV6 – BGP – Telnet – SSH – FTP – RPC	9
Unit V	**Case Studies** Cisco Packet Tracer – Network Simulator 2 – GNS3	9
	Practical Component	
Exercises	1. a) Study of different types of network cables and practically implement cross wired cable and straight through cable using clamping tool b) Study of network devices and network IP in detail 2. Study of network IP and practically connect the computers in LAN a) Study of basic network command and network configuration commands b) Configure a network topology using CPT 3. Configure a network using Distance vector/Link state routing protocol 4. Simulation of Sliding Window Protocol 5. Half Duplex Chat Using UDP 6. Full Duplex Chat Using TCP/IP	30
	Recommended Learning Resources	
References	1. John Galbraith, "Network Programming in Python: The Basic", First Edition, BPB Publications, 2022.	

Syllabus Design: Dr. S.K.V. Jayakumar, Professor, PUDoCS

Year	IV	Course Code: CSCS404 Course Title: Artificial Intelligence	Credits	4
Sem.	VII		Hours	75
			Category	C
Course Prerequisites, if any	Basic Programming Skills			
Internal Assessment Marks: 25	End Semester Marks: 75	Duration of ESA (Theory): 03 hrs. Duration of ESA (Practical): 03 hrs.		
Course Outcomes	● Familiarize with the diverse traits of a problem-solving agent ● Explore methods for tackling problems amidst different constraints ● Implement AI techniques in various applications ● Grasp the distinct models of learning ● Develop an expert system			

Unit No.	Course Content	Hours
	Theory Component	
Unit I	**Introduction** Foundation and History of AI – Intelligent Agents – Agents and Environments – The Concept of Rationality – Nature of Environments – Structure of Agents – Problem Solving Agents – Examples	9
Unit II	**Searching** Searching for Solutions, Uniformed Search Strategies – Heuristics Search Strategies – Local Search Algorithms and Optimization Problems – Hill Climbing- Simulated Annealing – Local Beam Search – Genetic Algorithms – Optimal Decisions in Games – Alpha-Beta Pruning	9
Unit III	**Agents** Logical Agents – Knowledge-Based Agents – The Wumpus World – Logic – Propositional Logic – Propositional Theorem Proving – Effective Propositional Model Checking – Agents Based on Propositional Logic	9
Unit IV	**First Order Logic** Introduction – Syntax and Semantics – Inference – Propositional Vs First-Order Inference – Unification and Lifting – Forward Chaining – Backward Chaining – Resolution	9
Unit V	**Learning** Forms of Learning – Supervised Learning – Learning Decision Trees – Hypothesis – Theory of Learning – Prolog – Programs – Data Objects	9
	Practical Component	
Exercises	1. Implement Breadth First Search 2. Implement Depth First Search 3. Implement Tic-Tac-Toe game 4. Implement 8-Puzzle problem 5. Implement Water-Jug problem 6. Implement Monkey Banana Problem 7. Implement Alpha-Beta Pruning 8. Develop an expert system using Prolog	30
	Recommended Learning Resources	

Print Resources	1. S. Russell and P. Norvig, “Artificial Intelligence – A Modern Approach”, Pearson Education, Third Edition, 2010. 2. Max Bramer, Logic Programming with Prolog, Springer, 2005.
Syllabus Design: Dr. P. Shanthi Bala, Professor, PUDoCS	

Year	IV	Course Code: CSCS405 Course Title: Compiler Design	Credits	4
Sem.	VII		Hours	75
			Category	C
Course Prerequisites, if any	• Knowledge in any programming language such as Java or C • Knowledge in Assembly Programming, Basic Arithmetic, and Data Structures			
Internal Assessment Marks: 25	End Semester Marks: 75	Duration of ESA (Theory): 03 hrs. Duration of ESA (Practical): 03 hrs.		
Course Outcomes	• Understand the functional components of compilers • Apply knowledge of lexical analysis by implementing scanners • Analyze and differentiate between various parsing techniques • Evaluate and integrate syntax-directed definitions and type checking in compiler construction • Design and create components of a runtime environment and a code generator			
Unit No.	**Course Content**		**Hours**	
Theory Component				
Unit I	**Introduction** Language Processors – Structure of a Compiler – Evolution of Programming Languages – Applications of Compiler Technology – Tool based Approach to Compiler Construction		9	
Unit II	**Lexical Analysis** Interface with Input – Parser and Symbol Table – Tokens, Patterns and Lexemes – Difficulties in Lexical Analysis – Error Reporting – Regular Definitions – Transition Diagrams – Lex		9	
Unit III	**Syntax Analysis** CFGs – Ambiguity – Associatively – Precedence – Top-Down Parsing – Recursive – Descent Parsing – FIRST and FOLLOW – LL (1) Grammars – Predictive Parsing – Bottom-Up Parsing – LR Parsing		9	
Unit IV	**Syntax Directed Definitions** Inherited and Synthesized Attributes – Dependency Graphs – Ordering the Evaluation of Attributes – L and S Attributed Definitions – Type Checking		9	
Unit V	**Run Time Environments** Storage Organization – Stack Allocation of Space – Parameter Passing – Symbol Table – Dynamic Storage Allocation **Code Generation** Issues in the Design of a Code Generator – Addresses in the Target Code – Basic Blocks and Flow Graphs – Optimization of Basic Blocks – Code Generator – Peep Hole Optimization		9	
Practical Component				
Exercises	1. Using Lex or a similar tool, implement a lexical analyzer for a simple programming language or a subset of an existing language 2. Write a recursive descent parser in a programming language of your choice for a simple arithmetic expression grammar that includes addition, subtraction, multiplication, division, and parentheses. Ensure your parser handles operator precedence correctly 3. Implement a program that builds a parse tree for an expression and evaluates its attributes according to your definitions			

	4. Implement a simple type checker that can handle basic data types (integers, floats), type conversions, and function/operator overloading 5. Create a simulation of a runtime environment that demonstrates stack allocation, parameter passing, and dynamic storage allocation 6. Given a set of basic blocks, implement an optimization routine that applies peephole optimization techniques	
Recommended Learning Resources		
Print Resources	1. Alfred V. Aho, Monica S. Lam, Ravi Sethi and Jeffrey D. Ullman, "Compilers: Principles, Techniques, & Tools", Second Edition, Pearson Addison Wesley, 2023. 2. Allen I. Holub, "Compiler Design in C", First Edition, Pearson India, 2015.	
Syllabus Design: Dr. M. Sathya, Assistant Professor, PUDoCS		

Year	IV	Course Code: CSCS406 Course Title: Cyber Security	Credits	4
Sem.	VII		Hours	75
			Category	C
Course Prerequisites, if any	Basic Knowledge of Programming and Information Security Principles			
Internal Assessment Marks: 25	End Semester Marks: 75	Duration of ESA (Theory): 03 hrs. Duration of ESA (Practical): 03 hrs.		
Course Outcomes	• Learn the definitions and categories of cybercrimes • Comprehend the tools and techniques employed in cybercrimes • Examine the legal frameworks surrounding cybercrime legislation • Assess the effectiveness of cybersecurity measures • Examine current cyber threats and vulnerabilities			
Unit No.	**Course Content**		**Hours**	
Theory Component				
Unit I	**Introduction to Cybercrime** Cybercrime Definition – Cybercrime and Information Security – Classification of Cybercrimes – Email Spoofing, Spamming, Data Diddling, Web Jacking, Hacking, Password Sniffing – Categories of Cybercrime – Passive attack – Active attack – Reconnaissance		9	
Unit II	**Tools and Methods used in Cybercrime** Cyberstalking – Cybercafe and Cybercrimes – Botnets – Proxy Servers and Anonymizers – Password Cracking – Keyloggers and Spyware – DoS and DDoS attacks – Virus and Worms – Trojan horses and Backdoors – SQL injection – Steganography		9	
Unit III	**Mobile and Wireless Devices** Proliferation of Mobile and Wireless Devices – Trends in Mobility – Security Challenges Posed by Mobile Devices – Authentication Service Security – Attacks on mobiles and cellphones – Credits Card Frauds in mobile and Wireless Computing Era – Organizational measures for Handling Mobile		9	
Unit IV	**Phishing and Identify Theft** Buffer Overflow – Phishing: Methods of Phishing, Phishing Techniques, Spear Phishing, Types of Phishing Scams, Phishing Tool Kits and Spy Phishing, Phishing Countermeasures – Identify Theft (ID Theft): Types of Identify Theft, Techniques of ID theft – ID Theft Counter Measures – Personally Identifiable Information		9	
Unit V	**Cybercrime and Cyber Security Legal Perspectives** The Indian IT Act – Challenges to Indian Law and Cybercrime Scenario in India – Digital Signatures and The Indian IT Act – Amendments to the Indian IT Act – Cybercrime and Punishment		9	
Practical Component				
Exercises	1. Create a simple program that encrypts and decrypts a text message using a basic cipher (e.g., Caesar cipher). Demonstrate encryption of a given plaintext and then decryption back to the original text 2. Simulate a basic SQL injection attack against a sample web application. Demonstrate how unauthorized access to data can be obtained through poorly sanitized input fields. Show the effect of the attack and suggest mitigation strategies		30	

	3. Use a password cracking tool on a set of hashed passwords. Demonstrate the process of cracking by identifying weak passwords from the hash values. Discuss the importance of strong password policies 4. Set up and configure a basic firewall on a network or computer system. Demonstrate how to block and allow specific traffic types. Test the firewall setup by attempting to access the protected resources with varying types of network traffic 5. Analyze a set of emails to identify characteristics of phishing attempts. Explain the indicators of phishing and suggest methods for verifying the authenticity of suspicious emails. Discuss the impact of phishing attacks and preventive measures 6. Create virtualized network environments with cybersecurity simulation software, guiding participants through defense strategies against various cyber-attacks 7. Equip participants with forensic analysis tools, presenting simulated cyber-attack scenarios to investigate, analyze evidence, and prepare forensic reports	
Recommended Learning Resources		
Print Resources	1. Nina Godbole and Sumit Belapure, “Cyber Security: Understanding Cyber Crimes, Computer Forensics and Legal Perspectives”, First Edition, Wiley India Pvt. Ltd., 2011. 2. Anand Shinde, “Introduction to Cyber Security: Guide to the World of Cyber Security”, First Edition, Notion Press, 2021.	
Syllabus Design: Dr. M.Sathya, Assistant Professor, PUDoCS		

Year	IV	Course Code: CSCS407 Course Title: Internet of Things	Credits	4
Sem.	VII		Hours	75
			Category	C
Course Prerequisites, if any	Basic knowledge of programming and networking			
Internal Assessment Marks: 25	End Semester Marks: 75	Duration of ESA (Theory): 03 hrs. Duration of ESA (Practical): 03 hrs.		
Course Outcomes	• Understand IoT fundamentals, including design, protocols, and technologies • Explore domain-specific applications such as home automation and industry • Learn about M2M applications and system management • Develop IoT systems using platforms like Raspberry Pi • Manage IoT server and cloud infrastructure, focusing on security			

Unit No.	Course Content	Hours
	Theory Component	
Unit I	**Introduction** Definition, Characteristics of IoT, Physical Design of IoT, Protocols, Logical Design of IoT, IoT Enabled Technologies, IoT Levels and Templates	9
Unit II	**Domain Specific IoT Applications** Home Automation, City, Environment, Energy, Retail, Logistics, Agriculture, Industry, health and Lifestyle	9
Unit III	**M2M and IoT System Management** M2M Applications, Software Defined Networks, Network Function Virtualization. Need for IoT System Management, Simple Network Management Protocol, IoT System Management with NETCOZF-YANG	9
Unit IV	**Developing IoT Systems** IoT Platforms Design Methodology, Steps for IoT Design, Case Study on IoT System for Weather Monitoring, Introduction to Raspberry PI, Interfaces (serial, SPI, I2C), Programming Raspberry Pi, IoT Devices	9
Unit V	**IoT Server and Cloud Management** Introduction to Cloud Storage Models and Communication APIs, Webserver – Web Server for IoT, Cloud for IoT, Security Management in an IoT System	9
	Practical Component	
Exercisers	1. Identify and list different types of IoT devices and their functionalities 2. Sketch a physical design for a home automation system using IoT devices 3. Compare and contrast different IoT protocols such as MQTT, CoAP, and HTTP 4. Set up a basic communication protocol between two IoT devices using MQTT 5. Discuss the role of cloud computing in enabling IoT solutions 6. Implement a simulation of the home automation system using IoT platforms like Arduino or Raspberry Pi 7. Investigate and compare M2M applications in industries such as healthcare and logistics 8. Program a Raspberry Pi to collect weather data from sensors and display it on a web server	30

	9. Explore different cloud storage models (e.g., public, private, hybrid) and their suitability for IoT applications 10. Implement security measures such as encryption and authentication in an IoT system using cloud-based services	
Recommended Learning Resources		
Print Resources	1. Arshdeep Bahga and Vijay Madisetti, "Internet of Things - A Hands-on Approach", First Edition, Orient Blackswan Private Limited, 2015. 2. Rajesh Singh, Anita Gehlot, Bhupendra Singh, Sushabhan Choudhury, " Internet of Things (IoT) Enabled Automation in Agriculture", Second Edition, CRC Press, 2022.	
Syllabus Design: Dr. T. Vengattaraman, Associate Professor, PUDoCS		