Course Type: - Major
Semester: - 6th
Paper Title: - **CRYPTOGRAPHY AND NETWORK SECURITY**
Paper Code: - CAPC1622M
Credit Weightage: - THEORY -04; PRACTICALS- 02
Batch: - 2023

Course Objective:

- Explain the objectives of information security.
- Explain the importance and application of each of confidentiality, integrity, authentication and availability.
- Understand various cryptographic algorithms and basic categories of threats to computers and networks.
- Describe public-key cryptosystem and enhancements made to IPv4 by IPSec.
- Understand Intrusions and intrusion detection.
- Discuss the fundamental ideas of public-key cryptography.
- Discuss Web security and Firewalls.

Course Outcomes:

- Student will be able to understand basic cryptographic algorithms, message and web authentication and security issues.
- Ability to identify information system requirements for both of them such as client and server.
- Ability to understand the current legal issues towards information security.

UNIT – I

SECURITY CONCEPTS: Introduction, The need for security, Security approaches, Principles of security, Types of Security attacks, Security services, Security Mechanisms, A model for Network Security.
Cryptography Concepts and Techniques: Introduction, plain text and cipher text, substitution techniques, transposition techniques, encryption and decryption, symmetric and asymmetric key cryptography, steganography, key range and key size, possible types of attacks.

UNIT – II

Symmetric key Ciphers: Block Cipher principles, DES, AES, Blowfish, RC5, IDEA, Block cipher operation, Stream ciphers, RC4.
Asymmetric key Ciphers: Principles of public key cryptosystems, RSA algorithm, Elgamal Cryptography, Diffie-Hellman Key Exchange, Knapsack Algorithm.

UNIT – III

Cryptographic Hash Functions: Message Authentication, Secure Hash Algorithm (SHA-512), Message authentication codes: Authentication requirements, HMAC, CMAC, Digital signatures, Elgamal Digital Signature Scheme.
Key Management and Distribution: Symmetric Key Distribution Using Symmetric & Asymmetric Encryption, Distribution of Public Keys, Kerberos.

UNIT – IV

Transport-level Security: Web security considerations, Secure Socket Layer and Transport Layer Security, HTTPS, Secure Shell (SSH).
Wireless Network Security: Wireless Security, Mobile Device Security, IEEE 802.11 Wireless LAN, IEEE 802.11i Wireless LAN Security.

TEXT & REFERENCES:

1) Cryptography and Network Security - Principles and Practice: William Stallings, Pearson Education, 6th Edition.
2) Cryptography and Network Security: Atul Kahate, Mc Graw Hill, 3rd Edition.
1. C. P. Pfleeger, S. L. Pfleeger; Security in Computing, Prentice Hall of India, 2006.

TUTORIALS - CRYPTOGRAPHY AND NETWORK SECURITY (CAPC1622M)

LIST OF EXPERIMENTS:

1. Write a program that contains a string (char pointer) with a value 'Hello world'. The program should XOR each character in this string with 0 and displays the result.
2. Write a program that contains a string (char pointer) with a value 'Hello world'. The program should AND or and XOR each character in this string with 127 and display the result.
3. Write a program to perform encryption and decryption using the following algorithms
   a) Ceaser cipher **b)**. Substitution cipher **c)**. Hill Cipher
4. Write a program to implement the DES algorithm logic.
5. Write a program to implement the Blowfish algorithm logic.
6. Write a program to implement the Rijndael algorithm logic.
7. Write the RC4 logic in Java Using Java cryptography; encrypt the text "Hello world" using Blowfish. Create your own key using Java key tool.
8. Write a program to implement RSA algorithm.
9. Implement the Diffie-Hellman Key Exchange mechanism using HTML and JavaScript.
10. Write a program to calculate the message digest of a text using the SHA-1 algorithm.
11. Write a program to calculate the message digest of a text using the MD5 algorithm.
12. Demonstrate the use of Network tools: ping, ipconfig, ifconfig, tracert, arp, netstat, whois.

Course Type: - Major
Paper Title: **ARTIFICIAL INTELLIGENCE**
Credit Weightage: - THEORY -04; PRACTICALS- 02

Semester: - 6th
Paper Code: - CAPC2622M
Batch: - 2023

Course Objective:

- To learn the distinction between optimal reasoning vs. human-like reasoning.
- To understand the concepts of state space representation, exhaustive search, heuristic search together with the time and space complexities.
- To know about the various knowledge representation methods

Course Outcomes:

- Ability to formulate an efficient problem space for a problem expressed in natural language.
- Select a search algorithm for a problem and estimate its time and space complexities.
- Understand knowledge representation methods and apply approximate reasoning.

UNIT – I

INTRODUCTION TO ARTIFICIAL INTELLIGENCE AND PROBLEM-SOLVING AGENT: Problems of AI, AI technique, Tic– Tac – Toe problem. Intelligent Agents, Agents & environment, nature of environment, structure of agents, goal-based agents, utility-based agents, learning agents. Defining the problem as state space search, production system, problem characteristics, and issues in the design of search programs. Problem solving agents, searching for solutions; uniform search strategies: breadth first search, depth first search, depth limited search, bidirectional search, comparing uniform search strategies.

UNIT – II

SEARCH TECHNIQUES AND CONSTRAINT SATISFACTION PROBLEMS: Heuristic search strategies Greedy best -first search, A* search, AO* search, memory bounded heuristic search: local search algorithms & optimization problems: Hill climbing search, simulated annealing search, local beam search.

Local search for constraint satisfaction problems. Adversarial search, Games, optimal decisions & strategies in games, the minimax search procedure, alpha-beta pruning, additional refinements, iterative deepening.

UNIT-III

KNOWLEDGE REPRESENTATION: predicate logic- logic programming, semantic nets- frames and inheritance, constraint propagation, representing knowledge using rules, rules-based deduction systems. Reasoning under uncertainty, review of probability, Baye's probabilistic interferences and dempster-Shafer theory.

UNIT-IV

EXPERT SYSTEMS: - Introduction, basic concepts, structure of expert systems, the human element in expert systems how expert systems works, problem areas addressed by expert systems, expert systems success factors, types of expert systems, expert systems and the internet interacts web, knowledge engineering, scope of knowledge, difficulties in knowledge acquisition, methods of knowledge acquisition. Typical expert systems - MYCIN, DART, XOON, Expert systems shells.

TEXT & REFERENCES:

1) Artificial Intelligence A Modern Approach, Stuart Russell and Peter Norvig, 3rd Edition, Pearson Education.
2) Artificial Intelligence, 3rdEdn. , E.Rich and K.Knight (TMH).
3) Tom M. Mitchell- Machine Learning - McGraw Hill Education.
4) Christopher M. Bishop Pattern Recognition and Machine Learning - Springer, 2nd edition
5) Trevor Hastie, Robert Tibshirani, and Jerome Friedman - The Elements of Statistical Learning: Data Mining, Inference, and Prediction - Springer, 2nd edition.
6) Marc Peter Deisenroth, A. Aldo Faisal, Cheng Soon Ong, Mathematics for Machine Learning, Cambridge University Press.

Course Type: - Major
Semester: - 6th
Paper Title: - **THEORY OF COMPUTATION**
Paper Code: - **CAPC3622M**
Credit Weightage: - THEORY -04; PRACTICALS- 02
Batch: - 2023

Course Objective:

- To provide introduction to some of the central ideas of theoretical computer science from the perspective of formal languages.
- To introduce the fundamental concepts of formal languages, grammars and automata theory.
- Classify machines by their power to recognize languages.
- To understand deterministic and non-deterministic machines.
- Introduce the major concepts of language translation and compiler design and impart the knowledge of practical skills necessary for constructing a compiler.
- Topics include phases of compiler, parsing, code optimization techniques, intermediate code generation, code generation.

Course Outcomes:

- Able to understand the concept of abstract machines and their power to recognize the languages.
- Able to employ finite state machines for modelling and solving computing problems.
- Able to design context free grammars for formal languages.
- Demonstrate the ability to design a compiler given a set of language features.
- Demonstrate the knowledge of patterns, tokens & regular expressions for lexical analysis.
- Design algorithms to do code optimization in order to improve the performance of a program in terms of space and time complexity.
- Design algorithms to generate machine code.

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.

Regular Expressions: Finite Automata and Regular Expressions, Applications of Regular Expressions, Algebraic Laws for Regular Expressions, Conversion of Finite Automata to Regular Expressions, Equivalence of DFA, NFA and REs.

UNIT – II

CONTEXT FREE GRAMMARS AND PARSING: 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. Chomsky Normal Form.

UNIT – III

PUSHDOWN 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, Equivalence of PDA and CFG.

UNIT – IV

TURING MACHINES: Introduction to Turing Machine, Formal Description, Instantaneous description, The language of a Turing machine.

RECURSIVE AND RECURSIVELY ENUMERABLE LANGUAGES (REL): Properties of recursive and recursively enumerable languages, Universal Turing machine, The Halting problem, Undecidable problems about TMs. Context sensitive language and linear bounded automata (LBA), Chomsky hierarchy

TEXT & REFERENCES:

1) John E. Hopcroft, Rajeev Motwani, Jeffrey D. Ullman (2007), Introduction to Automata Theory Languages and Computation, Pearson.
2) Introduction to Computer Theory, Daniel I.A. Cohen, Wiley India
3) K. L. P Mishra, N. Chandrashekaran (2003), Theory of Computer Science-Automata Languages and Computation, 2nd edition, Prentice Hall of India, India.
4) P. Linz, An Introduction to Formal Language and Automata 4th edition Publication Jones Bartlett.
5) NPTEL Course on Theory of Computation @ https://nptel.ac.in/courses/106104148
6) YouTube TOC- @ https://www.youtube.com/playlist?list=PLBlnK6fEyqRgp46KUv4ZY69yXmpwKOIev

TUTORIALS - THEORY OF COMPUTATION (CAPC3822M)

LIST OF TUTORIALS:

1. ……

Course Type: - Minor
Semester: - 6th
Paper Title: - **PROGRAMMING WITH PYTHON**
Paper Code: - **ACPC1622N**
Credit Weightage: - THEORY -04; PRACTICALS- 02
Batch: - 2023

Course Objective:

- To introduce students to Python programming and its applications in data science and machine learning.
- To provide a comprehensive understanding of essential Python concepts, data structures, and control flow.
- To equip students with the skills to work with popular Python libraries for data manipulation, analysis, and visualization.
- To enable students to apply Python programming and data science knowledge to real-world projects and problem-solving.

Course Outcomes:

- Demonstrate proficiency in Python programming, including variables, data types, functions, and control flow structures.
- Utilize NumPy to perform numerical computing and array operations efficiently.
- Understand the basics of machine learning and implement various ML models using Scikit-learn.
- Apply Python and data science skills to conduct exploratory data analysis and draw meaningful conclusions from datasets.

UNIT – I

Introduction to Python Programming: Introduction to Python and its features, Installing Python and development environments (Jupiter Notebook). Basic Python syntax: data types, Type conversion and casting, variables and constants , arithmetic operators, comparison operators, logical operators. Combining operators to form complex expressions. Control statements: if-else, loops (while, for), conditional expressions (ternary operator) for concise code. Functions and Modules: Defining functions in Python, Function parameters, return values, and recursion. Introduction to modules and the standard library, Importing and using modules in Python programs. Working with strings.

UNIT – II

PYTHON DATA STRUCTURES AND FILE HANDLING: Lists: Creating, accessing, and modifying lists. writing concise and efficient list comprehensions for creating lists based on existing data. Tuples, Sets and Dictionaries: Understanding unique elements and set operations (union, intersection, difference). Dictionaries: Associating key-value pairs for efficient data retrieval. Reading and writing binary files for textual & non-textual data (e.g., images, audio). File Handling in Python. Exception handling: try, except, finally blocks. Introduction to NumPy arrays for Efficient Numerical Computing: Array Creation and Manipulation, single and Multi-dimensional Arrays, Element-wise Operations, Mathematical Functions.

UNIT – III

Data Manipulation with Pandas: Introduction to Pandas library for data manipulation and analysis, Series and Data Frame objects for handling data, Data indexing, selection, and filtering, Data cleaning, handling missing values, and data transformation. Data Visualization with Matplotlib and Seaborn: Introduction to data visualization, Creating basic plots with Matplotlib, Enhancing visualizations with Seaborn, Customizing plots for effective data representation.

UNIT – IV

Exploratory Data Analysis (EDA): Understanding the importance of EDA in data science, Statistical summaries and data distributions, Univariate and bivariate analysis using visualizations, Correlation and feature relationships. Machine Learning in Python: Overview of Scikit-learn library, Data pre-processing and feature scaling, Building and training machine learning models, Evaluating and tuning model performance.

TEXT & REFERENCES:

1. Python Crash Course, Eric Matthes, No Starch Press.
2. Learning Python,Mark Lutz, O'Reilly.
3. How to Think Like a Computer Scientist: Learning with Python, Allen Downey, Jeff Elkner and Chris Meyers, SoHo Books, 2009.
4. Introduction to Machine Learning with Python, Andreas C. Müller & Sarah Guido, O'Reilly.
5. Python for Data Analysis, Wes McKinney, O'Reilly.

LAB. - PROGRAMMING WITH PYTHON (ACPC1622N)

LIST OF LAB. ASSIGNMENTS :

1. Write a Python program to calculate the area of a rectangle given its length and width.
2. Implement a function that takes a list of numbers as input and returns the sum of all the elements in the list.
3. Create a program that generates a random number between 1 and 100 and lets the user guess it. Provide appropriate feedback based on their guess.
4. Write a Python program to check if a given number is even or odd.
5. Create a program that prints the Fibonacci series up to a given number using a loop.
6. Write a Python function that takes a string as input and returns the reverse of the string.
7. Create a program that counts the number of vowels and consonants in a given sentence.
8. Implement a function that takes a sentence and returns the sentence with each word capitalized.
9. Write a Python program that takes a list of numbers as input and finds the maximum and minimum values.
10. Write a Python program to read a text file and count the number of words in it.
11. Create a program that reads a CSV file and calculates the average of a specific column.
12. Implement a function that takes a list of names and writes them to a new file, one name per line.
13. Create a module with a function to check if a number is prime or not.
14. Implement a program that imports the module and uses the prime-checking function to find all prime numbers in a given range.
15. Write a Python program that takes user input and handles exceptions for invalid input.
16. Implement a function that reads a number from a file and handles file-not-found exceptions.
17. Create a program that asks the user to enter two numbers and divide them, handling division by zero error.
18. Create a NumPy array with random integers and perform basic array operations like reshaping, flattening, and transposing.
19. Use Pandas to filter and slice data based on specific conditions.
20. Apply data cleaning techniques like handling missing values, removing duplicates, and transforming data.
21. Create line plots, scatter plots, and bar plots to visualize various datasets using Matplotlib.
22. Use Seaborn to create visually appealing statistical visualizations like box plots, violin plots, and pair plots.
23. Plot time series data with appropriate annotations and labels.
24. Perform EDA on a real-world dataset to understand data distributions, correlations, and feature relationships.
25. Use Matplotlib and Seaborn to create various visualizations that aid in data exploration.