Department of Computer Science and Engineering

Data Science
3290501

COURSE OBJECTIVES

- To provide the fundamental knowledge of Data Sciences, along with essential Python programming skills..
- Apply data manipulation, statistical analysis, and visualization techniques using Python libraries like NumPy and pandas.
- Develop, implement, and evaluate machine learning models while using statistical methods to derive insights and validate results.

Unit – I

Introduction to Data Science: Introduction, Definition, applications of Data Science, Impact of Data Science, Data Analytics Life Cycle, role of Data Scientist.

Basics of Python: Essential Python libraries, Python Introduction- Features, Identifiers, Reserved words, Indentation, Comments, Built-in Data types and their Methods: Strings, List, Tuples, Dictionary, Set, Type Conversion- Operators. Decision Making: Looping-Loop Control statement, Math and Random number functions. User defined functions.

Vectorized Computation: The NumPy ndarray- Creating ndarrays- Data Types for ndarrays- Arithmetic with NumPy Arrays- Basic Indexing and Slicing.

Unit-II

Data Analysis (with Pandas): Series, DataFrame, Essential Functionality: Dropping Entries, Indexing, Selection, and Filtering- Function Application and Mapping- Sorting and Ranking. Summarizing and Computing Descriptive Statistics – Mean, Standard Deviation, Skewness and Kurtosis. Unique Values, Value Counts, and Membership. Reading and Writing Data in Text Format.

Unit-III

Exploratory Data Analysis and Visualisation: Handling Missing Data, Data Transformation: Removing Duplicates, Transforming Data Using a Function or Mapping, Replacing Values, Detecting and Filtering Outliers, Functions in pandas. Plotting with pandas: Line Plots, Bar Plots, Histograms and Density Plots, Scatter or Point Plots.

Unit-IV

Introduction to Machine Learning: Types of Learning, Linear Regression- Simple Linear Regression, Implementation, plotting and fitting regression line, Logistic Regression, K-Nearest Neighbors (KNN), K-Means Clustering.

Department of Computer Science and Engineering

Unit-V
Model Evaluation Metrics: Accuracy, Precision, Recall, F1-Score
Hypothesis Testing: Mean and Variance Tests, p-value, Errors, Z-Test, t-Test, Paired t-Test, and F-Test, Analysis of Variance (ANOVA) and Contingency Table Analysis

---

RECOMMENDED BOOKS

1. Cathy O'Neil and Rachel Schutt , "Doing Data Science", O'Reilly, 2015.
2. David Dietrich, Barry Heller, Beibei Yang, "Data Science and Big data Analytics", EMC 2013
3. Artificial Intelligence: A Modern Approach by Stuart J. Russell and Peter Norvig, Prentice Hall.
4. Pattern Recognition and Machine Learning, Christopher M. Bishop
5. James, Gareth, et al. An introduction to statistical learning. Vol. 112. New York: springer, 2013.

COURSE OUTCOMES

After completion of this course, the students would be able to:

CO1: Analyze Data Science concepts and apply Python programming for data tasks, including data manipulation with NumPy.
CO2: Analysis of the data for applying various statistical modeling approaches.
CO3: Develop expertise in managing missing data and assessing the impact of visualizations on data insight communication.
CO4: Design and implement machine learning algorithms and assess model performance.
CO5: Develop statistical tests and **evaluate** machine learning models.

Course Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
CO1	3	2	1	1	3	2	-	-	1	2	1	1	3	2
CO2	3	3	2	2	3	2	-	1	1	2	2	1	3	3
CO3	2	2	2	1	3	2	-	1	1	3	3	2	3	2
CO4	3	3	3	2	3	3	-	1	1	2	3	2	3	3
CO5	3	3	2	2	3	3	1	1	1	2	2	2	3	3

1-Slightly; 2 - Moderately; 3 – Substantially

Department of Computer Science and Engineering

Data Science
3290501

List of Experiments

1. Perform Creation, indexing, slicing, concatenation and repetition operations on Python built-in data types: Strings, List, Tuples, Dictionary, Set
2. Solve problems using decision and looping statements.
3. Apply Python built-in data types: Strings, List, Tuples, Dictionary, Set and their methods to solve any given problem
4. Handle numerical operations using math and random number functions.
5. Manipulation of NumPy arrays- Indexing, Slicing, Reshaping, Joining and Splitting.
6. Computation on NumPy arrays using Universal Functions and Mathematical methods.
7. Import a CSV file and perform various Statistical and Comparison operations on rows/columns.
8. Create Pandas Series and DataFrame from various inputs.
9. Import any CSV file to Pandas DataFrame and perform the following:
    1. Visualize the first and last 10 records
    2. Get the shape, index and column details
    3. Select/Delete the records(rows)/columns based on conditions.
    4. Perform ranking and sorting operations.
    5. Do required statistical operations on the given columns.
    6. Find the count and uniqueness of the given categorical values.
    7. Rename single/multiple columns.
10. Import any CSV file to Pandas DataFrame and perform the following:
    1. Handle missing data by detecting and dropping/ filling missing values.
    2. Transform data using different methods.
    3. Detect and filter outliers.
    4. Perform Vectorized String operations on Pandas Series.
    5. Visualize data using Line Plots, Bar Plots, Histograms, Density Plots and Scatter Plots.
11. Use the scikit-learn package in python to implement the regression model and its related methods.

Department of Computer Science and Engineering

Course Outcomes (COs)

CO1: Apply fundamental Python programming constructs such as data types, control structures, and functions to design ethical and efficient solutions for real-life problems.

CO2: Analyze and process structured and unstructured data using Python libraries like NumPy and Pandas to derive meaningful insights while considering societal relevance and responsible data handling.

CO3: Develop real world data science applications using Python

----

Course Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
CO1	3	3			2	2		2		2	2		3	2
CO2	3	3		2	2	2		2		2	2		3	3
CO3	3	2	3	2	2	2			3	3	2	2	3	3
CO4	3	3			2	2		2		2	2		3	2
CO5	3	3		2	2	2		2		2	2		3	3

1-Slightly; 2 - Moderately; 3 – Substantially

Department of Computer Science and Engineering

Data Science
3290501

list of skill-based mini project

- Exploratory Data Analysis (EDA): Perform an in-depth analysis of a dataset, including data cleaning, visualization, and statistical analysis to gain insights and understand the underlying patterns and relationships.
- Predictive Modeling: Build a machine learning model to predict a specific outcome or target variable based on a given dataset. This could include classification, regression, or time series forecasting tasks.
- Natural Language Processing (NLP): Develop a text classification or sentiment analysis model using techniques such as tokenization, word embeddings, and recurrent neural networks (RNNs) to analyze and understand text data.
- Image Recognition: Create an image recognition system using convolutional neural networks (CNNs) to classify or identify objects, faces, or patterns in images.
- Recommendation System: Build a recommendation engine that suggests personalized recommendations to users based on their preferences and behavior, using collaborative filtering or content-based filtering techniques.
- Clustering Analysis: Implement clustering algorithms such as k-means, hierarchical clustering, or DBSCAN to group similar data points together and discover hidden patterns or segments within a dataset.
- Time Series Analysis: Analyze time-dependent data, such as stock prices or weather data, using techniques like autoregressive integrated moving average (ARIMA), exponential smoothing, or recurrent neural networks (RNNs).
- Anomaly Detection: Develop an anomaly detection system that can identify unusual or suspicious patterns in data, which can be useful for fraud detection, network intrusion detection, or outlier detection.
- Social Media Sentiment Analysis: Use data from social media platforms to analyze public sentiment towards specific topics, brands, or events using natural language processing techniques and sentiment analysis algorithms.
- Data Visualization Dashboard: Create an interactive dashboard using libraries like Plotly or Dash to visualize and explore data, providing users with an intuitive interface to interact with and gain insights from the data.

Please Note: *Each project has to be submitted by a group of 1 or 2 students, and each group will be assigned only one project.*

* * * * * * * * * * *

Department of Computer Science and Engineering

DATA SCIENCE
2290501 (OLD)

COURSE OBJECTIVES

- To provide the fundamental knowledge of Data Sciences.
- To analyse the working of various techniques used in Data Sciences.
- To understand the basic representation and exploratory data analysis used in Data Sciences.

Unit-I

Introduction to Data Science: Introduction, Definition, applications of Data Science, Impact of Data Science, Data Analytics Life Cycle, role of Data Scientist.

Basics of Python: Essential Python libraries, Python Introduction- Features, Identifiers, Reserved words, Indentation, Comments, Built-in Data types and their Methods: Strings, List, Tuples, Dictionary, Set, Type Conversion- Operators. Decision Making: Looping-Loop Control statement, Math and Random number functions. User defined functions, function arguments & its types.

Unit-II

Vectorized Computation: The NumPy nd-array- Creating nd-arrays- Data Types for nd-arrays- Arithmetic with NumPy Arrays- Basic Indexing and Slicing, Boolean Indexing, Transposing Arrays. Universal Functions: Fast Element, Wise Array Functions, Mathematical and Statistical Methods – Sorting Unique and Other Set Logic.

Unit-III

Data Analysis: Series, Data Frame, Essential Functionality: Dropping Entries, Indexing, Selection, and Filtering- Function Application and Mapping- Sorting and Ranking. Summarizing and Computing Descriptive Statistics – Mean, Standard Deviation, Skewness and Kurtosis. Unique Values, Value Counts, and Membership. Reading and Writing Data in Text Format.

Unit-IV

Inferential Statistics in Data Science: Types of Learning, Linear Regression- Simple Linear Regression, Implementation, plotting and fitting regression line. Multiple Linear Regression, Introduction, implementation, comparison with simple linear regression, Correlation Matrix, F-Statistic, Identification of significant features. Polynomial regression.

Unit-V

Exploratory Data Analysis and Visualisation: Handling Missing Data, Data Transformation: Removing Duplicates, Transforming Data Using a Function or Mapping, Replacing Values, Detecting and Filtering Outliers, Functions in pandas. Plotting with pandas: Line Plots, Bar Plots, Histograms and Density Plots, Scatter or Point Plots.

RECOMMENDED BOOKS

1. Cathy O'Neil and Rachel Schutt , "Doing Data Science", O'Reilly, 2015.
2. David Dietrich, Barry Heller, Beibei Yang, "Data Science and Big data Analytics", EMC 2013
3. Artificial Intelligence: A Modern Approach by Stuart J. Russell and Peter Norvig, Prentice Hall.
4. Pattern Recognition and Machine Learning, Christopher M. Bishop

COURSE OUTCOMES

After completion of this course, the students would be able to:

CO1. **Define** basic concepts of Data Sciences and Python.
CO2. **Identify** various methods for the representation and manipulation of vectors.
CO3. **Analysis** of the data for applying various statistical modeling approaches.
CO4. **Develop** the skill to use statistical measures to identify significant features, evaluate model performance and perform regression analysis.
CO5. **Develop** expertise in managing missing data and assessing the impact of visualizations on data insight communication.

Course Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
CO1	1	3												
CO2			2				3						1	
CO3					2				2			2	2	
CO4					1								1	
CO5					2							1	3	1

1-Slightly; 2 - Moderately; 3 – Substantially

Networking with TCP/IP
3290502

COURSE OBJECTIVES

- To understand TCP/IP Internetworking and Addressing.
- To understand framing, Routing, Address resolution and Error reporting mechanism used in the Internet
- To understand the working of Application layer protocols
- To Troubleshoot networking issues

Unit-1

TCP/IP model, Addressing- Physical, logical and port addressing, IPv4 addresses: Classful addressing, Classless addressing. Special addresses, DHCP and NAT. Subnetting and Supernetting.

Unit-2

IP Datagram- format, options, fragmentations, checksum, IPsec. Address Resolution Protocol (ARP), Reverse address resolution protocol (RARP). Internet Control message protocol (ICMP).

Unit-3

TCP: TCP Reliable data transfer, Connection Establishment & Release, TCP Frame, Header Checksum, Sliding Window Concept for error control, congestion control and TCP timers. UDP: Format, Pseudo header, Encapsulation, Checksum, Multiplexing & Demultiplexing. Stream Control Transmission Protocol.

Unit-4

Routing Protocols- RIP, OSPF and BGP, Application Layer: DNS, FTP, TFTP, Mail Transfer protocols, TELNET, HTTP.

Unit- 5

IPv6 Protocol, ICMPv6, IPv6 addressing, Voice over IP, RTP, SNMP, Internet security and Firewall: Internet Security, IP security, Firewall Implementation, Study of network packet analyzer tools: Wireshark, CISCO packet Tracer etc. Scanner Tools: Nmap, Nessus etc.

Internet of Things (IoT) Networking**,** 5G and Next-Generation Mobile Networks, Edge and Fog Computing in Networking, Zero Trust Network Architecture

---

RECOMMENDED BOOKS

- Data and Computer Communication - W. Stalling, Pearson
- Internetworking with TCP/IP - Vol. - I - D.E. Comer, PHI
- Data Communication & Networking -B.A. Forouzan
- ISDN and Broad band ISDN with Frame Relay & ATM - W. Stalling
- LANs - Keiser

COURSE OUTCOMES

After completion of this course, the students would be able to:

CO1. Illustrate the architecture of the TCP/IP model and demonstrate addressing schemes used in network communication.

CO2. Identify and explain the role of core Internet layer protocols including IP, ARP, ICMP, and IPsec.

CO3. Analyze the features of TCP, UDP, and SCTP transport protocols and their mechanisms for reliable communication.

CO4. Explain the operation of routing and application layer protocols in data communication.

CO5. Simulate and evaluate network topologies and security protocols using modern network analysis tools.

Course Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
CO1	3	2		1	2	1				1		1	2	1
CO2	3	3			2							1	2	1
CO3	3	3	2	2	2							2	2	1
CO4	3	3	2	2	2		1			1		2	2	2
CO5	3	3	3	3	3	1	2	1	2	2	2	3	2	3

1 - Slightly; 2 - Moderately; 3 – Substantially

NETWORKING WITH TCP/IP
2290502 (OLD)

COURSE OBJECTIVES

- To understand TCP/IP Internetworking and Addressing.
- To understand framing, Routing, Address resolution and Error reporting mechanism used in the Internet
- To understand the working of Application layer protocols
- To Troubleshoot networking issues

Unit-1

TCP/IP model, Addressing- Physical, logical and port addressing, IPv4 addresses: Classful addressing, Classless addressing. Special addresses, DHCP and NAT. Subnetting and Supernetting.

Unit-2

IP Datagram- format, options, fragmentations, checksum, IPsec. Address Resolution Protocol (ARP), Reverse address resolution protocol (RARP). Internet Control message protocol (ICMP).

Unit-3

TCP: TCP Reliable data transfer, Connection Establishment & Release, TCP Frame, Header Checksum, Sliding Window Concept for error control, congestion control and TCP timers. UDP: Format, Pseudo header, Encapsulation, Checksum, Multiplexing & Demultiplexing. Stream Control Transmission Protocol

Unit-4

Routing Protocols- RIP, OSPF and BGP, Application Layer: DNS, FTP, TFTP, Mail Transfer protocols, TELNET, HTTP.

Unit- 5

IPv6 Protocol, ICMPv6, IPv6 addressing, Voice over IP, RTP, SNMP, Internet security and Firewall: Internet Security, IP security, Firewall Implementation, Study of network packet analyzer tools: Wireshark, CISCO packet Tracer etc. Scanner Tools: Nmap, Nessus etc.

RECOMMENDED BOOKS

- Data and Computer Communication - W. Stalling, Pearson
- Internetworking with TCP/IP - Vol. - I - D.E. Comer, PHI
- Data Communication & Networking -B.A. Forouzan
- ISDN and Broad band ISDN with Frame Relay & ATM - W. Stalling
- LANs - Keiser

COURSE OUTCOMES

After completion of this course, the students would be able to:

CO1. **Illustrate** the basic functionality of TCP/IP layers.
CO2. **Identify** various network layer protocol
CO3. **Analyze** the working of Transport layer protocols
CO4. **Explain** the working of Application layer protocols
CO5. **Simulate** network protocols & Topologies

Course Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
CO1	1							1				2		
CO2				1			1	1		1		2		
CO3		2	2	2	2		1	1	2		1	2	3	2
CO4		2		1			1	1		1		2	2	
CO5	1	3	3	2	2			1	2	1	1	2	3	2

1 - Slightly; 2 - Moderately; 3 – Substantially

Software Design & Development
3290503

COURSE OBJECTIVES

- Understand and apply software development methodologies, including design process models, SCRUM, and Agile practices.

- Develop and analyze software design artifacts, such as UML diagrams, to support effective system design.

- Manage software development projects by employing planning, scheduling, quality assurance, and risk management techniques.

Unit-I

Introduction to Software Design Process Models: Software design process models, Iterative, Incremental, Agile practices. Characteristics of software projects, project attributes, project constraints. project baseline, project charter, Stakeholders, Feasibility Study, Cost-benefit Analysis.

Unit-II

Software Design & Development Approaches: Importance of software design, Software development life cycle, Principles of good design, Structural View, Class Diagrams, Dynamic behavioral view, Sequence diagrams, UML diagrams, Project and Product Life Cycles, role of project manager, System view of project management, Barry Boehm: W5HH principle.

Unit- III

Fundamentals of Agile & Agile Scrum Framework: Genesis of Agile, Introduction and background, Agile Manifesto and Principles, Lean Software, Agile project management, Design and development practices in Agile projects, Agile Tools, Problem Agile Solves, Introduction to Scrum, Project phases Product backlog, Sprint backlog, Iteration planning, Scrum and Kanban. User story definition, Characteristics and content of user stories, Burn down chart, Sprint planning.

Unit-IV
Techniques of Project Scheduling: Work Breakdown Structure (WBS), activities sequencing, network diagrams, activity duration estimation, schedule development, Gantt Charts, Critical path method (CPM), Program evaluation & review technique (PERT), concept of slack time, schedule control.

Unit-V

Quality and Risk Management: Cost budgeting, cost control, earned value management, project portfolio management. Project Quality Management: Quality Planning, quality Assurance, Quality control, Tool & techniques for quality control. Pareto Analysis, Six Sigma. CMM, ISO Standards, Juran Methodology, Human Resource Management, responsibility assignment metrics, resource loading, resource levelling, Risk Management planning, Expected Monetary Value, Decision tree, Releases vs. version, Introduction to DevOps and Continuous Practices.

RECOMMENDED BOOKS

- Bob Hughes, Mike Cottrell and Rajib Mall, Software Project Management (5"' cd.), Tata McGraw Hill, 2009. ISBN 978-0071072748.
- Cooperative Software Development – Dr. Amy Ko.
- Agile Software Development with Scrum, Ken Schawber, Mike Beedle, Pearson.
- Agile Software Development, Principles, Patterns and Practices, Robert C. Martin, Prentice Hall.
- Agile Software Development: The Cooperative Game, Alistair Cockburn, Addison Wesley.

COURSE OUTCOMES

After completion of course students will be able to:

CO1: **Identify** software design process models and its applications
CO2: **Investigate** various software design & development approaches.
CO3. **Execute** Agile practices in software projects such as scrum and sprints
CO4**: Implement** Project Scheduling techniques such as CPM and PERT.
CO5: **Recognize** Quality Assurance and Control Techniques and investigate the risks involved in software design & development.

Course Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
CO1	3	2	3	2	3	1	1	1	1	1	1	1	3	1
CO2	2	3	3	2	2	1	1	–	–	1	1	1	2	3
CO3	3	3	2	2	2	1	1	–	–	1	1	3	3	2
CO4	2	2	3	2	3	1	1	1	1	1	1	1	2	3
CO5	3	2	2	2	3	1	1	1	1	1	2	1	2	3

1 - Slightly; 2 - Moderately; 3 – Substantially

Software Design & Development
3290503

List of Experiments

1. Create a class and use case diagram for your project.
2. Develop a use case and sequence diagram.
3. Design a sequence and activity diagram.
4. Develop a component and object diagram.
5. Create a state machine diagram
6. Learn and create Epic, Story and Tasks
7. Manage Agile boards and build Roadmaps
8. Backlogs and Integrate WBS Gantt Chart.
9. Case Study of Online Grocery Shopping.
10. Case Study of Online Movie ticket booking
11. Develop an SRS for your project.

Lab Outcomes

CO1: design and interpret various UML diagrams including class, use case, and sequence activity diagrams.

CO2: Apply Agile methodologies by creating epics, user stories, tasks, and managing Agile boards and roadmaps.

CO3: Analyze real-world systems through case studies and document them using UML diagrams and SRS.

Software Design & Development
3290503

List of Skill Based Mini Project

1. Create a class diagram to model a simple library system, including classes for books, authors, and borrowers.
2. Develop a use case diagram for an online shopping system, identifying actors and use cases.
3. Design a sequence diagram for a login system, showing the interaction between user, login screen, and database.
4. Model an activity diagram for a payment processing system, highlighting actions and decision points
5. Develop a component diagram for a web-based application, showing components and their interactions
6. Design an object diagram for a simple banking system, showing objects and their relationships.
7. Create a To-Do Daily Task Management Project and set priorities
8. Create an SRS for a project management software with features like task assignment, progress tracking, and team collaboration.
9. Create an SRS for a banking software with features like account management, transaction processing, and security measures.
10. Develop an SRS for a food delivery mobile app with features like restaurant listing, menu management, and order tracking.

Software Design & Development
2290509(OLD)

COURSE OBJECTIVES

- To understand the nature of software design process models, SCRUM and Agile practices.
- Develop skills in creating and analyzing software design artifacts such as UMLdiagrams.
- Manage software development projects through effective planning and scheduling techniques.
- To understand concept of software quality assurance and risk management process.

Unit-I

Introduction to Software Design Process Models: Software design process models, Iterative, Incremental, Agile practices. Characteristics of software projects, project attributes, project constraints. project baseline, project charter, Stakeholders, Feasibility Study, Cost-benefit Analysis.

Unit-II

Software Design & Development Approaches: Importance of software design, Software development life cycle, Principles of good design, Structural View, Class Diagrams, Dynamic behavioral view, Sequence diagrams, UML diagrams, Project and Product Life Cycles, role of project manager, System view of project management, Barry Boehm: W5HH principle.

Unit- III

Fundamentals of Agile & Agile Scrum Framework: Genesis of Agile, Introduction and background, Agile Manifesto and Principles, Lean Software, Agile project management, Design and development practices in Agile projects, Agile Tools, Problem Agile Solves, Introduction to Scrum, Project phases Product backlog, Sprint backlog, Iteration planning, Scrum and Kanban. User story definition, Characteristics and content of user stories, Burn down chart, Sprint planning.

Unit-IV

Techniques of Project Scheduling: Work Breakdown Structure (WBS), activities sequencing, network diagrams, activity duration estimation, schedule development, Gantt Charts, Critical path method (CPM), Program evaluation & review technique (PERT), concept of slack time, schedule control

Unit-V

Quality and Risk Management: Cost budgeting, cost control, earned value management, project portfolio management. Project Quality Management: Quality Planning, quality

Assurance, Quality control, Tool & techniques for quality control. Pareto Analysis, Six Sigma. CMM, ISO Standards, Juran Methodology, Human Resource Management, responsibility assignment metrics, resource loading, resource levelling, Risk Management planning, Expected Monetary Value, Decision tree, Releases vs. version.

RECOMMENDED BOOKS

- Bob Hughes, Mike Cottrell and Rajib Mall, Software Project Management (5"' cd.), Tata McGraw Hill, 2009. ISBN 978-0071072748.
- Cooperative Software Development – Dr. Amy Ko.
- Agile Software Development with Scrum, Ken Schawber, Mike Beedle, Pearson.
- Agile Software Development, Principles, Patterns and Practices, Robert C. Martin, Prentice Hall.
- Agile Software Development: The Cooperative Game, Alistair Cockburn, Addison Wesley.

COURSE OUTCOMES

After completion of course students will be able to:

CO1: **Identify** software design process models and its applications
CO2: **Investigate** various software design & development approaches.
CO3. **Execute** Agile practices in software projects such as scrum and sprints
CO4**: Implement** Project Scheduling techniques such as CPM and PERT.
CO5: **Recognize** Quality Assurance and Control Techniques and investigate the risks involved in software design & development

Course Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
CO1	3	2	3	2	3	1	1	1	1	1	1	1	3	1
CO2	2	3	3	2	2	1	1	–	–	1	1	1	2	3
CO3	3	3	2	2	2	1	1	–	–	1	1	3	3	2
CO4	2	2	3	2	3	1	1	1	1	1	1	1	2	3
CO5	3	2	2	2	3	1	1	1	1	1	2	1	2	3

1 - Slightly; 2 - Moderately; 3 – Substantially

Compiler Design
3290504

COURSE OBJECTIVES

- To learn finite state machines and context free grammar.
- To learn, various phases of compiler
- To understand process of compiler implementation.

Unit-I

Overview of Translation Process: Introduction to Compiler, Major Data Structures in Compiler, Other Issues in Compiler Structure, BOOT Strapping and Porting, Compiler Structure: Analysis-Synthesis Model of Compilation, Various Phases of a Compiler, Compiler Design Tools.

Unit-II

Lexical Analysis: Input Buffering, Symbol Table, Token, Recognition of Tokens, Lexeme and Patterns, Difficulties in Lexical Analysis, Error Reporting and Implementation. Regular Grammar & Language Definition, Transition Diagrams, Design of a Typical Scanner using LEX.

Unit-III

Syntax Analysis: Context Free Grammars (CFGs), Ambiguity, Basic Parsing Techniques: Top Down Parsing, Recursive Descent Parsing, Transformation on the Grammars, Predictive Parsing LL(1) Grammar, Bottom-UP Parsing, Operator Precedence Parsing, LR Parsers (SLR, CLR, LALR), Design of a Typical Parser Using YACC.

Unit-IV

Semantic Analysis: Compilation of Expression, Control, Structures, Conditional Statements, Various Intermediate Code Forms, Syntax Directed Translation, Memory Allocation and Symbol Table Organizations, Static and Dynamic Array Allocation, String Allocation, Structure Allocation etc., Error Detection Indication and Recovery, Syntax and Semantic Errors.

Department of Computer Science and Engineering

Unit-V

Code Generation and Code Optimization: Issues, Basic Blocks and Flow Graphs, Register Allocation, Code Generation, DAG Representation of Programs, Code Generation from DAGS, Peep-hole Optimization, Code Generator Generators, Specification of Machine. Code Optimization: Source of Optimizations, Optimization of Basic Blocks, Loops, Global Data Flow Analysis, Solution to Iterative Data Flow Equations, Data Flow Analysis of Structured Flow Graphs.

RECOMMENDED BOOKS

- Compilers: Principles, Techniques and Tools, V. Aho, R. Sethi and J. D. Ullman, Pearson Education.
- Compiler Construction: Principles and Practice, K.C. Louden, Cengage Learning.

COURSE OUTCOMES

After completion of this course, the students would be able to:

CO1. **Build** an understanding of the working phases and concepts of a compiler.

CO2. **Apply** Lex and Yacc tools to develop lexical analyzers and parsers.

CO3. **Compare** and apply various parsing techniques for syntax analysis.

CO4. **Perform** semantic analysis, symbol table management, and memory allocation.

CO5. **Implement** code generation and optimization techniques for efficient execution.

Course Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
CO1	3	1	-	-	1	-	-	-	-	-	-	1	2	-
CO2	2	-	-	-	3	-	-	-	1	2	-	1	2	1
CO3	3	2	-	-	-	-	-	-	-	2	-	1	2	-
CO4	3	2	-	-	-	-	-	-	-	1	-	1	1	2
CO5	3	2	2	-	-	-	-	-	1	2	2	1	3	3

1 - Slightly; 2 - Moderately; 3 – Substantially

Department of Computer Science and Engineering

COMPILER DESIGN

2290504 (OLD)

COURSE OBJECTIVES

- To learn finite state machines and context free grammar.
- To learn, various phases of compiler
- To understand process of compiler implementation.

Unit-I

Overview of Translation Process: Introduction to Compiler, Major Data Structures in Compiler, Other Issues in Compiler Structure, BOOT Strapping and Porting, Compiler Structure: Analysis-Synthesis Model of Compilation, Various Phases of a Compiler, Compiler Design Tools.

Unit-II

Lexical Analysis: Input Buffering, Symbol Table, Token, Recognition of Tokens, Lexeme and Patterns, Difficulties in Lexical Analysis, Error Reporting and Implementation. Regular Grammar & Language Definition, Transition Diagrams, Design of a Typical Scanner using LEX.

Unit-III

Syntax Analysis: Context Free Grammars (CFGs), Ambiguity, Basic Parsing Techniques: Top Down Parsing, Recursive Descent Parsing, Transformation on the Grammars, Predictive Parsing LL(1) Grammar, Bottom-UP Parsing, Operator Precedence Parsing, LR Parsers (SLR, CLR, LALR), Design of a Typical Parser Using YACC.

Unit-IV

Semantic Analysis: Compilation of Expression, Control, Structures, Conditional Statements, Various Intermediate Code Forms, Syntax Directed Translation, Memory Allocation and Symbol Table Organizations, Static and Dynamic Array Allocation, String Allocation, Structure Allocation etc., Error Detection Indication and Recovery, Syntax and Semantic Errors.

Unit-V

Code Generation and Code Optimization: Issues, Basic Blocks and Flow Graphs, Register Allocation, Code Generation, DAG Representation of Programs, Code Generation from DAGS, Peep-hole Optimization, Code Generator Generators, Specification of Machine. Code Optimization: Source of Optimizations, Optimization of Basic Blocks, Loops, Global Data Flow Analysis, Solution to Iterative Data Flow Equations, Data Flow Analysis of Structured Flow Graphs.

RECOMMENDED BOOKS

- Compilers: Principles, Techniques and Tools, V. Aho, R. Sethi and J. D. Ullman, Pearson Education.
- Compiler Construction: Principles and Practice, K.C. Louden, Cengage Learning.

COURSE OUTCOMES

After completion of this course, the students would be able to:

CO1. **Build** the concept of working of compiler.
CO2. **Apply** the knowledge of lex & yacc tool to develop a scanner & parser.
CO3. **Examine** various parsing techniques and their comparison.
CO4. **Implemen**t various code generation and code optimization techniques.
CO5. **Analyze** different tools and techniques for designing a modern compiler.

Course Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
CO1	2	2	1	1	2	-	-	-	-	-	2	1	1	-
CO2	3	-	-	-	2	-	-	-	1	-	1	2	2	1
CO3	3	2	-	2	2	-	-	-	-	1	1	1	1	-
CO4	2	3	-	3	-	-	-	-	-	-	-	1	1	2
CO5	2	1	3	2	3	-	-	-	1	2	2	1	3	2

1 - Slightly; 2 - Moderately; 3 – Substantially

Department of Computer Science and Engineering

Design Pattern

3290505

COURSE OBJECTIVES

- To provide the fundamental concepts and principles of design patterns in software development.
- Explore the different types of design patterns and their classifications: creational, structural, and behavioral.
- Learn how to analyze and identify design pattern opportunities in software design and architecture.

Unit-I

Introduction to Design Patterns: Overview of design patterns, Importance of design patterns in software development, Types of design patterns: creational, structural, and behavioral, Object-Oriented Principles in Python, Role of Python Features in Implementing Design Patterns, UML diagrams for design patterns, Common design principles and SOLID principles.

Unit-II

Creational Design Patterns: Singleton pattern: Definition and purpose of the Singleton pattern, single instance and global access, Factory pattern: Factory pattern and its role in creating objects, Abstract factory pattern: Abstract Factory pattern using interfaces and abstract classes, Prototype pattern, Case study for creational design pattern.

Unit-III

Structural Design Patterns: Adapter pattern: Definition and purpose of the Adapter pattern, interfaces and the need for adaptation, Decorator pattern: Decorator pattern and its role in dynamically adding behavior to objects, Facade pattern: interface to a complex subsystem, Composite pattern: Composite pattern using component and leaf classes, recursive and non-recursive traversal techniques, Bridge pattern: decoupling abstractions from their implementations, Case study for structural design pattern.

Unit-IV

Behavioral Design Patterns: Observer pattern: subject and observer interfaces, Strategy pattern: strategy interfaces and concrete strategies, Template method pattern: Template Method pattern using abstract classes and concrete implementations, Command pattern: encapsulating requests as objects, decoupling requesters and receivers, State pattern: state interfaces, concrete states, and context objects, Case study for Behavioral design pattern.

Unit-V

Advanced Design Patterns: Iterator pattern: iterator interfaces and concrete iterators, Proxy pattern: surrogate or placeholder for another object, Mediator pattern: mediator interfaces and concrete mediators, Visitor pattern: visitor interfaces, concrete visitors, and visitable objects. Memento pattern: memento objects, originator objects, and caretaker objects.

RECOMMENDED BOOKS

- "Head First Design Patterns" by Eric Freeman, Elisabeth Robson, Bert Bates, and Kathy Sierra.
- "Design Patterns: Elements of Reusable Object-Oriented Software" by Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides
- "Design Patterns Explained: A New Perspective on Object-Oriented Design" by Alan Shalloway and James Trott
- "Design Patterns in Java" by Steven John Metsker and William C. Wake
- "Design Patterns in Python" by Rahul Verma
- "Modern C++ Design: Generic Programming and Design Patterns Applied" by Andrei Alexandrescu

Department of Computer Science and Engineering

COURSE OUTCOMES

After completion of the course students would be able to:

CO1. **Identify** and classify design patterns based on their purpose and characteristics.
CO2. **Implement** design patterns using appropriate programming languages and frameworks.
CO3. **Analyze** software design problems and select appropriate design patterns to address them.
CO4. **Understand** and adhere to best practices when utilizing design patterns in software development.
CO5**. Evaluate** the effectiveness and efficiency of design pattern implementations in software projects.

Course Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
CO1	3	3	3	3	3	1	1	1	1	1	1	3	3	3
CO2	3	3	3	3	3	1	1	1	1	1	1	3	3	3
CO3	3	3	3	3	3	2	1	1	1	1	1	3	3	3
CO4	3	3	2	3	3	1	2	1	1	1	1	3	3	3
CO5	3	3	3	3	3	1	1	1	1	1	2	3	3	3

1 - Slightly; 2 - Moderately; 3 – Substantially

माधव प्रौद्योगिकी एवं विज्ञान संस्थान, ग्वालियर (म.प्र.), भारत
MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR (M.P.), INDIA
Deemed to be University
(Declared under Distinct Category by Ministry of Education, Government of India)
NAAC ACCREDITED WITH A++ GRADE

Department of Computer Science and Engineering

Design Pattern
2290505(OLD)

COURSE OBJECTIVES

- To provide the fundamental concepts and principles of design patterns in software development.
- Explore the different types of design patterns and their classifications: creational, structural, and behavioral.
- Learn how to analyze and identify design pattern opportunities in software design and architecture.

Unit-I

Introduction to Design Patterns: Overview of design patterns, Importance of design patterns in software development, Types of design patterns: creational, structural, and behavioral, UML diagrams for design patterns, Common design principles and SOLID principles.

Unit-II

Creational Design Patterns: Singleton pattern: Definition and purpose of the Singleton pattern, single instance and global access, Case study, Factory pattern: Factory pattern and its role in creating objects, Abstract factory pattern: Abstract Factory pattern using interfaces and abstract classes, Builder pattern, Prototype pattern.

Unit-III

Structural Design Patterns: Adapter pattern: Definition and purpose of the Adapter pattern, interfaces and the need for adaptation, Decorator pattern: Decorator pattern and its role in dynamically adding behavior to objects, Facade pattern: interface to a complex subsystem, Composite pattern: Composite pattern using component and leaf classes, recursive and non-recursive traversal techniques, Bridge pattern: decoupling abstractions from their implementations.

Unit-IV

Behavioral Design Patterns: Observer pattern: subject and observer interfaces, Strategy pattern: strategy interfaces and concrete strategies, Template method pattern: Template Method pattern using abstract classes and concrete implementations, Case study, Command pattern: encapsulating requests as objects, decoupling requesters and receivers, State pattern: state interfaces, concrete states, and context objects.

Unit-V

Advanced Design Patterns: Iterator pattern: iterator interfaces and concrete iterators, Proxy pattern: surrogate or placeholder for another object, Mediator pattern: mediator interfaces and concrete mediators, Visitor pattern: visitor interfaces, concrete visitors, and visitable objects. Memento pattern: memento objects, originator objects, and caretaker objects.

RECOMMENDED BOOKS

- "Head First Design Patterns" by Eric Freeman, Elisabeth Robson, Bert Bates, and Kathy Sierra.
- "Design Patterns: Elements of Reusable Object-Oriented Software" by Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides
- "Design Patterns Explained: A New Perspective on Object-Oriented Design" by Alan Shalloway and James Trott
- "Design Patterns in Java" by Steven John Metsker and William C. Wake
- "Design Patterns in Python" by Rahul Verma
- "Modern C++ Design: Generic Programming and Design Patterns Applied" by Andrei Alexandrescu

Department of Computer Science and Engineering

COURSE OUTCOMES

After completion of the course students would be able to:

CO1. **Identify** and classify design patterns based on their purpose and characteristics.
CO2. **Implement** design patterns using appropriate programming languages and frameworks.
CO3. **Analyze** software design problems and select appropriate design patterns to address them.
CO4. **Understand** and adhere to best practices when utilizing design patterns in software development.
CO5**. Evaluate** the effectiveness and efficiency of design pattern implementations in software projects.

Course Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
CO1	3	3	3	3	3	1	1	1	1	1	1	3	3	3
CO2	3	3	3	3	3	1	1	1	1	1	1	3	3	3
CO3	3	3	3	3	3	2	1	1	1	1	1	3	3	3
CO4	3	3	2	3	3	1	2	1	1	1	1	3	3	3
CO5	3	3	3	3	3	1	1	1	1	1	2	3	3	3

1 - Slightly; 2 - Moderately; 3 – Substantially