Skill Enhancement Courses (SEC)

Course Code	Course Name	Level	L	T	P	C	CIE	SEE	Total	Pre-requisite
241AI033	Programming with C++	IC			2	2	50	50	100	-
241CS017	Object Oriented Analysis & Design using UML	IC			2	2	50	50	100	ASE
241AI034	Web Application Development using MERN Stack	AC			2	2	50	50	100	-
241IT008	Bigdata Spark	AC			2	2	50	50	100	-
241CS019	CI/CD using DevOps	AC			1	1	100	-	100	-
Total					**9**	**9**				

PROGRAMMING with C++

	L	T	P	C
Course Code: 241AI033	**0**	**0**	**2**	**2**

Course Outcomes:
At the end of the course, student will be able to:

CO1: Apply C++ programming concepts recursion, functions, and scope resolution in problem-solving.

CO2: Design programs using classes, encapsulation, access specifiers, constructors, destructors, and operator overloading

CO3: Analyse forms of inheritance and polymorphism to build flexible and reusable code components.

CO4: Utilize C++ features templates, exception handling, and standard template library (STL) containers to create generic, robust, and high-performance applications.

CO5: Develop a deep understanding of memory management, pointers, and virtual base classes to solve problems involving complex inheritance and runtime behaviours in C++.

Mapping of Course Outcomes with Program Outcomes:

CO/PO	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11
CO1	3	2	2	1	1				1		1
CO2	3	3	2	1	2				1		1
CO3	3	2	3	2	1						1
CO4	3	3	2	2	3				1		1
CO5	3	3	3	2	3						1

Mapping of Course Outcomes with Program Specific Outcomes:

CO/PSO	PSO1	PSO2
CO1	2	
CO2	2	
CO3	2	
CO4	2	
CO5	2	

Practice:

1. Fundamental concepts of C++
    a. **Concepts Covered**: Recursion is a process

Develop a C++ Program to Find Factorial of a Given Number Using Recursion.

b. **Concepts Covered**: arguments to a function, Call by Value and Call by Reference.
Develop a C++ Program to Demonstrate Call by Value and Call by Reference

2.

a. Scope Resolution and Namespaces
Concepts Covered: Global vs Local variable, Scope resolution operator, Declaring and using custom namespaces, using directive vs fully qualified names
Develop a C++ program that demonstrates the use of the scope resolution operator and namespaces.

b. Inline Functions
Concepts Covered: Definition, Benefits and Syntax of inline functions, Situations where inline functions are not recommended
Create a C++ program that illustrates the use of inline functions

3.

a. classes, objects, and encapsulation
Concepts Covered: Defining and creating a class, Declaring objects, Data members and member functions, Basic encapsulation principles
Develop a C++ program that models a Bank Account using a class. The class include data members account number, name, balance and member functions deposit, withdraw, display balance.

b. Access Specifiers: public and private
Concepts Covered: Access specifiers, Data hiding and encapsulation, Member access control
Create a C++ program that illustrates the difference between the public and private access specifiers.

c. this Pointer
Concepts Covered: this pointer, Resolving naming conflicts, Returning the current object.
Develop a C++ program that uses the this pointer to refer to the current object

4.

a. Function Overloading
Concepts Covered: Function overloading, Function signatures, Use cases of function overloading
Create a C++ program that demonstrates function overloading by defining multiple functions with the same name but different parameter types or counts.

b. Default Arguments in Functions

Concepts Covered: Syntax for default parameters, Rules and limitations of default arguments, Function call variations using default values
Develop a C++ program that illustrates the use of default arguments in functions.

c. Friend Function
Concepts Covered: Friend functions and their declaration, Accessing private members from non-member functions, Controlled access and encapsulation
Create a C++ program that uses a friend function to access the private data of a class

5.

a. Constructors and Destructors
Concepts Covered: Default constructor, Destructor, Constructor and destructor invocation timing, Object lifecycle management
Create a C++ program that demonstrates the use of constructors and destructors in a class.

b. Constructor Overloading
Concepts Covered: Constructor overloading, Different ways of object initialization, Function overloading principles applied to constructors
Develop a C++ program that illustrates constructor overloading.

c. Copy Constructor
Concepts Covered: Copy constructor syntax and use, Passing objects by value, When the copy constructor is invoked
Write a C++ program that illustrates the use of a copy constructor

6.

a. Overloading Unary and Binary Operators using Member Functions
Concepts Covered: Operator overloading syntax, Overloading unary operators and binary operators, Importance of return types and chaining
Develop a C++ program that demonstrates how to overload both unary and binary operators using member functions.

b. Overloading Unary and Binary Operators using Friend Functions
Concepts Covered: Syntax and declaration of friend functions, Accessing private members outside the class, Difference between member and friend function, Operator overloading with friend functions.
Create a C++ program to demonstrate operator overloading for unary and binary operators using friend functions

7.

a. Exploring Various Forms of Inheritance in C++
Concepts Covered: Single Inheritance, Multiple Inheritance, Multi-level Inheritance, Hierarchical Inheritance, Hybrid Inheritance

Develop C++ programs to demonstrate different forms of inheritance

b. Order of Execution of Constructors and Destructors in Inheritance
Concepts Covered: Constructor execution order, Destructor execution order, Constructor and destructor chaining
Develop a C++ program that illustrates the order of execution for constructors and destructors in the context of inheritance.

8.

a. Illustrating Pointers to a Class
Concepts Covered: Pointers to objects, Dereferencing and member access, Memory management
Develop a C++ program that demonstrates how to use pointers to access and manipulate objects of a class.

b. Illustrating Virtual Base Class
Concepts Covered: Diamond problem, Virtual base class, Order of constructor and destructor calls
Develop a C++ program to demonstrate the concept of virtual base classes in the context of multiple inheritance, which resolves ambiguity in the inheritance hierarchy

9.

a. Virtual Functions in C++
Concepts Covered: Virtual functions, Function overriding, Dynamic binding
Develop a C++ program to demonstrate the use of virtual functions to achieve dynamic dispatch and enable runtime polymorphism.

b. Runtime Polymorphism in C++
Concepts Covered: Runtime polymorphism, Base class pointer/reference, Overriding methods
Develop a C++ program that illustrates runtime polymorphism using virtual functions

10.

a. Function Templates in C++
Concepts Covered: Function template, Template argument deduction
Develop a C++ program that demonstrates the use of function templates to create functions that can work with different data types.

b. Template Classes in C++
Concepts Covered: Template class ,Code reusability
Develop a C++ program that demonstrates template classes, which allow creating classes that can work with any data type

11.

a. Handling Exceptions in C++
Concepts Covered: Exception handling mechanism, try block, throw statement, catch block.

Develop a C++ program that demonstrates exception handling using try, throw, and catch blocks.

b. Using Multiple Catch Statements in C++
Concepts Covered: Multiple catch blocks, Handling various exception types, Exception hierarchy
Develop a C++ program to illustrate the use of multiple catch statements, where different types of exceptions are caught and handled differently.

12.

a. Implementing List, Vector, and their Operations
Key Concepts: STL , List and Vector Operations
Develop a C++ program to implement List and Vector containers and perform basic operations such as insertion, deletion, traversal.

b. Implementing Deque and its Operations
Key Concepts: Deque operations
Implement Deque in C++ and demonstrate basic operations.

c. Implementing Map and Map Operations
Key Concepts: Map operations
Implement Map and demonstrate operations such as insertion, deletion, access, and searching

Additional Practice:

1. Develop a C++ Program for Flight Booking System
2. Develop a Qt Application Containing Slider and Spin Box(Slider Responds to Spin Box Changes)
3. Develop a Qt Application for Creating a Text Pad
4. C++ Program for a Guessing Game with Asterisks

This program presents a guessing game where the user must guess letters of a mystery word, represented initially by asterisks. The program allows 3 incorrect guesses. After each incorrect guess, the remaining chances are displayed.
A program with maximum of 20 characters, user will be guessed and will show only asterisks (*) on the screen.
The user will input one character at a time. And for every correct character, the asterisk will be replaced by that character until all the characters or the mystery word/s will reveal.
Your program will accept a maximum three (3) errors or mistakes in entering/inputting character otherwise the mystery word/s will be viewed.
Sample Output: Output: *******
Enter your character: e Output: ***e**e
Enter your character: a Output: sorry! the character is not existing. you still have 2 chances
Enter your character: s Output: s**e**e
Enter your character: c Output: sc*e*ce

Enter your character: i Output: scie*ce
Enter your character: n Output: science

Reference Books:

1	C++ Primer Plus by Stephen Prata, Sixth Edition, Pearson, ISBN: 978-9332546189
2	C++ GUI Programming with Qt4, Jasmin Blanchette, Mark Summerfield, Second Edition, Prentice Hall Press, ISBN: 978-0132354165
3	C++ for Programmers, Paul J. Deitel, Harvey M. Deitel, Pearson, ISBN: 978-0137001309

Web Links:

1	http://en.cppreference.com/w/cpp/links/libs
2	https://www.daniweb.com/digital-media/ui-ux-design/threads/113591/trying to-run-ac-program-through-a-web-link
3	http://www.yolinux.com/TUTORIALS/LinuxTutorialC++.html
4	https://github.com/fffaraz/awesome-cpp
5	http://www.techsystemsembedded.com/cpp_links.ph

Object Oriented Analysis & Design using UML
(Common to CSE, IT AIML & CSE(DS))

	L	T	P	C
Course Code: 241CS017	**0**	**0**	**2**	**2**

Course Outcomes:
At the end of the course, student will be able to:

CO1: Explain the importance of system analysis and design in solving complex problems.
CO2: Compare object-oriented approach with traditional approach in system analysis and design.
CO3: Analyze the importance of modeling and design of various applications.
CO4: Compare various object relationships.
CO5: Show the role and function of each UML model in developing object- oriented software.

Mapping of Course Outcomes with Program Outcomes:

CO/PO	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11
CO1	2	1			3			2	2		
CO2	3	1			2			2	2		
CO3	3		2		2			2	2		
CO4	3	2	1		2			2	2		
CO5	2	3			2			2	2		

Mapping of Course Outcomes with Program Specific Outcomes:

CO/PSO	PSO1	PSO2
CO1	2	
CO2	2	
CO3	3	
CO4	3	
CO5	3	

Practice:

1. List of Case Studies:
   ATMApplication.
   Library ManagementSystem.
   Online Book Shop.
   Customer SupportSystem.
   Point ofSale.
   Familiarization with RationalRose

2. For each case study:
   2.1) Identify and analyze events
   2.2) Identify Usecases
3. For each case study:
   3.1) Develop event table
   3.2) Identify &analyze domain classes
4. For each case study:
   4.1) Represent use cases and a domain class diagram using Rational Rose
   4.2) Develop CRUD matrix to represent relationships between use cases and problem domain classes
5. For each case study:
   5.1) Develop Use case diagrams
   5.2) Develop elaborate Use case descriptions &scenarios
6. For each case study
   6.1) develop system sequence diagrams
7. For each case study:
   7.1) Develop highlevel sequence diagrams for each usecase
   7.2) Develop Detailed Sequence Diagrams / Communication diagrams for each use case showing interactions among all the threelayerobjects
8. For each case study:
   8.1) Develop highlevel sequence diagrams for each usecase1
   8.2) Develop Detailed Sequence Diagrams / Communication diagrams for each use case showing interactions among all the threelayerobjects
9. For each case study:
   9.1) Develop highlevel sequence diagrams for each usecase
   9.2) Develop Detailed Sequence Diagrams / Communication diagrams for each use case showing interactions among all the threelayerobjects
10. For each case study:
    10.1) Develop highlevel Collaboration diagrams for each usecase
    10.2) Develop detailed collaboration diagrams for each use case showing interactions among all the threelayerobjects
11. For each case study:
    11.1) Develop Use case Packages.
    11.2) Develop component diagrams.
    11.3) Develop sample diagrams for other UML diagrams state chart diagrams, activity diagrams and deployment diagrams.
12. Design UML models for the following application.RestaurantSystem.

Additional practice:

Design UML models for the following Case studies.

1. Banking Application System.
2. Railway Reservation System
3. Online Movie Ticket BookingSystem.

Text Books:

1 Object-oriented analysis and design using UML, Mahesh P. Matha,PHI, ISBN: 9788120333222
2 Head first object-oriented analysis and design, Brett D. McLaughlin, Gary Pollice, Dave West,O‟Reilly. ISBN: 9780596008673

Reference Books:

1 Object-oriented analysis and design with the Unified process, John W. Satzinger, Robert B. Jackson, Stephen D. Burd, CengageLearning, ISBN: 978-0619216436
2 The Unified modeling language Reference manual, James Rumbaugh, Ivar Jacobson, Grady Booch, Addison Wesley, ISBN: 9780321245625
3 Object Oriented Analysis & Design, AtulKahate, The McGraw-Hills Companies, ISBN: 9780070583764

Web Links:

1 https://onlinecourses.nptel.ac.in/noc24_cs40/preview
2 https://www.quora.com/in/What-are-the-best-website-to-study- UML-for-beginners
3 https://creately.com/lp/uml-diagram-tool/

WEB Application Development using MERN Stack

	L	T	P	C
Course Code: 241AI034	**0**	**0**	**2**	**2**

Course Outcomes:
At the end of the course, student will be able to:

- **CO1:** Make use of HTML elements, CSS styles and their attributes for designing static web pages.
- **CO2:** Apply JavaScript to develop dynamic web pages and validate forms.
- **CO3:** Build a basic web server using Node.js and also working with Node Package Manager (NPM) & Make use of Typescript to optimize JavaScript code using strict type checking
- **CO4:** Make use of router, template engine and authentication using sessions to develop application in Express JS & components, props, stats and render data in ReactJS
- **CO5:** Build a single page application using RESTful APIs in ExpressJS. & Make use of MongoDB queries

Mapping of Course Outcomes with Program Outcomes:

CO/PO	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11
CO1	2	2	3		2			1	2		2
CO2	2	3	2		2			1	2		2
CO3	2	2	3	1	2			1	2		2
CO4	2	3	2	2	2			1	2		2
CO5	2	2	3	1	2			1	2		2

Mapping of Course Outcomes with Program Specific Outcomes:

CO/PSO	PSO1	PSO2
CO1	2	
CO2	2	
CO3	2	
CO4	2	
CO5	2	

Practice:

1. Lists, Links, Images, Tables, Forms.
   a. Write a HTML program, to explain the working of lists.
   Note: It should have an ordered list, unordered list, nested lists and ordered list in an unordered list and definition lists.

b. Write a HTML program, to explain the working of hyperlinks using <a> tag and href, target Attributes.
   Note: Use text to link https://www.aec.edu.in/
   Use image to link https://www.aec.edu.in/?p=Gallery
c. Create a HTML document that has your image and your friend's image with a specific height and width. Also when clicked on the images it should navigate to their respective profiles.
d. Write a HTML program, to explain the working of tables. (use tags: <table>, <tr>, <th>, <td> and attributes: border, rowspan, colspan)
e. Write a HTML program, to explain the working of forms by designing Registration form. (Note: Include text field, password field, number field, date of birth field.

2. HTML 5 and Cascading Style Sheets, Types of CSS & CSS Box Model
   a. Write a HTML program, that makes use of <article>, <aside>, <figure>, <figcaption>, <footer>, <header>, <main>, <nav>, <section>, , tags.
   b. Write a HTML program, to embed audio and video into HTML web page.
   c. Write a program to demonstrate the various ways you can reference a color in CSS.
   d. Write a program, to explain the importance of CSS Box model using
      i. Content ii. Border iii. Margin iv. padding

3. Selector forms & JavaScript-internal and & I/o
   a. Write a program to apply different types of selector forms
      i. Simple selector (element, id, class, group, universal)
      ii. Combinator selector (descendant, child, adjacent sibling, general sibling)
      iii. Write a program to embed internal and external JavaScript in a web page.
      iv. Write a program to explain the different ways for displaying output.

4. Javascript Pre-defined and User-defined Objects, Conditional & Loop Statements
   a. Write a program using document, window, math & string object properties and methods.
   b. Write a program which asks the user to enter three integers, obtains the numbers from the user and outputs HTML text that displays the larger number followed by the words "LARGER NUMBER" in an information message dialog. If the numbers are equal, output HTML text as "EQUAL NUMBERS".
   c. Write a program to print 1 to 10 numbers using for, while and do-while loops.

5. Javascript Functions and Events & Node.js
   a. Design a appropriate function should be called to display
      i. Factorial of that number
      ii. ii. Fibonacci series up to that number
      iii. Write a program to show the workflow of JavaScript code executable by creating web server in Node.js.
      iv. Write a program to transfer data over http protocol using http module.
      v. Create a text file src.txt and add the following content to it. (HTML, CSS, Javascript, Typescript, MongoDB, Express.js, React.js, Node.js)
6. Typescript
   a. Write a program to understand simple and special types.
   b. Write a program to understand function parameter and return types.

7. ExpressJS – Routing, HTTP Methods,Templating
   a. Write a program to define a route, Handling Routes, Route Parameters, Query Parameters and URL building.
   b. Write a program to accept data, retrieve data and delete a specified resource using http methods.
   c. Write a program using templating engine.

8. ExpressJS – Cookies, Sessions, Authentication, Database.
   a. Write a program for session management using cookies and sessions.
   b. Write a program for user authentication.
   c. Write a program to connect MongoDB database using Mangoose and perform CRUD operations
9. ExpressJS – Templating, Form Data, Cookies, Sessions
   a. Write a program using templating engine.
   b. Write a program to work with form data.
   c. Write a program for session management using cookies and sessions.

10. ReactJS – Render HTML, JSX, Props and States
    a. Write a program to render HTML to a web page.
    b. Write a program for writing markup with JSX.
    c. Write a program to work with props and states.

11. ReactJS – Conditional Rendering, Rendering Lists, React Router
    a. Write a program for conditional rendering.
    b. Write a program for rendering lists.
    c. Write a program for routing to different pages using react router.

12. ReactJS – Hooks, Sharing data between Components, To-do list and Quiz & MongoDB – Installation, Configuration
    a. Write a program to understand the importance of using hooks.
    b. Write a program for sharing data between components.

c. Design to-do list application.
d. Install MongoDB and configure ATLAS

Cornerstone Project:

Project 1 : Student Management System (MERN CRUD App)

- **Implementation mainly focuses on**
  - Create, update, view, and delete student records using MongoDB, Express, React, Node.js.
- **Documentation must contain**
  - Problem Definition, Objectives, Analysis& Design, Implementation and Sample Code, Output screens, Conclusion&Enhancement

Project 2: Online Food Ordering System Using MERN

- **Implementation mainly focuses on**
  - Menu display, cart, orders, user login, admin panel.
- **Documentation must contain**
  - Problem Definition, Objectives, Analysis& Design, Implementation and Sample Code, Output screens, Conclusion & Enhancement

Project 3: Personal Portfolio with Admin Dashboard (MERN + Authentication)

- **Implementation mainly focuses on**
  - Edit portfolio sections via a protected admin panel.
- **Documentation must contain**
  - Problem Definition, Objectives, Analysis& Design, Implementation and Sample Code, Output screens, Conclusion & Enhancement

Project 4: To-Do List Application with User Login (MERN + Sessions/JWT)

- **Implementation mainly focuses on**
  - Add, edit, delete tasks, user registration/login, MongoDB storage.
- **Documentation must contain**
  - Problem Definition, Objectives, Analysis& Design, Implementation and Sample Code, Output screens, Conclusion & Enhancement

Project 5: Event Registration Portal (HTML + CSS + JS + MERN Backend)

- **Implementation mainly focuses on**
  - Users register for events; admin can view/manage registrations.
- **Documentation must contain**

- Problem Definition, Objectives, Analysis& Design, Implementation and Sample Code, Output screens, Conclusion &Enhancement

Project 6: E-Commerce Product Catalog Using MERN

- **Implementation mainly focuses on**
  - Product listings, categories, product details, cart basics.
- **Documentation must contain**
  - Problem Definition, Objectives, Analysis & Design, Implementation and Sample Code, Output screens, Conclusion & Enhancement

Project 7: College Information & Admission Portal (MERN Full Stack)

- **Implementation mainly focuses on**
  - Forms, tables, student admission data, admin management.
- **Documentation must contain**
  - Problem Definition, Objectives, Analysis & Design, Implementation and Sample Code, Output screens, Conclusion & Enhancement

Project 8: Blog Posting Platform Using MERN (CRUD + File Upload)

- **Implementation mainly focuses on**
  - Users create posts, add images, edit, comment, delete.
- **Documentation must contain**
  - Problem Definition, Objectives, Analysis & Design, Implementation and Sample Code, Output screens, Conclusion & Enhancement

Project 9: Movie Review and Rating System Using MERN

- **Implementation mainly focuses on**
  - Users can review movies, give ratings, view reviews list.
- **Documentation must contain**
  - Problem Definition, Objectives, Analysis & Design, Implementation and Sample Code, Output screens, Conclusion & Enhancement

Project 10: Employee Record Management (MERN CRUD)

- **Implementation mainly focuses on**
  - Employee details, salary, department, search filters.
- **Documentation must contain**
  - Problem Definition, Objectives, Analysis & Design, Implementation and Sample Code, Output screens, Conclusion & Enhancement

Project 11: Notes Application Using MERN with Authentication

- **Implementation mainly focuses on**
  - Create/edit notes, categorize them, private notes per user.
- **Documentation must contain**
  - Problem Definition, Objectives, Analysis & Design, Implementation and Sample Code, Output screens, Conclusion & Enhancement

Project 12. Online Quiz Application Using MERN (MCQ Based)

- **Implementation mainly focuses on**
  - Admin adds questions; users take quiz; store scores.
- **Documentation must contain**
  - Problem Definition, Objectives, Analysis & Design, Implementation and Sample Code, Output screens, Conclusion & Enhancement

Project 13: Library Book Management System

- **Implementation mainly focuses on**
  - Add books, issue/return system, student records, dashboards.
- **Documentation must contain**
  - Problem Definition, Objectives, Analysis & Design, Implementation and Sample Code, Output screens, Conclusion & Enhancement

Project 14: Hospital Appointment Booking System

- **Implementation mainly focuses on**
  - Users book appointments; doctors manage schedules.
- **Documentation must contain**

Problem Definition, Objectives, Analysis & Design, Implementation and Sample Code, Output screens, Conclusion & Enhancement

Reference Books:

1 Programming the World Wide Web, Robet W Sebesta, Pearson, 7th Edition. ISBN 139789332518827

2 Pro MERN Stack: Full Stack Web App Development with Mongo, Express, React, and Node, Vasan Subramanian, APress, O'Reilly, 2nd edition. ISBN 9781484226520

Web Links:

1 https://infyspringboard.onwingspan.com/en/app/toc/lex_17739732834840810000_shared/overview (HTML5)
2 https://infyspringboard.onwingspan.com/en/app/toc/lex_18109698366332810000_shared/overview (Javascript)
3 https://infyspringboard.onwingspan.com/web/en/app/toc/lex_auth_013843290195951 61635073_shared/overview (Express Js)
4 https://infyspringboard.onwingspan.com/en/app/toc/lex_auth_01317716929471283 2113_shared/overview (MongoDB)

Note:

- Students must submit the Certificate of Completion offered by any Industry.
- The cornerstone project team size shall be four (4) students.
- Students may select any one of the above prescribed projects or a project of their own choice with the prior permission of the Course instructor.
- The SEE – Lab shall be evaluated for 50 marks based on project implementation, oral presentation, 10-minute video presentation, report and viva voce.
- The video presentation should consists of the working procedure of the project along with contribution of each student for a minimum of 2 minutes.
- Finally, the Source code of the cornerstone Project has to be pushed into the Students GitHub repository.

Bigdata Spark
(Common to CSE, IT, AIML & CSE(DS))

	L	T	P	C
Course Code: 241IT008	**0**	**0**	**2**	**2**

Course Outcomes:
At the end of the course, student will be able to:

CO1: Demonstrate Hadoop Installation process.
CO2: Illustrate Hadoop architecture
CO3: Compare Bigdata models
CO4: Implement various Scalable Algorithms
CO5: Illustrate Applications of Bigdata

Mapping of Course Outcomes with Program Outcomes:

CO/PO	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11
CO1	3	2	2	1	2	1					
CO2	2	2	2	1	2						
CO3	2	2	3	1	2	1					
CO4	2		2	1	3						
CO5	2		2	1	3	1					

Mapping of Course Outcomes with Program Specific Outcomes:

CO/PSO	PSO1	PSO2
CO1	2	
CO2	2	
CO3	2	
CO4	2	
CO5	2	

Practice:

1. Set up and Configuration Hadoop Using Cloudera Creating a HDFS System with minimum 1 Name Node and 1 Data Nodes HDFS Commands Self-Learning Topics: Set up Hadoop in Linux Environment

2. Map Reduce Programming Examples Word Count. Union, Intersection and Difference. Matrix Multiplication. Self-Learning Topics: Natural Join Programming Example

3. Implement an iterative PageRank graph algorithm in MapReduce

4. Perform an efficient semi-join in MapReduce. Hint: Perform a semi-join by having the mappers load a Bloom filter from the Distributed Cache, and then filter results from the actual MapReduce data source by performing membership queries against the Bloom filter to determine which data source records should be emitted to the reducers

5. Hive: Introduction Creation of Database and Table, Hive Partition, Hive Built in Function and Operators, Hive View and Index. Self-Learning Topics: Configure Hive Metastore to MySQL

6. Install and Run Hive then use Hive to create, alter, and drop databases, tables, views, functions, and indexes

7. Pig: Pig Latin Basic Pig Shell, Pig Data Types, Creating a Pig Data Model, Reading and Storing Data, Pig Operations Self-Learning Topics

8. Analytics at Large Scale: Libraries of algorithms include SparkMLlib, H20; integrations with TensorFlow and PyTorch

9. Visualization: Connect to data, Build Charts and Analyze Data, Create Dashboard, Create Stories using Tableau Self-Learning Topics: Tableau using web

10. Mongo DB: Installation and Creation of database and Collection CRUD Document: Insert, Query, Update and Delete Document. Self-Learning Topics: HBASE Commands

Additional Practice:

1. Spark: RDD, Actions and Transformation on RDD , Ways to Create -file, data in memory, other RDD. Lazy Execution, Persisti RDD Self-Learning Topics: Machine Learning Algorithms like K-Means using Spark

2. Spark joins: Consider a scenario where 2 datasets of a leading retail client to be joined with one another using Spark joins. Customer dataset: Sales dataset: Schema Details: 101 ravi 1 102 keerth 2 101 Syam 1 101 Geetha 1 103 Dawn 3 101 ravi 1 102 keerth 2 101 Syam 1 101 Geetha 1 103 Dawn 3) 66 Customer schema (customer id,customer name,product id) Sales Schema (product id,product name and price) Join both datasets with common key Product id and print customer id, customer name, product name and price.

3. Matrix-Vector Multiplication by Map Reduce

4. Implement an iterative PageRank graph algorithm in MapReduce

Note: The student must Complete & Submit a Big Data 101 & Big Data - 201 Certificate Courses offered by Infosys Spring board at the end of the Practice Session.

Text Books:

1. Understanding Big data, Chris Eaton, Dirk Deroos et al., McGraw Hill, ISBN : 9789339221270.
2. Professional Hadoop Solutions, Boris Lublin sky, Kevin t. Smith, Alexey Yakubovich, Wiley, ISBN: 9788126551071.

Reference Books:

1. Spark in Action, Marko Bonaci and Petar Zecevic, Manning, ISBN: 978-1617292606
2. PySpark SQL Recipes: With HiveQL, Data frame and Graph frames, Raju Kumar Mishra and Sundar Rajan Raman, A press Media, ISBN: 978-1484243343

Web Links:

1. https://infyspringboard.onwingspan.com/en/app/toc/lex_auth_01256841991858585686_shared/overview
2. https://infyspringboard.onwingspan.com/en/app/toc/lex_auth_01258388119638835242_shared/overview
3. https://infyspringboard.onwingspan.com/en/app/toc/lex_auth_01260526842300825 61692_shared/overview

CI/CD using Devops
(Common to CSE, IT, AIML & CSE (DS))

Semester:	**L**	**T**	**P**	**C**
Course Code: 241CS019	**0**	**0**	**1**	**1**

Course Outcomes:
At the end of the course, student will be able to:

CO1: Summarize the life cycle of agile software development and Develop the fundamental programs using Devops
CO2: Analyze the adoption of Devops in Projects and Process
CO3: Demonstrate the types of continuous integration and continuous delivery in Devops
CO4: Build an automated CI/CD pipeline using a stack of tools
CO5: Develop basic branching and merging in GIT

Mapping of Course Outcomes with Program Outcomes:

CO/PO	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11
CO1	1	3	1		2				2	2	1
CO2	1	2	1		3				2	2	1
CO3	1	3	2	2	2				2	2	1
CO4		1	3		2				2	2	1
CO5	2	1	3		3				2	2	2

Mapping of Course Outcomes with Program Specific Outcomes:

CO/PSO	PSO1	PSO2
CO1	1	
CO2	1	
CO3	2	
CO4	2	
CO5	3	

Practice:

1. Get an understanding of the stages in software development lifecycle, the process models, values and principles of agility and the need for agile software development. This will enable you to work in projects following an agile approach to software development.

2. Get a working knowledge of using extreme automation through XP programming practices of test first development, refactoring and automating test case writing.

3. It is important to comprehend the need to automate the software development lifecycle stages through DevOps. Gain an understanding of the capabilities required to implement DevOps, continuous integration and continuous delivery practices.

4. Configure the web application and Version control using Git using Git commands and version control operations.

5. Configure a static code analyzer which will perform static analysis of the web application code and identify the coding practices that are not appropriate. Configure the profiles and dashboard of the static code analysis tool.

6. Write a build script to build the application using a build automation tool like Maven. Create a folder structure that will run the build script and invoke the various software development build stages. This script should invoke the static analysis tool and unit test cases and deploy the application to a web application server like Tomcat.

7. Configure the Jenkins tool with the required paths, path variables, users and pipeline views.

8. Configure the Jenkins pipeline to call the build script jobs and configure to run it whenever there is a change made to an application in the version control system. Make a change to the background color of the landing page of the web application and check if the configured pipeline runs.

9. Create a pipeline view of the Jenkins pipeline used in Exercise 8. Configure it with user defined messages.

10. In the configured Jenkins pipeline created in Exercise 8 and 9, implement quality gates for static analysis of code.

11. In the configured Jenkins pipeline created in Exercise 8 and 9, implement quality gates for static unit testing.

12. In the configured Jenkins pipeline created in Exercise 8 and 9, implement quality gates for code coverage.

Additional Practice:

1. To fetch and synchronize git repository
2. To perform basic branching and merging in Git
3. To implement form validation

Reference Books:

1 Learning Continuous Integration with Jenkins: A beginner's guide to implementing Continuous Integration and Continuous Delivery using Jenkins - Nikhil Pathania ,Packt publication [https://www.amazon.in/Learning-Continuous-Integration-Jenkins-Pat, ISBN: 978-1785284830

2 Jenkins 2 – Up and Running: Evolve Your Deployment Pipeline for Next Generation Automation – Brent Laster, O'Reilly publication, ISBN: 978-1491979594 [https://www.amazon.in/Jenkins-2-Running-Brent-Laster/dp/1491979593]

Web Links:

1 https://www.geeksforgeeks.org/what-is-ci-cd/

2 https://github.com/nkatre/Free-DevOps-Books-1/blob