St. Xavier's College (Autonomous), Ahmedabad

Syllabus of Semester – II of the following departments under Faculty of Data Science based on Undergraduate Curriculum Framework to be implemented from the Academic Year 2025-26.

DEPARTMENT OF DATA SCIENCE

Category: III

BSc. Artificial Intelligence and Machine Learning (Hons.)

Major Course-3: Cognitive Strategies for Problem Decomposition and Reasoning

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course Title & Code	Credit Distribution of The Course			Eligibility Criteria	Prerequisite(s) of the Course (if any)
	Lecture	Tutorial	Practical / Practice		
Introduction to Searching Strategies in AI	4	0	0	10 + 2 from a recognized board in any stream	Nil

Course Objectives:

After successful completion of the course, students will be able to:

- CO-1: Apply problem decomposition methods and rule-based approaches to solve AI problems.
- CO-2: Analyze game-playing strategies using Minimax algorithm and Alpha–Beta pruning techniques.
- CO-3 Evaluate logical propositions and perform deductive reasoning using propositional and first-order logic.
- CO-4 Develop simple knowledge-based systems and Prolog programs using inference and unification methods.

Learning Outcomes:

At the end of the course, learners will be able to:

1. Explain the concepts of problem decomposition, AND–OR graphs, and rule-based systems.
2. Describe the principles of game theory and AI techniques used in strategic games.
3. Construct truth tables and interpret logical connectives, entailment, and proofs.
4. Represent knowledge using first-order logic and perform reasoning through chaining and Prolog queries.

UNIT	Topics / Subtopics	HOURS
1	**Problem Decomposition and Rule-Based Systems:** Introduction to problem decomposition, Problem decomposition with And–Or graphs, AO* Algorithm (with simple examples), Case studies: DENDRAL, Symbolic Integration (introductory only)	15
2	**Game Theory and AI in Games:** Basics of game theory: Prisoner's dilemma, cooperative vs. competitive games, Board games and strategies (Chess as example), Limited lookahead and evaluation functions, Minimax algorithm with examples, Alpha–Beta pruning (basic working and advantages), Introduction to AlphaGo, AlphaZero (conceptual, no technical details), Games with chance: Backgammon, Scrabble (overview only), AI challenges in contract bridge (discussion only).	15
3	**Deduction and Propositional Logic:** Logical connectives (AND, OR, NOT, IMPLIES, etc.), Truth tables and evaluation of propositions, Concept of entailment and proof, Soundness and completeness (conceptual only)	15
4	**First-Order Logic and Deductive Reasoning:** Terms, domains, and atomic formulas, Quantifiers, formulas, and sentences, Deduction in FOL: forward chaining and backward chaining, Introduction to Prolog programming (facts, rules, and queries), Unification (simple examples), Resolution refutation method (basic idea only), Equality in FOL (conceptual overview)	15

Textbook

A first course in Artificial Intelligence by Deepak Khemani

Reference Books

- Artificial Intelligence a modern Approach by Stuart Russelll and Peter Norvig
- Artificial Intelligence by Kelvin Knight, Eliane Rich and Sivashankar B Nair (3rd Edition), MedTech Science Press

St. Xavier's College (Autonomous), Ahmedabad

Syllabus of Semester – II of the following departments under Faculty of Data Science based on Undergraduate Curriculum Framework to be implemented from the Academic Year 2025-26.

DEPARTMENT OF DATA SCIENCE

BSc. Artificial Intelligence and Machine Learning (Hons.)

Category: III

Major Course-4: Foundations of Object-Oriented Programming with C++

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course Title & Code	Credit Distribution of The Course			Eligibility Criteria	Prerequisite(s) of the Course (if any)
	Lecture	Tutorial	Practical / Practice		
Logic Building with C Programming	2	0	4	10 + 2 from a recognized board in any stream	Basic knowledge of Programming

Course Objectives:

By the end of this course, students will be able to understand:

- The foundational concepts of Object-Oriented Programming (OOP) and differentiate it from Procedural Programming.
- The syntax and usage of C++ constructs like data types, keywords, and header files.
- The importance and application of function overloading, default arguments, and inline functions in C++.
- The concepts of classes, objects, access specifiers, and the 'this' pointer for effective data encapsulation.
- The principles of inheritance, virtual functions, and polymorphism for code reusability and flexibility.
- The advanced concepts like operator overloading, templates, and dynamic memory management in C++.

Learning Outcomes:

Upon successful completion, students will be able to implement:

- Clearly differentiate between procedure-oriented and object-oriented programming approaches.
- Demonstrate a theoretical understanding of C++ syntax, including keywords, header files, and reference variables.
- Analyze and explain the role of constructors, destructors, and memory management techniques.
- Articulate the concept of inheritance, its types, and the use of virtual functions for runtime polymorphism.

- Explain operator overloading, its rules, and its applications in enhancing the readability of C++ programs.
- Discuss the benefits and limitations of templates for achieving code reusability and type safety.

UNIT	Topics/Subtopics	Hours
1	**Overview of Object-Oriented Programming** ▪ Introduction to Object Oriented Programming ▪ Procedure Oriented and Object-Oriented ▪ Difference Between C and C++ ▪ C++ Output/ Input ▪ Keywords in C++ ▪ New style of header file specification ▪ Reference Variables in C++ ▪ The bool Data type ▪ Importance of function prototyping in C++ ▪ Function Overloading ▪ Default Arguments ▪ Inline Function ▪ Scope Resolution Operator **Classes and Object** ▪ Structure in C++ ▪ Access Specifier ▪ Classes ▪ Objects in C++ ▪ Characteristics of Access Specifier ▪ Function outside a class ▪ Initialization of variable in C++ ▪ Arrow Operator ▪ 'this' pointer **More on Classes and Objects** ▪ Member Functions and Data Members ▪ Friend Functions ▪ Friend Class ▪ Array of Class Object ▪ Passing Class Objects to Function ▪ Returning Objects from Functions ▪ Nested Classes ▪ Namespaces **Dynamic Memory Management** ▪ Introduction ▪ Dynamic Memory Allocation Using "new" ▪ Dynamic Memory Deallocation ▪ "Set_New_Handler" Function	15

Constructor and Destructor

- Constructor
- Characteristics of Constructor
- Types of Constructors
- Destructor

Characteristics of Destructor

- Writing basic C++ programs with loops, conditions and functions
- Creating Simple classes and objects along with access specifiers
- Implementing
  - Friend Function and Friend Class,
  - DMM
  - constructors and its types
  - destructors.

2 30

Inheritance and Virtual Functions

- Introduction
- Advantages of Inheritance
- 'Protected' Access specifier
- Inheritance using different access specifier
- Initialization of Base class members through derived class object
- Different forms of Inheritance
- Function Overriding
- Introduction
- Pointers to derived class
- Rules for virtual function
- Internals of Virtual Functions
- Pure virtual function
- Virtual Base class
- Virtual destructor

3

- Abstract class

15

- Limitations of virtual
- Function Early binding v/s Late binding

Operator Overloading

- Introduction
- Operators that can be overloaded
- Overloading Unary Operator using member Functions
- Overloading Unary Operator using friend Functions
- Overloading Binary Operator using member Functions
- Overloading Binary Operator using friend Functions
- Why Overload Operators using friend function?
- Rules for Operator Overloading

Constructor- Destructor Invocation

- Introduction
- Order of Invocation of Constructors and destructors
- Destructors in Action

- Type Conversions

Templates

- Introduction
- Function Templates
- Function Templates with multiple parameters
- Overloading Function Template
- Class Template
- Class Template with multiple parameters
- Nested Class Templates
- Advantages of using Templates

- Practical implementation on:
  - Inheritance and its types
  - Virtual functions
  - Operator Overloading
  - Constructor and destructor in inheritance
  - Templates

4		30

Textbook: Object Oriented Programming with C++, Publication: Pearson by Subhash KU

Reference Books:

1. Object-Oriented Programming with C++ (Second Edition)
   Publication: PHI By Poornachandra Sarang

2. Object Oriented Programming using C++
   Publication: Cengage Learning By Joyce Farrell

3. Object Oriented Programming In C++
   Publication: Wiley India edition By Rajesh K. Shukla

Sample Programs for Object-Oriented Programming with C++

1. Write a program to calculate the area of circle, rectangle and square using function overloading
2. Write a program to demonstrate the use of default arguments in function overloading.
3. Write a program to demonstrate the use of returning a reference variable.
4. Create a class student which stores the detail about roll no, name, marks of 5 subjects, i.e. science, Mathematics, English, C, C++. The class must have the following:
   a. Get function to accept value of the data members.
   b. Display function to display values of data members.
   c. Total function to add marks of all 5 subjects and store it in the data members named total.

5. Create a function power () to raise a number m to power n. the function takes a double value fo m and int value for n, and returns the result correctly. Use the default value of 2 for n to make the function calculate squares when this argument is omitted. Write a main that gets the values of m and n from the user to test the function.

6. Write a basic program which shows the use of scope resolution operator.

7. Write a C++ program to swap the value of private data members from 2 different classes.

8. Write a program to illustrate the use of **this** pointer.

9. An election is contested by five candidates. The candidates are numbered 1 to 5 and the voting is done by marking the candidate number on the ballot paper. Write a program to read the ballots and count the votes cast for each candidate using an array variable count. In case a number is read outside the range of 1 to 5, the ballot should be considered as a 'spoilt ballot' and the program should also count the number of spoilt ballots.

10. Write a program to call member functions of class in the main function using pointer to object and pointer to member function.

11. Using friend function find the maximum number from given two numbers from two different classes. Write all necessary functions and constructors for the program.

12. Using a friend function, find the average of three numbers from three different classes

13. Write all necessary member functions and constructor for the classes.

14. Define currency class which contains rupees and paisa as data members. Write a friend function named AddCurrency ( ) which add 2 different Currency objects and returns a Currency object. Write parameterized constructor to initialize the values and use appropriate functions to get the details from the user and display it.

15. Create Calendar class with day, month and year as data members. Include default and parameterized constructors to initialize a Calendar object with a valid date value. Define a function AddDays to add days to the Calendar object. Define a display function to show data in "dd/mm/yyyy" format.

16. Create a class named 'String' with one data member of type char *, which stores a string. Include default, parameterized and copy constructor to initialize the data member. Write a program to test this class.

17. Write a base class named Employee and derive classes Male employee and Female Employee from it. Every employee has an id, name and a scale of salary. Make a function ComputePay (in hours) to compute the weekly payment of every employee. A male employee is paid on the number of days and hours he works. The female employee gets paid the wages for 40 hours a week, no matter what the actual hours are. Test this program to calculate the pay of employee.

18. Create a class called scheme with scheme_id, scheme_name, outgoing_rate, and message_charge. Derive customer class form scheme and include cust_id, name and mobile_no data. Define necessary functions to read and display data. Create a menu driven program to read call and message information for a customer and display the detail bill.

19. Write a program with use of inheritance: Define a class publisher that stores the name of the title. Derive two classes book and tape, which inherit publisher. Book class contains member data called page no and tape class contain time for playing. Define functions in the appropriate classes to get and print the details.

20. Create a class account that stores customer name, account no, types of account. From this derive classes cur_acc and sav_acc to include necessary member function to do the following:
    - Accepts deposit from customer and update balance
    - Compute and Deposit interest
    - Permit withdrawal and Update balance.

21. Write a base class named Employee and derive classes Male employee and Female Employee from it. Every employee has an id, name and a scale of salary. Make a function ComputePay (in hours) to compute the weekly payment of every employee. A male employee is paid on the number of days and hours he works. The female employee gets paid the wages for 40 hours a week, no matter what the actual hours are. Test this program to calculate the pay of employee.

22. Create a class vehicle which stores the vehicleno and chassisno as a member. Define another class for scooter, which inherits the data members of the class vehicle and has a data member for a storing wheels and company. Define another class for which inherits the data member of the class vehicle and has a data member for storing price and company. Display the data from derived class. Use virtual function.

23. Create a base class shape. Use this class to store two double type values that could be used to compute the area of figures. Derive two specific classes called triangle and rectangle from the base shape. Add to the base class, a member function get_data() to initialize the base class data members and another member function display_area() to compute and display the area of figures. Make display_area() as a virtual function and redefine this function in the derived class to suit their requirements.

24. Write a program to demonstrate the use of pure virtual function.

25. For multiple inheritance, write a program to show the invocation of constructor and destructor.

26. Create a class string with character array as a data member and write a program to add two strings with use of operator overloading concept.

27. Create a class distance which contains feet and inch as a data member. Overhead = =, <and> operator for the same class. Create necessary functions and constructors too.

28. Create a class MARIX of size mxn. Overload + and – operators for addition and subtraction of the MATRIX.

29. Define a class Coord, which has x and y coordinates as its data members. Overload ++ and – operators for the Coord class. Create both its prefix and postfix forms

30. Create one class called Rupees, which has one member data to store amount in rupee and create another class called Paise which has member data to store amount in paise. Write a program to convert one amount to another amount with use of type conversion.

31. Create two classes Celsius and Fahrenheit to store temperature in terms of Celsius and Fahrenheit respectively. Include necessary functions to read and display the values. Define conversion mechanism to convert Celsius object to Fahrenheit object and vice versa. Show both types of conversions in main function.

32. Write a program to create a function template for finding maximum value contained in an array.

33. Write a program to create a class template for the 'Array' class. 3 Create a template for the bubble sort function.

34. Write a program to illustrate the use of insertion and extraction operators for Text mode Input/Output.

35. Write a program to illustrate the use of put(), get() and getline() functions for Text mode Input/Output.

36. Write a program to illustrate the use of read() and write() functions for Binary mode Input/Output.

37. Write a program to illustrate the use of manipulators in file handling.

38. Write a program to illustrate the use of file pointer manipulation functions.
39. Write down a program to Copy source file 'source.txt' to destination file.
40. A file contains a list of telephone numbers in the following format:

    (a) Ram 47890
    (b) Krishna 878787
    …….
    (n)_____

The names contain only one word and the names and telephone numbers are separated by white space. Write a Program to read the tel.dat file and display the content. The names should be left justified and the number right-justified.