Provisional Syllabus

Curriculum for Diploma in Artificial Intelligence [Choice Based Credit System]

Semester	Subject Code		Credit
I	AICOR01T	Programming Fundamental using Python	4
	AICOR01P	Programming Fundamental using Python	2
	AICOR02T	Computer Fundamental	6
	GE1	Mathematics for AI / Introduction to Artificial Intelligence	6
II	AICOR03T	Data Structure	4
	AICOR03P	Data Structure	2
	AICOR04T	Computer System Architecture	6
	GE2	Python Programming for AI/Machine Learning Fundamentals	6
III	AICOR05T	Advanced Python Programming for AI	4
	AICOR05P	Advanced Python Programming for AI	2
	AICOR06T	Deep Learning Basics	6
	GE3	Natural Language Processing (NLP)/Computer Vision Basics	6
IV	AICOR07T	AI Ethics and Responsible AI	6
	AICOR08P	AI Applications and Project	6
	GE4	Advanced AI Algorithms/Big Data Analytics for AI	6
Total Number Of Courses		16	72

Course Objectives

1. **Knowledge of Computer Fundamentals and Python Programming:**
    - Impart knowledge about basic computer fundamentals.
    - Teach Python programming required for implementing AI and ML applications.
2. **Understanding History and Principles of AI:**
    - Provide an understanding of the history of AI.
    - Teach the basic principles of modern AI.
3. **Basics of Machine Learning, Artificial Neural Networks, and Deep Learning:**
    - Educate students on the basics of machine learning.
    - Introduce artificial neural networks.
    - Cover fundamental concepts of deep learning.
4. **Applications of AI and ML/DL:**
    - **Explain the uses of AI and ML/DL in various fields:**
        - Natural Language Processing (NLP)
        - Speech recognition
        - Image Processing & Computer Vision
        - Weather predictions
        - Medicine and healthcare
        - Economics
        - eCommerce
        - Government law and policy-making
        - Environmental sustainability
        - Chatbots and ChatGPT
5. **Ethics in AI:**
    - Discuss the ethical considerations in AI.
6. **Practical Experience with AI and ML Tools:**
    - Provide hands-on experience in handling various AI and ML tools.
    - Implement real-world applications using these tools.

Course Outcomes

Upon successful completion of this course, students will be able to:

1. **History and Foundations of AI:**
    - Demonstrate an understanding of the history of AI and its foundations.
2. **Applications Awareness and Understanding:**
    - Demonstrate awareness and a fundamental understanding of various applications of AI and ML in real-world scenarios including:
        - Natural Language Processing (NLP)
        - Speech recognition
        - Image Processing & Computer Vision
        - Weather predictions
        - Medicine and healthcare
        - Economics
        - eCommerce
        - Government law and policy-making
        - Environmental sustainability
        - Chatbots and ChatGPT
3. **Proficiency in Developing AI and ML Applications:**
    - Demonstrate proficiency in developing various real-world AI and ML applications using existing Python-based tools.
4. **Discussion and Societal Impact:**

- **Demonstrate the ability to engage in discussions about AI and ML, including:**
  - Current scope and limitations
  - Impact on society

Foundation of AI: -

History of AI, What is natural intelligence? What is Artificial Intelligence(AI)? Strong AI vs. weak AI.

AI agent, An architecture of an AI agent(a block diagram and short description of each component).

● Relationships between AI, Machine Learning(ML), and Deep Learning(DL).
● What is Machine Learning? Difference between traditional programming and Machine Learning. Different types of Machine Learning. Advantages of ML over DL.
● Basic steps of ML system design- problem understanding, data acquisition, Features, Data representation, modeling using approaches like rule-based, supervised learning, unsupervised learning, and Reinforcement Learning(short description of each modeling approach with simple examples).

Concept of Supervised Learning :-

● Supervised learning - a block diagram with a short description,
● regression and classification with simple examples.
● Common supervised classifiers- K-Nearest Neighbour search algorithm (in detail),
● Decision tree classifier (basic idea only, no induction algorithm),

Preliminary Concept of Artificial Neural Network :-

Neural Network- biological motivation, comparison between Artificial Neuron and Human Neuron.
● Artificial Neuron as a processing unit. Perceptron learning rule for updating weights of an artificial neuron. Limitation of a perceptron in solving XOR problem,
● Multilayer feedforward neural network (only a diagram showing interconnections among neurons at multiple layers).
● High-level description of Forward pass and backward pass of the backpropagation (BP) algorithm used for training.
● Multilayer feedforward neural network (No mathematical derivation).
● How is deep learning (DL) related to Artificial Neural Networks? Difference between shallow and deep learning.

This First section mentioned in above is focus for the 6 months a.i. course

CORE COURSES ARTIFICIAL INTELLIGENCE (DIPLOMA)

AICOR01T : Programming Fundamental using Python **Theory: 60 Lectures**

1. **Introduction to Python** **(5 Lectures)**

    a. History and evolution of Python
    b. Installing Python and setting up the development environment
    c. Basics of procedural programming in Python using the main() function
    d. Writing and executing simple Python programs

2. **Variables, Data Types, and Basic Input/Output** **(8 Lectures)**

    a. Declaring, defining, and initializing variables in Python
    b. Understanding data types (integer, float, string, boolean) in Python
    c. Using named constants and keywords
    d. Input and output operations in Python (input(), print())
    e. Using basic built-in functions and modules (e.g., math module)

3. **Operators and Expressions** **(6 Lectures)**

    a. Arithmetic operators (+, -, *, /, %, etc.)
    b. Comparison operators (==, !=, <, >, <=, >=)
    c. Logical operators (and, or, not)
    d. Bitwise operators (&, |, ^, ~, <<, >>)
    e. Operator precedence and associativity in Python

4. **Control Structures: Conditionals** **(7 Lectures)**

    a. Using if statements for conditional execution
    b. Nested if-else statements and elif ladder
    c. Ternary conditional operator (conditional expression)

5. **Control Structures: Loops** **(8 Lectures)**

    a. Using while loop for repetitive execution
    b. Using for loop with range() function
    c. Nested loops and loop control statements (break, continue)
    d. Looping techniques and patterns

6. **Functions in Python** **(8 Lectures)**

    a. Understanding functions and their benefits
    b. Defining and calling functions in Python
    c. Function arguments (positional, keyword, default, variable-length)
    d. Returning values from functions
    e. Scope of variables (global vs. local scope)

7. **Lists, Tuples, and Dictionaries** **(8 Lectures)**

    a. Introduction to data structures in Python
    b. Lists: Creating, indexing, slicing, adding/removing elements
    c. Tuples: Creating, accessing elements, immutability
    d. Dictionaries: Creating, accessing values, dictionary methods

8. Strings and File Handling (5 Lectures)

a. Working with strings: String methods, formatting strings
b. Reading from and writing to files in Python
c. File modes, file objects, file handling operations

AICOR01P : Programming Fundamental using Python Practical: 60 Lectures

1. Write a Python program to print the sum and product of digits of an integer.
2. Write a Python program to reverse a number.
3. Write a Python program to compute the sum of the first n terms of the series S = 1 + 1/2 + 1/3 + 1/4 + ...
4. Write a Python program to compute the sum of the first n terms of the series S = 1 - 2 + 3 - 4 + 5 ...
5. Write a Python function that checks whether a given string is a palindrome or not. Use this function to check if a string entered by the user is a palindrome.
6. Write a Python function to determine whether a given number is prime or not. Use the same function to generate prime numbers less than 100.
7. Write a Python program to compute the factors of a given number.
8. Write a Python program with a macro that swaps two numbers. Use the macro in the program.
9. Write a Python program to print a triangle of stars based on the number of lines entered by the user.
10. Write a Python program to perform various actions on an array entered by the user, such as printing even-valued elements, odd-valued elements, sum and average of elements, maximum and minimum element, removing duplicates, printing in reverse order, and providing a menu to the user to choose these options.
11. Write a Python program that prints a table indicating the number of occurrences of each alphabet in the text entered as command line arguments.
12. Write a Python program that swaps two numbers using pointers.
13. Write a Python program where a function is passed the address of two variables and then alters their contents.
14. Write a Python program that takes the radius of a circle as input from the user, computes the area and circumference, and displays the values from the main program.
15. Write a Python program to find the sum of n elements entered by the user using dynamic memory allocation.
16. Write a menu-driven Python program to perform various operations on strings, such as showing the address of each character, concatenating strings, comparing strings, calculating string length using pointers, converting case, counting vowels, and reversing the string.
17. Write a Python program that merges two ordered arrays of integers to get an ordered array.
18. Write a Python program to display the Fibonacci series using recursion and iteration.
19. Write a Python program to calculate the factorial of a number using recursion and iteration.
20. Write a Python program to calculate the GCD of two numbers with and without recursion.
21. Create a Matrix class using templates in Python. Write a menu-driven program to perform matrix operations such as sum, difference, product, and transpose.
22. Create a Python program using a structure for Student containing fields like Roll No., Name, Class, Year, and Total Marks. Create 10 students and store them in a file.
23. Write a Python program to retrieve student information from the file created in the previous question and print it in a specific format.
24. Open the file created in a previous question, append five more records, and print all records.
25. Write a menu-driven Python program to create a sequential file, print records, and add records.
26. Write a Python program to copy the contents of one text file to another file after removing all whitespaces.

AICOR02T : Computer Fundamental Theory : 75 Lectures

1. Introduction (6 Lectures)

a. Functional Units of a Computer System

b. Different Types of Computers (Desktops, Laptops, Servers, etc.)
c. Software and Hardware Overview
d. Types of Software: System Software and Application Software
e. Introduction to Operating Systems (OS)
f. Basic Functions of an Operating System
g. Different Types of Operating Systems (Windows, macOS, Linux, etc.)

2. Data Representation (8 Lectures)

a. Base or Radix in Number Systems
b. Number Systems: Decimal, Binary, Octal, and Hexadecimal
c. Conversion Between Number Systems
d. Binary Arithmetic: Addition and Subtraction
e. Integer Representation: Signed Magnitude, 1's Complement, 2's Complement
f. Character Representation in Computers
g. Floating Point Representation in Computers

3. Computer Memory (6 Lectures)

a. Overview of Computer Memory Hierarchy
b. Primary Memory: RAM (Random Access Memory) and ROM (Read-Only Memory)
c. Secondary Storage: Hard Drives, Solid-State Drives (SSDs), Flash Drives, etc.
d. Cache Memory: Levels and Functions
e. Virtual Memory and Paging
f. Memory Management in Operating Systems

4. Input and Output Devices (6 Lectures)

a. Types of Input Devices: Keyboards, Mice, Touchscreens, etc.
b. Types of Output Devices: Monitors, Printers, Speakers, etc.
c. Peripheral Devices: Scanners, Webcams, Microphones, etc.
d. Device Drivers and Interfaces
e. Human-Computer Interaction (HCI) Principles

5. Computer Networks (8 Lectures)

a. Introduction to Computer Networks
b. Types of Networks: LAN (Local Area Network), WAN (Wide Area Network), WLAN (Wireless LAN), etc.
c. Network Topologies: Bus, Star, Ring, Mesh, etc.
d. Network Protocols: TCP/IP, HTTP, FTP, etc.
e. Network Security Basics: Firewalls, Encryption, Authentication

6. Internet Basics (8 Lectures)

a. History and Evolution of the Internet
b. Understanding URLs, Domain Names, and IP Addresses
c. Web Browsers and Search Engines
d. Email Communication: Clients and Servers
e. Online Safety and Security: Phishing, Malware, Password Management

7. Software Basics (8 Lectures)

a. Types of Software: System Software (Operating Systems, Utility Programs) and Application Software
b. Word Processing Software: Microsoft Word, Google Docs, etc.
c. Spreadsheet Software: Microsoft Excel, Google Sheets, etc.
d. Presentation Software: Microsoft PowerPoint, Google Slides, etc.
e. Database Management Software: Microsoft Access, MySQL, etc.

AICOR03T : Data Structure **Theory: 60 Lectures**

1. Introduction to Data Structures (6 Lectures)

- **A.** Data Object and Abstract Data Type (ADT)
- **B.** Definition, types, and properties of data structures
- **C.** Basics of data types and their significance in programming

2. Arrays (5 Lectures)

- **A.** Single and Multi-dimensional Arrays
- **B.** Sparse Matrices (Array and Linked List Representation)
- **C.** Operations on arrays, such as searching and sorting

3. Stacks (5 Lectures)

- **A.** Introduction to stacks and their operations
- **B.** Implementing stacks using arrays and linked lists
- **C.** Infix, Prefix, and Postfix expressions and their conversion using stacks

4. Linked Lists (8 Lectures)

- **A.** Singly, Doubly, and Circular Linked Lists
- **B.** Operations on linked lists (insertion, deletion, traversal)
- **C.** Applications of linked lists and their advantages over arrays

5. Queues (5 Lectures)

- **A.** Introduction to queues and their operations (enqueue, dequeue)
- **B.** Circular queue, Deque (Double-ended queue), Priority Queue
- **C.** Implementing queues using arrays and linked lists

6. Recursion (5 Lectures)

- **A.** Understanding recursion and recursive algorithms
- **B.** Recursive definition of problems and their implementation
- **C.** Advantages, limitations, and internal stack implementation of recursion

7. Trees (20 Lectures)

- **A.** Introduction to trees as a hierarchical data structure
- **B.** Binary Trees, Binary Search Trees (BST), operations like insertion, deletion, and traversal
- **C.** AVL Trees for balanced binary search trees, concepts of height-balanced trees

8. Searching and Sorting (5 Lectures)

- **A.** Linear Search and Binary Search algorithms
- **B.** Comparison of linear and binary search efficiency
- **C.** Sorting algorithms: Selection Sort, Insertion Sort, Bubble Sort, their comparison, and performance analysis

9. Hashing (5 Lectures)

- **A.** Introduction to Hashing and its applications
- **B.** Collision resolution techniques: Open Addressing (Linear Probing, Quadratic Probing), Separate Chaining

C. Choosing a Hash Function and handling collisions, Perfect Hashing Function overview

AICOR03P : Data Structure Lab **Practical: 60 Lectures**

1. Write a program to search an element from a list. Give user the option to perform Linear or Binary search. Use Template functions.
2. WAP using templates to sort a list of elements. Give user the option to perform sorting using Insertion sort, Bubble sort or Selection sort.
3. Implement Linked List using templates. Include functions for insertion, deletion and search of a number, reverse the list and concatenate two linked lists (include a function and also overload operator +).
4. Implement Doubly Linked List using templates. Include functions for insertion, deletion and search of a number, reverse the list.
5. Implement Circular Linked List using templates. Include functions for insertion, deletion and search of a number, reverse the list.
6. Perform Stack operations using Linked List implementation.
7. Perform Stack operations using Array implementation. Use Templates.
8. Perform Queues operations using Circular Array implementation. Use Templates.
9. Create and perform different operations on Double-ended Queues using Linked List implementation.
10. WAP to scan a polynomial using linked list and add two polynomial.
11. WAP to calculate factorial and to compute the factors of a given no. (i)using recursion,
(ii) using iteration
12. (ii) WAP to display Fibonacci series (i)using recursion, (ii) using iteration
13. WAP to calculate GCD of 2 number (i) with recursion (ii) without recursion
14. WAP to create a Binary Search Tree and include following operations in tree:
i. Insertion (Recursive and Iterative Implementation)
ii. Deletion by copying
iii. Deletion by Merging
iv. Search a no. in BST
v. Display its preorder, postorder and inorder traversals Recursively
vi. Display its preorder, postorder and inorder traversals Iteratively
vii. Display its level-by-level traversals
viii. Count the non-leaf nodes and leaf nodes
ix. Display height of tree
x. Create a mirror image of tree
xi. Check whether two BSTs are equal or not
15. WAP to convert the Sparse Matrix into non-zero form and vice-versa.
16. WAP to reverse the order of the elements in the stack using additional stack.
17. WAP to reverse the order of the elements in the stack using additional Queue.
18. WAP to implement Diagonal Matrix using one-dimensional array.
19. WAP to implement Lower Triangular Matrix using one-dimensional array.
20. WAP to implement Upper Triangular Matrix using one-dimensional array.
21. WAP to implement Symmetric Matrix using one-dimensional array.
22. WAP to create a Threaded Binary Tree as per inorder traversal, and implement operations like finding the successor / predecessor of an element, insert an element, inorder traversal.
23. WAP to implement various operations on AVL Tree.

Recommended Books:

1. Adam Drozdek, "Data Structures and algorithm in C", Third Edition, Cengage Learning, 2012.
2. SartajSahni, Data Structures, "Algorithms and applications in C", Second Edition, Universities Press, 2011.
3. Aaron M. Tenenbaum, Moshe J. Augenstein, YedidyahLangsam, "Data Structures Using C and C:, Second edition, PHI, 2009.
4. Robert L. Kruse, "Data Structures and Program Design in C", Pearson,1999.
5. D.S Malik, Data Structure using C,Second edition, Cengage Learning, 2010.
6. Mark Allen Weiss, "Data Structures and Algorithms Analysis in Java", Pearson Education, 3rd edition, 2011

7. Aaron M. Tenenbaum, Moshe J. Augenstein, YedidyahLangsam, "Data Structures Using Java, 2003.
8. Robert Lafore, "Data Structures and Algorithms in Java, 2/E", Pearson/ Macmillan Computer Pub, 2003
9. John Hubbard, "Data Structures with JAVA", McGraw Hill Education (India) Private Limited; 2 edition, 2009
10. Goodrich, M. and Tamassia, R. "Data Structures and Algorithms Analysis in Java", 4th Edition, Wiley, 2013
11. Herbert Schildt, "Java The Complete Reference (English) 9th Edition Paperback", Tata McGraw Hill, 2014.
12. 12. D. S. Malik, P.S. Nair, "Data Structures Using Java", Course Technology, 2003.

AICOR04T : Computer System Architecture **Theory: 75 Lectures**

1. Introduction **(5 Lectures)**

Combinational logic circuit and sequential logic circuit (basic concepts with example)

2. Data Representation and Basic Computer Arithmetic **(13 Lectures)**

Number systems, complements, fixed and floating point representation, character representation, addition, subtraction, magnitude comparison, multiplication and division algorithms for integers

3. Basic Computer Organization and Design **(15 Lectures)**

Registers, bus system, instruction set, timing and control, instruction cycle, memory reference, input -output and interrupt, Interconnection Structures, Bus Interconnection design of basic computer.

4. Central Processing Unit (20 Lectures)

Register organization, arithmetic and logical micro-operations, stack organization, micro programmed control. Instruction formats, addressing modes, instruction codes, machine language, assembly language, input output programming, RISC, CISC architectures, pipelining and parallel architecture.

5. Memory Organization (12 Lectures)

Main Memory, Auxiliary Memory, Associative memory, Cache memory,mapping.

6. Input-Output Organization (10 Lectures)

Input / Output: External Devices, I/O Modules, Programmed I/O, Interrupt-Driven I/O, Direct Memory Access, I/O Channels, I/OP

Recommended Books:

1. M. Mano, Computer System Architecture, Pearson Education, 1992
2. A. J. Dos Reis, Assembly Language and Computer Architecture using C and JAVA, Course Technology, 2004
3. W. Stallings, Computer Organization and Architecture Designing for Performance, 8 Edition, Prentice Hall of India, 2009
4. Carl Hamacher, Computer Organization, Fifth edition, McGrawHill, 2012.

AICOR05T : Advanced Python Programming for AI **Theory: 60 Lectures**

1. **Introduction to Python for AI** **(5 Lectures)**

    a. Overview of Python programming language
    b. Python syntax, variables, and data types
    c. Control structures: loops and conditional statements
    d. Functions and modules in Python

2. **Data Handling and Manipulation in Python (10 Lectures)**

    a. Working with data structures: lists, tuples, dictionaries
    b. File handling and input/output operations in Python
    c. Data manipulation techniques using libraries like NumPy and Pandas

3. **Object-Oriented Programming in Python (15 Lectures)**

    a. Introduction to object-oriented programming (OOP) concepts
    b. Classes, objects, methods, and attributes in Python
    c. Inheritance, polymorphism, and encapsulation in Python OOP

4. **Advanced Python Concepts (15 Lectures)**

    a. Decorators and their usage in Python
    b. Generators and iterators for efficient memory usage
    c. Exception handling and error management in Python

5. **Working with External Libraries for AI (15 Lectures)**

    a. Introduction to AI libraries like TensorFlow, Keras, and Scikit-Learn
    b. Integration of external libraries with Python for AI development
    c. Hands-on exercises and projects using AI libraries

Recommended Books:

Mark Lutz, "Learning Python," O'Reilly Media
Wes McKinney, "Python for Data Analysis," O'Reilly Media
Sebastian Raschka, "Python Machine Learning," Packt Publishing
François Chollet, "Deep Learning with Python," Manning Publications

AICOR05P : Advanced Python Programming for AI Practical : 60 Lectures

1. Write a program to calculate the factorial of a number using recursion.
2. Implement a function to perform linear search on a list and return the index of the element if found.
3. Implement a function to perform binary search on a sorted list and return the index of the element if found.
4. Create a program to sort a list of integers using the bubble sort algorithm.
5. Develop a function to find the GCD (Greatest Common Divisor) of two numbers using Euclid's algorithm.
6. Write a program to calculate the Fibonacci series up to a specified limit.
7. Implement a basic calculator program that can perform addition, subtraction, multiplication, and division operations.
8. Create a function to check if a given number is prime or not.
9. Develop a program to convert a decimal number to binary and vice versa.
10. Write a function to find the sum of digits of a given number recursively.
11. Implement a simple text-based tic-tac-toe game where two players can take turns to make moves.
12. Create a program to generate a random password of a specified length containing letters, numbers, and special characters.
13. Develop a function to reverse a given string using recursion.
14. Write a program to count the frequency of each word in a given text file.
15. Implement a function to calculate the area of various geometric shapes (circle, rectangle, triangle) based on user input.
16. Develop a program to find the square root of a number using the Newton-Raphson method.
17. Write a function to check if a given string is a palindrome or not.

18. Implement a simple student management system using dictionaries to store student information (name, roll number, marks, etc.) and perform CRUD operations.
19. These exercises cover a range of fundamental programming concepts such as recursion, sorting algorithms, mathematical calculations, file handling, string manipulation, and basic data structures. They provide a hands-on approach to learning Python programming while also preparing students for more advanced topics in AI development.

AICOR06T : Deep Learning Basics **Theory : 75 Lectures**

1. **Introduction to Deep Learning** **(5 Lectures)**
   a) What is deep learning?
   b) Brief history and evolution of deep learning
   c) Applications of deep learning in various fields
   d) Overview of neural networks and their role in deep learning
   e) Introduction to popular deep learning frameworks like TensorFlow and PyTorch
2. **Neural Networks Fundamentals** **(10 Lectures)**
   a) Basics of artificial neurons and activation functions
   b) Structure and architecture of neural networks (input layer, hidden layers, output layer)
   c) Feedforward propagation and backpropagation algorithms
   d) Understanding loss functions and optimization algorithms (e.g., gradient descent)
3. **Deep Learning Models** **(15 Lectures)**
   a) Introduction to different types of deep learning models:
   b) Fully connected neural networks
   c) Convolutional Neural Networks (CNNs) for image data
   d) Recurrent Neural Networks (RNNs) for sequential data
   e) Use cases and examples of deep learning models in practice
4. **Training Deep Learning Models** **(10 Lectures)**
   a) Data preprocessing techniques for deep learning tasks
   b) Model training, validation, and testing procedures
   c) Hyperparameter tuning and regularization techniques
   d) Evaluating model performance metrics (accuracy, precision, recall, F1 score)
5. **Applications of Deep Learning** **(10 Lectures)**
   a) Deep learning applications in computer vision: Image classification, Object detection, Image segmentation

b)Deep learning applications in natural language processing (NLP): Text classification, Sentiment analysis, Named entity recognition

6. **Deep Learning Tools and Libraries (10 Lectures)**

   a) Hands-on introduction to deep learning frameworks:

   b) TensorFlow basics and implementation of neural networks

   c) PyTorch basics and building deep learning models

   d) Keras for rapid prototyping of deep learning architectures

7. **Challenges and Future Trends in Deep Learning (5 Lectures)**

   a) Common challenges in deep learning (overfitting, vanishing gradients, etc.)

   b) Recent advancements and trends in deep learning research

   c) Ethical considerations and biases in deep learning algorithms

Recommended Books:

"Deep Learning" by Ian Goodfellow, Yoshua Bengio, and Aaron Courville

"Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow" by Aurélien Géron

"Deep Learning with Python" by François Chollet

AICOR06T : AI Ethics and Responsible AI **Theory : 75 Lectures**

1. **Introduction to AI Ethics (5 Lectures)**
   a. Definition and importance of AI ethics
   b. Ethical considerations in AI development and deployment
   c. Examples of ethical dilemmas in AI
2. **Foundations of Responsible AI (10 Lectures)**
   a. Principles of responsible AI design (fairness, transparency, accountability)
   b. Bias and fairness in AI algorithms
   c. Transparency in AI decision-making processes
3. **Ethical AI Development (15 Lectures)**
   a. Data privacy and security in AI systems
   b. Ethical considerations in data collection, storage, and usage
   c. Ethical implications of AI-powered decision-making
4. **AI and Social Impact (10 Lectures)**
   a. AI's impact on society, economy, and job market
   b. Ethical issues in AI applications (surveillance, misinformation, social biases)
   c. AI and inequality: addressing societal challenges

5. **Regulatory and Legal Frameworks for AI** **(10 Lectures)**
    a. Overview of global AI regulations and guidelines
    b. Ethical AI policies and standards
    c. Compliance and ethical audits in AI projects
6. **AI Ethics in Practice** **(10 Lectures)**
    a. Case studies of ethical AI implementations and failures
    b. Best practices for integrating ethics into AI projects
    c. Ethical decision-making frameworks for AI professionals

Recommended Resources:

"Ethics of Artificial Intelligence" by Nick Bostrom, Oxford University Press

"Responsible AI: A Global Policy Framework" by The IEEE Global Initiative on Ethics of Autonomous and Intelligent Systems

"The Age of AI: And Our Human Future" by Henry A. Kissinger, Eric Schmidt, and Daniel Huttenlocher

"Ethics and Artificial Intelligence: The Moral Compass of a Machine" by Vincent C. Müller (Editor), Springer