Course Outline Artificial Intelligence
Department of Computer Science
Faculty of Computing & Information Technology
Hafiz Hayat Campus, University of Gujrat

	Title	**Artificial Intelligence**
	Code	**CS-329**
	Credit hours	**3 (2+1)**
	Prerequisite	None
	Category	Core
	Course Description	This course will introduce the basics of artificial intelligence (AI), its scope and application domain. The course will cover topics such as the history of AI, intelligent agents, state-space problem representations, uninformed and heuristic search, game playing and adversarial search, logical agents, constraint satisfaction problems, Genetic Algorithms, Decision Tree, Probabilistic Reasoning and Neural Network etc.
	Aims & Objectives	• To introduce the principles of AI methods importance, and application of Artificial Intelligence techniques in solving problems • To equip students with the developments, justifications, implementation, and use of representational, formalism and search methods. • Understanding and Applying to Problem Solving Using Intelligent Adversarial Games Strategies. • Understanding and Applying to Problem Solving Using Statistical Learning. • Understanding and Applying Problem Solving Using Probabilistic Methods.
	Learning Outcomes	• Enabling students to study advanced courses in the field such as machine learning, neural networks, and probabilistic decision making. • Ability to apply knowledge representation, reasoning, and learning techniques to real-world problems by developing programs using the Python language.
	Text Book	• Artificial Intelligence: A modern approach. Russell and Norvig, 3rd edition Pearson Education Series in AI

	Reference Books & Material	• Luger, George & Stubblefield, William, Artificial Intelligence: Structures and Strategies for Complex Problem Solving (6th ed.), • Mathematical Methods in Artificial Intelligence. (Edward A. Bender). • Principals of Artificial Intelligence and Expert Systems Development. (David W. Rolston) • Nils J Nilson, Artificial Intelligence – A New Synthesis, Morgan Kaufman Publishers, Elsevier, USA. • Patrick Henry Winston, Artificial Intelligence, Third Edition, Pearson Education Series in AI.
	Grading Breakup and Policy	Assignment(s): 10%Quizzes: 5% Project: 10%Midterm Examination: 25% Final Examination: 50%
	Plagiarism Policy:	Collaboration and group work is encouraged but each student is required to submit his/her own contribution(s). Your writings must be your own thoughts. Cheating and plagiarism will not be tolerated and will be referred to the HoD & Dean for appropriate action(s).

16 Week Plan

Week	Lectures	Theory Topics	Source (Book, Chapter No)	Recommended Learning Activities & Lab Work
1	1, 2	Introduction to AI: History, Applications, Future. Intelligent Agents: Agents, Environments, Rationality, Types.	A) Chapter 1, B) Chapter 2	Lab Work: Introduction to Python for AI. Set up the development environment and write basic scripts for data handling.
2	3, 4	Knowledge Representation with AI: Propositional Logic, Predicate Calculus.	B) Chapter 2, A) Chapter 7	Lab Work: Implement a simple agent in Python that demonstrates rational behavior in a defined environment.
3	5, 6	Uninformed Search: State-Space Search, DFS, BFS, Iterative Deepening, Uniform-Cost Search.	B) Chapter 3	Lab Work: Implement and visualize DFS and BFS to solve a search problem, such as finding a path in a maze.
4	7, 8	Evaluating Search Strategies: Complexity, Completeness, Optimality. Informed Search: Heuristically Informed Methods.	A) Chapter 3	Lab Work: Compare the performance of uninformed vs. informed search by measuring solution cost and time.
5	9, 10	Informed Search: Greedy Search and A* Search.	A) Chapter 3	Lab Work: Implement A* search for a pathfinding problem (e.g., 8-puzzle).
6	11, 12	Informed Search (Continued): Hill Climbing, Beam Search. Best First Search.	A) Chapter 3	Lab Work: Implement Hill Climbing to find a local optimum for a given problem.
7	13, 14	Adversarial Search: Game Playing, Min-Max Procedure.	A) Chapter 4, 5	Lab Work: Develop a basic two-player game (e.g., Tic-Tac-Toe) and implement the Min-Max algorithm.
8	15, 16	Game Playing (Continued): Static Evaluation Function, Alpha-Beta Pruning.	A) Chapter 4, 5	Lab Work: Enhance the previous week's game with an Alpha-Beta Pruning implementation to improve its efficiency.
9	17, 18	Expert Systems: Intro, Knowledge Base, Components, Types, Applications, Forward/Backward Chaining.	A) Chapter 7, 8, 9	Lab Work: Build a small rule-based expert system in Python using an if-elif-else structure to model a simple decision-making process.
10	19, 20	Machine Learning Fundamentals: Supervised Learning, Decision Tree, Information Theory, Feature Selection,	A) Chapter 9	Lab Work: Develop a decision tree classifier from scratch and apply it to a classification dataset.

		Entropy, Information Gain.		
11	21, 22	Advanced Decision Trees: Gain Ratio, Gini Index. Probabilistic Reasoning: Bayesian Theorem, Handling Uncertainty.	A) Chapter 10, 13	Lab Work: Implement a Naïve Bayes classifier to handle a simple classification task with uncertainty.
12	23, 24	Machine Learning for Classification: Naïve Bayes, SVM (Optional). Probabilistic Reasoning for Continuous Variables.	A) Chapter 19, 20 + online resources	Lab Work: Use a machine learning library (e.g., scikit-learn) to apply and compare a Naïve Bayes and SVM model to a new dataset.
13	25, 26	Neural Networks: Intro to ANN, Applications, Topologies, Single-Layer Perceptron (SLP).	A) Chapter 11	Lab Work: Use a deep learning library like TensorFlow or PyTorch to build and train a basic feedforward neural network.
14	27, 28	Backpropagation Algorithm: Multi-Layer Perceptron (MLP), Implementation for AND, OR, XOR.	A) Chapter 12	Lab Work: Implement an MLP from scratch in Python to solve the XOR problem, demonstrating backpropagation.
15	29, 30	Genetic and Emergent ML: Introduction to Evolutionary Computing, Genetic Algorithms (GA), Genetic Operators.	A) Chapter 12	Lab Work: Implement a Genetic Algorithm to solve a classic optimization problem, such as the Eight Queens Problem.
16	31, 32	Introduction to Deep Learning: Convolutional Neural Networks (CNN), RNN. Planning, Computer Vision, and Clustering overview.	Handouts	Lab Work: A project-based lab where students work on a small-scale final project, such as an image classifier using a pre-trained CNN.

Lab Exercises in Detail

Part 1: Foundations and Problem-Solving (Weeks 1-9)

This section focuses on essential programming skills and the implementation of classic AI search algorithms.

Week 1: Introduction to Python for AI

- **Objective:** Learn Python basics and core libraries for AI.
- **Key Concepts:** Python syntax, data types, control flow, functions.
- **Libraries: NumPy** for numerical operations and **Pandas** for data manipulation.
- **Lab Work:** Set up a Python environment (Jupyter Notebook or Google Colab).
- **Exercise:** Write a Python script to demonstrate basic operations and list manipulations. Implement matrix operations using NumPy.
- **Assignment:** Develop a Python script to perform basic operations on NumPy arrays.

Week 2: Uninformed Search Algorithms (DFS, BFS)

- **Objective:** Implement **Depth-First Search (DFS)** and **Breadth-First Search (BFS)**.
- **Key Concepts:** State-space search, graph traversal.
- **Lab Work:** Write Python code to implement both DFS and BFS.
- **Exercise:** Apply both algorithms to a sample state-space graph or a simple maze and compare their output paths.

Week 3: Comparing Search Strategies

- **Objective:** Evaluate different search strategies based on time, space complexity, optimality, and completeness.
- **Key Concepts:** Complexity analysis, optimality, completeness.
- **Lab Work:** Test DFS, BFS, and Uniform-Cost Search on the same graph or problem.

- **Exercise:** Analyze and report on the performance of each algorithm, noting their strengths and weaknesses.

*Week 4: Informed Search Algorithms (A Search)**

- **Objective:** Implement **A* search** and compare it with Greedy Search.
- **Key Concepts:** Heuristics, cost function, admissibility.
- **Lab Work:** Write a Python program for A* search.
- **Exercise:** Implement A* search to solve a pathfinding problem, such as finding the optimal path through a maze. Compare its performance with a Greedy Search on the same problem in terms of nodes explored.

Week 5: Heuristic Methods (Hill Climbing, Beam Search)

- **Objective:** Understand heuristic search methods and their limitations.
- **Key Concepts:** Local optima, greedy approach.
- **Lab Work:** Implement Hill Climbing and Beam Search in Python.
- **Exercise:** Apply these algorithms to a simple optimization problem, such as finding the maximum value in a simple cost landscape. Analyze how different heuristics affect performance.

Week 6: Game Playing and Adversarial Search (Minimax)

- **Objective:** Implement the **Minimax algorithm** for a two-player game.
- **Key Concepts:** Game trees, adversarial search, best move selection.
- **Lab Work:** Create a basic two-player game like Tic-Tac-Toe.
- **Exercise:** Implement the Minimax algorithm for the game to allow the computer to make optimal moves.

Week 7: Static Evaluation and Alpha-Beta Pruning

- **Objective:** Improve game search efficiency using **Alpha-Beta Pruning**.
- **Key Concepts:** Search tree optimization, static evaluation function.
- **Lab Work:** Modify the Tic-Tac-Toe implementation from the previous week.

- **Exercise:** Add Alpha-Beta Pruning to the Minimax algorithm and compare the performance with the standard implementation.

Week 8: Introduction to Expert Systems

- **Objective:** Understand the basics of expert systems and their components.
- **Key Concepts:** Knowledge base, inference engine, rule-based systems.
- **Lab Work:** Create a simple rule-based system using Python.
- **Exercise:** Develop a program that can make decisions based on a set of predefined rules and user input (e.g., a simple medical or animal diagnosis system).

Part 2: Machine Learning and Advanced AI (Weeks 10-16)

This section focuses on practical machine learning models, from decision trees to neural networks, and introduces optimization techniques.

Week 10: Decision Trees for Machine Learning

- **Objective:** Implement a **Decision Tree** classifier.
- **Key Concepts:** Supervised learning, classification, entropy, information gain.
- **Lab Work:** Use the **Scikit-learn** library.
- **Exercise:** Build and evaluate a decision tree on a classification dataset (e.g., the Iris dataset).

Week 11: Probabilistic Reasoning and Naive Bayes

- **Objective:** Implement probabilistic reasoning and the **Naive Bayes classifier**.
- **Key Concepts:** Probability, Bayes' theorem, classification.
- **Lab Work:** Use Scikit-learn for the implementation.
- **Exercise:** Build a Naive Bayes model for a classification task, such as spam detection.

Week 12: Artificial Neural Networks (ANN)

- **Objective:** Build a simple neural network using **TensorFlow** or **Keras**.
- **Key Concepts:** ANN structure, feedforward networks, neurons, layers.
- **Lab Work:** Build a feedforward neural network for a simple classification problem.
- **Exercise:** Implement a neural network for classifying handwritten digits using the MNIST dataset.

Week 13: Backpropagation and MLP

- **Objective:** Understand the **backpropagation algorithm** and **Multi-Layer Perceptrons (MLP)**.
- **Key Concepts:** Training a neural network, loss function, gradient descent.
- **Lab Work:** Implement a simple MLP with backpropagation.
- **Exercise:** Create an MLP that can solve a binary classification problem (e.g., the XOR problem).

Week 14: AI Model Optimization and Hyperparameter Tuning

- **Objective:** Optimize AI models using techniques like grid search and random search.
- **Key Concepts:** Hyperparameters, model optimization.
- **Lab Work:** Implement hyperparameter tuning using Scikit-learn.
- **Assignment:** Optimize the hyperparameters of a machine learning model for a classification task.

Week 15: Genetic Algorithms

- **Objective:** Understand and implement a **Genetic Algorithm**.
- **Key Concepts:** Evolutionary computing, chromosomes, fitness, crossover, mutation.
- **Lab Work:** Implement a Genetic Algorithm to solve a basic optimization problem.
- **Exercise:** Solve a classic optimization problem like the Traveling Salesman Problem using a Genetic Algorithm.

Week 16: Final Lab Project and Submission

- **Objective:** Apply all learned techniques to a real-world AI project.

- **Key Concepts:** Problem-solving, project management, integration of AI techniques.
- **Lab Work:** Develop a final project (e.g., an image classifier, a chatbot, or a medical diagnosis system).
- **Assignment:** Present the completed project along with a detailed report.

Grading Policy

- Lab Assignments: 40%
- Final Lab Project: 60%

References and Further Reading

- **Books:**
    - "Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow" by Aurélien Géron.
    - "Deep Learning" by Ian Goodfellow, Yoshua Bengio, and Aaron Courville.
- **Online Resources:**
    - Coursera: "Machine Learning" by Andrew Ng.
    - TensorFlow and Keras documentation.