University of Kotli, AJK

Department of CS&IT

Assignment # 1
AI- Class: BS(IT)-7th
Total marks 10
Due Date: **10/02/26**
Course Instructor: Mrs. Naila Javed

Task 1.

1. What is Artificial Intelligence? How does it differ from Machine Learning?

2. Describe the Turing Test. What is its significance in AI?

3. What are the main categories of AI? Explain with examples.

Task 2.

1. Define an Intelligent Agent. What are its components?

2. Explain the PEAS (Performance, Environment, Actuators, Sensors) model with an example agent.

3. What are the types of Agents? Explain:

- Simple Reflex Agent
- Model-Based Reflex Agent
- Goal-Based Agent
- Utility-Based Agent

Task 3.

1. What are the characteristics of an environment in AI?

2. Explain the following environmental properties with examples:

- Fully Observable vs Partially Observable
- Deterministic vs Stochastic
- Episodic vs Sequential
- Static vs Dynamic
- Discrete vs Continuous
- Single Agent vs multi-Agent

Task 4.

1. Design a simple AI agent for a vacuum cleaner. Describe its PEAS components and environmental properties.

2. Compare a human driver and a self-driving car as AI agents. How do their environments differ?

3. For a chess-playing AI, what type of agent is it? What are its environmental properties?

Task 5.

Design and develop an AI Agent:

Problem: Create a simple AI agent to solve the "Tire Change" problem.

Description:

- The agent is a robot tasked with changing a flat tire on a car.

- The environment consists of the car, a flat tire, a spare tire, and tools (jack, wrench).

- The agent's goal is to replace the flat tire with the spare tire.

Requirements:

1. Define the agent's PEAS components.

2. Implement the agent's decision-making process using a simple algorithm (e.g., simple reflex or goal-based).

3. Write a Python program to simulate the agent's actions.

Deliverables:

- PEAS components description

- Algorithm pseudocode

- Python code implementation

Example Environment:

- Car with flat tire

- Spare tire

- Tools (jack, wrench)

^^^^^^^^^^^^^^^^^^^^^^^^

University of Kotli, AJK

Department of CS&IT

Assignment # 2

AI---Class: BS(IT)-7th

Total marks 10

Due Date: **20/02/26**

Course Instructor: Mrs. Naila Javed

Task 01

1. What is the key difference between informed and uninformed search algorithms? Give an example of each.

2. Explain why Breadth-First Search (BFS) is guaranteed to find the shortest path in an unweighted graph.

3. What is the main advantage of A* over Greedy Best-First Search?

4. Describe a scenario where Depth-First Search (DFS) would be preferred over BFS.

5. How does the heuristic function impact the performance of A* algorithm?

6. Compare Uniform Cost Search (UCS) and A* in terms of optimality and completeness.

7. What is the "informedness" in informed search? How does it affect search efficiency?

8. Explain the concept of "completeness" in search algorithms. Give an example of a complete and an incomplete algorithm.

9. How does Iterative Deepening Depth-First Search (IDDFS) combine benefits of BFS and DFS?

10. When would you choose Greedy Best-First Search over A*

Task 02

1. Given a map with cities and distances, apply A* to find the shortest path from City A to City B.

2. Explain why BFS is suitable for finding the shortest path in an unweighted graph.

3. For a maze-solving robot, which search algorithm would you use and why?

Task 03

1. What are the properties of a good search algorithm?

2. Compare the following:

   - Completeness

   - Optimality

   - Time Complexity

   - Space Complexity

For BFS, DFS, and A* algorithms.

Task 04 (challenging Task)

1. 8-Puzzle Problem: Implement A* to solve the 8-puzzle game.

2. Tic-Tac-Toe: Use Minimax algorithm to create an AI opponent.

3. Maze Solver: Use BFS or A* to navigate a maze.

4. N-Queens Problem: Implement a search algorithm to place N queens on a chessboard.

Task 05

Implement Breadth-First Search (BFS) to traverse the following graph and find the shortest path from Node A to Node G.

Graph:

A -> B, C

B -> D, E

C -> F

D -> G

E -> G

F -> G

Requirements:

1. Write a Python function bfs(graph, start, goal) that:

- Takes graph (as adjacency list), start node, and goal node
- Returns the shortest path from start to goal

2. Test with start=A, goal=G

Output: Shortest path: A -> B -> D -> G (or equivalent)

/* Top position is always vacant for HARD WORKERS */

Bonus Marks :

Students who are willingly present their work in class will get bonus marks inshaAllah.

Types of Searches in AI:

In AI (especially **searching in problem-solving**), algorithms are commonly split into:

- **Uninformed (Blind) search**: don't use extra knowledge about how close you are to the goal.
- **Informed (Heuristic) search**: use a **heuristic** (an "estimate") to guide the search toward the goal faster.

Uninformed (Blind) Search Algorithms

These only know:

- the start state
- how to generate neighbors (successors)
- how to test the goal

List:

1. **Breadth-First Search (BFS)**
2. **Depth-First Search (DFS)**
3. **Depth-Limited Search (DLS)**
4. **Iterative Deepening Depth-First Search (IDDFS / IDS)**
5. **Uniform Cost Search (UCS)** (Dijkstra-style for graphs with costs)
6. **Bidirectional Search** (search from start and goal toward each other)

Informed (Heuristic) Search Algorithms

These use a heuristic function like **h(n)** = estimated distance/cost from node *n* to goal.

List:

1. **Greedy Best-First Search** (uses only **h(n)**)
2. **A*** Search (uses **f(n) = g(n) + h(n)**)
3. **IDA*** (Iterative Deepening A*)
4. **RBFS** (Recursive Best-First Search)
5. **SMA*** (Simplified Memory-bounded A*)
6. **Hill Climbing**
7. **Beam Search**
8. **Simulated Annealing** (often treated as heuristic/optimization search)

Quick memory trick

- **Uninformed**: "No clue where goal is" → explores systematically (BFS/DFS/UCS).
- **Informed**: "Has a clue" (heuristic) → tries to move toward goal (Greedy/A*/Hill climbing