National University of Computer and Emerging Sciences
Lahore Campus

AI Lab (AL2002)
Date: May 29the 2025
Total Questions: 3

Final Exam

Total Time (Hrs): 2.5
Total Marks: 45

Roll No ____________ Section ____________ Student Signature ____________

Submission Path: \\exam\Final Exam\Artificial Intelligence\YourSection

---

Attempt all the questions. No quires shell be entertained. If you have any confusion, write the assumptions in your code as comments. Drag and Drop the file into your respective submission folder, do not try to open it. Write your code in a single rollNo.ipynb file. The folder will close automatically on time, hence submission on time, submission of correct rollNo.ipynb file is your responsibility. If you fail to do so, you will be awarded with zero marks.

Q 1: Simple Linear Regression Using NumPy and Matplotlib (No sklearn) 15 Marks

You are provided with a CSV file named `student_scores.csv`, which contains two columns: `Hours_Studied` and `Exam_Score`. You are required to implement simple linear regression from scratch using only NumPy and Matplotlib.

(a) **Load the Dataset:** Read the dataset using NumPy and display the first five rows.

(b) **Summary Statistics:** Calculate and display summary statistics including mean, minimum, and maximum for each column.

(c) **Scatter Plot:** Create a scatter plot with `Hours_Studied` on the x-axis and `Exam_Score` on the y-axis. Add axis labels and a title.

(d) **Linear Regression Model:** Implement the following steps to fit the linear regression line:

- Compute the slope (m) and intercept (c) using:

$$m = \frac{n\sum xy - \sum x \sum y}{n\sum x^2 - (\sum x)^2}, \qquad c = \frac{\sum y - m\sum x}{n}$$

- Use the line equation $\hat{y} = mx + c$ to generate predictions.
- Overlay the regression line on the scatter plot.

(e) **Model Evaluation:** Calculate and print the following metrics:

- **Mean Squared Error (MSE):**

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

- **R-squared (R^2) Score:**

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \overline{y})^2}$$

(f) **Prediction:** Use the trained model to predict the exam score for a student who studied for 7.5 hours. Substitute $x = 7.5$ into the regression line equation $\hat{y} = mx + c$ to find the predicted score.

CLO #: Implement and evaluate basic machine learning algorithms using fundamental programming libraries.

Q 2: Genetic Algorithm to Maximize a Mathematical Function (15 Marks)

You are required to implement a Genetic Algorithm (GA) from scratch to maximize the following function:

$$f(x) = x \cdot \sin(10\pi x) + 1.0, \quad \text{where } x \in [0, 1]$$

(a) **Initialization:**

- Initialize a population of size 10.
- Each individual is a floating-point value $x \in [0, 1]$.

(b) **Fitness Function:**

- Define fitness as $f(x) = x \cdot \sin(10\pi x) + 1.0$
- This function ensures the fitness is always non-negative.

(c) **Selection:**

- Use roulette wheel selection to probabilistically select parents based on their fitness values.

(d) **Crossover:**

- Apply arithmetic crossover between two parents by averaging or interpolating their x values to produce two new individuals

(e) **Mutation:**

- Apply mutation to each gene with a small probability (e.g., 0.1).
- Mutation adds a small random noise sampled uniformly from the range $[-0.05, 0.05]$.

(f) **Generations:**

- Run the GA for 50 generations.
- In each generation:
  - Evaluate the fitness of all individuals.
  - Print the best fitness value of the current generation.

(g) **Final Output:**

- After 50 generations, print the best solution x and its corresponding fitness $f(x)$.

CLO #: Implement evolutionary optimization algorithms for solving non-linear objective functions.

Q 3: Pathfinding – DFS and BFS Algorithms (15 Marks)

A maze is represented as a 2D grid where 0 indicates an open cell and 1 indicates a wall. The objective is to navigate from a given start point to a goal point using two classical uninformed search strategies: Depth-First Search (DFS) and Breadth-First Search (BFS).

(a) **Maze Representation:** Use the following 6×6 grid:

$$\texttt{maze} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Start: $(0, 0)$ **Goal:** $(5, 5)$

(b) **Algorithm Implementation:**

- Implement the Depth-First Search (DFS) algorithm to find a path from start to goal.
- Implement the Breadth-First Search (BFS) algorithm to find the shortest path from start to goal.

(c) **Evaluation Metrics:**

- Display the complete path found by each algorithm as an ordered list of coordinates.
- Print the total number of nodes visited during the search process.

- Compute and print the path cost, defined as the number of steps (or length of the path).

(d) **Comparison and Discussion:**

- Compare the performance of DFS and BFS with respect to:
  - Path optimality (shortest path vs possible suboptimal path)
  - Search efficiency (nodes explored)
  - Memory and time usage characteristics (depth vs breadth preference)
- Conclude which algorithm performs better for this specific maze and justify your answer based on the outputs observed.

CLO #: Analyze and implement uninformed search algorithms to solve maze-based pathfinding problems.