Shiraz University
Department of Computer Science
Course Title: Evolutionary Computing (2025)
Assignment 1: The 8-Queens Problem
Due Date: [Set by instructor]

Goal

While implementation may be supported by general-purpose programming tools, the analytical, comparative, and explanatory sections must reflect the student's own understanding.

Problem Overview

The 8-Queens problem requires placing 8 queens on a chessboard such that none attack each other. You will implement a Genetic Algorithm (GA) tailored for this problem and evaluate its performance under different conditions and modifications.

Tasks

1. GA Implementation

Implement a GA to solve the 8-Queens problem using these baseline settings:

- **Representation**: Permutation of 0 to 7 (rows)
- **Crossover**: Cut-and-Fill
- **Mutation**: Swap
- **Parent Selection**: Best 2 of Random 5
- **Population Size**: 100
- **Offspring per generation**: 2
- **Initialization**: Random
- **Termination**: Solution found or after 10,000 evaluations

Requirement: Modular, readable, and documented code. Use **Python** or **MATLAB** with only **NumPy**, **Matplotlib**, and **Pandas**.

---

2. Parameter Sensitivity

- Try **3 mutation probabilities**: 20%, 50%, 100%
- Try **2 recombination rates**: 50%, 100%

- Implement mutation in two modes: **swap** and **bitwise mutation** (if applicable)
- Report convergence speed and success rate for each case
- Discuss which settings worked best in your runs and why. Include graphs.

---

3. Crossover Strategy Exploration

- Compare **Cut-and-Fill** crossover with **PMX (Partially Mapped Crossover)**
- Implement both and analyze:
    - Which one leads to faster or more stable convergence?
    - How does the solution quality vary between the two?
- Additionally:
    - Implement and compare **2-cut** and **3-cut** crossover
    - Discuss: Does increasing the number of cuts lead to significantly different results?

---

4. Survival Strategy Comparison

- Implement and compare:
    - **Generational Replacement** (all replaced)
    - **Elitism** (2 best survive)
- For each, run the algorithm 3 times and record results
- Analyze:
    - Which converges faster?
    - Which finds better-quality solutions more consistently?
    - When might one be preferred over the other?

---

5. Scalability Study

- Extend your code to solve **N-Queens** for N = 10, 12, 20
- For each N:
    - Report number of generations, evaluations, and solution quality
    - Compare scaling behavior: what gets worse and why?

Submission Requirements

- **One ZIP file**: Lastname-StudentNumber.zip
- Must include:
    - **PDF report**
    - **All source code** (readable and runnable)
    - **Figures/plots as needed**
- Submit via **Quera**

Report Guidelines

- Explain your implementation choices clearly
- Discuss results beyond just showing graphs
- Reflect on what worked, what didn't, and what you learned

Academic Honesty Note

Use of tools like GPT is allowed only for code assistance and concept explanation. Analytical sections and experiments must be your own. Identical results or explanations will be flagged.

Good luck | Think evolutionary | Submit thoughtfully