Cover Page

Project Title: Genetic Algorithms for Function Optimization

Date of Submission: ……………………………………………

Shared Folder: ………………………………………………….

	Student Name (In Arabic)	Student ID	Level	Department
1				
2				
3				
4				
5				
6				

Introduction and Overview

1. **Project Idea**:

   - The project implements a genetic algorithm (GA) to optimize mathematical functions, specifically benchmark optimization problems like Sphere, Ackley, and Rosenbrock functions.
   - These functions represent complex optimization challenges in AI and are widely used to evaluate algorithm performance.

2. **Applications**:

- Optimization problems in engineering design, scheduling, and machine learning.
- Examples:
  - Optimization in neural networks (hyperparameter tuning).
  - Industrial applications for cost minimization or performance maximization.

3. **Similar Applications**:

- MATLAB Optimization Toolbox.
- Python libraries like DEAP and PyGAD.

4. **Literature Review**:

- [1] Goldberg, D. E. "Genetic Algorithms in Search, Optimization, and Machine Learning."
- [2] "Benchmark Functions for Global Optimization," Surjanovic and Bingham.
- [3] Ackley's "A Connectionist Approach to Numerical Optimization."
- [4] Griewank, "Generalized Descent for Global Optimization."
- [5] Raschka, S., and Mirjalili, "Python Machine Learning."

Proposed Solution & Dataset

1. **Main Functionalities**:

- User-defined input for benchmark functions and parameters.
- GA runs iteratively, evolving solutions to optimize the function.
- Visual feedback through fitness history plots.

2. **Dataset**:

- Benchmark functions (Sphere, Ackley, Rosenbrock, etc.).
- These are mathematical equations, so no external dataset is required.

Applied Algorithms

1. **Genetic Algorithm Process**:

- **Initialization**: A random population of genomes is generated.
- **Fitness Evaluation**: Each genome is evaluated using the benchmark function.
- **Selection**: The top-performing genomes are selected for breeding.
- **Crossover**: Pairs of genomes are combined to create new solutions.
- **Mutation**: Genomes are randomly altered to maintain diversity.
- **Termination**: The algorithm stops when the fitness goal is achieved or the maximum number of generations is reached.

2. **Key Parameters**:

- Population size: Determines how many solutions are explored per generation.
- Mutation rate: Controls the randomness in genome alteration.
- Goal fitness: Defines the optimization target.

Experiments & Results

1. **Experiment Design**:

   - Run the algorithm on five benchmark functions (Sphere, Ackley, Rosenbrock, Rastrigin, and Griewank).
   - Use different parameter combinations (e.g., population size, mutation rate) for analysis.

2. **Results**:

   - Plot fitness evolution over generations for each function.
   - Tabulate best solutions for each benchmark function.

Analysis, Discussion, and Future Work

1. **Analysis**:

   - Identify patterns in convergence rates across different benchmark functions.
   - Discuss parameter impacts (e.g., larger populations improve diversity but increase runtime).

2. **Advantages and Disadvantages**:

   - Advantages: Simple, adaptable, and effective for diverse optimization problems.
   - Disadvantages: Slow convergence for high-dimensional or complex landscapes.

3. Future Work:

- Implement advanced selection methods (e.g., tournament selection).
- Parallelize computations to speed up execution.
- Explore hybrid algorithms combining GAs with other optimization techniques.