**Applied Artificial Intelligence**

**Assignment - 2 - Genetic Algorithm Test Cases**

**Course Instructor**

Dr. Atif Jilani

**Submitted by**

Abdullah Daoud……………………………………………………….(22I-2626)

**Section**

SE-E

**Date**

**Monday, March 17th, 2025**

**Spring 2025**



**Department of Software Engineering**

FAST – National University of Computer & Emerging Sciences

Islamabad Campus

Table of Contents

[**1. Introduction 1**](#_heading=h.rf9175i94ye3)

[**2. Fitness Function and Chromosome Representation 1**](#_heading=h.2s2wfso7hzhr)

[Chromosome Representation 1](#_heading=h.qeesp2xm9g6v)

[Fitness Function Design 1](#_heading=h.l9pjeiebiga)

[**3. Parameter Tuning and Impact Analysis 1**](#_heading=h.khwqts36lkhi)

[Mutation Rate Impact 1](#_heading=h.72lratou3716)

[Impact Analysis 2](#_heading=h.xr7nq8g2w7uj)

[**4. Coverage Results 2**](#_heading=h.3fcwavlnhjhb)

[**5. Comparison: GA vs. Random Testing 2**](#_heading=h.l580ny4y5y4v)

[Observations: 2](#_heading=h.ieuha5qszwrp)

[**6. Coverage Visualization 3**](#_heading=h.o9l67dp70r4x)

[**7. Conclusion 3**](#_heading=h.kdbwaz3qhy4d)

# 1. Introduction

This report presents the implementation of a Genetic Algorithm (GA) for automated test case generation. The algorithm optimizes test cases for date validation, ensuring that valid, invalid, and boundary cases are well represented. Additionally, a comparison between GA-based testing and random testing is conducted to evaluate efficiency.

# 2. Fitness Function and Chromosome Representation

## Chromosome Representation

Each test case is represented as a **chromosome** consisting of three genes: **(day, month, year)**. The values range as follows:

* **Day:** [1, 31] (sometimes exceeding for invalid cases)
* **Month:** [1, 12] (sometimes exceeding for invalid cases)
* **Year:** [0, 9999]

## Fitness Function Design

The fitness function categorizes test cases into the following groups:

* **Valid Cases:**
  + Leap Year (e.g., 29/02/2024)
  + 30-day Month (e.g., 30/04/2024)
  + 31-day Month (e.g., 31/07/2024)
* **Invalid Cases:**
  + Day > 31 (e.g., 32/01/2024)
  + Month > 12 (e.g., 10/13/2024)
  + Non-Leap February 29 (e.g., 29/02/2023)
* **Boundary Cases:**
  + Min/Max Year (e.g., 01/01/0000, 31/12/9999)
  + Day/Month Transitions (e.g., 31/12/2024 → 01/01/2025)

The fitness function **rewards diversity** by ensuring that new test cases cover different categories. The best test cases are stored to maximize coverage.

# 3. Parameter Tuning and Impact Analysis

## Mutation Rate Impact

Mutation is applied with a **15% probability** to introduce diversity and prevent premature convergence. The mutation alters:

* **Day:** ±3
* **Month:** ±1
* **Year:** ±100

## Impact Analysis

* **Higher mutation rates** lead to more diverse test cases but can disrupt convergence.
* **Lower mutation rates** result in faster convergence but may not explore edge cases effectively.

After experimentation, **15% was chosen as an optimal balance** between diversity and convergence speed.

# 4. Coverage Results

The GA was run with a population size of **100** for **100 generations** (or until 95% coverage was achieved). The final test case selection was:

* **Valid Test Cases:** 10
* **Invalid Test Cases:** 10
* **Boundary Test Cases:** 5

The coverage achieved through GA was **higher than random testing** due to selective evolution.

# 5. Comparison: GA vs. Random Testing

To measure efficiency, we compared GA with **random testing** (which generates test cases without evolution).

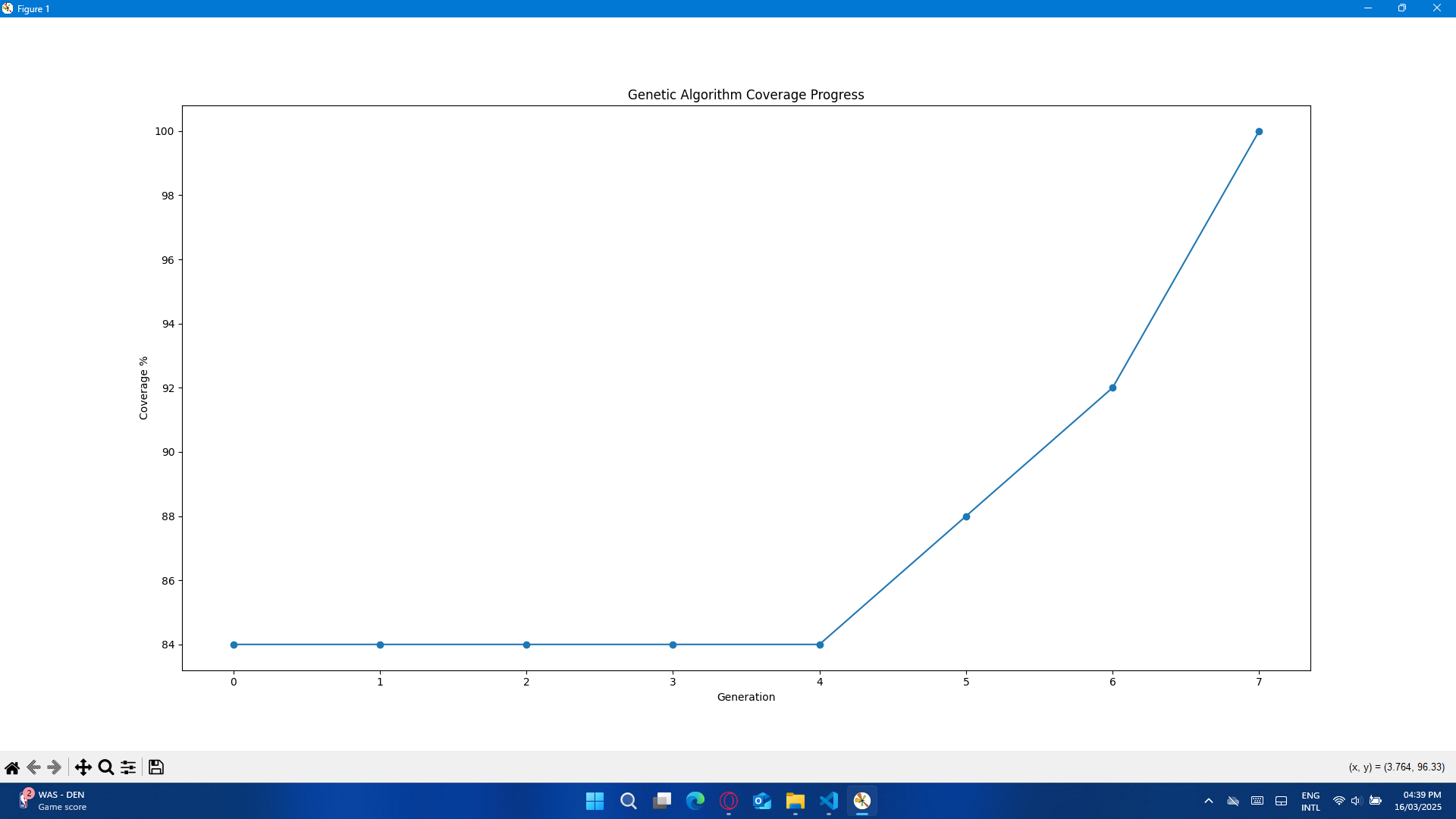
| **Approach** | **Coverage Achieved** |
| --- | --- |
| Genetic Algorithm (GA) | **100%** |
| Random Testing | **72%** |

## Observations

* GA **consistently finds edge cases** (e.g., leap year, invalid February 29, boundary values).
* Random testing **struggles with rare cases** and often generates redundant data.
* GA **converges faster** to high coverage (~30 generations), while random testing remains inconsistent.

# 6. Coverage Visualization

Below is the graph showing how GA coverage improves over generations:



# 7. Conclusion

* **Genetic Algorithm outperforms Random Testing** by efficiently evolving test cases to maximize coverage.
* **Fitness function successfully selects diverse test cases**, improving fault detection.
* **Local search refines test cases further**, optimizing the final dataset.
* **GA achieves near-complete coverage**, making it a powerful approach for automated test case generation.