Course Scheduling using Genetic Algorithm

Team Members

B Manish Reddy	- CB.EN.U4CSE17409
Kiran S. Raj	- CB.EN.U4CSE17430
Sanjay Tharagesh R S	- CB.EN.U4CSE17453
SriShilesh P S	- CB.EN.U4CSE17458

Course Scheduling using Genetic Algorithm

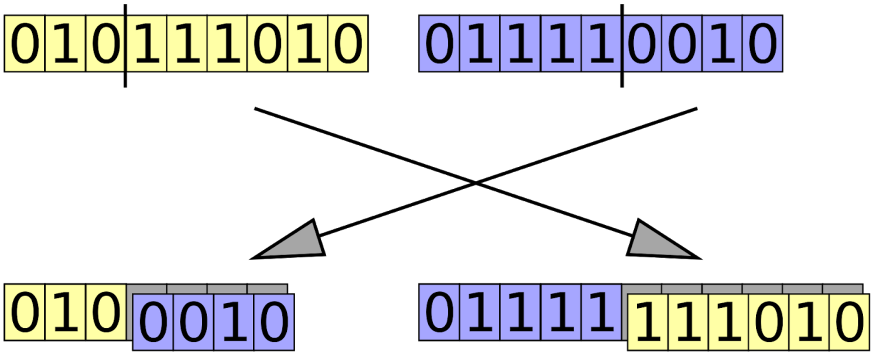
Project Description

Implementation of Course Scheduling plan for educational universities using a Genetic algorithm based on natural selection. When the Courses, Departments, Rooms, Room capacity and the Instructor details are provided, the program will generate a new schedule without any conflicts using the Genetic algorithm.

Objective.

The main objective of this project is to demonstrate our solution to the course scheduling problem. Course scheduling done manually is a tedious process, where the required fields are collected, and the schedule creation must be done without any conflicts. There are usually a very large number of feasible solutions for it. Manual creation of it may lead to imperfections and partiality in favor of a particular person or a group of people. Using Genetic algorithm, which revolves around the process of natural selection, we optimize this problem by choosing the best solution among the available. The solution thus obtained may not be the most perfect but strives to be the most accurate and closest to the desired aim. Thus a course schedule will be created based on the specified fields without any conflicts.

Workflow.



Genetic Algorithm works on Darwin's theory of "Survival Of The Fittest". With that in mind, with the help of genetic algorithm cross mutations takes place to obtain the most desired result.

The working process:-

- Initially, the program is set up to create a set of records with a random value which eventually constitute a "Population". Each field inside it is called a "Chromosome".

- After the genetic algorithm is invoked, a “Fitness” function and a “Conflict” function are invoked.The fitness function calculates how close each chromosome is to our desired result. In case of similar redundant values in different chromosomes, the conflict function keeps count of this.

- A new generation is then iterated with the set of the new populations. The new population is “selected” from the older generation. The chromosomes with the best fitness and the least number of conflicts are carried forward. The best chromosomes are “cross-overed” with other chromosomes and some values are altered in order to “Mutate” the chromosome so as to reach closer to the desired output.

- In a similar fashion, the program continues until the fitness becomes the most desirable for the required output i.e. Fitness becomes 1.000.

- Thus the main motive of reaching the most desirable output is taken care of using the concept of genetic algorithm.

Object Oriented Programming Concepts

- **Classes and Objects**
  The software is developed using classes and objects in java. Separate classes are created for every required parameters such as Room, Instructor, Meeting time, Course, Department and for the Genetic algorithms.

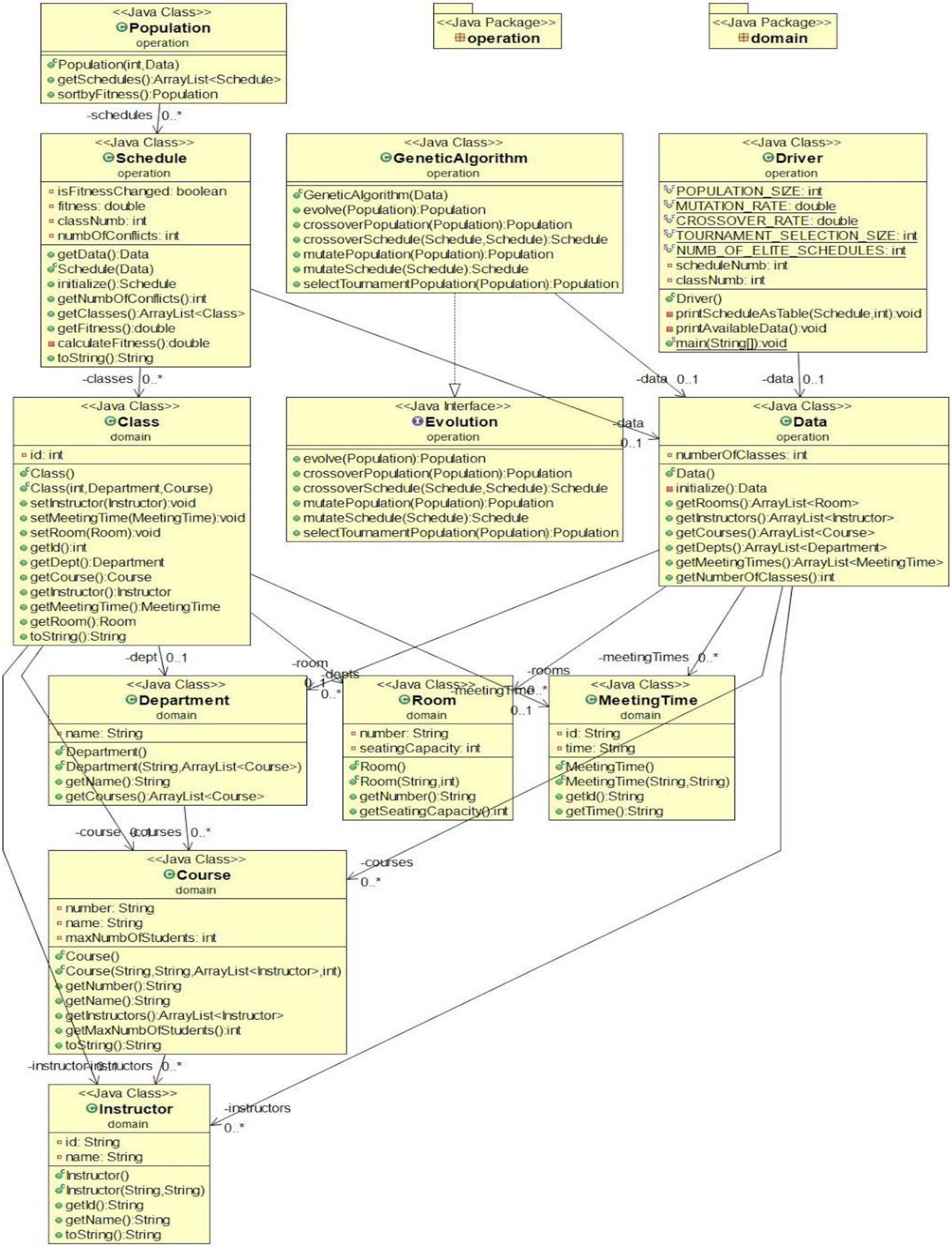
- **Packages**
  With the help of packages in Java. We were able to divide our project into small quantities and then be able to use it. The fundamental structure of quantities like courses, Instructor, Department, Room is put in a package. The other package consists of the required genetic algorithm which works on the classes in the first package.

- **Abstraction**
  We have used interfaces for scheduling through each generation. The interfaces for each department, course, room, instructor and meeting time are implemented in Genetic algorithm and the driver class.

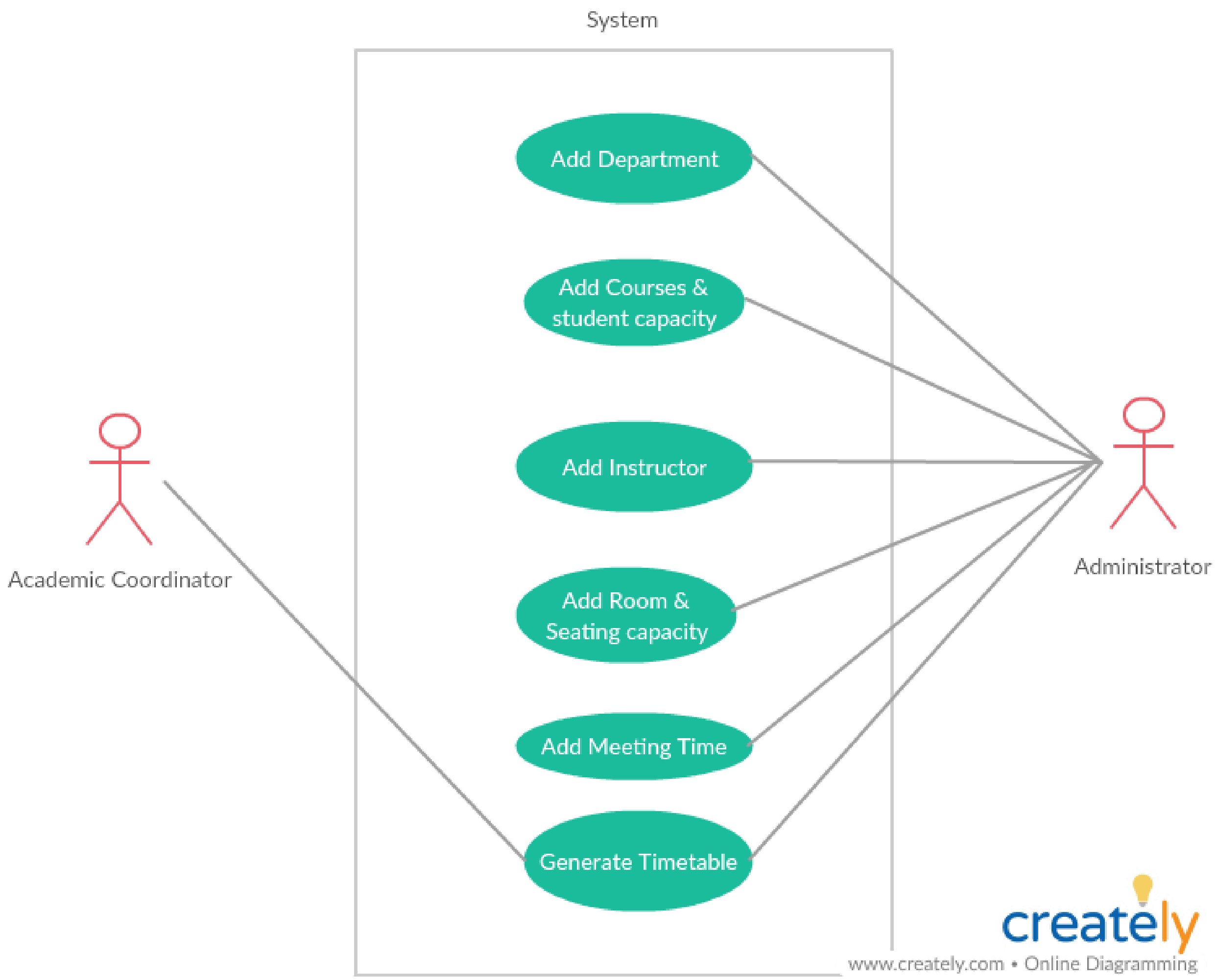
- **Polymorphism**
  The constructor methods are overloaded according to the type of parameters it handles. This enables to initialize the class objects with different attributes. Thus Polymorphism is incorporated into our project.

- **Data Encapsulation**
  In our project, the attributes and corresponding member functions are wrapped and encapsulated into a single unit called class. Thus these classes can be imported independently into another class and its member functions can be used

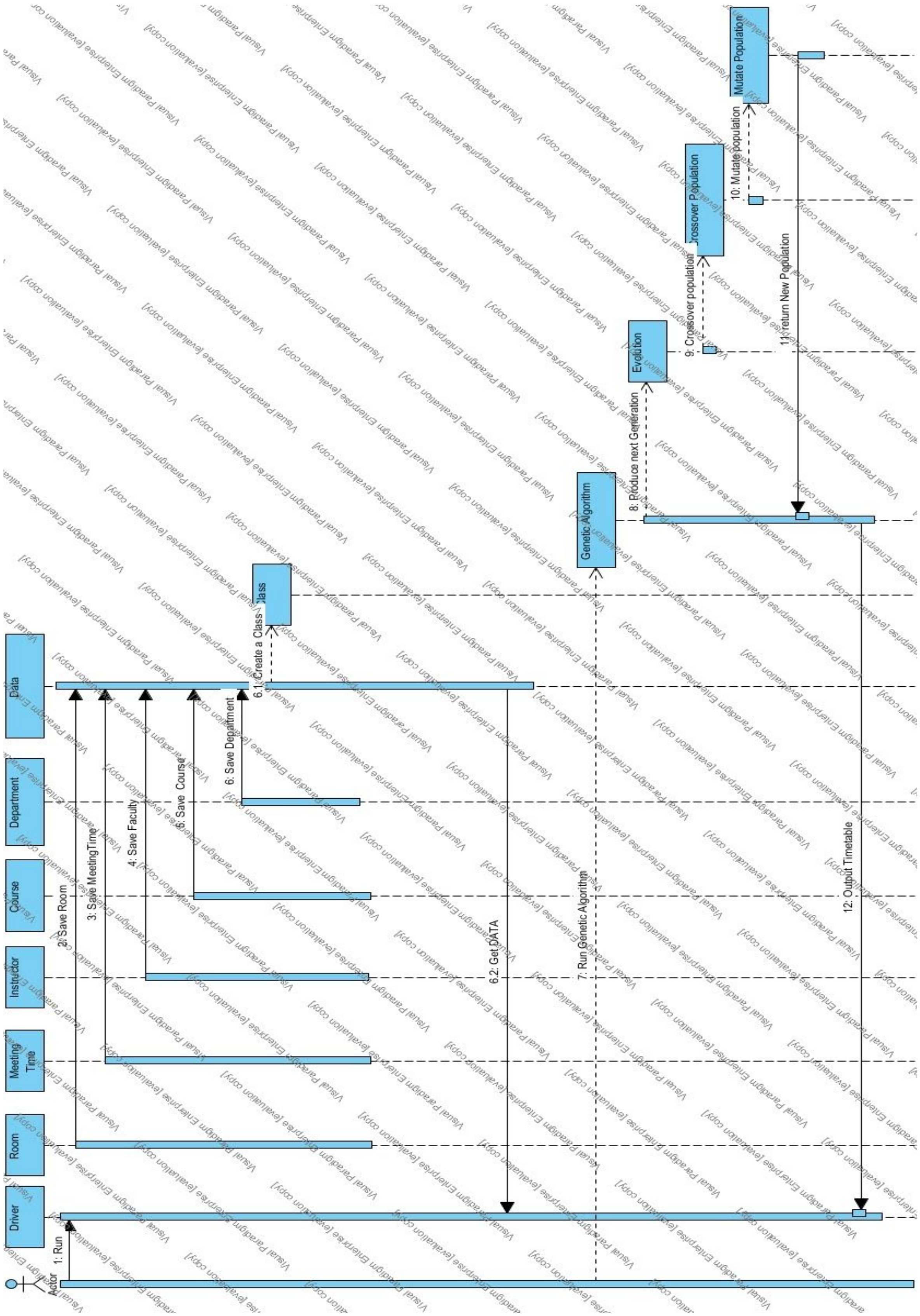
Class Diagram



Use Case Diagram



Sequence Diagram


Actor
Driver
Room
Meeting Time
Instructor
Course
Department
Data
Class
Genetic Algorithm
Evolution
Crossover Population
Mutate Population
1: Run
2: Save Room
3: Save MeetingTime
4: Save Faculty
5: Save Course
6: Save Department
6.1: Create a Class
6.2: Get DATA
7: Run Genetic Algorithm
8: Produce next Generation
9: Crossover population
10: Mutate population
11: return New Population
12: Output Timetable