National University of Computer and Emerging Sciences
Islamabad Campus

Assignment 03
GA

Exam Schedule Generation Using Genetic Algorithm

Submitted by: Ayesha Areej
Roll number: i221711
Date: 4 April 2025

Table of Contents

Introduction

In this assignment the implementation of a Genetic Algorithm (GA), to generate an optimized exam schedule. The objective is to create a timetable that satisfies hard constraints and optimize soft constraints. GA algorithm was used for this implementation, which involves initialization, fitness evaluation, selection, crossover, and mutation.

Problem

Exam scheduling is a highly constrained and complex problem that universities and institutions face regularly. The goal is to assign exams to time slots, rooms, and invigilators while ensuring that no student has overlapping exams and facility are not assigned multiple invigilation duties simultaneously. Additionally, Kumar constraints such as room capacity, fairness and invigilation assignments, and optimal spacing of exams must be considered.

Traditional methods, searching, such as manual scheduling or simple rule-based algorithms often fail to handle the complexity efficiently. These approaches can be slow comma error prone comma and may not find the most optimal solution comma leading to student and faculty dissatisfaction.

So, to solve this problem efficiently instead of using any other algorithm I decided to use the GA algorithm.

Why GA?

It's a genetic algorithm, which is an evolutionary based optimization method inspired by natural selection. It provides a powerful approach to solving scheduling problems due to the following advantages:

- GA explores a large solution space effectively, avoiding getting stuck in sub optimal solutions.
- GA optimizes both hard and soft constraints by evolving solutions across generations.
- It can be used for large scale scheduling problems without exhaustive manual intervention.
- True crossover and mutation, GA continuously refines schedules to ensure fairness and efficiency.
- Unlike brute force or heuristic methods, GA balances flexibility and computational efficiency.

Objectives

- Develop a GA based approach for automated exam scheduling.
- Ensure all exams are assigned with no student conflicts.
- Optimize invigilator assignments and room allocations.
- Minimize gaps between exams for students.
- Implement roulette wheel selection for choosing schedules.

Problem Formulation

As discussed above the problem includes a set of courses, students, rooms, and faculty members. For all these characteristics some constraints were given to the problem:

Hard Constraints

- No student conflict: a student cannot take two exams at the same time.
- Faculty constraints: A faculty member cannot immediate 2 exams simultaneously.
- Room capacity: the number of students assigned to a room should not exceed its capacity.
- Exam timing: exams should only be scheduled between 9:00 AM to 5:00 PM.
- No consecutive invitations: frugality should not have back-to-back invigilation duties.

Soft Constraints

- A common break on Friday from 1:00 PM to 2:00 PM for all students and teachers.
- If a student is enrolled in both management and computer science management course exam, then MG should be scheduled before CS exam.
- A two hour break in the week ensure that half of the faculty is free and one slot and the rest in another for faculty meetings.

- ## GA Implementation

The breakdown of the GA implementation:

- ### Representation of Solutions (Chromosomes)

- Each chromosome consists of a complex exam schedule.

Fittest Population 1 (Fitness Score: 70):

Course Code	Course Name	Teacher	Day	Time Slot	Room
CS217	Object Oriented Programming	Bilal Khalid	Tuesday	04:00-05:00	C303
EE227	Digital Logic Design	Adnan Tariq	Thursday	02:00-03:00	C303
CS211	Discrete Structures	Shoaib Mehboob	Monday	03:00-04:00	C301
SE110	Intro to Software Engineering	Waseem Shahzad	Monday	11:00-12:00	C303
CS118	Programming Fundamentals	Waqas Munir	Wednesday	09:00-10:00	C301
CS219	Database Systems	Usman Ashraf	Friday	04:00-05:00	C310
CS220	Operating Systems	Mehwish Hassan	Wednesday	04:00-05:00	C309
CS302	Design & Analysis of Algorithms	Gul e Aisha	Thursday	10:00-11:00	C308
CY2012	Digital Forensics	Zainab Abaid	Monday	02:00-03:00	C305
CS307	Computer Networks	Usman Rashid	Wednesday	11:00-12:00	C308
CS328	Software Engineering	Tayyaba Zainab	Monday	10:00-11:00	C302
EE229	Computer Organization and Assembly Language	Hamda Khan	Monday	10:00-11:00	C310
AI2011	Programming for AI	Amna Irum	Wednesday	09:00-10:00	C304
DS3011	Big Data Analytics	Asma Nisa	Tuesday	10:00-11:00	C307
CS328	Software Engineering	Khadija Farooq	Wednesday	01:00-02:00	C305
CS218	Data Structures	Noor ul Ain	Monday	11:00-12:00	C304
EE229	Computer Organization & Assembly Language	Muhammad Usman	Monday	04:00-05:00	C308
CS211	Discrete Structures	Hassan Mustafa	Wednesday	04:00-05:00	C303
MT224	Differential Equations	Tayyaba Zainab	Friday	11:00-12:00	C303
SS113	Pakistan Studies	Naveed Ahmad	Monday	10:00-11:00	C303
MG220	Marketing Management	Mehwish Hassan	Thursday	10:00-11:00	C310
MG223	Fundamentals of Management	Behjat Zuhaira	Tuesday	09:00-10:00	C304
SS111	Islamic and Religious Studies	Shams Farooq	Wednesday	04:00-05:00	C304
SS152	Communication & Presentation Skills	Waseem Shahzad	Tuesday	10:00-11:00	C306
SS118	Psychology	Noreen Jamil	Tuesday	10:00-11:00	C305

- A chromosome consists of genes, where each gene corresponds to an exam slot.

Course Code	Course Name	Teacher	Day	Time Slot	Room
CS217	Object Oriented Programming	Bilal Khalid	Tuesday	04:00-05:00	C303
EE227	Digital Logic Design	Adnan Tariq	Thursday	02:00-03:00	C303
CS211	Discrete Structures	Shoaib Mehboob	Monday	03:00-04:00	C301
SE110	Intro to Software Engineering	Waseem Shahzad	Monday	11:00-12:00	C303
CS118	Programming Fundamentals	Waqas Munir	Wednesday	09:00-10:00	C301
CS219	Database Systems	Usman Ashraf	Friday	04:00-05:00	C310
CS220	Operating Systems	Mehwish Hassan	Wednesday	04:00-05:00	C309
CS302	Design & Analysis of Algorithms	Gul e Aisha	Thursday	10:00-11:00	C308
CY2012	Digital Forensics	Zainab Abaid	Monday	02:00-03:00	C305
CS307	Computer Networks	Usman Rashid	Wednesday	11:00-12:00	C308
CS328	Software Engineering	Tayyaba Zainab	Monday	10:00-11:00	C302
EE229	Computer Organization and Assembly Language	Hamda Khan	Monday	10:00-11:00	C310
AI2011	Programming for AI	Amna Irum	Wednesday	09:00-10:00	C304
DS3011	Big Data Analytics	Asma Nisa	Tuesday	10:00-11:00	C307
CS328	Software Engineering	Khadija Farooq	Wednesday	01:00-02:00	C305
CS218	Data Structures	Noor ul Ain	Monday	11:00-12:00	C304
EE229	Computer Organization & Assembly Language	Muhammad Usman	Monday	04:00-05:00	C308
CS211	Discrete Structures	Hassan Mustafa	Wednesday	04:00-05:00	C303
MT224	Differential Equations	Tayyaba Zainab	Friday	11:00-12:00	C303
SS113	Pakistan Studies	Naveed Ahmad	Monday	10:00-11:00	C303
MG220	Marketing Management	Mehwish Hassan	Thursday	10:00-11:00	C310

- ## GA Algorithm Components
  - ### File reading

    At the start the data from the files were read and stored in arrays on which the further work had to be done.

```
generation_fitness_history = []
# Read the CSV files
courses_df = pd.read_csv("courses.csv")
student_course_df = pd.read_csv("studentCourse.csv")
student_names_df = pd.read_csv("studentNames.csv")
teachers_df = pd.read_csv("teachers.csv")
```

  - ### Logging fitness Evaluation

    Created the file to log the constraints checked, unchecked from the fitness assignment.

```
 # Write to log file
with open("log.txt", "w") as log_file:  # First clear the file
    log_file.write("Population Constraints\n")
```

  - ### Population Initialization
  - Generate an initial population of 5 random schedules.
  - Basic constraints were the valid rooms and timeslots to follow.
  - The data to create the population was obtained from the arrays in which data was stored.
  - At the start the schedules were totally random.

```python
def generate_random_timetable():
    timetable = []
    for _, course in courses_df.iterrows():
        teacher = random.choice(teachers_df["Names"].tolist())
        day = random.choice(days)
        time_slot = random.choice(time_slots)
        room = random.choice(rooms)
        timetable.append({
            "Course Code": course["Course Code"],
            "Course Name": course["Course Name"],
            "Teacher": teacher,
            "Day": day,
            "Time Slot": time_slot,
            "Room": room
        })
    return timetable

# Generate 5 populations of timetables
#populations = [generate_random_timetable() for _ in range(5)]

# Generate student exam schedules
def generate_student_exam_schedule(timetable):
    student_schedule = []
    for _, student in student_course_df.iterrows():
        course_schedule = next((entry for entry in timetable if entry["Course Code"] == student["Course Code"]), None)
        if course_schedule:
            student_schedule.append({
                "Student Name": student["Student Name"],
                "Course Code": student["Course Code"],
                "Course Name": course_schedule["Course Name"],
                "Day": course_schedule["Day"],
                "Time Slot": course_schedule["Time Slot"],
                "Room": course_schedule["Room"]
            })
    return student_schedule
```

Fitness Evaluation

- This function evaluates that the created timetable is how efficient or unto the constraints or demands.
- The fitness is calculated like there are penalties for not handling hard constraints, which is 10 per constraint and bonus score is added for the optimization/soft constraint 10 for each. So,

  Fitness score = 60 – (hard-constraint-penalty) + (optimization-bonus)

```python
def evaluate_fitness(timetable):
    fitness = 60  # Initial fitness score for the whole table
    teacher_schedule = {}
    student_schedule = {}
    fitness_log = []

    # Initialize violation flags for hard constraints
    hard_constraints_violated = {
        'missing_courses': False,
        'student_overlap': False,
        'invalid_day': False,
        'invalid_time': False,
        'teacher_conflict': False,
        'consecutive_duty': False
    }

    # First pass: Collect all schedules
    for entry in timetable:
        day = entry["Day"]
        time_slot = entry["Time Slot"]
        teacher = entry["Teacher"]
        course_code = entry["Course Code"]

        # Track schedules
        teacher_schedule.setdefault(day, {}).setdefault(time_slot, []).append(teacher)

        # Track student schedules
        for _, student in student_course_df.iterrows():
            if student["Course Code"] == course_code:
                student_schedule.setdefault(student["Student Name"], {}).setdefault(day, []).append(time_slot)

    # HARD CONSTRAINT 1: Every course must have an exam scheduled
    scheduled_courses = set(entry["Course Code"] for entry in timetable)
    required_courses = set(courses_df["Course Code"])
    if scheduled_courses != required_courses:
        if not hard_constraints_violated['missing_courses']:
            fitness -= 10
            hard_constraints_violated['missing_courses'] = True
            fitness_log.append("Penalty: -10 (Not all courses are scheduled)")

    # HARD CONSTRAINT 2: No student should have overlapping exams
    for student, schedule in student_schedule.items():
        for day, times in schedule.items():
            if len(times) != len(set(times)):
                if not hard_constraints_violated['student_overlap']:
                    fitness -= 10
                    hard_constraints_violated['student_overlap'] = True
                    fitness_log.append("Penalty: -10 (Student exam overlap detected)")
                break

    # HARD CONSTRAINT 3: Exams only on weekdays
    valid_days = set(["Monday", "Tuesday", "Wednesday", "Thursday", "Friday"])
    for entry in timetable:
        if entry["Day"] not in valid_days:
            if not hard_constraints_violated['invalid_day']:
```

- Selection
- The method for the selection was Roulette Wheel selection.
- First calculated the cumulative fitness of all schedules.
- Then assign the selection probabilities proportional scores.
- Then random schedules are picked based on weighted probability.
- Then this selected population is used for the next generation.

```
def roulette_wheel_selection(fitness_results):
    fitness_scores = [score for _, score, _ in fitness_results]
    min_score = min(fitness_scores)

    # Adjust scores to be positive for roulette wheel selection
    adjusted_scores = [score - min_score + 1 for score in fitness_scores]
    total_fitness = sum(adjusted_scores)

    probabilities = [score / total_fitness for score in adjusted_scores]
    selected_indices = np.random.choice(len(fitness_results), size=2, replace=False, p=probabilities)
    return [fitness_results[i] for i in selected_indices]
```

- Crossover Operation
- It is used to generate new schedules by combining the parts of two parent schedules.
- One-point crossover is applied in this case and at random positions.
- Main reason is to create the randomness.
- Child = Parent1 [0:k] + Parent2 [k:N]
- K is the crossover point.
- N is the length of the chromosomes.

```python
def crossover(parent1, parent2, crossover_rate=0.8):
    """
    Performs crossover between two parent timetables based on their fitness scores.
    Preserves the best attributes from both parents with higher probability.
    """
    if random.random() > crossover_rate:
        return parent1.copy(), parent2.copy()

    # Evaluate fitness of both parents
    fitness1, _ = evaluate_fitness(parent1)
    fitness2, _ = evaluate_fitness(parent2)

    # Calculate probability weights based on fitness
    total_fitness = fitness1 + fitness2
    p1_weight = fitness1 / total_fitness

    offspring1 = []
    offspring2 = []

    # For each course, select attributes from either parent based on their fitness
    for i in range(len(parent1)):
        entry1 = parent1[i]
        entry2 = parent2[i]

        # Create new entries for offspring
        new_entry1 = {}
        new_entry2 = {}

        # Keep course code and name constant
        new_entry1['Course Code'] = entry1['Course Code']
        new_entry1['Course Name'] = entry1['Course Name']
        new_entry2['Course Code'] = entry1['Course Code']
        new_entry2['Course Name'] = entry1['Course Name']

        # For each attribute, choose from parents based on their fitness
        for attr in ['Teacher', 'Day', 'Time Slot', 'Room']:
            # Use weighted probability to select from which parent to inherit
            if random.random() < p1_weight:
                new_entry1[attr] = entry1[attr]
                new_entry2[attr] = entry2[attr]
            else:
                new_entry1[attr] = entry2[attr]
                new_entry2[attr] = entry1[attr]

        offspring1.append(new_entry1)
        offspring2.append(new_entry2)

    return offspring1, offspring2
```

- Mutate operation
- Mutation was done by a very small probability which is 10%, 0.1.
- Mutation modifies the exam time, room, invigilator or anything.
- Mutation is necessary for premature convergence or the same solutions.

```python
    return populations
def mutate(timetable, mutation_rate=0.1):
    """Randomly mutates an attribute of an exam schedule with a given probability."""
    for entry in timetable:
        if random.random() < mutation_rate:
            entry["Teacher"] = random.choice(teachers_df["Names"].tolist())  # Change teacher
        if random.random() < mutation_rate:
            entry["Day"] = random.choice(days)  # Change exam day
        if random.random() < mutation_rate:
            entry["Time Slot"] = random.choice(time_slots)  # Change time slot
        if random.random() < mutation_rate:
            entry["Room"] = random.choice(rooms)  # Change room
    return timetable
```

- Termination Condition
- The program is terminated after 20 generations can be modified.
- The best schedule is based on the highest fitness achieved.

```
# Number of generations
num_generations = 10
```

• Summary of Key parameters

Parameter	Value
Population size	5 per generation
Generations	20 changeable
Fitness assignment	60 – (hard-constraint-penalty) + (optimization-bonus)
Selection Method	Route wheel
Mutation Rate	10% / 0.1
Cross over	One-point

• Analysis of the code

- Creation of random 5 populations -> exam schedules.
- Fitness assigned to them.

```
====== Generation 1 ======

Generation Fitness Scores Before Selection:
+--------------+-----------------+
|   GENERATION |   FITNESS SCORE |
+==============+=================+
|            1 |              70 |
+--------------+-----------------+
|            2 |              70 |
+--------------+-----------------+
|            3 |              60 |
+--------------+-----------------+
|            4 |              70 |
+--------------+-----------------+
|            5 |              50 |
+--------------+-----------------+
```

- Selected two populations with highest fitness and displayed the population.

Fittest Population 1 (Fitness Score: 70):

Course Code	Course Name	Teacher	Day	Time Slot	Room
CS217	Object Oriented Programming	Bilal Khalid	Tuesday	04:00-05:00	C303
EE227	Digital Logic Design	Adnan Tariq	Thursday	02:00-03:00	C303
CS211	Discrete Structures	Shoaib Mehboob	Monday	03:00-04:00	C301
SE110	Intro to Software Engineering	Waseem Shahzad	Monday	11:00-12:00	C303
CS118	Programming Fundamentals	Waqas Munir	Wednesday	09:00-10:00	C301
CS219	Database Systems	Usman Ashraf	Friday	04:00-05:00	C310
CS220	Operating Systems	Mehwish Hassan	Wednesday	04:00-05:00	C309
CS302	Design & Analysis of Algorithms	Gul e Aisha	Thursday	10:00-11:00	C308
CY2012	Digital Forensics	Zainab Abaid	Monday	02:00-03:00	C305
CS307	Computer Networks	Usman Rashid	Wednesday	11:00-12:00	C308
CS328	Software Engineering	Tayyaba Zainab	Monday	10:00-11:00	C302
EE229	Computer Organization and Assembly Language	Hamda Khan	Monday	10:00-11:00	C310
AI2011	Programming for AI	Amna Irum	Wednesday	09:00-10:00	C304
DS3011	Big Data Analytics	Asma Nisa	Tuesday	10:00-11:00	C307
CS328	Software Engineering	Khadija Farooq	Wednesday	01:00-02:00	C305
CS218	Data Structures	Noor ul Ain	Monday	11:00-12:00	C304
EE229	Computer Organization & Assembly Language	Muhammad Usman	Monday	04:00-05:00	C308
CS211	Discrete Structures	Hassan Mustafa	Wednesday	04:00-05:00	C303
MT224	Differential Equations	Tayyaba Zainab	Friday	11:00-12:00	C303
SS113	Pakistan Studies	Naveed Ahmad	Monday	10:00-11:00	C303
MG220	Marketing Management	Mehwish Hassan	Thursday	10:00-11:00	C310
MG223	Fundamentals of Management	Behjat Zuhaira	Tuesday	09:00-10:00	C304
SS111	Islamic and Religious Studies	Shams Farooq	Wednesday	04:00-05:00	C304
SS152	Communication & Presentation Skills	Waseem Shahzad	Tuesday	10:00-11:00	C306
SS118	Psychology	Noreen Jamil	Tuesday	10:00-11:00	C305
MT205	Probability and Statistics	Noor ul Ain	Thursday	09:00-10:00	C301

Fittest Population 2 (Fitness Score: 70):

Course Code	Course Name	Teacher	Day	Time Slot	Room
CS217	Object Oriented Programming	Javaria Imtiaz	Tuesday	10:00-11:00	C305
EE227	Digital Logic Design	Gul e Aisha	Wednesday	01:00-02:00	C309
CS211	Discrete Structures	Amna Irum	Tuesday	04:00-05:00	C306
SE110	Intro to Software Engineering	Muhammad Usman	Tuesday	12:00-01:00	C304
CS118	Programming Fundamentals	Naveed Ahmad	Tuesday	11:00-12:00	C302
CS219	Database Systems	Shafaq Riaz	Monday	04:00-05:00	C304
CS220	Operating Systems	Zohaib Iqbal	Thursday	10:00-11:00	C303
CS302	Design & Analysis of Algorithms	Shoaib Mehboob	Monday	11:00-12:00	C306
CY2012	Digital Forensics	Khadija Farooq	Friday	04:00-05:00	C303
CS307	Computer Networks	Zainab Moin	Thursday	11:00-12:00	C303
CS328	Software Engineering	Farah Jabeen Awan	Thursday	03:00-04:00	C305
EE229	Computer Organization and Assembly Language	Muhammad Asim	Thursday	03:00-04:00	C305
AI2011	Programming for AI	Nagina Safdar	Monday	12:00-01:00	C303
DS3011	Big Data Analytics	Behjat Zuhaira	Wednesday	10:00-11:00	C310
CS328	Software Engineering	Noreen Jamil	Tuesday	02:00-03:00	C310
CS218	Data Structures	Asif Naeem	Tuesday	03:00-04:00	C310
EE229	Computer Organization & Assembly Language	Kashif Munir	Thursday	09:00-10:00	C310
CS211	Discrete Structures	Waqas Munir	Thursday	11:00-12:00	C303
MT224	Differential Equations	Kashif Munir	Tuesday	09:00-10:00	C306
SS113	Pakistan Studies	Asif Naeem	Wednesday	03:00-04:00	C301
MG220	Marketing Management	Zainab Moin	Monday	10:00-11:00	C304
MG223	Fundamentals of Management	Behjat Zuhaira	Monday	09:00-10:00	C301
SS111	Islamic and Religious Studies	Shahzad Mehmood	Wednesday	03:00-04:00	C304
SS152	Communication & Presentation Skills	Shafaq Riaz	Wednesday	04:00-05:00	C303
SS118	Psychology	Waseem Shahzad	Thursday	12:00-01:00	C301

- The constraints details were logged in the txt file.

```
Population Constraints

=== Generation 1 ===

Fittest Population 1 (Fitness Score: 70):
Penalty: -10 (Student exam overlap detected)
Reward: +10 (Common break on Friday 1-2 PM maintained)
Reward: +10 (Adequate faculty meeting slots available)

Fittest Population 1 (Fitness Score: 70):
Penalty: -10 (Student exam overlap detected)
Reward: +10 (Common break on Friday 1-2 PM maintained)
Reward: +10 (Adequate faculty meeting slots available)

Fittest Population 2 (Fitness Score: 70):
Penalty: -10 (Student exam overlap detected)
Reward: +10 (Common break on Friday 1-2 PM maintained)
Reward: +10 (Adequate faculty meeting slots available)

Fittest Population 2 (Fitness Score: 70):
Penalty: -10 (Student exam overlap detected)
Reward: +10 (Common break on Friday 1-2 PM maintained)
Reward: +10 (Adequate faculty meeting slots available)

=== Generation 2 ===

Fittest Population 1 (Fitness Score: 70):
Penalty: -10 (Student exam overlap detected)
Reward: +10 (Common break on Friday 1-2 PM maintained)
Reward: +10 (Adequate faculty meeting slots available)

Fittest Population 1 (Fitness Score: 70):
Penalty: -10 (Student exam overlap detected)
Reward: +10 (Common break on Friday 1-2 PM maintained)
Reward: +10 (Adequate faculty meeting slots available)

Fittest Population 2 (Fitness Score: 70):
Penalty: -10 (Student exam overlap detected)
Reward: +10 (Common break on Friday 1-2 PM maintained)
Reward: +10 (Adequate faculty meeting slots available)

Fittest Population 2 (Fitness Score: 70):
Penalty: -10 (Student exam overlap detected)
Reward: +10 (Common break on Friday 1-2 PM maintained)
Reward: +10 (Adequate faculty meeting slots available)

=== Generation 3 ===

Fittest Population 1 (Fitness Score: 80):
Penalty: -10 (Student exam overlap detected)
Reward: +10 (Common break on Friday 1-2 PM maintained)
Reward: +10 (All MG courses before CS courses)
Reward: +10 (Adequate faculty meeting slots available)

Fittest Population 1 (Fitness Score: 80):
```

- Then after crossover and mutation new populations were created names as generation2 which repeats the same process.

```
====== Generation 2 ======

Generation Fitness Scores Before Selection:
+--------------+-----------------+
|   GENERATION |   FITNESS SCORE |
+==============+=================+
|            1 |              60 |
+--------------+-----------------+
|            2 |              60 |
+--------------+-----------------+
|            3 |              70 |
+--------------+-----------------+
|            4 |              70 |
+--------------+-----------------+
|            5 |              60 |
+--------------+-----------------+

Fittest Population 1 (Fitness Score: 70):
+---------------+-----------------------------------------+-----------------+-----------+-------------+--------+
| Course Code   | Course Name                             | Teacher         | Day       | Time Slot   | Room   |
+===============+=========================================+=================+===========+=============+========+
| CS217         | Object Oriented Programming             | Bilal Khalid    | Tuesday   | 10:00-11:00 | C306   |
+---------------+-----------------------------------------+-----------------+-----------+-------------+--------+
| EE227         | Digital Logic Design                    | Gul e Aisha     | Wednesday | 02:00-03:00 | C309   |
+---------------+-----------------------------------------+-----------------+-----------+-------------+--------+
| CS211         | Discrete Structures                     | Shoaib Mehboob  | Monday    | 04:00-05:00 | C308   |
+---------------+-----------------------------------------+-----------------+-----------+-------------+--------+
| SE110         | Intro to Software Engineering           | Waseem Shahzad  | Monday    | 12:00-01:00 | C303   |
+---------------+-----------------------------------------+-----------------+-----------+-------------+--------+
| CS118         | Programming Fundamentals                | Waqas Munir     | Wednesday | 09:00-10:00 | C302   |
+---------------+-----------------------------------------+-----------------+-----------+-------------+--------+
| CS219         | Database Systems                        | Usman Ashraf    | Friday    | 04:00-05:00 | C310   |
+---------------+-----------------------------------------+-----------------+-----------+-------------+--------+
| CS220         | Operating Systems                       | Mehwish Hassan  | Wednesday | 10:00-11:00 | C309   |
+---------------+-----------------------------------------+-----------------+-----------+-------------+--------+
| CS302         | Design & Analysis of Algorithms         | Gul e Aisha     | Friday    | 10:00-11:00 | C308   |
+---------------+-----------------------------------------+-----------------+-----------+-------------+--------+
| CY2012        | Digital Forensics                       | Khadija Farooq  | Friday    | 10:00-11:00 | C303   |
+---------------+-----------------------------------------+-----------------+-----------+-------------+--------+
| CS307         | Computer Networks                       | Usman Rashid    | Wednesday | 11:00-12:00 | C308   |
+---------------+-----------------------------------------+-----------------+-----------+-------------+--------+
| CS328         | Software Engineering                    | Asif Naeem      | Thursday  | 03:00-04:00 | C308   |
+---------------+-----------------------------------------+-----------------+-----------+-------------+--------+
| EE229         | Computer Organization and Assembly Language | Muhammad Asim   | Thursday  | 03:00-04:00 | C309   |
+---------------+-----------------------------------------+-----------------+-----------+-------------+--------+
| AI2011        | Programming for AI                      | Amna Irum       | Monday    | 09:00-10:00 | C304   |
+---------------+-----------------------------------------+-----------------+-----------+-------------+--------+
| DS3011        | Big Data Analytics                      | Javaria Imtiaz  | Wednesday | 10:00-11:00 | C307   |
+---------------+-----------------------------------------+-----------------+-----------+-------------+--------+
| CS328         | Software Engineering                    | Noreen Jamil    | Friday    | 02:00-03:00 | C305   |
+---------------+-----------------------------------------+-----------------+-----------+-------------+--------+
| CS218         | Data Structures                         | Noor ul Ain     | Wednesday | 03:00-04:00 | C304   |
+---------------+-----------------------------------------+-----------------+-----------+-------------+--------+
```

- This will continue until the termination condition has met.

```
====== Generation 10 ======

Generation Fitness Scores Before Selection:
+--------------+-----------------+
|   GENERATION |   FITNESS SCORE |
+==============+=================+
|            1 |              80 |
+--------------+-----------------+
|            2 |              60 |
+--------------+-----------------+
|            3 |              80 |
+--------------+-----------------+
|            4 |              70 |
+--------------+-----------------+
|            5 |              70 |
+--------------+-----------------+

Fittest Population 1 (Fitness Score: 80):
+---------------+-----------------------------------------+----------------+-----------+-------------+--------+
| Course Code   | Course Name                             | Teacher        | Day       | Time Slot   | Room   |
+===============+=========================================+================+===========+=============+========+
| CS217         | Object Oriented Programming             | Noreen Jamil   | Tuesday   | 09:00-10:00 | C305   |
+---------------+-----------------------------------------+----------------+-----------+-------------+--------+
| EE227         | Digital Logic Design                    | Kashif Munir   | Wednesday | 02:00-03:00 | C309   |
+---------------+-----------------------------------------+----------------+-----------+-------------+--------+
| CS211         | Discrete Structures                     | Shoaib Mehboob | Thursday  | 09:00-10:00 | C306   |
+---------------+-----------------------------------------+----------------+-----------+-------------+--------+
| SE110         | Intro to Software Engineering           | Sara Aziz      | Monday    | 09:00-10:00 | C303   |
+---------------+-----------------------------------------+----------------+-----------+-------------+--------+
| CS118         | Programming Fundamentals                | Waseem Shahzad | Tuesday   | 11:00-12:00 | C306   |
+---------------+-----------------------------------------+----------------+-----------+-------------+--------+
| CS219         | Database Systems                        | Bilal Khalid   | Tuesday   | 12:00-01:00 | C304   |
+---------------+-----------------------------------------+----------------+-----------+-------------+--------+
| CS220         | Operating Systems                       | Mehwish Hassan | Wednesday | 04:00-05:00 | C303   |
+---------------+-----------------------------------------+----------------+-----------+-------------+--------+
| CS302         | Design & Analysis of Algorithms         | Tayyab Nadeem  | Friday    | 11:00-12:00 | C306   |
+---------------+-----------------------------------------+----------------+-----------+-------------+--------+
| CY2012        | Digital Forensics                       | Ameen Chilwan  | Friday    | 10:00-11:00 | C305   |
+---------------+-----------------------------------------+----------------+-----------+-------------+--------+
| CS307         | Computer Networks                       | Zohaib Iqbal   | Wednesday | 03:00-04:00 | C308   |
+---------------+-----------------------------------------+----------------+-----------+-------------+--------+
| CS328         | Software Engineering                    | Naveed Ahmad   | Friday    | 10:00-11:00 | C302   |
+---------------+-----------------------------------------+----------------+-----------+-------------+--------+
| EE229         | Computer Organization and Assembly Language | Hassan Raza    | Wednesday | 03:00-04:00 | C306   |
+---------------+-----------------------------------------+----------------+-----------+-------------+--------+
| AI2011        | Programming for AI                      | Sajid Khan     | Friday    | 09:00-10:00 | C308   |
+---------------+-----------------------------------------+----------------+-----------+-------------+--------+
| DS3011        | Big Data Analytics                      | Asma Nisa      | Wednesday | 10:00-11:00 | C302   |
+---------------+-----------------------------------------+----------------+-----------+-------------+--------+
| CS328         | Software Engineering                    | Hassan Mustafa | Thursday  | 02:00-03:00 | C304   |
+---------------+-----------------------------------------+----------------+-----------+-------------+--------+
| CS218         | Data Structures                         | Shams Farooq   | Friday    | 03:00-04:00 | C304   |
+---------------+-----------------------------------------+----------------+-----------+-------------+--------+
| EE229         | Computer Organization & Assembly Language | Kashif Munir   | Thursday  | 04:00-05:00 | C301   |
```

- Then there is the summary of all the generations with their fittest individuals.

```
====== Generation Summary ======
+--------------+------------------------+------------------------+
|   GENERATION |   FITTEST POPULATION 1 |   FITTEST POPULATION 2 |
+==============+========================+========================+
|            1 |                     70 |                     70 |
+--------------+------------------------+------------------------+
|            2 |                     70 |                     70 |
+--------------+------------------------+------------------------+
|            3 |                     80 |                     70 |
+--------------+------------------------+------------------------+
|            4 |                     80 |                     80 |
+--------------+------------------------+------------------------+
|            5 |                     80 |                     70 |
+--------------+------------------------+------------------------+
|            6 |                     80 |                     70 |
+--------------+------------------------+------------------------+
|            7 |                     70 |                     70 |
+--------------+------------------------+------------------------+
|            8 |                     70 |                     70 |
+--------------+------------------------+------------------------+
|            9 |                     80 |                     80 |
+--------------+------------------------+------------------------+
|           10 |                     80 |                     80 |
+--------------+------------------------+------------------------+
```

- Then the best timetable with the highest fitness score is displayed.

Best Timetable (Fitness Score: 80):

Course Code	Course Name	Teacher	Day	Time Slot	Room
CS217	Object Oriented Programming	Noreen Jamil	Tuesday	09:00-10:00	C305
EE227	Digital Logic Design	Kashif Munir	Tuesday	02:00-03:00	C309
CS211	Discrete Structures	Shoaib Mehboob	Thursday	09:00-10:00	C306
SE110	Intro to Software Engineering	Sara Aziz	Monday	10:00-11:00	C303
CS118	Programming Fundamentals	Waseem Shahzad	Thursday	11:00-12:00	C301
CS219	Database Systems	Bilal Khalid	Tuesday	03:00-04:00	C304
CS220	Operating Systems	Mehwish Hassan	Wednesday	04:00-05:00	C303
CS302	Design & Analysis of Algorithms	Tayyab Nadeem	Friday	11:00-12:00	C306
CY2012	Digital Forensics	Faisal Cheema	Friday	10:00-11:00	C305
CS307	Computer Networks	Sidra Khalid	Wednesday	03:00-04:00	C308
CS328	Software Engineering	Naveed Ahmad	Friday	10:00-11:00	C302
EE229	Computer Organization and Assembly Language	Usman Rashid	Wednesday	03:00-04:00	C306
AI2011	Programming for AI	Noor ul Ain	Friday	09:00-10:00	C308
DS3011	Big Data Analytics	Asma Nisa	Thursday	01:00-02:00	C302
CS328	Software Engineering	Hassan Mustafa	Thursday	11:00-12:00	C304
CS218	Data Structures	Shams Farooq	Friday	03:00-04:00	C304
EE229	Computer Organization & Assembly Language	Kashif Munir	Wednesday	04:00-05:00	C301
CS211	Discrete Structures	Hassan Mustafa	Friday	11:00-12:00	C302
MT224	Differential Equations	Tayyaba Zainab	Friday	11:00-12:00	C303
SS113	Pakistan Studies	Subhan Ullah	Friday	04:00-05:00	C302
MG220	Marketing Management	Mehwish Hassan	Monday	10:00-11:00	C301
MG223	Fundamentals of Management	Behjat Zuhaira	Tuesday	12:00-01:00	C307
SS111	Islamic and Religious Studies	Sadia Nauman	Wednesday	03:00-04:00	C310

MG223	Fundamentals of Management	Behjat Zuhaira	Tuesday	12:00-01:00	C307
SS111	Islamic and Religious Studies	Sadia Nauman	Wednesday	03:00-04:00	C310
SS152	Communication & Presentation Skills	Muhammad Usman	Tuesday	04:00-05:00	C306
SS118	Psychology	Umair Arshad	Thursday	01:00-02:00	C307
MT205	Probability and Statistics	Mehboobullah	Thursday	01:00-02:00	C305

- Then there is the Constraint analysis of the fittest individual.

```
====== Constraints Analysis for Best Timetable ======

Essential Requirements (Hard Constraints):
----------------------------------------
✓ 1. All courses scheduled
✓ 2. No student exam overlaps
✓ 3. Weekdays only
✓ 4. Valid exam times (9 AM - 5 PM)
✓ 5. No teacher conflicts
✓ 6. No consecutive duties

Optimization Criteria (Soft Constraints):
----------------------------------------
✓ 1. Common break (Friday 1-2 PM)
o 2. No back-to-back exams
✓ 3. MG courses before CS courses
✓ 4. Faculty meeting slots available

Compliance Summary:
Hard Constraints: 100.0% satisfied
Soft Constraints: 75.0% satisfied
Final Fitness Score: 80
```

Summary

The assignment explores the implementation of a (GA) for generating an optimized exam schedule, ensuring that both hard and soft constraints are met. Traditional scheduling methods, such as manual scheduling or simple rule-based algorithms, often struggle with complexity, leading to inefficiencies and dissatisfaction among students and faculty. To address this, the assignment leverages GA, an evolutionary optimization technique inspired by natural selection, which efficiently explores large solution spaces while balancing flexibility and computational efficiency.

The GA implementation begins with chromosome representation, where each gene corresponds to an exam slot. A fitness evaluation function is used to assess schedules, penalizing violations of hard constraints, such as student exam overlaps, faculty availability, and room capacity, while rewarding optimizations like scheduling preferences and break times. Roulette Wheel selection is employed to choose schedules for the next generation based on their fitness scores. One-point crossover is applied to combine elements from two parent schedules, introducing randomness, while mutation, with a 10% probability, modifies elements like exam times, rooms, or invigilators to prevent premature convergence. The algorithm terminates after 20 generations, selecting the best schedule with the highest fitness score.

The process results in an optimized exam timetable that satisfies the requirements while minimizing conflicts. The final schedule is analyzed for constraint compliance, ensuring an efficient and fair allocation of exams, rooms, and invigilators. Through this approach, the assignment demonstrates how GA can effectively tackle complex scheduling problems that traditional methods struggle to handle.