National University
of computer and emerging sciences

Assignment Report

CS-2002 Artificial Intelligence

Project Report

Javaria Habib

21i-2485

Z

Submitted to: Usama Imtiaz

Department of Computer Science BS(CS)

FAST-NUCES Islamabad

Code Logic:

Overall the code is using initial population of time tables, encoding them in binary and then applying a genetic algorithm on it to calculate fitness function , crossover, mutation etc to calculate further generations of timetable. Each time table is a chromosome and each population has 10 chromosomes. It is considering the constraints as mentioned in question such as solving clashes between professors, room capacity and section schedules etc.

Main functionality:

Initially the constants are initialized with the number of courses, professors, weeks, time slots per day etc. Population size is determining the timetables used in genetic algorithms.Mutation rate is determining the probability of mutation occurring during genetic algorithm.

Initial dictionary is initialized with a course allocation table to each section and courses and professors .

Encoding:It converts different entities and attributes into binary strings.

decode_timeslot() and decode_day(): Decode binary strings back into human-readable format.initialize_population(num_timetables): Generates an initial population of timetables, ensuring no clashes occur.

Fitness function calculates fitness of each timetable based on number of clashes so gives mostly 0 and using this method fitness function will be an inverse or negative of the sum of all the conflicts/clashes.

GA :

Selection is done based on fitness function calculation , two of the fittest parents are selected for crossover.Crossover is combination of two parents selected that are timetable crossover point is random and then mutation performed which are random .For mutation each course scheduled in the timetable, it checks if a mutation should occur based on the MUTATION_RATE.Randomly selects new values for the day, timeslot, and room for both days of the course.

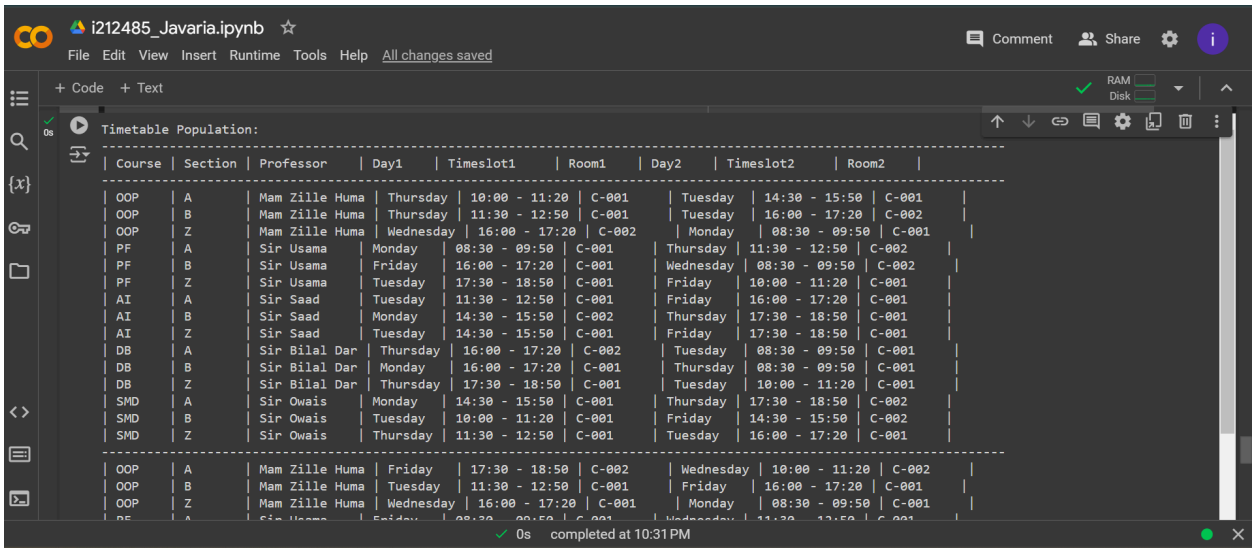
check_clashes():

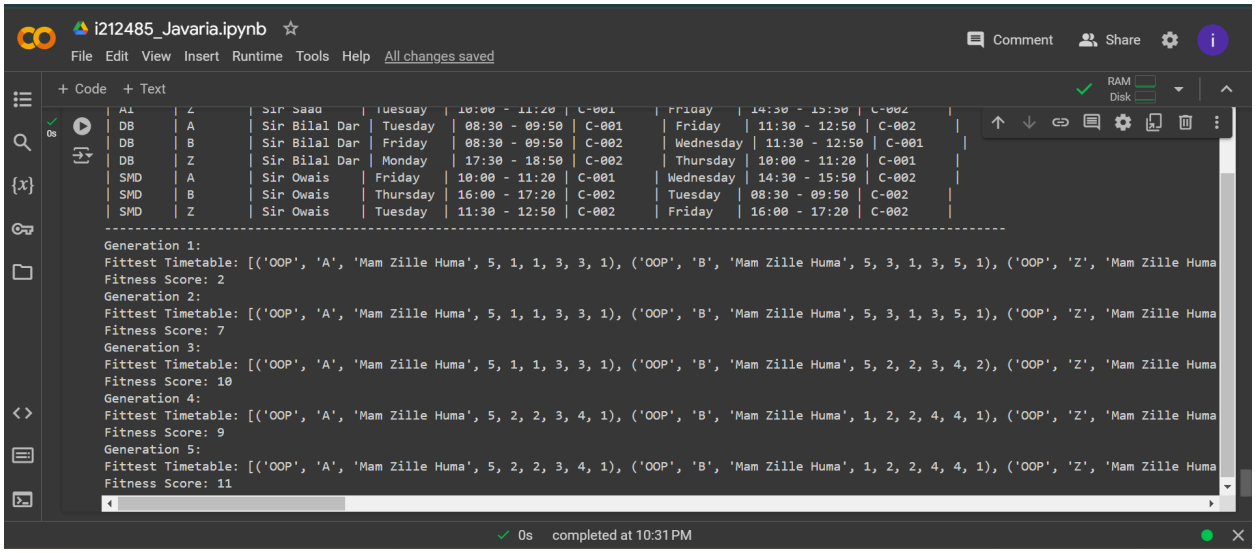
It is taking input of timetables and dictionaries used to check clashes For each course in the timetable, it checks if the assigned time slots for both days of the course overlap with any other courses scheduled for the same professor. If so, it indicates a clash.If a clash is detected, it returns True otherwise it updates professor schedule dictionary with new schedule . Sme is done with sections, courses etc.

Main :

It runs a loop for a specified number of generations which is 5 and for each generation it uses GA to perform fitness evaluation, crossover and mutation to get offspring. Then offspring replace the old population and the fittest timetable of generation is printed with its fitness score.

Output :





In first output it shows the timetable population and in second it prints the generations and their fittest scores.