Solving the schedule problem with Genetic Algorithms

1. **Genetic algorithms depend on…**
    - Parents are randomly selected solutions
    - Crossover is a constructed operator on a solution
    - The mutation is a random change in the solution
    - Fitness evaluator to determine between solutions

2. **Pseudo code**

    Initialize a Population of Individuals
    While Stop Criterion not met:
    Selection of Individuals to Combine
    Application of Crossover Operator
    Application of Mutation Operator
    Application of Local Search Heuristics
    Evaluation of Fitness of the Newly Created Individuals
    Update Population
    Endwhile

3. **Notation**
    - Subject S activities A scheduled, a = 1, …, A
    - R rooms, r = 1, …, R
    - D days, d = 1, ..., D
    - Each day has 4 shifts/times denoted by T = {t1, t2, t3, t4}
    - A list L of students participating in that activity
    - We will refer to specific instances by the 5-tuple (S, A, R, D, T, L)

4. **Formulation - Simplifying assumptions**

5. **Formulation – Constraints**

    Hard constraints cannot be violated
    - Every activity of every subject has a room and time. A feasible schedule equals 1000 points

    Soft constraints are those which may be violated but with an associated penalty cost
    - For every subject of x activities, you get 10 points off if they are scheduled on x minus 1 days, 20 if x minus 2 days and 30 for x minus 3 days
    - 1 point off for every student that doesn't fit in the room
    - 1 point off for every student that has two subjects at the same time
    - 50 points off if the biggest room is used from 17:00 to 19:00 (t5)
    - Bonus point: add 20 points when activities are spread within the week

6. **Formulation – Objective function**

- Let F be a linear function that takes in a schedule matrix and outputs an integer fitness value
- We will maximize the points for the schedule