# Special Topics Project II

The Traveling Salesman Problem (TSP) is a classic optimization problem in computer science and operations research. It involves finding the shortest possible route that visits a given set of cities exactly once and returns to the starting city.

**Problem Statement:** Given a list of cities and the distances between each pair of cities, find the shortest possible route that visits each city exactly once and returns to the starting city.

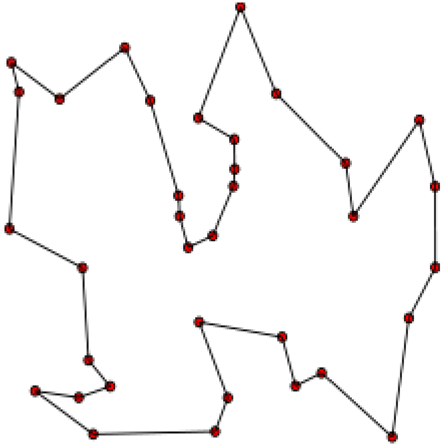

Image Source: Wikipedia Url: https://en.wikipedia.org/wiki/Travelling_salesman_problem)

The TSP is inherently combinatorial in nature because it involves finding the optimal combination of city sequences from a given set. The number of possible routes grows exponentially with the number of cities.

**Combinatorial Explosion:** For a graph with n nodes (cities), there are (n-1)! possible permutations (sequences) of the nodes. This number grows very rapidly as n increases. For example, with 10 cities, there are 9! = 362,880 possible routes.

This combinatorial explosion makes it impractical to solve the TSP using brute-force search for large instances.
Implications:

- The combinatorial nature of the TSP makes it challenging to find exact solutions efficiently.
- This has led to the development of various approximation algorithms and heuristics that provide near-optimal solutions.
- The TSP is often used as a benchmark for testing the performance of optimization algorithms.

## Notes

1. Use the dataset provided with the project. Show the 2D result of your solution indicating starting and final cities. The cityData.txt file lists the Ids of the cities along with coordinates which you can use for plotting. The intercityDistance.txt file stores the distances for each city couple.
2. You can use any language for implementation. Use of a toolbox (MATLAB Optimization Toolbox etc.) is not permitted. You can use a library, for instance PyMOO is Python library that you can use, likewise MOEA is a Java framework with similar functionality which you can also prefer.
3. You need to demonstrate your implementation and results with a 10-minute video demonstration. In addition, you need to submit a report (max. 5 pages, pdf) of your project results. The title page for your report must include the link to your video.
4. While showing your results, you must demonstrate results for 5 different starting cities.
5. In the libraries mentioned, there are several variants of evolutionary algorithms. You should try with at least 2 methods.
6. Your code should include parts that improve code readability, such as variable naming and comment lines.
7. You should develop your project in groups of no more than 2 people. All group members must be present for the demo.
8. Deadline will be announced on the E-Kampus system.

**Prof. Dr. Gazi Erkan BOSTANCI**