Contents

Section 1: Overview of Assessment

This assignment assesses the following module learning outcomes:

- MO1 Apply AI concepts and processes to formulate appropriate representations and algorithms to solve a range of tasks, taking into account the context in which they are being solved.
- MO2 Select, apply, evaluate, and then refine (as needed) an appropriate AI- based search algorithm to solve one or more problems, taking account of ethical, legal and practical issues.

The assignment is worth **100%** of the overall mark for the module.

Broadly speaking, the assignment requires you to submit individual coursework that involves submitting one or more pieces of source code to solve a specified problem using both a single search algorithm and an evolutionary algorithm, and comparing and evaluating the performance of both algorithms using performance metrics.

The assignment is described in more detail in section 2.

This is an individual assignment.

Working on this assignment will significantly enhance your learning in the **AI Search and Optimization** module by providing practical experience in implementing both a single search algorithm and an evolutionary algorithm. It will allow you to apply theoretical concepts in a real-world context, deepening your understanding of how different algorithms operate and their respective strengths and weaknesses. Additionally, the assignment will foster your problem-solving abilities by requiring you to tune hyper-parameters and balance exploration and exploitation within each algorithm. Exposure to real-world challenges, such as optimizing algorithms for efficiency and robustness, will help you appreciate how these methods are applied outside the classroom.

If you have questions about this assignment, please

- Post them to the discussion Forum on Blackboard.
- Raise them with module tutors during one of the timetabled sessions
- Email Dr Mastoi or Dr Aydin.

Section 2: Task Specification

The assessment is designed to consolidate your familiarity with domain-specific vocabulary, well-known problems, established approaches, and algorithm evaluation techniques. It is divided into two stages: in the first stage, you will work on a single-member search algorithm, and in the second stage, you will work on an evolutionary algorithm.

Problem Definition: The **Traveling Salesman Problem** is a classic optimization problem in which the objective is to find the shortest route for an agent who needs to visit a list of cities exactly once and return to the starting point.

Algorithm Implementation:

Stage 1:

Single-Solution-Driven Search Algorithm: Implement a search algorithm that operates with a single candidate solution at a time, adjusting it iteratively to improve its fitness according to the objective function (total distance travelled). The goal is to minimize the total distance travelled, subject to the constraint that each city is visited exactly once and return to your starting point. You have a choice of three possible algorithms that you can use to do this. You will need to select **ONE** algorithm only. The algorithmic options available to you are:

1. A Hill Climber variant (such as Steepest Ascent, Stochastic, Random Restart or Iterated Local Search)
2. Simulated Annealing
3. Tabu Search

Data File Format
You have been provided with a single data file for this problem. The data file, **cities.csv**, contains the coordinates or distance matrix for a set of cities. For this assessment, you have been given 50 cities to visit. The data file is available in the Assessments folder of the Module Blackboard page.

Task Requirement
Your task is to develop a Python solution for the **Traveling Salesman Problem** (TSP) using one of the single-member search algorithms we have covered in the module. You are also expected to evaluate the performance of your algorithm using appropriate visualisation(s) and accompanying explanation.

Your Python solution is required to:

- Read the CSV data file into memory.
- Find out (near) optimal solution for visiting all cities exactly once and return to your starting point. using your chosen search algorithm.
- Output to the console the total distance of the shortest route.
- In an external file, record the sequence of cities for the best route found.
- You are requested to submit your Python solution and your evaluation document to Blackboard.

Stage 2:

Population-Driven Search Algorithm: you are required to design and implement an Evolutionary Algorithm (for example, a genetic algorithm) to solve the **Traveling Salesman Problem (TSP)** and conduct a set of experiments to evaluate its performance in contrast to simple local search and other algorithmic approaches you should have already developed.

Performance Evaluation: Execute both algorithms on the provided optimization problem. Measure and compare their performance based on criteria such as solution quality, convergence speed, and

computational efficiency. Provide a detailed analysis of the strengths, weaknesses, and applicability of each algorithm to the problem domain.
Additional Notes: You are encouraged to explore variations or enhancements to the algorithms to improve performance, provided they are clearly documented and justified.

Mandatory Points:

1. You are required to implement both algorithms and compare their performance in addressing the optimization challenge.
2. You will need to evaluate each algorithm's solution quality, computation time, and robustness to determine how effectively it solves the problem.
3. You must critically evaluate the performance of both algorithms, highlighting their strengths and weaknesses, and discuss why one algorithm may outperform the other for the given problem.
4. In addition to the implementation, you must document your process, provide the code, and use charts or tables to support your comparison.
5. The final submission should include your quality code and a detailed report summarizing your analysis, findings, and conclusions with a short video recording.

Section 3: Deliverables

Item	**Detail**	**Date & Submission Mechanism**
Source Code solution in Python	Source code for both AI search algorithms must adhere to PEP8 style guidelines, ensuring clean, readable, and maintainable code. It should include clear and concise comments and docstrings to explain the purpose and functionality of functions and classes	Via Blackboard
Technical Report (2000-2500 words)	You should then compile and submit a report that describes: • Your chosen methodology for comparing algorithms. • The results obtained and your analysis of them – using figures/graphs and appropriate statistical hypothesis tests to illustrate and substantiate your analysis. ○ Your recommendations about which algorithm should be deployed in different contexts. Contexts might typically be defined by factors such as: how much run-time is available, the number of cities to be visited, and other characteristics of problem instances you find from the literature. • How you managed legal and ethical issues around licensing conditions for any code and/or datasets you use, and whether these would be different if you were operating in a commercial setting.	Via Blackboard
A short 10-minute video presentation(recorded)	You are required to submit a 10-minute presentation, which you will deliver to a general public audience and Data Scientists. Your presentation should cover the following points: 1. Design of Choices with justification 2. Evaluation of the methodology 3. Result analysis and discussion on which algorithm is optimal and why?	Via Blackboard