NATIONAL UNIVERSITY OF SINGAPORE
DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

ACADEMIC YEAR 2025-2026
SEMESTER 1

EE2213: Introduction to AI

ASSIGNMENT 2: L4

Assignment 2 (6% of total grade)
Submission Deadline: 19 September 2025 (Friday), 23:59

Problem Statement:

You are given a map of cities connected by roads. The distances between directly connected cities are known and represented as edge costs. Each city also has a known geographic location represented by 2D coordinates (e.g., latitude and longitude on a simplified plane).

Your task is to implement the A* search algorithm in Python to find the shortest driving route from a specified start city to a destination city.

Your implementation should:

- Use the Euclidean distance between city coordinates as a heuristic estimate of remaining distance to the destination.
- Find the route with the minimum total driving distance.
- Return the list of cities along the shortest path, including the start and destination cities, as well as the total distance traveled.

Assume the Euclidean heuristic is consistent.

Hint: you may find heapq library useful.

Input format (we will use this fixed format during grading):

- The road network is represented as a dictionary where each city maps to its neighboring cities and the distance to each neighbor.
- The coordinates of each city are given as (x, y) tuples in a separate dictionary.

For example:

```
road_map = {
  'A': {'B': 4, 'C': 6, 'D': 3},
  'B': {'F': 6},
  'C': {'D': 2, 'E': 3, 'F': 3},
```

```
    'D': {'E': 4},
    'E': {'F': 1},
    'F': {}
}

city_coordinates = {
    'A': (1, 2),
    'B': (2, 5),
    'C': (5, 2),
    'D': (7, 4),
    'E': (9, 5),
    'F': (10, 8)
}
```

Deliverable:

Write a Python function A2_MatricNumber (road_map, city_coordinates, start_city, destination_city) that returns:

- The shortest path as a list of city names (including start and destination), and
- The total distance of this shortest route.

Instructions:

Submit a file named **A2_StudentMatriculationNumber.py** (replace "StudentMatriculationNumber" with your own your student matriculation number).

This file should contain one Python function with the following signature: (def A2_MatricNumber(road_map, city_coordinates, start_city, destination_city))

Your function should take the following inputs:

- road_map: a dictionary representing the graph (adjacency list with distances)
- city_coordinates: a dictionary mapping city names to their 2D coordinates (used for heuristics)
- start_city: the starting city (string)
- destination_city: the goal city (string)

Your function must return:

1. The shortest path as a list of city names (including start and destination) *(type: list)*

2. The total cost of the shortest total path (as a number) *(type: float)*

Do not redefine road_map, city_coordinates, start_city, destination_city inside the function. We will do it in the main calling part.

You are required to implement A* search by yourself. The use of built-in A* search functions (such as those in the networkx library) is not allowed.

Rename both your file and your function using your student matriculation number.
For example, if your matriculation ID is **A1234567R**, then:

- Filename should be **"A2_A1234567R.py"**
- Function name should be **"A2_A1234567R"**

Please use the provided template (A2_StudentMatriculationNumber.py) and do not comment out any lines. You can use the heuristic function in the template to compute heuristics. The template also contains helpful hints to guide your implementation.

Please do NOT zip/compress your file.

Please ensure you test your code before submission. You are encouraged to create your own test files to check whether your code produces the correct output. For grading, we will use a variety of test cases. Therefore, it's strongly recommended that you design and run multiple test cases to confirm your code works correctly across different graph structures.

You must follow instructions strictly; marks will be deducted otherwise due to large class size.

Marks Allocation:

- Correct shortest distance: 3%
- Correct output path: 3%

The way we would run your code might be something like this (main calling part):

>> import A2_A1234567R as grading

>> #code to define road_map, city_coordinates, start_city, destination_city --- you do not need to redefine them inside your function

>> path, cost = grading.A2_A1234567R(road_map, city_coordinates, start_city, destination_city)

Please ensure your filenames and function names follow the specified format exactly.

Important Notes:

Please **replace** both "StudentMatriculationNumber" and "MatricNumber" in your **filename and function name** with your **actual matriculation number**.

Submission folder: Canvas→EE2213→Assignments→A2

Deadline:
23:59, 19 September 2025 (Friday)
No extensions will be granted.

Late submission policy:

- Up to 12 hours late: 80% of your grade
- 1 day late: 50% of your grade
- 2 days late: 30% of your grade
- More than 2 days late: No marks (0%)