COMENIUS UNIVERSITY IN BRATISLAVA
FACULTY OF MATHEMATICS, PHYSICS AND INFORMATICS
DEPARTMENT OF APPLIED INFORMATICS

Introduction to Artificial Intelligence

Exercise 2 - Graph search - UC & A*

October 2, 2025

GRAPH SEARCH

The goal is to implement searching algorithms **Uniform-cost search** and **A***, and use them to find the shortest path (in km) between stops of Bratislava public transport.

Note: In this exercise we do not consider specific lines, just connections between them (so not "first i take line 39, then switch to 4", but "from Zoo i will go to Lanfranconi, from there to Botanical garden...").

Algorithm 1 - general tree (or graph) searching algorithm

```
1:  procedure UNIVERSAL SEARCH(start_node)
2:      Open ← start_node
3:      Explored ← {}
4:      while Open ≠ ∅ do
5:          N ← Open.pop()
6:          if goal_reached(N) then
7:              return N
8:          add N to Explored
9:          Add all neighbours of N to Open
10:     Search unsuccessful, finish
```

Difference between DFS, BFS, Uniform-Cost, A* and others is just in the data structure used for OPEN list and how we add nodes to it.

Program:

File ba_mhd_db.json containts data - stops and their connections. You do not need to load or parse the file, all necessary methods are already included in the code.
File bamhd.py contains basic code structure, including examples showing how to use the pre-implemented functions, it is recommended to have a look at them. **Use class BusStop to represent the nodes**.

Implement Uniform-cost search and A* search to find the shortest route (in km) between two stops. Also consider the case where the stops are not connected, for example return ['Route not found']. Print how many nodes of graph the algorithms added to the *Open* list and the length

of the found route (just feed data to the print that is already prepared in the code).

Sanity check: routes "Zoo - Aupark"/"VW - Astronomicka" rout lengths are = 3.89km/18.08km.