Lab 6 - Shortest Path Algorithms

1. Implement Dijkstra's algorithm using any graph representation you want: [2]

```
RELAX(u, v, w)
1  if v.d > u.d + w(u, v)
2      v.d = u.d + w(u, v)
3      v.π = u

INITIALIZE-SINGLE-SOURCE(G, s)
1  for each vertex v ∈ G.V
2      v.d = ∞
3      v.π = NIL
4  s.d = 0

DIJKSTRA(G, w, s)
1  INITIALIZE-SINGLE-SOURCE(G, s)
2  S = ∅
3  Q = G.V
4  while Q ≠ ∅
5      u = EXTRACT-MIN(Q)
6      S = S ∪ {u}
7      for each vertex v ∈ G.Adj[u]
8          RELAX(u, v, w)
```

2. Implement the Floyd-Warshall algorithm using any graph representation you want: [2]

```
FLOYD-WARSHALL(W)
1  n = W.rows
2  D^(0) = W
3  for k = 1 to n
4      let D^(k) = (d_ij^(k)) be a new n × n matrix
5      for i = 1 to n
6          for j = 1 to n
7              d_ij^(k) = min(d_ij^(k-1), d_ik^(k-1) + d_kj^(k-1))
8  return D^(n)
```

3. Implement the A* algorithm using any graph representation you want: [1]

$$d(v) \leftarrow \begin{cases} \infty & \text{if } v \neq S \\ 0 & \text{if } v = S \end{cases}$$

```
Q := the set of nodes in V, sorted by d(v) + h(v)
while Q not empty do
  v ← Q.pop()
  for all neighbours u of v do
    if d(v) + e(v, u) ≤ d(u) then
      d(u) ← d(v) + e(v, u)
    end if
  end for
end while
```

Hint: You can add 2d coordinates to the nodes in the graph and use the Euclidean distance between each node and a destination node as the function h

Note: Leave a comment with the text PB1, PB2.A.II, ... PB10 above every function that implements the respective lab task. (upper case text, no space between the text and the problem number)

References

[1] Siyang Chen. *The A* Search Algorithm*. https://courses.cs.duke.edu/fall11/cps149s/notes/a_star.pdf. Accessed: 2025-03-31.

[2] Thomas H Cormen et al. *Introduction to algorithms*. MIT press, 2022.