Report: Assignment 1
by Maria Karpova m.karpova@innopolis.university DSAI-04
To begin the explanation, I would like to introduce my Github repository

1. **A* algorithm idea**

   The key idea is from my A* code (in repository named "BASE A*" A star.cpp) that I have written as a homework from Lab 5. The algorithm uses a heuristic search to find the shortest path from the starting position to the goal, it evaluates each cell using the function f(n) = g(n) + h(n), where g(n) is the cost of the path from the start, h(n) is the heuristic estimate to the goal. The algorithm maintains a priority queue by processing the most promising cells first, which ensures that the optimal path is found with acceptable heuristics. This implementation uses the Manhattan distance as a heuristic, and the algorithm dynamically recalculates the danger zones depending on the condition of the ring and the mithril.

2. **Backtracking algorithm idea**

   The key idea is from my Backtracking code (in repository named "DFS" Main.java) that I have written as a homework at the Week 10 Coding exercises. The algorithm uses a depth-first search (DFS) to find a path. The algorithm recursively explores all possible directions of movement, marking the visited cells and returning when a dead end is reached. The implementation uses the color labeling of cells (WHITE - not visited, GRAY - in the process of processing, BLACK - processed). Although this method does not guarantee finding the shortest path, it systematically explores all possible routes and finds a valid path if it exists.

3. **Statical comparison**

	A* V1	A* V2	Backtracking V1	Backtracking V2
Wins	970	939	968	931
Losses	30	61	32	69
Average path, steps	20.9979	21.0021	167.281	166.466
Median, sec	0.00018405	0.000184	4.67e-05	4.685e-05
Standard Deviation	7.97483e-05	9.34769e-05	1.61436e-05	1.78728e-05
Mean, sec	0.000200534	0.000202641	4.71e-05	4.72e-05
Mode, sec	≈ 0.000184	≈ 0.000184	≈ 4.67e-05	≈ 4.69e-05
MIN Time, sec	6.92e-05	6.77e-05	1.44e-05	1.34e-05
MAX Time, sec	0.0007729	0.0009358	0.0001355	0.0001332

   In variant 1 Backtracking is faster for about 6.53409e-05 s, and in variant 2 Backtracking is faster for about 6.60762e-05 s. Backtracking is faster than A* because it uses simple depth-first search without the expensive operations of maintaining a priority queue and heuristic calculations, which eliminates the computational burden typical of A*, which guarantees an optimal path at the cost of significant time spent on sorting and evaluating nodes.

   The comparison between the two Frodo's perception variants reveals a notable pattern: both algorithms achieved higher success rates in Variant 1 compared to Variant 2, despite the larger perception radius in the latter. The expanded perception range in Variant 2, while providing more environmental information, appears to create more challenging decision-making scenarios.

4. **PEAS**

Performance measure:
- Destroy the ring in Mount Doom
- End Mordor's Era
- Find the shortest safe solution
- Avoid detection and death from enemies and survive the journey
- Efficiently use the mithril and ring
- Complete the mission with optimal resource management

Environment:
- Map 13x13 grid, indexed by numbers in range of [0, 12]
- Partially observable, sequential, dynamic, discrete, multi-agent, stochastic
- Frodo limited to Moore neighborhood perception (radius 1 or 2)
- Threat zones change based on ring and mithril status

Actuators:
- Activate and deactivate ring using 'r' and 'rr' commands
- Systematically discover map through perception and movement
- Calculate optimal and/or safe routes using search algorithms
- Move orthogonally to adjacent cells (North/South/East/West)

Sensors:
- Discover mithril when within perception range
- Know Gollum's initial position and learn Mount Doom location after Gollum contact
- Track current position, ring/mithril status, and known map areas

5. **Impossible maps (created using Excel)**

FRODO	.	.	.	.	.	.	.	.	.	.	.	.
.	WATCHTOWER	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	URUK-HAI	.
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	NAZGUL	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	ORC
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	GOLLUM
.	.	.	.	.	.	.	.	.	.	.	MOUNT	.
.	.	.	.	.	.	.	.	.	.	.	.	.
ORC	.	.	.	.	.	.	.	.	.	.	.	.

FRODO	.	WATCHTOWER	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	URUK-HAI
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.		.	.	.	.	.	GOLLUM	.
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	NAZGUL	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	ORC
.	.	.	.	.	.	.	.	ORC	.	.	.	.
.	MOUNT	.	.	.	.	.	.	.	.	.	.	.

FRODO	.	.	.	.	.	.	.	.	.	.	.	.
.	NAZGUL	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.
WATCHTOWER	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	GOLLUM	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	URUK-HAI	.	.	.	.	ORC	.	.	.	.
.	.	.	ORC	.	.	.	.	.	MOUNT	.	.	.

FRODO	.	.	.	.	.	.	.	.	.	.	.	.
.	WATCHTOWER	.	.	.	.	.	.	.	.	.	.	.
.	ORC	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.
.	URUK-HAI	.	.	.	.	.	NAZGUL	.	.	.	.	.
.	.	GOLLUM	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	ORC	.
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	MOUNT	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.

FRODO	.	.	.	.	.	.	.	.	.	.	.	.
NAZGUL	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	MOUNT	WATCHTOWER	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	ORC	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	GOLLUM	.	.	.	.	ORC	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.