TITLE

Informed Search using Python.

OBJECTIVES

- Solving problems by search
- Discuss the A* algorithm.
- Compare different heuristic functions
- Learn how to formalize a problem as a search problem.

PREREQUISITES

- Python 3+.
- Online Python IDE
- Search Problem Solver Framework
- Manhattan distance
- Euclidian distance
- Course Chapter 6

RESOURCES

- Course Slides
- Python Tutorial
- Python Tuples

LAB

Read first:

8-puzzle problem.

In search.py at line 422, you will be able to see the implementation for the 8-puzzle problem using the Python-based framework used also on the previous laboratory.

The code is well commented, so you have to analyze the code.

- Based on Code Skeleton implement the followings:
  - Implement the Manhattan distance heuristic function for the 8-puzzle problem. In the eight_puzzle.py file, you will find two EightPuzzle subclasses. Pick EightPuzzleMht and override the *h* method to return the sum of Manhattan distances of all tiles related to the goal state.
  - Using main.py code, try to compare various heuristic functions for the 8-puzzle problem. Try to understand how the *compare_searchers* function. What does this function return?
  - Extend the 8-puzzle problem in order to solve the 15-puzzle problem.
  - Calculate branching factor for each heuristic function

$$N + 1 = 1 + (b^*) + (b^*)^2 + \ldots + (b^*)^d \quad ,where\ N - the\ number\ of\ nodes\ expanded, d - depth\ of\ solution$$

$$Low\ branching\ factor\ means\ better\ heuristic.$$