

**CENG 461: Artificial Intelligence**  
**Homework #1**  
**Depth First, Breadth First and Uniform Cost Search**

**DUE DATE: 11.11.2018 - 23:55**

### 1. Implementation



Suppose there is a maze in which the agent can move one tile at a time, horizontally or vertically. It starts at a given starting tile and searches for the goal tile. Each action costs 1 movement point. Additionally, the destination tiles may take additional movement points as stated on the tiles. The maze also has walls that are represented with '-' and cannot be passed through. The agent can sense the walls and does not consider the action to go towards them.

For any given maze, you are expected to program an agent (in PYTHON) to perform:

- a) breadth first search
- b) depth first search
- c) uniform cost search, for which the costs will be taken as the total cost in movement points:
  - i. not taking extra move points into consideration
  - ii. taking extra move points into consideration
- d) A\* search using two sensible distance heuristics that you determine and which are
  - i. inadmissible
  - ii. admissible

An example maze is given below:

-	0	1	0	1
-	-	2	1	2
2	-	3	0	-
0	2	1	1	-
1	0	3	1	-

 : Starting tile       : Goal tile

This can be hard-coded as a 2D list at the beginning of your program. But, the agent should perform even if the map values and configuration are modified.

### 2. Reporting

- a. Explain how you decide on the heuristics you determine.
- b. Comment on your findings with each approach (a, b, c-i, c-ii, d-i and d-ii):
  - Can the agent find all possible paths from the starting tile to the goal tile? How can this be made possible?
  - Can the agent find the path from the starting tile to the goal tile with minimum number of actions? If so, how? If not, what are the causes?
  - Can the agent find the path that it reaches the goal tile with the minimum cost in movement points? If so, how? If not, what are the causes?
  - What would happen if additional movement points were negative?