Section 1)

1. Search Strategies

Briefly explain, compare and contrast the Breadth First Search, Depth First Search, Iterative Deeping Search and the AStar Search in the context of search trees.

BFS:
- BFS traverses a search tree in the order of: left, right and then down to the next layer.
- It is complete in a finite b space.
- It has a time complexity of O(b^d) since it traverses entire breadth up until the first layer that contains the goal state
- It has a space complexity of O(b^d) as it save every node that it generates
- BFS is an optimal algorithm

DFS:
- DFS traverses a search tree in the order of: bottom and then to the right
- It is complete in a finite m space.
- It has a time complexity of O(b^m) since it traverses to the bottom layer (potentially) until it finds a goal state
- It has a space complexity of O(b*m) as it save every node that it generates but is layer by layer first
- DFS is not an optimal algorithm

IDS:
- IDS traverses a search tree using the DFS algorithm but for each iteration has a layer-depth restriction.
- It is complete.
- It has a time complexity of O(b^d) since it traverses entire breadth up until the first layer that contains the goal state
- It has a space complexity of O(b*d) as it save every node that it generates
- IDS is an optimal algorithm

AStar:
- A* search takes advantage of a heuristic and cost function to determine the order in which it traverses a search tree
- It is complete when a solution exists
- It has an exponential time complexity, dependent on the heuristic function
- It saves every node that it generates
- AStar search is a n optimal algorithm, it goes down the “cheapest” paths (through contours) first when in search of a goal state.

Section 2)

Instructions:

a) Use the files in Assignment4.zip.
b) A4_Base.py file contains the class definition for Graph, GraphProblem and a hardcoded state transition dictionary for the graph example we discussed in class. DO NOT EDIT THIS FILE.
c) A4_ToDo.py consists of the skeleton code, which should be completed. Your implementation goes in this file.
d) A4_Tests.py consists of test cases for the assignment. DO NOT EDIT THIS FILE. To run this file in PyCharm, open the file, select "Run" from the menu bar, and select "run pytest in A4_Tests.py".
e) You only need to submit the A4_ToDo.py

1. (30 points) Implement the AStar search by completing the function

   ```
   def astar_search(problem, h=None):
   ```

   NOTE: Your code must print out f(n) = h(n) + path_cost values for nodes as they are explored/explanded.

2. (30 points) Implement the AStarGraph sub-class which inherits the GraphProblem class. Add a heuristic function h which computes the straight line distance between cities using Romania_map.locations.

   NOTE : Your implementation must print out the h values as and when they are computed.

3. (10 points) For passing all test cases in A4_Tests.py