Lab 01: Search

I. The project objectives

Research, implement, and present graph search algorithms

II. Requirements

- The project is conducted individually.
- Deadline and submission method can be found on Moodle.
- Implement the following graph search algorithms:

    1. Breadth First Search (BFS)
    2. Depth First Search (DFS)
    3. Uniform-Cost Search (UCS)
    4. Greedy Best First Search (GBFS)
    5. A*

    Your task is to write the code for the BFS, DFS, UCS, GBFS, and Astar functions that have already been defined in the file student_functions.py. Do not modify other functions or files (though you may add new functions or files if necessary).
    Note:

    - With Greedy Best First Search, choosing h = edge weight
    - With A*, h = eclidean_distance(pos[current vertex], pos[Goal])
    - Read the code of the example_func() function in the search_algorithm.py file to understand how to color the vertices and edges of the graph.
    - To update the graph's state during the search, call the function graphUI.updateUI().

- Provide at least 5 test cases and illustrate each test case in the report. Test case example:

    - Input:

```
4 0
0 3 7 6 9 0
3 0 8 0 3 6
5 7 0 4 0 0
9 0 8 0 6 0
7 2 0 2 0 1
0 4 0 0 8 0
```

    - Notes:
        - Starting at node 4 and Goal is node 0.
        - edge[0][1] denotes the edge between node 0 and node 1.
        - 3 is the edge weight of edge[0][1].

- (0,0) is the pos[0].
- (1,1) is the pos[1].

III. Report

- Content:
  - **Student Information:** Full name, student ID, etc
  - **Completion Level of Each Requirement:** Self-assessment of the project on a scale of 1 to 10.
  - **Presentation of Basic Theories:** Including concepts, complexity, properties, etc., of each implemented algorithm.
  - **Comparison of UCS and A* Algorithms:** Highlighting the differences between Uniform Cost Search and A*.
  - **Bonus for Additional Search Algorithms:** Extra points if additional search algorithms, beyond the four specified (BFS, DFS, UCS, GBFS, A*), are implemented.
- Source Code: Code must have comments and be clearly organized.
- Videos: Screen recordings for each algorithm applied to a test case. For example, at least four videos (BFS, DFS, UCS, A*) for one test case. Submit via a Google Drive link stored in the file Video.txt.

  For example:

```
Test case 1:
+ BFS: link
+ DFS: link
+ ...
```

- Submission requirements:

```
<StudentID>.zip
|   <StudentID>.pdf
|   Video.txt
└──Source
    |student_functions.py
    |main.py
    |...
```

Example:

```
23120027.zip
|   23120027.pdf
|   Video.txt
└──Source
    |student_functions.py
    |main.py
    |...
```

IV. Assessment

- For each completed algorithm, you will receive 1.5 points for the code and 0.5 points for the report.
- Test cases that are not reported will not be graded.
- Any referenced report/source code must be clearly cited at the end of the report.
- Cheating and plagiarism will receive a grade of 0 for the course.
- Folders submitted that do not meet the requirements will not be graded.
- If you submit late after the deadline, 30% of the points will be deducted. After one day past the deadline, no submissions will be accepted for any reason.

V. Contact

Contact teacher via email at ntthuhang0131@gmail.com