DS251

Assignment 1

Due Date - 25th September EOD

Q1. CONVERSION ORDER [10 Marks]

Given a list of words as a dictionary, you are tasked to find the smallest order that converts one word to another word using the dictionary.

The order $word1 \rightarrow word2 \rightarrow word3 \rightarrow ... \rightarrow wordn$ should satisfy the following conditions :

- All adjacent words should differ by a letter
- All words should be of the same length
- All words should be unique
- Except *word* 1 , all words should exist in the dictionary

Use the following uninformed search algorithm to solve the problem.

1. Breadth First Search
2. Iterative Deepening Depth First Search

Input

- First line describes the dictionary length K.
- Second line describes the start and end word separated by space. Do note the second word belongs to the dictionary.
- Following K lines give the different words in the dictionary.

Output

- Every word in the transition separated by space.
- Do note that the last word should match with the target word.

Example

Input

```
5
sky sun
spy
soy
son
sun
```

```
sum
```

output

```
sky soy son sun
```

Q2. Drone Delivery [10 Marks]

You are the lead engineer at a logistics company that operates autonomous delivery drones for package deliveries in urban areas. The drones navigate a grid-like city environment to reach specific delivery destinations while avoiding obstacles and optimizing delivery time. The city is represented as an 8x8 grid, where each cell represents a location in the city. The grid is defined as follows:

```
. . . . . . # .
. . # # # . # .
. . # . # . # .
. . # # # . # .
. . # . # . # .
. . # . # . # .
. . . . . . # .
. . . . . . . .
```

```
. : Open space where the drone can navigate.
# : Obstacles or impassable terrain that must be avoided.
```

You are tasked with developing a path planning algorithm for the delivery drone to navigate from the starting point (1, 1) to the delivery destination (7, 7) in minimum amount of time, while taking one step of 1 unit at a time and avoiding obstacles.

1. Greedy Best First Search
2. A* search

Constraints :

- Heuristic: **Manhattan distance** between the current cell and the goal cell.
- Boundary: The drone should stay within the boundaries of the grid and the drone cannot move in diagonal directions, It can only move horizontally or vertically to adjacent grid cells.

Input

- A 8x8 matrix of "." and "#", where "." represents empty space, and "#" represents obstacle.

Output

- Minimum travel time to reach (7, 7)

Example

Input

```
...... #.
..###.#.
..#.#.#.
..###.#.
..#.#.#.
..#.#.#.
...... #.
...... ..
```

Output

```
14
```

Q3. Path-finding in a Maze with Terrain Costs [15 Marks]

Imagine you have a grid-based maze representing a terrain with varying levels of difficulty to traverse. The maze contains different types of terrain(Grass = G, dirt = D, rocks = R), each with a specific traversal cost (e.g., grassy fields are easy to traverse, while rocky areas are more challenging). The cost is defined for the target terrain.

Target Terrain	Cost
Grass	1
Dirt	2
Rock	5

The terrain is represented as a grid, where each cell can be one of several terrain types, each with an associated traversal cost. You have a specific starting point and a goal point within the maze and you can move in four cardinal directions (up, down, left, right) from one cell to another.

```
G D R G F
G R R D G
D G G R G
G R G G D
S G D R G

G - GRASS
D - DUST
R - ROCKS
S - START
F - FINISH
```

Find the path from the starting point to the goal point that minimizes the total traversal cost while obeying the rules of movement (i.e., you cannot move through walls or impassable terrain) using the following Informed searches.

1. A* Search
2. Weighted A* Search

   Note : For each of the algorithms, mention the following details.

   - **State Representation** : Define the state representation, which includes the current position (cell) and the cumulative traversal cost.
   - **Heuristic Function (h)** : Develop a heuristic function that estimates the cost from the current cell to the goal. A common choice might be the Euclidean distance or the Manhattan distance between the two cells, weighted by the terrain type.
   - **Cost Function(g)** : Define the cost function to keep track of the cumulative traversal cost from the start to the current cell.

Input

- First line contains the dimension of the terrain, No. Of rows(N) and No. Of columns(M)
- N lines follow, where each line has M characters, representing the cell terrain.

Output

- The minimum cost form S(start) to F(Finish).

Q4. Robot Navigation [15 Marks]

Consider the problem of finding the shortest path between two points on a plane that has convex polygonal obstacles (see Fig below). This is an idealization of the problem a robot has to solve to navigate its way around in a crowded environment. Based on this, answer the following questions.

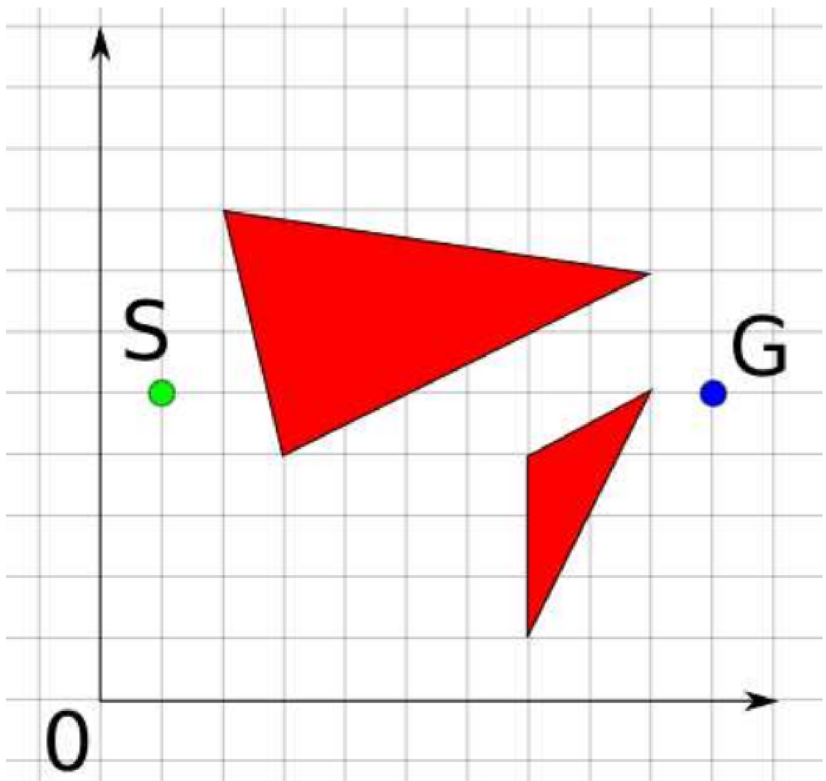

Figure 1: Robot Map

1. Suppose the state space consists of all positions (x, y) in the plane. How many states exist? How many paths are there to the goal?
2. We are interested in the shortest path to the goal. This runs along the corners of the polygons and therefore consists of line segments that connect the polygon's corners. We formulate the state space to contain the corners of all polygons as well as the start and goal coordinates. State the full successor function for the states (1, 5) (start) and (3, 4) in the problem figure given above.
3. We will now use hill-climbing in the same setting, that is, planar robot

navigation among polygonal obstacles. We assume that the obstacles do not touch each other.

4. Explain how hill-climbing would work as a method of reaching a particular end point. Is it guaranteed to find the path?
5. Show how non-convex obstacles can result in a local maximum for the hill-climber, using an example

Instructions for submission

- Make Separate folder for each question.
- Each folder should have the following files.
  - Readme.md/.txt
  - Writeup.md/.txt
  - code.py (optional)
- Zip all the question folders in a single zip file and name it as, `GroupX_rollno1_rollno2_rollno3_rollno4.zip`