National University of Computer and Emerging Sciences

Lab Manual 07
CL461-Artificial Intelligence Lab

Course Instructor	Saif ul Islam
Lab Instructor (s)	Abdul Rehman Sukhan Amir
Section	BCS-6D
Semester	Spring 2024

Background

Anato was chilling in his room waiting for the new "Attack On Titans" episode to air when suddenly he hears an emergency broadcast on the T.V. that zombies have attacked the city, although he is disappointed that why did it have to be a zombie attack and not a titans attack, regardless, Anato now has to find a way to get to one of the safe bunkers that the government have announced.

You must design a code that can find the quickest and safest route to a bunker, so that Anato can get to the safe bunker before the AOT episode airs and not miss his episode. [hopefully, the bunker will have internet otherwise Anato will be very sad (ToT)]

Function Structures:

You must make functions that take the following parameters:

- **Reading maps from a text file:** Write a function to read a matrix map from a text file. The function should take a filename as input and return a 2D array representing the map.

  Hint: The matrix can be represented using a binary matrix, where 0 represents an empty cell and 1 represents a wall.

- **Implementing Search Algorithm**: The function should take a 2D array, which showing the map of the city, 'z' will should areas where there are zombies and 'w' where there are survival items that can help kill the zombies to open new paths. 'A' will show Anato's position and 'S' will show the safe houses.

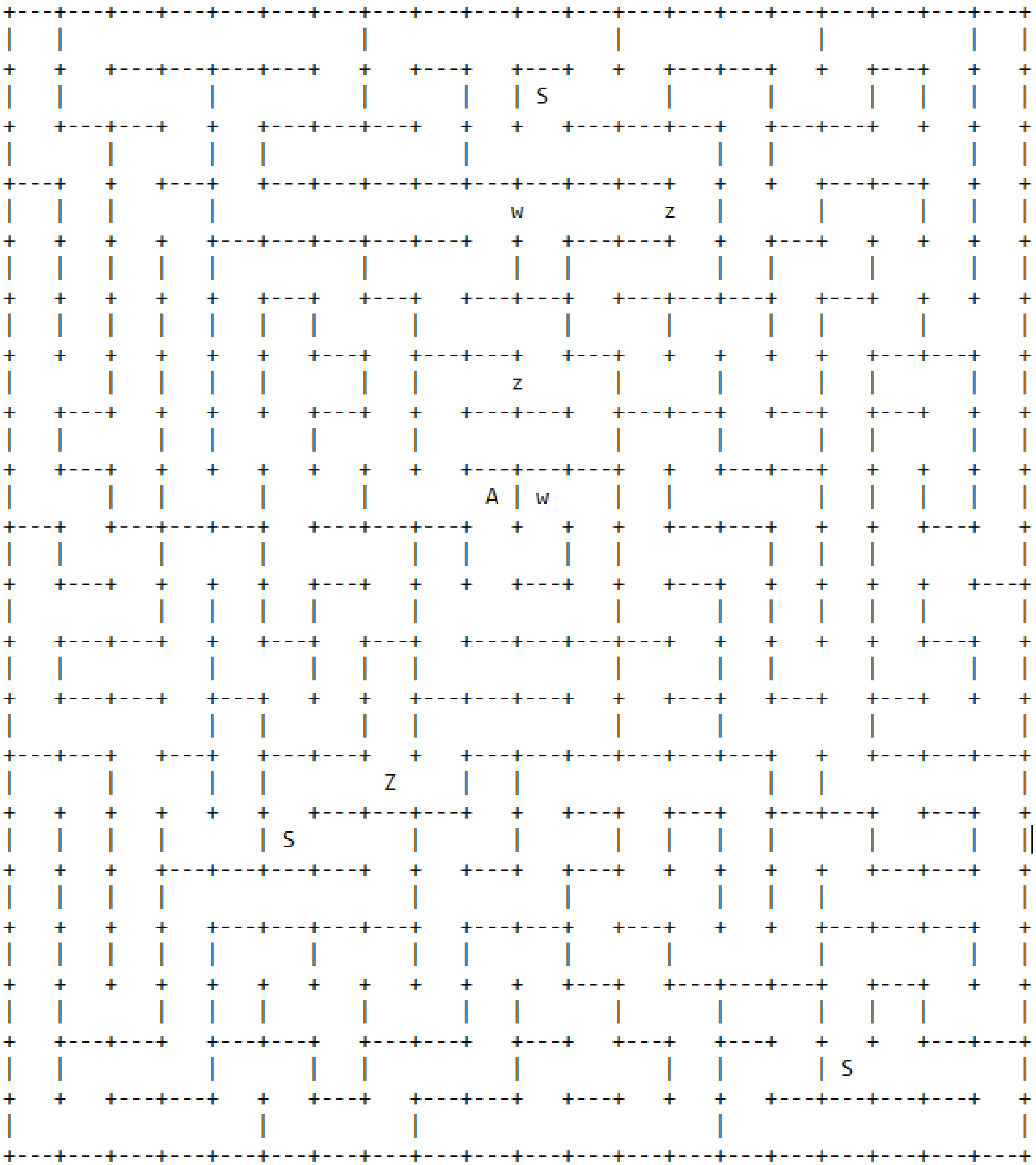
- **Returning the path to the goal state:** The function should return a list of coordinates representing the path from the starting point to the goal state. If there is no path, the function should return -1 and print "Oh, no. Anato is doomed and going to die in suspense without watching the Final episode."

Note: It is important to note that a map can have multiple goal states or safe bunkers, which means that different algorithms may produce different paths to reach different goal states.

Your task is to implement a path finder algorithm function for each algorithm:

- **Depth-First Search-DFS**
- **Breadth-First Search-BFS**).
- **Uniform Cost Search (UCS)**
- **A* Algorithm**

Each algorithm must carefully follow the rules to give the safest, quickest and shortest path.



```
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|   |                       |                   |               |           |   |
+   +   +---+---+---+---+   +   +---+   +---+   +   +---+---+   +   +---+   +   +
|   |           |           |       |   | S         |       |       |   |   |   |
+   +---+---+   +   +---+---+---+   +   +   +---+---+---+   +---+---+   +   +   +
|       |       |   |               |                   |   |               |   |
+---+   +   +---+   +---+---+---+---+---+---+---+---+   +   +   +---+---+   +   +
|   |   |       |                       w           z   |       |       |   |   |
+   +   +   +   +---+---+---+---+---+   +   +---+---+   +   +---+   +   +   +   +
|   |   |   |   |           |           |   |           |   |       |       |   |
+   +   +   +   +   +---+   +---+   +---+---+   +---+---+---+   +---+   +   +   +
|   |   |   |   |   |   |       |           |       |       |   |       |       |
+   +   +   +   +   +   +---+   +---+---+   +---+   +   +   +   +   +---+---+   +
|       |   |   |   |       |   |       z       |       |       |   |       |   |
+   +---+   +   +   +   +---+   +   +---+---+   +---+---+   +---+   +---+   +   +
|   |       |   |       |       |               |       |       |   |       |   |
+   +---+   +   +   +   +   +   +   +---+---+---+   +   +---+---+   +   +   +   +
|       |   |       |       |         A | w     |   |           |   |   |   |   |
+---+   +---+---+---+   +---+---+---+   +   +   +   +---+---+   +   +   +---+   +
|   |       |       |           |   |       |   |           |   |   |           |
+   +---+   +   +   +   +   +---+   +   +   +---+   +   +---+   +   +   +   +   +---+
|           |   |   |   |       |               |       |   |   |   |   |       |
+   +---+---+   +   +---+   +---+   +---+---+---+---+   +   +   +   +   +---+   +
|   |           |       |   |   |               |       |   |       |       |   |
+   +---+---+   +---+   +   +   +---+---+---+   +   +---+   +---+   +---+   +   +
|               |   |       |   |               |       |           |           |
+---+---+   +---+   +---+---+   +   +---+---+---+---+---+---+   +   +---+---+---+
|       |       |   |         Z     |   |                   |   |               |
+   +   +   +   +   +   +---+---+---+   +   +---+   +---+   +---+---+   +---+   +
|   |   |   |       | S         |       |       |   |   |   |       |       |   |
+   +   +   +---+---+---+---+   +   +---+   +---+   +   +   +   +   +---+---+   +
|   |   |   |                   |           |           |   |   |               |
+   +   +   +   +---+---+---+---+   +---+---+   +---+   +   +   +---+---+---+   +
|   |   |   |   |       |       |   |       |       |           |           |   |
+   +   +   +   +   +   +   +   +   +   +   +---+   +---+---+---+   +---+   +   +
|   |       |   |   |       |       |   |       |       |       |   |   |       |
+   +---+---+   +---+---+   +---+---+   +---+   +---+   +---+   +   +   +---+---+
|   |           |       |   |           |           |   |       | S             |
+   +   +---+---+   +   +---+   +---+---+   +---+   +   +   +---+---+---+---+   +
|                   |           |                       |                       |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
```

Rules:

1. The number of ‘w’s help kills the number of ‘z’s in the path.
2. The safety of the path can be considered as no of ‘w’s – no of ‘z’s on the path
3. The quickness of the path depends on the less no of turns in the path
4. The shortest of the path with depend on the travel distance
5. Apply appropriate values to each variable to give the most appropriate path.