

**Name: Hafsa Siddiqua, Mahnoor Mehmood**

**Roll No: 22I-0542, 22I-0460**

**Subject: AI (Artificial Intelligence).**

**Section: AI(C)**

**Assignment # 2**

Description:

We are stranded in an Enchanted Forest at Northwest Corner now without knowing the path we have to reach treasure with the minimum cost taken alongside the path.

Overview:

Now to solve this problem we are applying two different algorithms.

1. A\* search
2. UCS search

**UCS Search:**

It means getting to the goal or endpoint with minimal cost. It can sometimes consider long distances more efficient due to its constraint for cost. However, cost alone is a constraint that decides whether a solution is optimum or not.

**A\* Search:**

It means to get to the goal or endpoint with the shortest distance possible and with the minimum cost. These two constraints of shortest path and cost efficiency make it a much more optimum solution as compared to UCS.

Instructions:

Just hit run the cell it will give you the matrix its optimum path and its cost for both UCS and A\* search.

External libraries:

There is one main library which is being used in our code

**Numpy:**

It is a library used for comprehensive mathematical functions, random number generators, and arrays and is way more fast and versatile

* We are constantly utilizing numpy as we created the 2D grid using a numpy array of size 8,8. The only reason for using the numpy array was due to its subscription abilities that allow to access any index directly.
* Secondly, we need to create the grid randomly at each run and have to block areas randomly therefore we need to use random functions in numpy such as random.randint() or random.choice()

Optimization Details:

* The code implements a simple priority queue for storing nodes, ensuring optimal exploration.
* Redundant computations are avoided by storing grid dimensions and boundaries.
* Modular code structure improves readability, maintainability, and reusability.
* Named constants represent parameters and grid dimensions for easy adjustment.
* User input handling allows for selecting search algorithms and graceful program exit.
* Documentation and comments clarify the purpose of each component, enhancing code understandability.

Dynamic Elements Handling:

Randomization:

Group Collaboration Details: