Homework Assignment 03

ITCS 5153 – Fall 2024

Introduction

This assignment is a continuation from Programming Assignment 02. This assignment introduces students to the implementation of DFS and Uninformed Cost search algorithms using Python.

Objectives

Implementing a grid-based search algorithm in Python using functions, arrays, and 2D arrays. The assignment focuses on building a modular and structured program through step-by-step process.

Tasks

Use the code that you have implemented for Homework Assignment 02, and add the following:

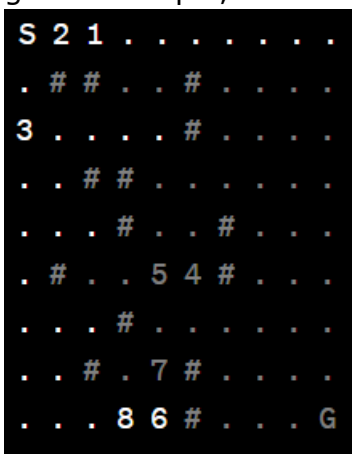
1- **Update User Input To Specify The Search Algorithm:**
   Prompt the user to input the search algorithm the wants to use find the path from Start to Goal nodes.
2- **Define Depth-First Search (DFS) Algorithm Structure:**
   Create a function for DFS that takes the grid, start, and goal positions as parameters. Initialize necessary data structures (e.g., stack, visited set, path array) within the DFS function.
3- **Implement DFS Loop:**
   Build the main loop of the DFS algorithm to explore neighboring cells and update the path array.
4- **Define Uninformed Cost Search Algorithm Structure:**
   Create a function for Uninformed Cost Search (e.g., Uniform Cost Search) that takes the grid, start, and goal positions as parameters. Initialize necessary data structures (e.g., priority queue, visited set, path array) within the cost search function.
5- **Implement Cost Search Loop:**
   Build the main loop of the Uninformed Cost Search algorithm to explore neighboring cells and update the path array.
6- **User Output and Metrics:**
   Print the final grid with the optimal paths marked for each algorithm.

Display the length of the optimal paths and the total number of cells visited/explored.

7- Comparison and Analysis:

Compare the performance of DFS, BFS, and Uninformed Cost Search in terms of path length and nodes explored.

Provide insights into the strengths and weaknesses of each algorithm in the given example, see below.




```
example_grid = [
['S','2','1','.','.','.','.','.','.','.'],
['.','#','#','.','.','.','#','.','.','.'],
['3','.','.','.','.','.','#','.','.','.'],
['.','.','#','#','.','.','.','.','.','.'],
['.','.','.','#','.','.','.','#','.','.'],
['.','#','.','.','.','5','4','#','.','.'],
['.','.','.','#','.','.','.','.','.','.'],
['.','.','#','.','.','7','#','.','.','.'],
['.','.','.','.','8','6','#','.','.','G']
]
```

What to submit

When you have completed all the programs in an assignment you will need to create a single zip file that includes your source code for each program. **The zip file should only include the .py files.** Upload and submit your zip file to Canvas in the corresponding assignment, in this case the ***Programming Assignment 3***. Make sure to use the *zip* format for your submission. Archive files in any other format (rar for example) do not meet this criterion.

Submit the Python program with step-by-step implementation using functions and arrays. Include any challenges faced during implementation and how they were addressed.

Note: This assignment focuses on implementing the different AI agent types. Students are encouraged to explore various goals and experiment with different scenarios to observe the agent's behavior.

Grading Criteria

Correctness of the step-by-step implementation.
Proper documentation and comments.
Functionality of the final program.

Grading Rubric

+20 points for submission
+10 points for the correct implementation of every step from 1 to 6
+20 points for the comparison step #7
-10 points if the submission instructions are not correctly followed