National University of Computer & Emerging Sciences
Karachi Campus
Artificial Intelligence
Programming Assignment #1

Instructions ***Each Problem of 10 points***

Due Date: The assignment is due by March 13, 2017 till midnight.

Submission: The assignment has to be submitted via slate website submission. You must submit the source code files with proper naming convention for example (Assignment No. 1 Problem No. 1) you should give CS401-Kxxxxxx-A1P1.cpp. You should copies all questions for the assignment in a single folder (named as your id e.g. K1xxxxx) and zipped it before uploading to slate.

Sample Input and Output files: The sample input and output files are available from course website at slate. Make sure that you have tested your programs against all the available input files and EXACT output file is produced.

Maze Puzzle

The Maze-Puzzle a rectangular maze where each cell is either empty (contains 0) or blocked (contains 1). There is a special cell designated as start cell and there are some cells designated as goal cells. Our search agent (Raju) need to find an escaping path from given a maze. There are some searching strategies that we need to follow in finding the optimal path. The aims of this maze puzzle is to understand different search strategies. Consider an overly simple example for this programming assignment:

Dimension 5 x 5
Initial State= starting cell (0, 0)
Goal State= goal cell (4, 4)
Optimal Path = {(0, 0), (1, 1), (2, 2), (3, 3), (4, 4)} with Path Cost = 5 when all cell movements have equal unit cost.

0	0	0	0	0
1	0	1	1	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

From the start cell there eight possible branching for the paths like Up, Down, Right, Left, LeftUp, LeftDown, RightUP and Right Down (Considering the diagonal moves).

Problem 1 Reading & Writing Maze Puzzle

In this problem you need to write a routine that read a maze from the input file. The first line of the input file contains the dimension of the maze in row, column fashioned like two integers m, n (where n, m >= 0 and < 500). The next line contains a starting cell in cell[i][j] format. The goal cell in cell [i][j] format is available in next line. From the next line you will get an n x m matrix of 0s and 1s, which is the actual maze. You need to read the maze and print it on the screen and rewrite it back onto another newly created file (output file).

Problem 2- Maze Puzzle (Uninformed-BFS)

In this problem you need to implement a solution to maze puzzle by using Breadth First Search (BFS), you are allowed to keep track of visited states. The operators that are available as per the description of the problem. The solution should print all sequence of moves (Operators used per line) that you apply to reach to a goal state. You can see the output file for problem 2.

Problem 3- Maze Puzzle (Uninformed-DFS)

In this problem you need to implement a solution to maze puzzle by using Depth First Search (DFS), you are allowed to keep track of visited states (search is a graph for this problem). The solution should print all sequence of moves that you apply to reach to a goal state.

Problem 4- Maze Puzzle (Uninformed-Iterative Deepening)

In this problem you need to implement a solution to maze puzzle by using Iterative Deepening (IDS), you are allowed to keep track of visited states. The solution should print all sequence of moves that you apply to reach to a goal state.

Problem 5-Maze Puzzle (Informed-Greedy Best First)

In this problem you need to implement a solution to maze puzzle by using Greedy Best First Search (GBFS), you are allowed to keep track of visited states. You are free to select your heuristic for the best criteria. The solution should print all sequence of moves that you apply to reach to a goal state.

Problem 6- Maze Puzzle (Informed-A*)

In this problem you need to implement a solution to maze puzzle by using A*, you are allowed to keep track of visited states. The heuristic for A* will be cost of reaching to an intermediate node and from there, cost of the diagonal distance to the goal state. Intuitively, this heuristic is

admissible. The solution should print all sequence of moves that you apply to reach to a goal state.

Input/output

First line of the input file contains the dimension of the maze in row, column fashioned like two integers m, n (where n, m >= 0 and < 500). The next line contains a starting cell in cell[i][j] format. The goal cell in cell [i][j] format is available in next line. From the next line you will get an n x m matrix of 0s and 1s, which is the actual maze.

Output file contains each move per line from the starting cell to the goal cell. The last line contains the solution cost as number of cell traveled.

Example-Problem 1

A1P1in1.txt	A1P1out1.txt
5 5 0 0 4 4 0 0 0 0 0 1 0 1 1 0 1 1 0 1 0 1 1 1 0 0 1 1 1 1 0	0 0 1 1 2 2 3 3 4 4 5

Coding Guide

You are allowed to program in C/C++ and Java Only. For each problem you need to submit one code file that should contains all the code. The code file should be named as CS401-Kxxxxxx-A1Px this is a main requirement for this assignment. Be careful while submitting the final assignment. There will be marks for following coding standards.

<The end>