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 07, 2016 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.

RGB-Puzzle

The RGB-Puzzle is a kind of sliding board puzzle similar to the 8-puzzle. The puzzle consists of an area divided into a grid, 5 by 5.On each grid square is a tile, expect for one square which remains empty. A tile that is next to the empty grid square can be moved into the empty space, leaving its previous position empty in turn. Tiles are colored in Red and Green, B represents a blank. The aim of the puzzle is to achieve a given configuration of tiles from a given (different-initial) configuration by sliding the individual tiles around the grid as described above. For example: The following diagram shows an initial and goal state of the problem.

R	R	R	G	R
R	G	R	R	R
R	G	G	G	G
R	G	G	R	R
R	R	R	B	R

Initial State

R	R	R	R	R
R	G	G	G	R
R	G	B	G	R
R	G	G	G	R
R	R	R	R	R

Goal State

Problem 1 Reading & Writing RGB -Puzzle

In this problem you need to write a routine that read an initial state and a goal state from a file. These two states are available in the same order in file, first an initial state as a 5 by 5 matrix and in the next line a goal state as 5 by 5 matrix. In the output file, you need to first output goal state first and then initial state.

Problem 2- RGB-Puzzle (Uninformed-BFS)

In this problem you need to implement a solution to RGB-puzzle by using Breadth First Search (BFS), you are allowed to keep track of visited states (RGB-puzzle search is a graph search). The operators that are available are (UP,DOWN, LEFT, RIGHT) for blank. 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- RGB-Puzzle (Uninformed-DFS)

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

Problem 4-RGB-Puzzle (Uninformed-Iterative Deepening)

In this problem you need to implement a solution to RGB-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-RGB-Puzzle (Informed-Greedy Best First)

In this problem you need to implement a solution to RGB-puzzle by using Greedy Best First Search (GBFS), you are allowed to keep track of visited states. The suggested Heuristic is h (s) = number of tiles that are not in the correct place (notcounting the blank).The solution should print all sequence of moves that you apply to reach to a goal state.

Problem 6-RGB-Puzzle (Informed-A*)

In this problem you need to implement a solution to RGB-puzzle by using A*, you are allowed to keep track of visited states. The heuristic for A* will be Hamming priority function, which can be defined as The number of blocks in the wrong position, plus the number of moves made so far to get to the search node. Intuitively, a search node with a small number of blocks in the wrong position is close to the goal, and we prefer a search node that has been reached using a small number of moves.The solution should print all sequence of moves that you apply to reach to a goal state.

Input/output

The input file contains two occurrence of a 5 x 5 board. Given as initial and goal state. For Problem-1 the output is in reverse that is goal state first and from next line initial state.

Example-Problem 1

A1P1in1.txt	A1P1out1.txt
R R R G R	R R R R R
R G R R R	R G G G R
R G G G G	R G B G R
R G G R R	R G G G R
R R R B R	R R R R R
R R R R R	R R R G R
R G G G R	R G R R R
R G B G R	R G G G G
R G G G R	R G G R R
R R R R R	R R R B R

<The end>