CPE/CSC 480 ARTIFICIAL INTELLIGENCE

MIDTERM FALL 2003

SECTION 1

PROF. FRANZ J. KURFESS

CAL POLY, COMPUTER SCIENCE DEPARTMENT

This is the first Fall 2003 midterm exam for the CPE/CSC 480 class. You may use textbooks, course notes, or other material, but you must formulate the text for your answers yourself. The use of calculators or computers is allowed for viewing documents or for numerical calculations, but not for the execution of algorithms or programs to compute solutions for exam questions. The exam time is 1 hour minutes.

Student Name:

Signature: **Date:**

PART 1: MULTIPLE CHOICE QUESTIONS

Mark the answer you think is correct. Unless otherwise noted, there is only one correct answer. Each question is worth 3 points.

a) What is an *omniscient* agent?
- ❑ An agent that always selects the best action under the given circumstances.
- ❑ An agent that performs reasoning steps to select the next action.
- ❑ An agent that can explain its choice of an action.
- ❑ An agent that is capable of predicting the actual outcome of an action.

b) In the context of search algorithms, which of the following issues addresses the question of *finding a solution* to the problem if one exists?
- ❑ completeness
- ❑ time complexity
- ❑ space complexity
- ❑ optimality

c) In the context of search algorithms, which of the following issues is determined by the *number of steps* it takes to find a solution?
- ❑ completeness
- ❑ time complexity
- ❑ space complexity
- ❑ optimality

d) What is *optimality* with respect to search algorithms?
- ❑ An estimate of how many search steps it takes to find a solution.
- ❑ An estimate of how much information the algorithm needs to store for finding a solution.
- ❑ Provided that a solution exists, the algorithm will find it.
- ❑ If there is a distinction between the quality of goal states, the agent will find the best one.

e) On the basis of the general search method as described in the text book, which simple strategy results in *uniform-cost*-first search?
- ❑ append newly generated nodes at the beginning of the fringe (search queue)
- ❑ append newly generated nodes at the end of the fringe (search queue)
- ❑ insert newly generated nodes in the fringe (search queue) according to their path cost (lowest values first)
- ❑ insert newly generated nodes in the fringe (search queue) according to their heuristic function (lowest values first)

PART 2: SHORT QUESTIONS

In this part of the exam, you should answer the questions in one paragraph per aspect.

1. Identify the type, and give a PEAS description for a backup agent. The task of the agent is to examine all files in a computer system periodically, compare them against a backup repository, and update the repository with new, modified or deleted files.

[10 points]

   a) What is the least sophisticated type of agent (reflex-based, goal-based, etc.) that is reasonably capable of performing this taks? Briefly explain your answer!

   b) Provide a PEAS description in the table.

Aspect	*Description*
P	
E	
A	
S	

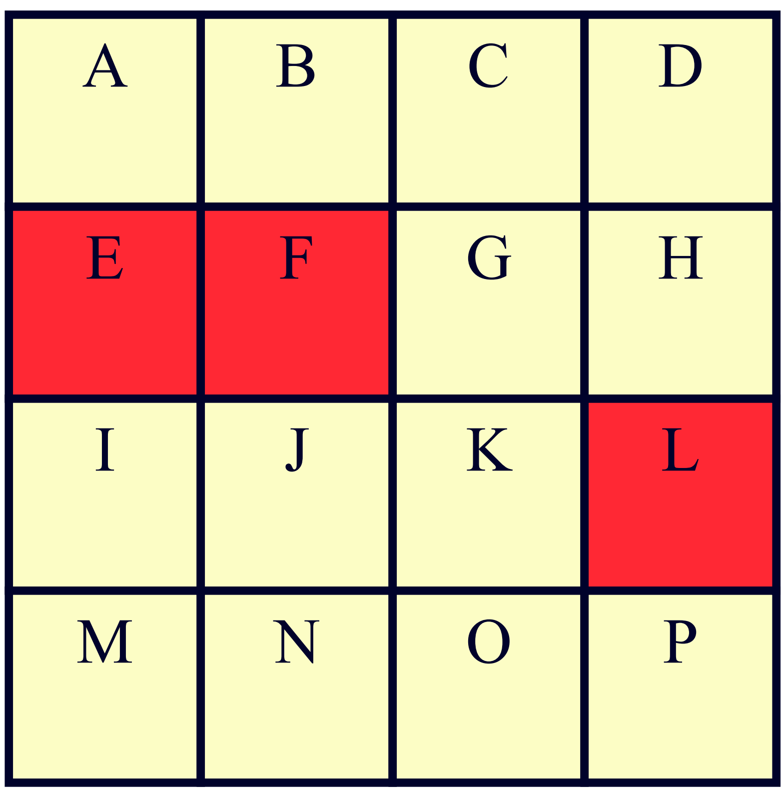
PART 3: GRID SEARCH

In this scenario, an agent is exploring a grid from the tile **A** designated as the starting point to the goal point, tile **O**. The shaded tiles are blocked. At each step, the agent can proceed from the current tile to one of the vertically or horizontally adjacent tiles; the agent can not move diagonally. If there are multiple successor tiles for a tile, the agent examines those tiles in alphabetical order.

In the following parts, you need to apply different search algorithms to solve this traversal problem. For each of the algorithms, you need to do the following tasks:

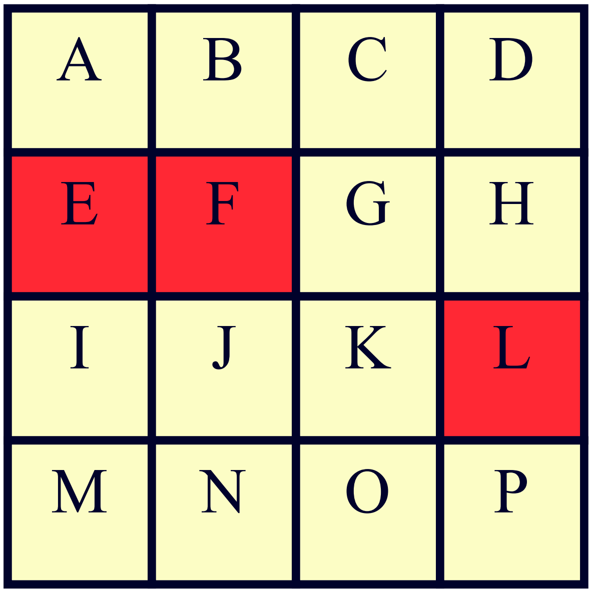
- Mark the sequence in which the tiles are visited. You can do this directly in the copy of the grid provided for each algorithm, e.g. by numbering the tiles in the order they are visited.
- Draw the corresponding search tree. It might be advisable to draw the complete search tree first on a separate sheet of paper, and then copy the relevant parts to the exam sheet.
- Fill in the table with the information about the search trace. Mark the newly added tiles in the frontier, e.g. by underlining them. (Note: The size of the table does not necessarily relate to the number of steps in the algorithm. Extend the table if necessary.)

You can assume that your algorithms avoid cycles by not re-visiting previously inspected tiles. For your convenience, the environment configuration is repeated on the next pages.



a) Traverse the environment from the starting point **A** to the goal **O** according to the *Depth-First Search* method.

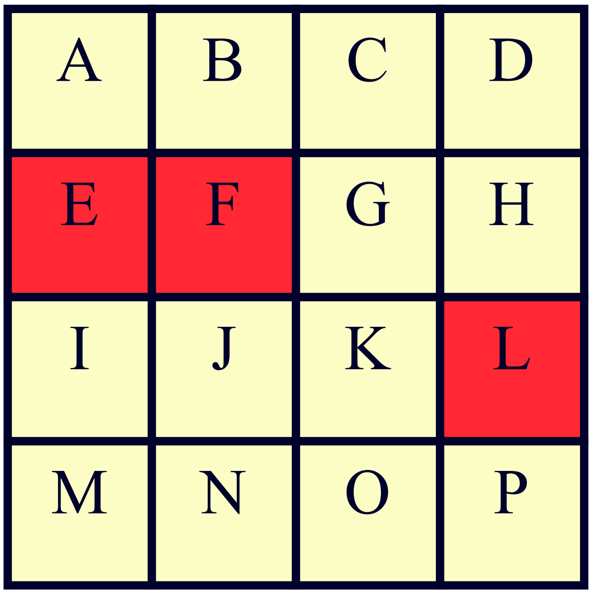
[10 points]



Step	*Current Tile*	*Frontier*
0		
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		
12		
13		
14		
15		
16		
17		
18		
19		
20		

Traverse the environment from the starting point A to the goal O according to the Breadth-First Search method.

[10 points]



Step	*Current Tile*	*Frontier*
0		
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		
12		
13		
14		
15		
16		
17		
18		
19		
20		

What are possible improvements for the above search methods for this type of environment (i.e. with a grid-based topology)?

[5 points]

Total Points: