CSCI-460 University of Southern California Homework 1
Due: 9/25/2003 11:59:59pm

Question 1 (50 points)
For each of the following agents, (a) give PAGE description, (b) identify the properties of the environment, and (c) identify the type of the problem.
1. Chess player.
2. Robot navigation avoiding obstacles
3. Spelling and grammar assistant.
4. Mathematician's theorem-proving assistant.
5. A robot assembling a machine from parts.

Question 2 (25points)
Give (a) the initial state, (b) the goal test, (c) the successor functions and (d) a possible cost function, for the following problems. Choose a formulation that is precise enough to attempt to solve the problem (but you DO NOT have to solve the problem).

1. Traveling salesman problem (TSP): Given a finite number of "cities" along with the cost of travel between each pair of them, find the cheapest way of visiting all the cities and returning to your starting point.

2. Mine sweeper: There is an N*N board considered as a minefield. You must locate all mines (bombs) in a minefield as quickly as possible by uncovering squares that do not contain a mine, and marking squares that do.

3. Phonebook: You have a standard phone book, and you want to know the phone number of a given person. You can turn pages, you can scan a page for a name, and you know the alphabet.

Question 3 (Programming problem - 75 points)
Look at the first Figure. We have a 10*10 board. The rows and columns are numbered from 0 to 9. The picture shows the way to calculate the next move of a new-knight: The pink square (4,4) is the current position of the new-knight, and the blue squares are the possible positions resulting from one of the possible moves from that pink square.

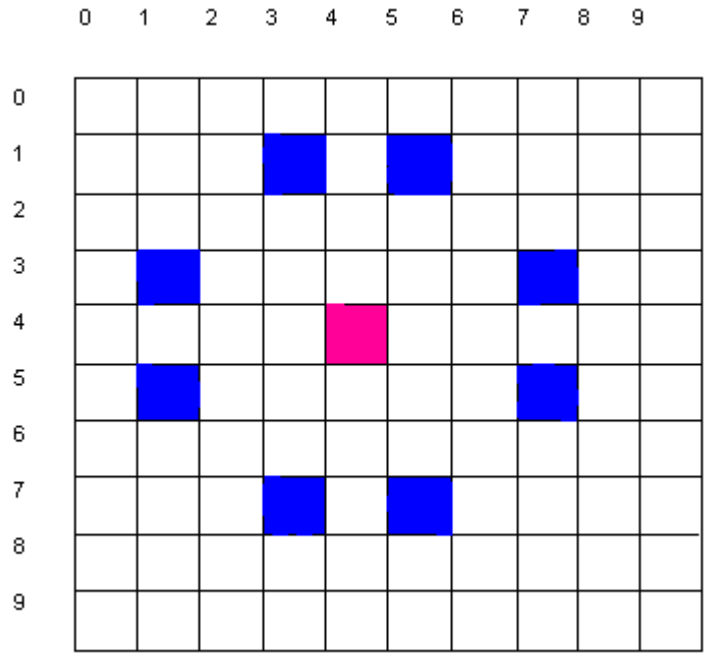

Figure1: Calculation of possible moves

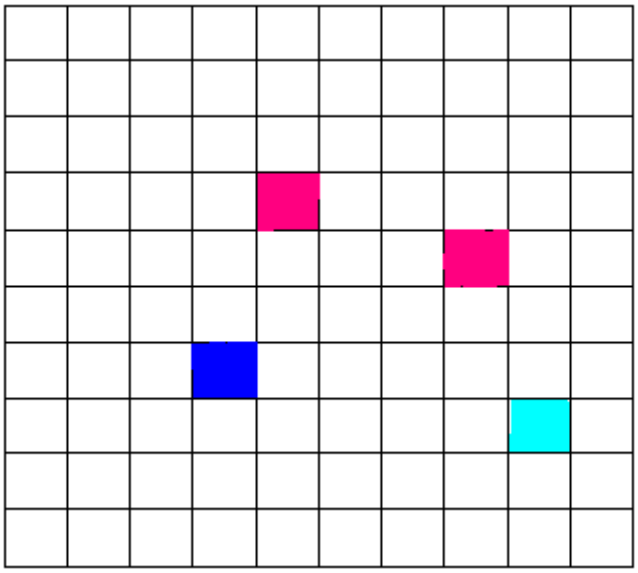

Figure2: Needed moves to reach destination-square

Now if you are given the current position of the new-knight and you want to take it to a destination square, it is possible to generate a sequence of moves to reach the destination. For example look at the second figure. If the current position is the light-blue square (7,8) and the destination is the blue square (6,3), the board

in Figure 2 shows one possible sequence of intermediate moves in red. There may be alternative paths to reach the destination square.

Write a program that will calculate the minimum number of moves required to go from a source-square to a destination square. This is a search problem. You have to implement it in both BFS and DFS.

Input: You will be given a text file called **problem.txt**. The file will be located in the current directory. The first line contains an integer **n** specifying the number of problems in the file. Each line will contain a problem. In each problem line there is an integer followed by 2 pairs of integers, separated by space. The first character specifies the type of search you have to use for this problem. If it is **1,** you have to run BFS for this problem. If it is **2**, you have to run DFS. The first pair of integers is the source position of the knight. Next pair specifies the destination. Please see the following sample **problem.txt** file.

Line number	***Contents:***
1.	2
2.	1 0 0 7 5
3.	2 9 0 1 6

Output: All outputs should be written in a file called **solution.txt** in the current directory. For each problem in **problem.txt**, there will be a **corresponding line** in **solution.txt** containing one integer: the minimum number of moves required for the corresponding problem. Put a number –1 if there is no solution. The sample **solution.txt** for the sample **problem.txt** is as follows:

Line number	***Contents:***
1.	3
2.	4

Caution: Please follow the following rules
1. Do not expect any command line argument(s).
2. Please make sure that your program compiles on aludra.usc.edu using the gcc/g++ compiler.
3. There can be any number of white spaces between fields in the input file.

Finally, answer the following questions:
1. Which of the search methods (BFS or DFS) works better and why?
2. What is the size of the state space?