CS 540 Fall 2005
Homework 1

Questions to Pradheep Elango (pradheep@cs.wisc.edu)

Due Date: **September 29, 2005**

There are 15 pages in this assignment. There are some written parts as well as a programming question. Please turn in all written parts together (we prefer typed answer scripts). Please write your **name** and **CS login ID** on every answer sheet.

Code submission. The solution to the programming problem should be coded in Java. Remember to turn in just your source code. The programming part should be turned in using the handin program as follows:
set path = ($path /s/handin/bin)
Then run the following command:
handin -c cs540-[your-section] -a [hw1] -d [your-directory]
Where [your-directory] is the directory where your files are currently located
and [your-section] is 1 if your instructor is Jerry Zhu,
2 if your instructor is Louis Oliphant.

Late policy. Homework must be handed in at the start of class (1:00pm) on the due date. Written portions should be handed in to me on the due date. Programmed portions should be handed in electronically. Late assignments will result in a 10% grade drop per day. No assignments may be handed in more than 3 days late (even when using late days). You have 3 free late days that can be used at any time throughout the semester.

Collaboration policy. You are to complete this assignment individually. However, you are encouraged to discuss the general algorithms and ideas in the class in order to help each other answer homework questions. You are also welcome to give each other examples that are not on the assignment in order to demonstrate how to solve problems. But we require you to

- not explicitly tell each other the answers
- not to copy answers
- not to allow your answers to be copied

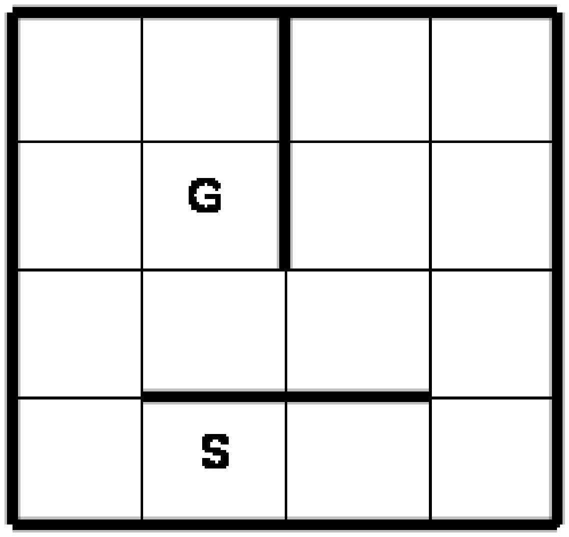
In those cases where you work with one or more other people on the general discussion of the assignment and surrounding topics, we suggest that you specifically record on the assignment the names of the people you were in discussion with. This will help resolve the situation where a mistake in general discussion led to a replicated weird error among multiple solutions.

Question 1

References (names of people I talked with regarding this problem or "none"):

The following figure shows a simple maze on a 4 X 4 checquered board, where the bold lines indicate walls. The task is to find a path from the start state S to the goal state G by exploration.

Figure for Question 1



The problem can be solved by formulating it as a search problem. Each square is a state, and the successor function is essentially a listing of the adjacent squares (that is, squares on the left, right, below or top of a given square). Since, a bold line indicates a wall, not all squares have the same number of adjacent squares. For this question, **assume** that the successor function lists the adjacent squares in this order, whenever possible: up, right, down, left. **To answer parts (a) - (e), use the sheets on pages 7 to 11**. For each of the parts (a) - (e), you should **write** down the order that states are visited (i.e. current node in general search algorithm), as indicated in the example on page 4. You should also ensure that you **avoid loops** by maintaining a CLOSED list as discussed in class.

(a) Simulate Depth-First Search on the figure by clearly writing down the order in which states are visited (expanded).

(b) Simulate Breadth-First Search similarly. Write down the order in which states are visited (expanded).

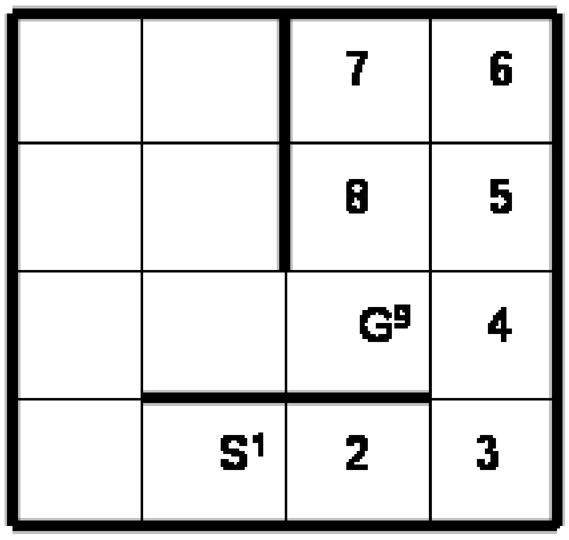
(c) Let's assume that the cost of going up is 10 units, the cost of turning left is 2 units, the cost of turning right is 4 units, and the cost of going down in 1 unit. Simulate Uniform Cost Search on the figure. **Write** down the order that states are visited (expanded) as well as the cost to reach any state (the g-values) you put in the priority queue.

(d) Simulate iterative deepening with a depth-limit on this figure. Proceed only till a depth limit of 3 (the root is at depth 0). Write down the order in which the states are visited (expanded). If a state is visited (expanded) more than once, then write all the order numbers for that state. For this part, assume that costs for turning in different directions are all unit costs, and that the successor functions lists the adjacent squares in the same order (up, right, down, left) as mentioned earlier.

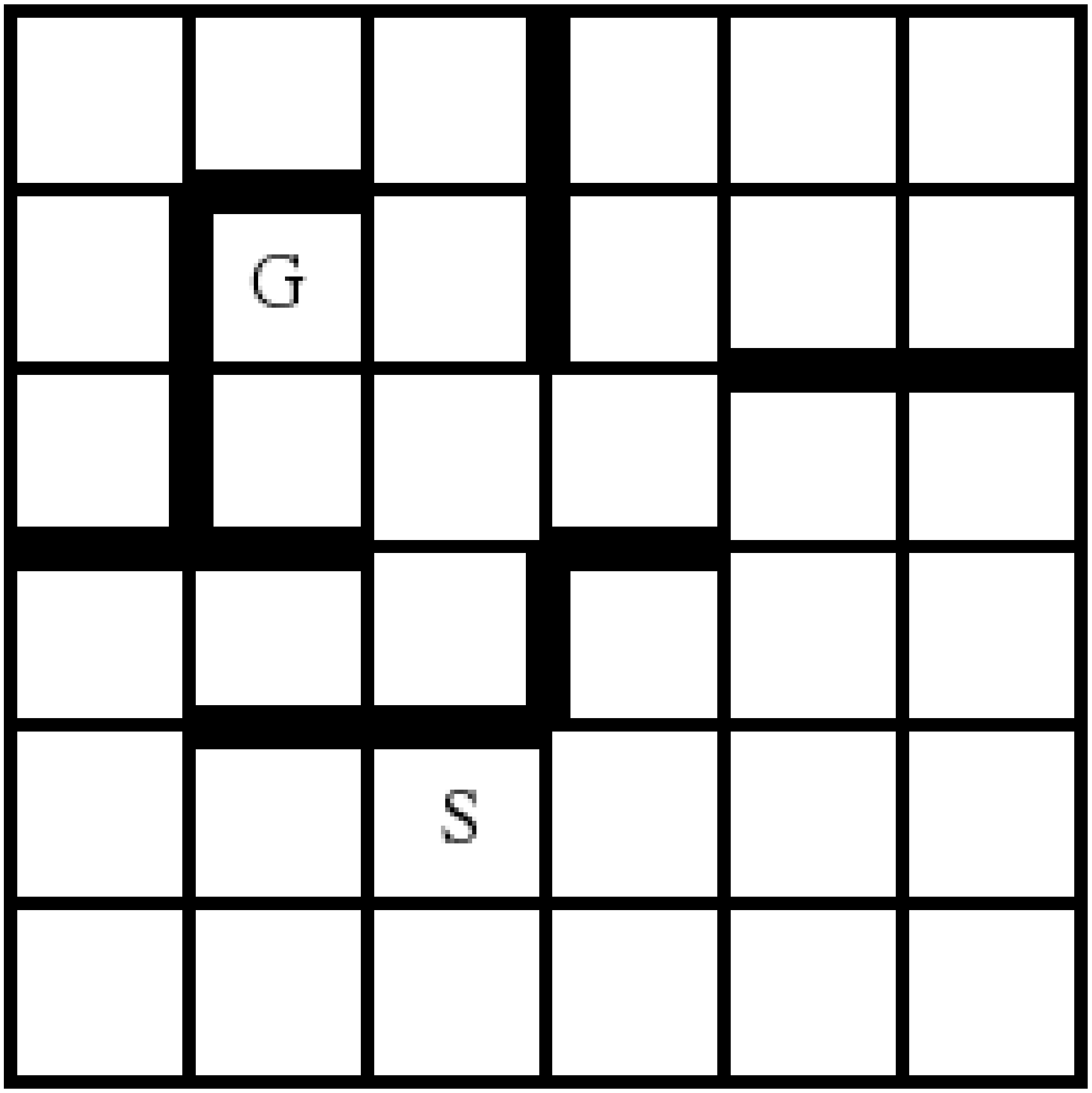
(e) Use Manhattan distance between the current state and the goal state, as a heuristic to perform A* search. For example, the Manhattan distance between the S and the G state in the figure on page 5 is 2, and similarly for the figure on page 6, it is 4. Assume costs for turning in the different directions as explained in part (c) above. Simulate A* search on this figure. Write the order in which the states are visited (expanded), as well as the g-values (the cost to reach the node), the h-values (the distance of the goal node from this node), and the f-values (the sum of g and h values for a given node) for each state that is visited.

(f) Give an **admissible** heuristic that **dominates** Manhattan Distance for 1(e).

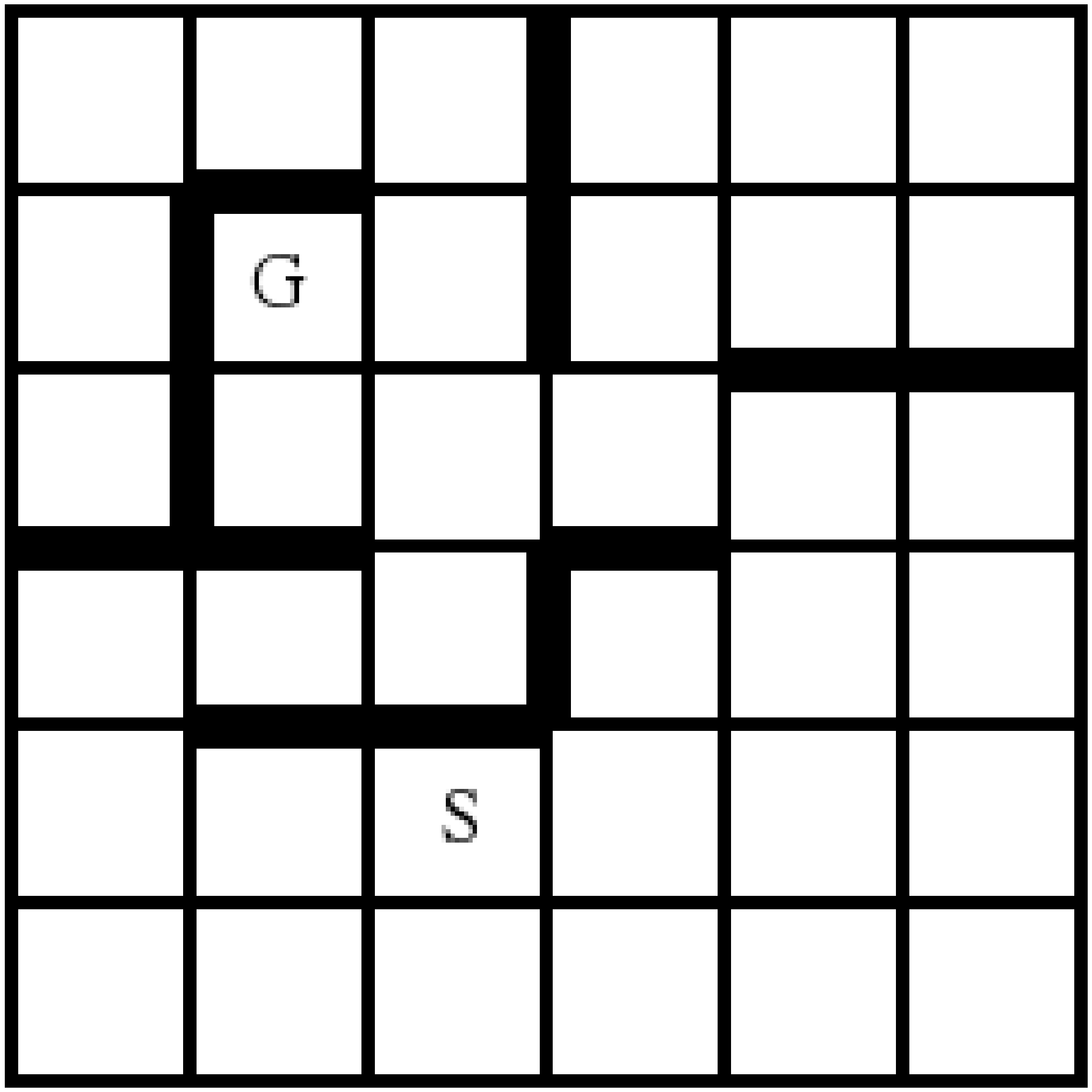
Example problem using DFS



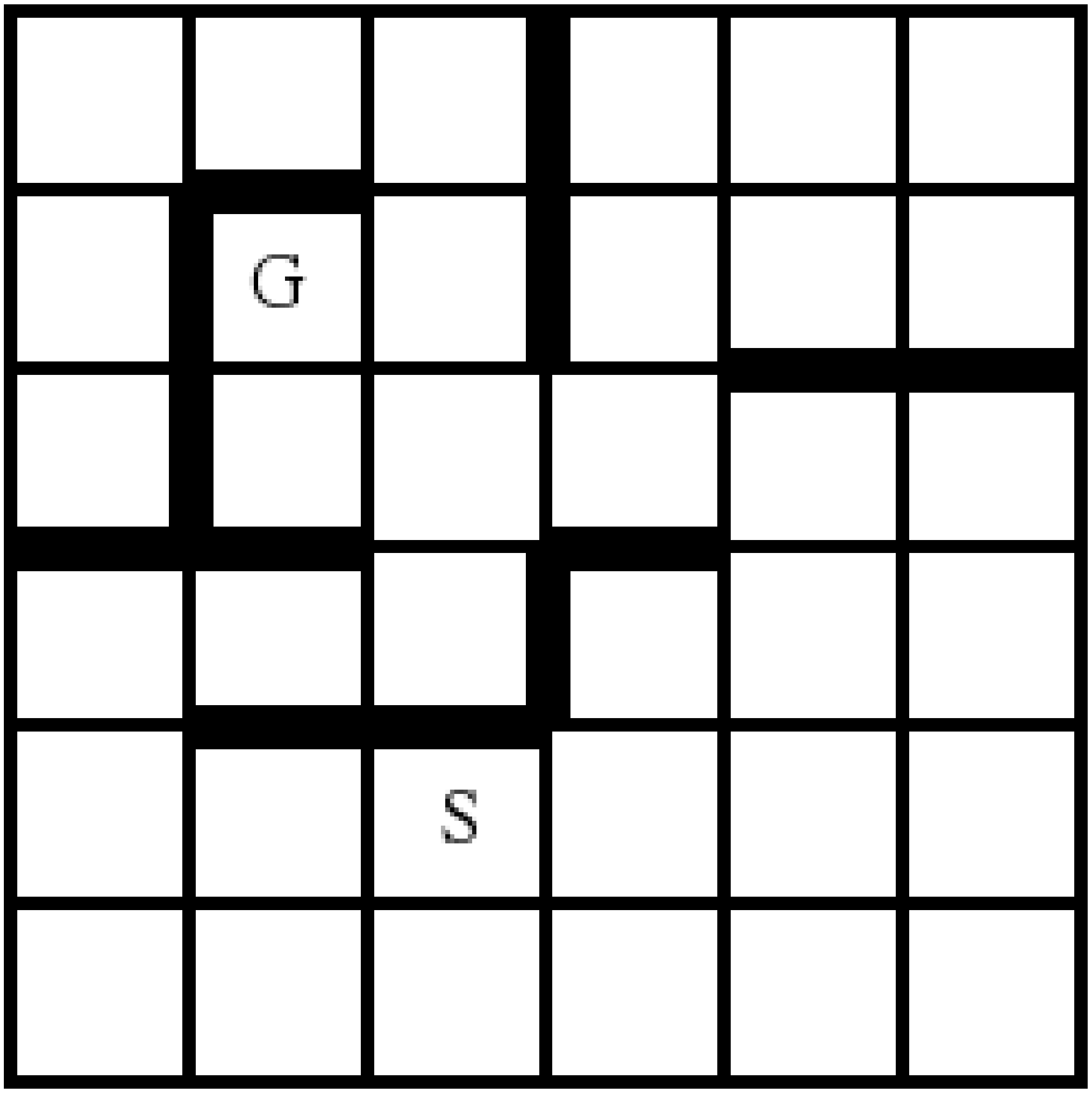
Use this for 1(a)



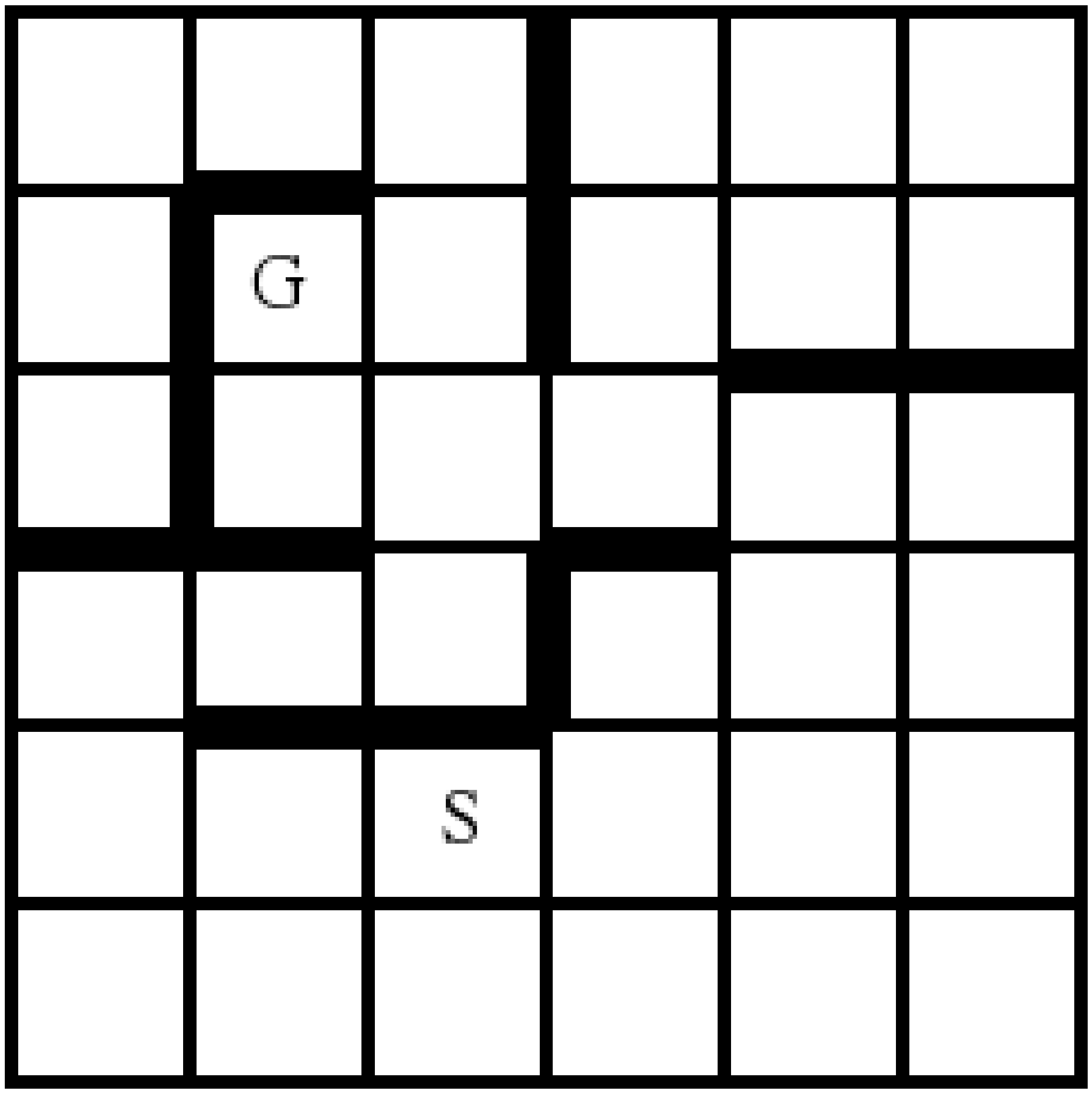
Use this for 1(b)



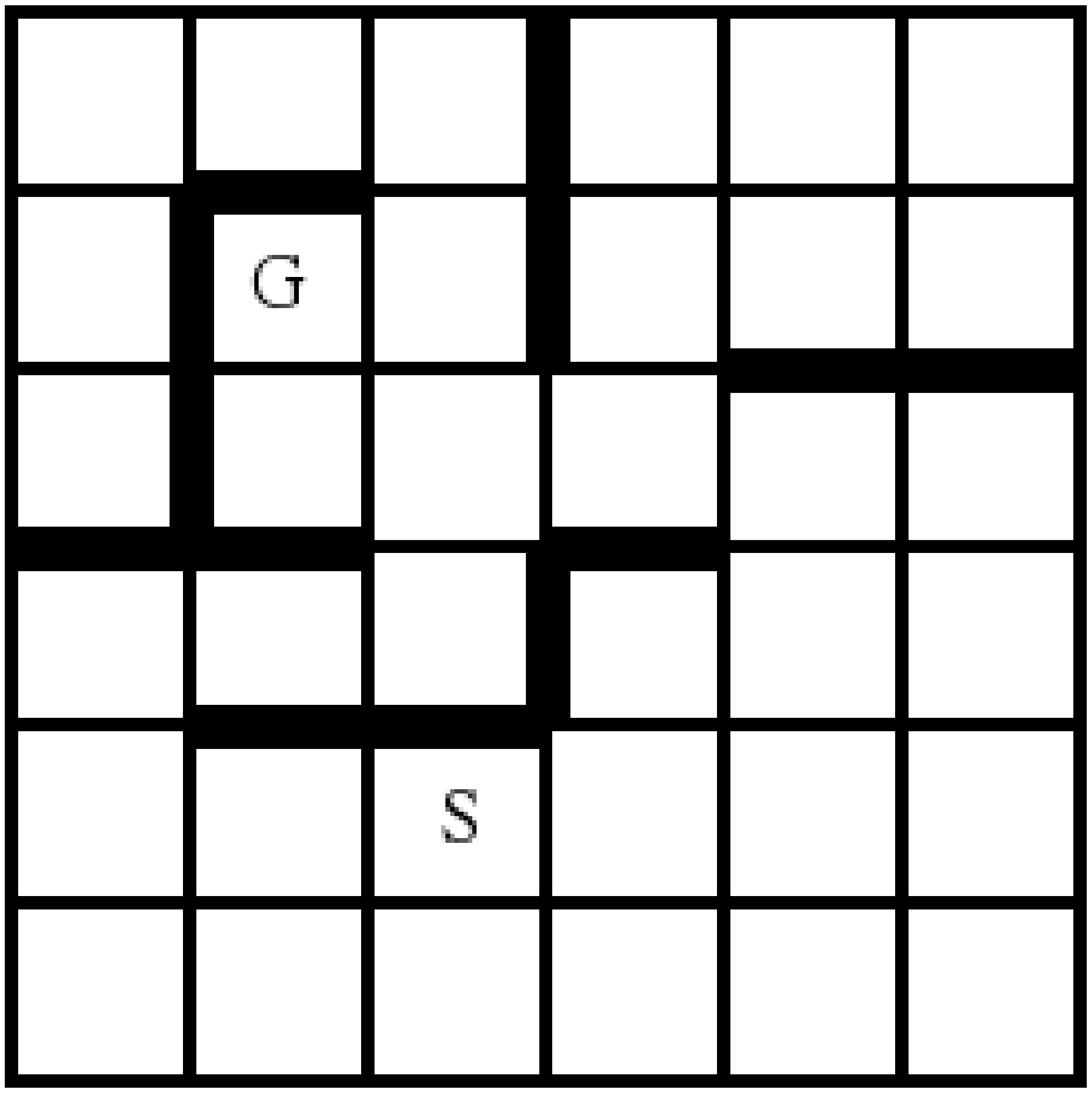
Use this for 1(c)



Use this for 1(d)



Use this for 1(e)



Question 2

References (names of people I talked with regarding this problem or ”none”):

In order to do Bidirectional Search, we need both a predecessor function and a successor function. Let us consider the Water-jugs problem, as discussed in class. Suppose one jug has a capacity of 7 gallons, and another jug has a capacity of 5 gallons, and the task is to exactly measure out 1 gallon. We can formulate this as a search problem, where each state is the amount of water in each jug. For this problem, let us assume that the states are restricted to integer amounts in each jug.
(a) Given the state (7,1) **enumerate** the different possible predecessor states.
(b) Given the same state (7,1) **enumerate** the different possible successor states.

Question 3

References (names of people I talked with regarding this problem or ”none”):

We can unify Breadth-First Search and Depth-First Search into a single search function using a priority queue. Outline a pseudo-code to unify both BFS and DFS. How would you define the priority function for each of the two methods? **Assume** that you are dealing with trees (so that there are no loops).

Question 4

References (names of people I talked with regarding this problem or ”none”):

Captchas are popularly used to tell a human from a robot. Suggest two general strategies for defeating captchas. Your answers should fit within half a page. For this problem, you may use the web to search for answers and describe various methods that are currently in use, or you could come up with a new method by yourself. You need to indicate how you came up with your answer but the credit for this question will not depend on it.

Question 5

References (names of people I talked with regarding this problem or ”none”):

Consider the following stream of characters:
“inwhichweseehowanagentcanfindasequenceofactionsthatachievesitsgoalswhen-nosingleactionwilldo”

On careful inspection, one observes that it is just the following sentence with all white spaces removed: “In which we see how an agent can find a sequence of actions that achieves its goals when no single action will do.”

This is the segmentation problem - breaking a stream of characters into a possible sentence. Formulate the segmentation problem as a search problem. Assume that there is no punctuation other than white space. **Define** the state space, initial state, the successor function, the goal test and the goal state clearly. **Assume** we have a function that indicates whether a string is a valid dictionary word or not, and also that we need not do any grammar checks.

This question has a programming task, and written parts which are based on the programming question. Please follow the instructions on the first page of this assignment to turn in the code. Turn in the written parts separately in class (preferably typed) along with the written parts from Questions 1 - 5

Question 6

References (names of people I talked with regarding this problem or "none"):

You are given a set of N numbers and a target. The task is to find an expression using a subset of the given set of N numbers and the 4 basic arithmetic operation (+, -, x, /) to reach the target.

E.g: Given the set of numbers 1,2,3,4,6 and target 53, one way to reach the target is:
$(2 + 3 + 4) * 6 - 1$

(a) Define a suitable search space representation for the problem, and briefly outline a strategy to solve the problem.

(b) Write a program to solve the problem, according to the program specification given below. You should implement different search strategies (BFS, DFS, IDS, UCS) for finding the best node. For UCS, use the following costs for the different operators, which roughly relates to the number of CPU cycles for a typical modern processor:
Addition: 4 units,
Subtraction: 4 units,
Multiplication: 6 units,
Division: 32 units.
The program should output one way of reaching the target in the usual arithmetic notation, using parantheses whenever necessary.

Program Specification : **Assume** that all input numbers and the target are positive integers. Each number may be used at most once in the expression. However, the set of input numbers may themselves have repeated numbers. Your program should not do division that will not result in integers.

The java program should run on any of the UNIX machines in CS. It will be invoked as follows:

java HW1 $\langle debug\rangle\langle Search\_Strategy\rangle\langle target\rangle\langle setofnumbers\rangle$

where

$\langle debug \rangle$ could be either 0 or 1. A value of 1 would mean that the program should output all states and the paths taken that it searches in the same order. (Whenever the program expands a node during search, it outputs the contents of the node and the label of the path it took to reach that node.)

$\langle Search\_Strategy \rangle$ could be either BFS, DFS, IDS, UCS

$\langle setofnumbers \rangle$ is just the input set of N numbers separated by a space.

$\langle target \rangle$ is the goal number that we want to reach

Example invocation: java HW1 1 BFS 6 1 2 3

Answer this part with respect to the problem described above and turn in the answers along with the answers to the other written questions.

(c) For a given set of N numbers, the 4 arithmetic operators, and a target, **answer** the following questions. Explain how you get each of your answers, indicating any assumptions that you made.

1. Calculate the branching factor at the root node (the initial state) in the search space.

2. Calculate the maximum depth possible for the search space.

3. Calculate the total number of leaf nodes in the search space. Calculate the total number of possible states (it is ok if your formula is expressed as a big sum).

4. Using any of the search strategies, will the entire state space ever be searched? If yes, then describe an input set. If no, provide reasonable arguments.

5. If every state has to be visited and if the program can visit 10 million states (10,000,000 states) per second, how long will the program run in the worst-case for a set of 100 numbers and a given target? What is the biggest N that the program can solve in 1 second, 1 minute, 1 hour, 1 day, 1 month, and 1 year?

6. Define a suitable heuristic for the problem. Is it admissible?

7. Suppose in our problem, instead of printing out some possible way of reaching the target, we always want to print the expression with the minimum number of operators. Using the same example as above, if we are given 1, 2, 3, 4, 6, the expression with the minimum number of operators will be "4 * 6" in order to reach 24. How will you use/modify your program to solve this problem?