



CENG 383 Algorithms - Programming Assignment 1


Board

You will be given an $n \times m$ board ($2 \leq n \leq 500$, $2 \leq m \leq 500$). Each cell of the board will be identified using its row and column indices: (i, j) . For simplicity, let's name the cell on i^{th} row and j^{th} column as $c_{i,j}$. As you might imagine, this board looks like a 2D array. The top leftmost cell of the board will be $c_{1,1}$ and bottom rightmost cell of the board will be $c_{n,m}$. A sample 3×4 board:

(1, 1)	(1, 2)	(1, 3)	(1, 4)
(2, 1)	(2, 2)	(2, 3)	(2, 4)
(3, 1)	(3, 2)	(3, 3)	(3, 4)

Knight

In one of $n * m$ cells, there will be a *knight*. There will be exactly 1 knight on the board (you do not need to make an error check on this). A sample 5×5 board with a knight in $c_{3,3}$:

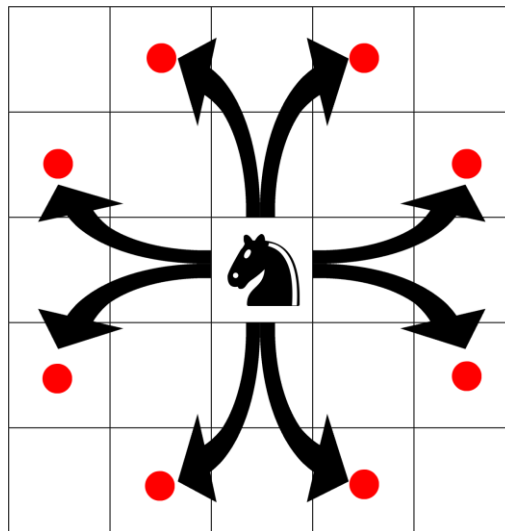
				



Knight Move

If you are familiar with chess game, then you might know what knight moves mean. Don't worry if you are not familiar. You do not have to know the rules of the chess game to finish this assignment.

From $c_{i,j}$ the knight can move to one of the following cells (in **one step**): $c_{i-2,j+1}$, $c_{i-1,j+2}$, $c_{i+1,j+2}$, $c_{i+2,j+1}$, $c_{i+2,j-1}$, $c_{i+1,j-2}$, $c_{i-1,j-2}$, $c_{i-2,j-1}$. The figure below illustrates the knight move better. Cells with red circles are possible knight moves. Knight **cannot** leave the board.

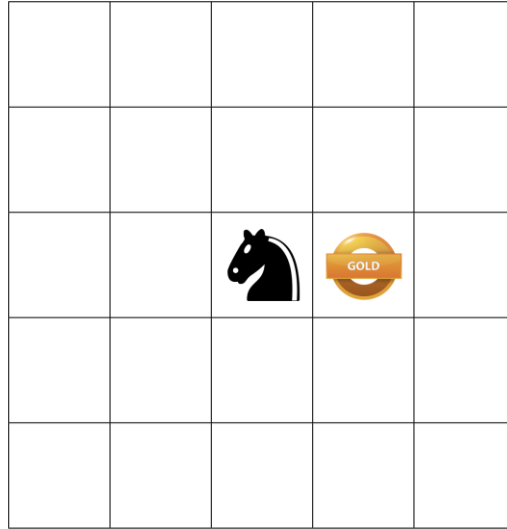


Gold

Besides the knight, there will be *gold* on the board. There will be exactly 1 gold on the board. You do not have to make an error check on this. A sample 5×5 board with the knight in $c_{3,3}$ and gold in cell $c_{3,4}$:



ÇANKAYA UNIVERSITY



In this assignment, you should find the length of the shortest path from the initial position of the knight to the cell where gold is located. In other words, you should find the **minimum number of steps for the knight to reach the gold**. For the example given above, a possible shortest path is $C_{3,3} \Rightarrow C_{2,5} \Rightarrow C_{1,3} \Rightarrow C_{3,4}$. The length of the shortest path is 3.

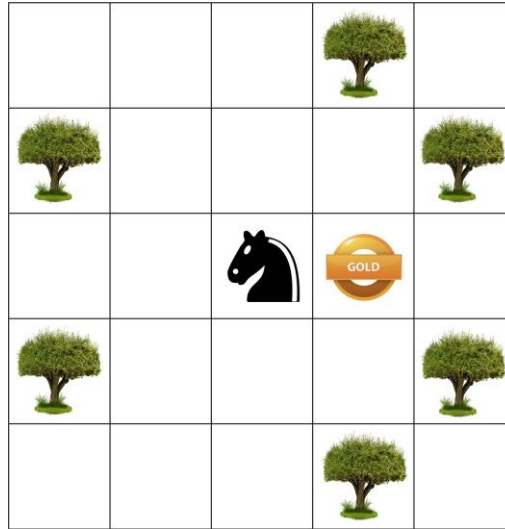
Obstacles: Trees

To make the problem slightly harder, let's insert a few trees into the board. They will act as an obstacle.

A cell with a tree **cannot** be used by the knight. Therefore, the shortest path should not pass through any cell which contains a tree. The sample board given above with 6 trees:



ÇANKAYA UNIVERSITY



Now, the knight cannot reach the gold in 3 steps. A possible shortest path from $c_{3,3}$ to $c_{3,4}$: $c_{3,3} \Rightarrow c_{1,2} \Rightarrow c_{2,4} \Rightarrow c_{3,2} \Rightarrow c_{5,3} \Rightarrow c_{3,4}$. The length of the shortest path is 5. There isn't any other path which is shorter than 5 steps.

Input / Output Specifications

The board will be given in an input file.

- The first line of the file will contain 2 integers: n and m .
- Each of the next n lines will contain m characters. j^{th} character of i^{th} line will contain an information about $c_{i,j}$.
- If j^{th} character of i^{th} line is **.**, $c_{i,j}$ is an empty cell.
- If j^{th} character of i^{th} line is **K**, $c_{i,j}$ contains the knight.
- If j^{th} character of i^{th} line is **G**, $c_{i,j}$ contains the gold.
- If j^{th} character of i^{th} line is **T**, $c_{i,j}$ contains a tree.
- The input file contains exactly one **K** and exactly one **G** characters.

The file version of the sample board given in previous section:



ÇANKAYA UNIVERSITY

```
5 5
...T.
T...T
..KG.
T...T
...T.
```

The program should read the given input text file and return the path of knight from the source to the target. For example, for the given input file, the result should be:

```
5 steps
c3,3 to c3,4: c3,3 -> c1,2 -> c2,4 -> c3,2 -> c5,3 -> c3,4
```

If there is no possible path from knight's initial position to the gold, the program should print:

```
No path to the target.
```



ÇANKAYA UNIVERSITY

Remarks

- **You MUST use the Graph API provided in the lecture (slides).** You are not allowed to use any other API for graph implementation in this assignment.
- The main function in assignment should take 1 command line argument, the name of input file.
- You can generate random boards and test your implementation.
- You can develop your solution in terms of the following classes:
 - **Graph.java**
 - This class should implement an empty undirected graph.
 - ✓ Constructor
 - loadGraph(String inputFile): Loads a board graph from an input.txt file.
 - ✓ Input: a String representing the text file
 - ✓ Output: void
 - **PathFinder.java**
 - This second class is responsible for finding the path and showing the result.

NOTE: Your implementation will be tested on different input files. You are recommended to work with smaller graphs first, and then try on larger graphs.

WHAT TO HAND IN

A zip file containing:

- The Java source files of your programs.
- The Java codes should be **WELL FORMATTED** and **WELL DOCUMENTED** as comments, as part of your grade will be based on the level of your comments.



ÇANKAYA UNIVERSITY

IMPORTANT

IMPORTANT NOTES: Do not start your homework before reading these notes!!!

1. This assignment is due by 23:55 on Saturday, April 6th.
2. You should upload your homework to **WebOnline** before the deadline. No hardcopy submission is needed. You should upload files and any additional files if you wrote additional classes in your solution as a **single zip file**.
3. The standard rules about late homework submissions apply. Please see the course syllabus for further discussion of the late homework policy as well as academic integrity.
4. For this assignment, you are allowed to use the codes given in our textbook and/or lecture slides. However, if necessary, you may define additional data members and member functions.
5. The submissions that do not obey these rules will not be graded.
6. To increase the efficiency of the grading process as well as the readability of your code, you must follow the following instructions about the format and general layout of your program.
7. Do not forget to write down your id, name, section, assignment number or any other information relevant to your program in the beginning of your Java files. Example:

```
//-----  
// Title: Pathfinder tester class  
// Author: Bora Celikkale  
// Description: This class tests the ...  
//-----
```

8. Since your codes will be checked without your observation, you should report everything about your implementation. Add detailed comments to



ÇANKAYA UNIVERSITY

your classes, functions, declarations etc. Make sure that you explain each function in the beginning of your function structure. Example:

```
//-----  
// Summary: Assigns a value to the variable  
//-----  
void setVariable(char varName, int varValue) {  
    // body of the function  
}
```

- Indentation, indentation, indentation...
- You are welcome to ask your questions on the **Office Hours** (Mondays 13.30-15:00 L117)

You do not need to prepare a PDF report, just document your code well. You may be asked to explain your code as a face-to-face demo session. The schedule will be announced later in that case.