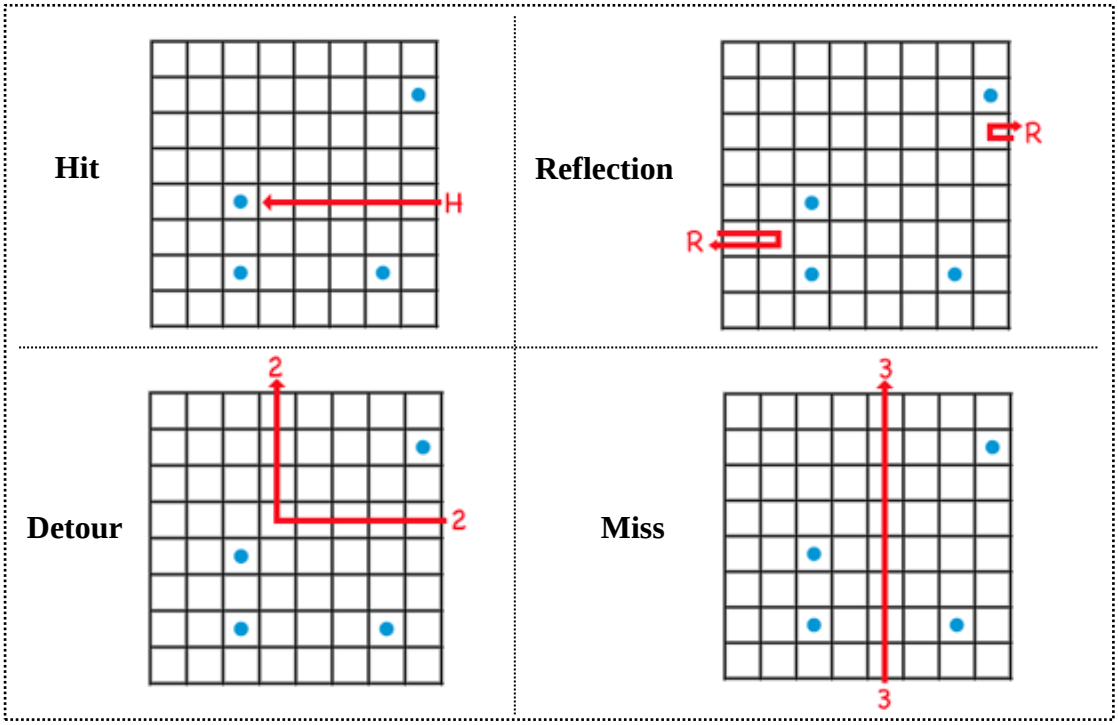
Lab 2: Black Box

Due: 18:00, 01 Feb 2013 (Fri) Full marks: 100

Introduction

In this lab, we are going to look at the interesting pencil and paper game called "Black Box"[1]. The game consists of an 8 × 8 grids with 4 atoms hiding on four distinct cells of the grid. The player fires rays into the grid by specifying the positions around the edges of the grid. The ray fires into the grid and interacts with the hidden atoms. The result can be one of the four outcomes (depicted below): a "Hit", a "Reflection", a "Detour" or a "Miss". The player has to guess the correct locations of the four atoms based on the outcomes of the different rays fired.



Program Specification

Create a new project in NetBeans named (`BlackBox`) and write a Java program (`BlackBox.java`) to implement the core functionality of "Black Box", which is the firing of rays into the grid, tracing the path of rays fired and determine the corresponding outcomes. For simplicity, we use the character '`.`' to denote an unoccupied cell, and '`@`' to denote an atom. The rows and columns are numbered from top to bottom and from left to right respectively, starting from 1 to 8. Figure 1 shows an example grid with 4 atoms.

[1] URL: http://www.papg.com/show?2XL9

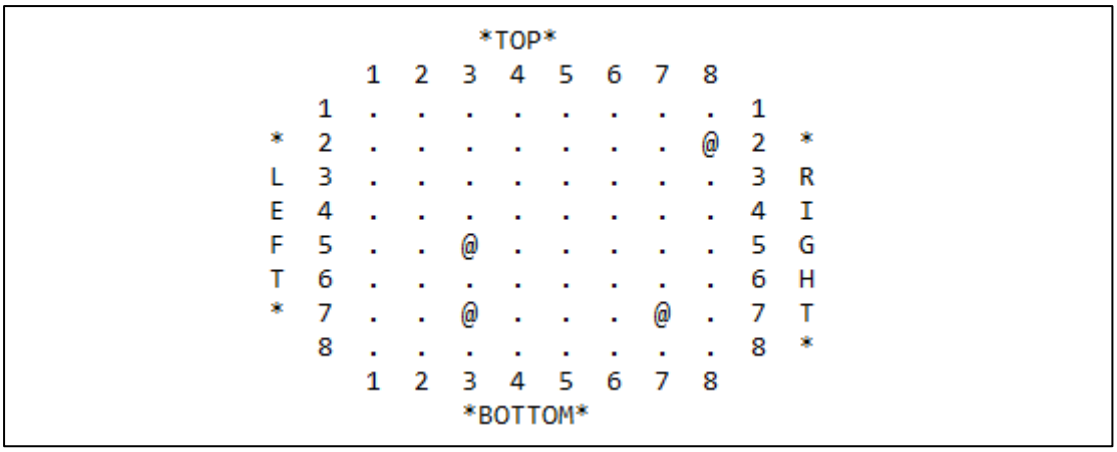


Figure 1 : An Example Grid with Four Atoms

Your Java program must execute the following steps in order.

1. The program must start by getting the locations of the four atoms from the user. Locations are represented by a pair of integers with values from 1 to 8, denoting the row and column coordinates. If the user's input is invalid, you must display a warning message and ask the user to try again until a valid input is made.
2. Next, the program must display the grid and the four atoms using the format as shown in Figure 1, and prompt the user to input an integer denoting the ray starting position. The integer value 1 means TOP, 2 means BOTTOM, 3 means LEFT, and 4 means RIGHT. The user can also terminate the program at this point by entering the value -1. If the user's input is invalid, you must display a warning message and ask the user to try again until a valid input is made.
3. If the user inputs a valid position (i.e. integer values 1 to 4), the program then asks the user to input another integer (with values from 1 to 8) denoting the particular row (if LEFT or RIGHT) or column (if TOP or BOTTOM) where the ray will fire. If the user's input is invalid, you must display a warning message and ask the user to try again until a valid input is made.
4. After determining the ray starting position, the ray fires into the grid and your program must trace the path of the ray. The path of the ray is just a straight line passing through the grid (from one edge of the grid to another). However, depending on the locations of the four atoms, the path may be altered according to the following four rules.
    a. If a ray hits an atom directly, the ray is absorbed (by the atom).
    b. There are two scenarios where a ray is being reflected. First, a ray is reflected if a ray passes next to an atom on the edge of the grid. Second, a ray is also reflected if it tries passing between a pair of atoms on the same row (if the ray is vertical) or same column (if the

ray is horizontal) one cell apart from each other. When a ray is being reflected, it returns back to the starting position.

c. A ray is being deflected if it passes next to an atom, in which case it will be deflected by 90 degrees away from that particular atom, and the path continues with the deflected direction.

5. Your program must display the row and column coordinates of the ray as it passing through the grid, and displays a message to indicate the path is being altered by any of the rules in Step 4.
6. The path of a ray terminated when it is absorbed by an atom or reached the edge of the grid. After a ray terminated, your program must determine the outcome based on the following rules.
    a. If a ray is absorbed by an atom, the outcome is 'Hit'.
    b. If a ray starting position is the same as the terminating position, the outcome is 'Reflection'.
    c. If a ray terminates on opposite edge as the starting position, and has the same column (if TOP or BOTTOM) or row (if LEFT or RIGHT). The outcome is 'Miss'.
    d. If a ray outcome does not fall under 'Hit', 'Reflection', and 'Miss', the starting position should be on a different row or a different column as the terminating position. The outcome is 'Detour'.
7. After finished tracing the path of a ray and determining the outcome, the program should go back to Step 2 to check whether the user still want to fire another ray.

Sample Program Run

Refer to CU e-Learning System for the provided sample program runs. Please take note that your program output must be exactly the same as the sample program runs (i.e., same text, same symbols, same letter case, same number of spaces, etc.). Otherwise, it will be considered as wrong, even if you have computed the correct result.

Submission

Submit the source file `BlackBox.java` to CU e-Learning System (i.e. the entry "lab02 – Black Box" under "Lab Works").

== END ==

