Robot Tennis

In Robot World a grand championship on tennis is about to take place. All the best robotic tennis players have already started practice for this great event and they are ready to give their best for the Robotic Cup! Unfortunately, due to the bad weather both the tennis field where the tournament was about to take place was ruined and the referee of the game got ill. Can you develop a program that could simulate the tennis field and the coach in order to host the tournament?



Task

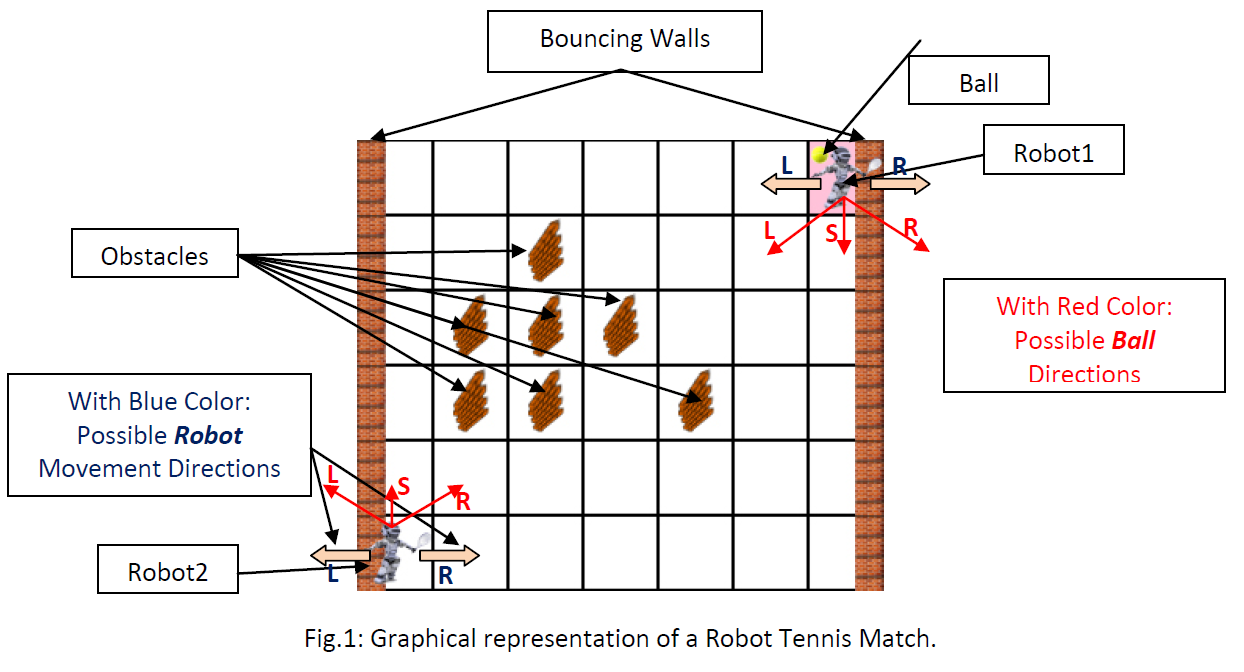
Your task is to develop a program that can efficiently simulate a tennis match between two robotic tennis players. The simulation should comply with the following rules of the game:

*(Note: please also refer to Fig.1 for a better understanding of the notations and the rules)*

1. The tennis court will be a 2 dimensional space comprised of ***n*** rows and ***m*** columns.
2. At every match, only two robots (***Robot1*** and ***Robot2*** hereafter) can participate.
3. Robot1 can be positioned at ***any column*** (even the rightmost or the leftmost column) but always at the ***first row*** of the tennis court whereas Robot2 can be positioned at ***any column*** but always at the ***last row*** of the tennis court.
4. On the rightmost and the leftmost side of the tennis court there exist ***bouncing*** walls which may change the direction of the ball when it collides with them.
5. Moreover, obstacles may exist within the tennis field. Obstacles *cannot* be positioned at the first and the last row of the tennis field, nor at the leftmost and rightmost columns of the tennis field (i.e. obstacles cannot co-exist neither with robots nor with bouncing walls). Obstacles always alter the direction of the ball.
6. Both robots can ***only*** move along the horizontal axis (i.e. only to the right or to the left of their current position) and never on the vertical axis.
7. No robot is allowed to stay still. At every step, at first the ball moves and then each robot should moves (on the horizontal axis) towards the ball, meaning that if the ball is positioned at a column *i* and the robot is at column *j*, and *i<j< em="">, then the robot should move to the left, whereas it should move to the right if i>j. In case that both the ball and a robot happen to be in the same column (i=j), the robot should try to move to the****right by default****. If this is not possible (because it is already at the rightmost side of the tennis field and moving to the right means colliding onto the bouncing wall) then it moves to the left side instead.</j<>*
8. *When a robot has the ball, it should always throw it towards its opponent at one of the three possible directions:*
   1. *Diagonally to the left (denoted by****L****hereafter),*
   2. *Diagonally to the right (denoted by****R****hereafter) or*
   3. *Straight ahead (denoted by****S****hereafter).*

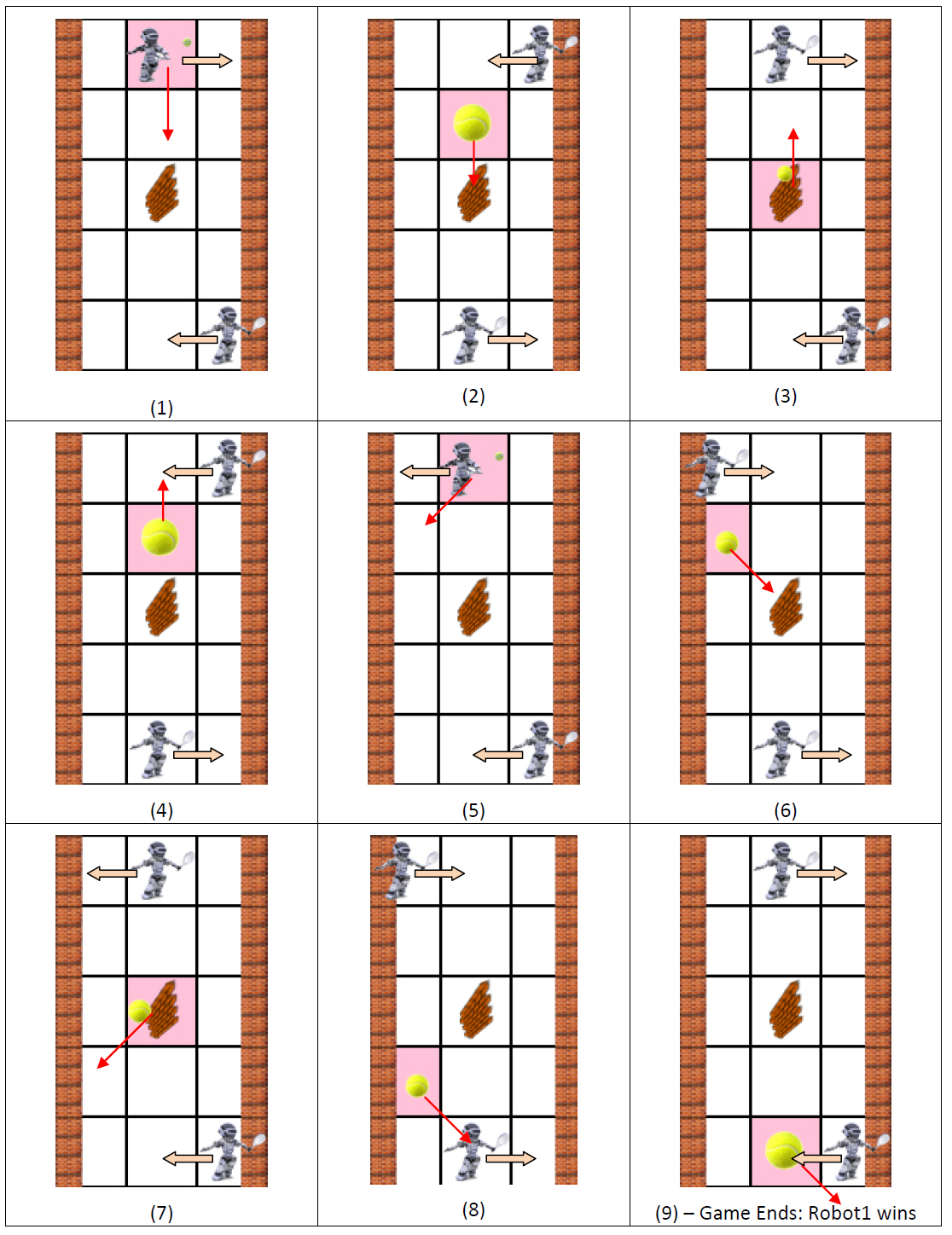
*The direction along which the robot should throw the ball will be decided according to an input sequence that will be provided as input at the beginning of the program. In particular, a sequence of directions (e.g. LRSSR) will be provided as input at the beginning of the program, and every time a Robot has to throw the ball towards its opponent, the next available direction will be chosen (e.g. for the example sequence LRSSR, the first Robot that catches the ball should throw it on the L direction towards it opponent, afterwards the R direction should be selected in order to throw the ball back to the opponent robot, then the S direction and so forth).*

1. *The ball should continue to move along the direction defined by the robot until one of the following happens:*
   1. ***The ball collides with a bouncing wall****: In this case the bouncing wall****may****change the ball direction to its opposite. More specifically, if the ball was moving on the R direction, it changes it to L and vice versa. If the ball was moving on the S direction, the bouncing wall does not change the ball’s direction (i.e. it will continue moving straight as if the bouncing wall was not there)*
   2. ***The ball collides with an obstacle****: In this case, the obstacle****always****changes the movement direction of the ball. In particular, if the ball was moving on the R direction when it collided with the obstacle, it should move on the L direction afterwards and vice versa. In case the ball was moving on the S direction, then the ball should continue moving straight but towards the opposite direction (e.g. if the ball was moving straight heading from Robot1 to Robot2, it should then continue moving straight but heading from Robot2 to Robot1).*
   3. ***The ball reaches a square where a Robot is positioned****: In this case, the Robot should throw the ball back to the opponent Robot using one of the available directions (L, R or S) and the ball should start moving along this direction afterwards.*
   4. ***The ball reaches the end of the tennis court and no Robot is positioned there****: In this case, the game ends and the Robot positioned at the other side of the tennis field is nominated winner of the match.*
2. *The game ends when:*
   1. ***The ball reaches the end of the tennis court and no Robot is positioned there****: In this case, the Robot positioned at the other side of the tennis field is nominated winner of the match.*
   2. ***The ball reaches the end of the tennis court and a Robot is positioned there****but****there are no remaining directions****at the provided input sequence so as for this Robot to throw the ball back to its opponent: In this case the game ends without a winner.*

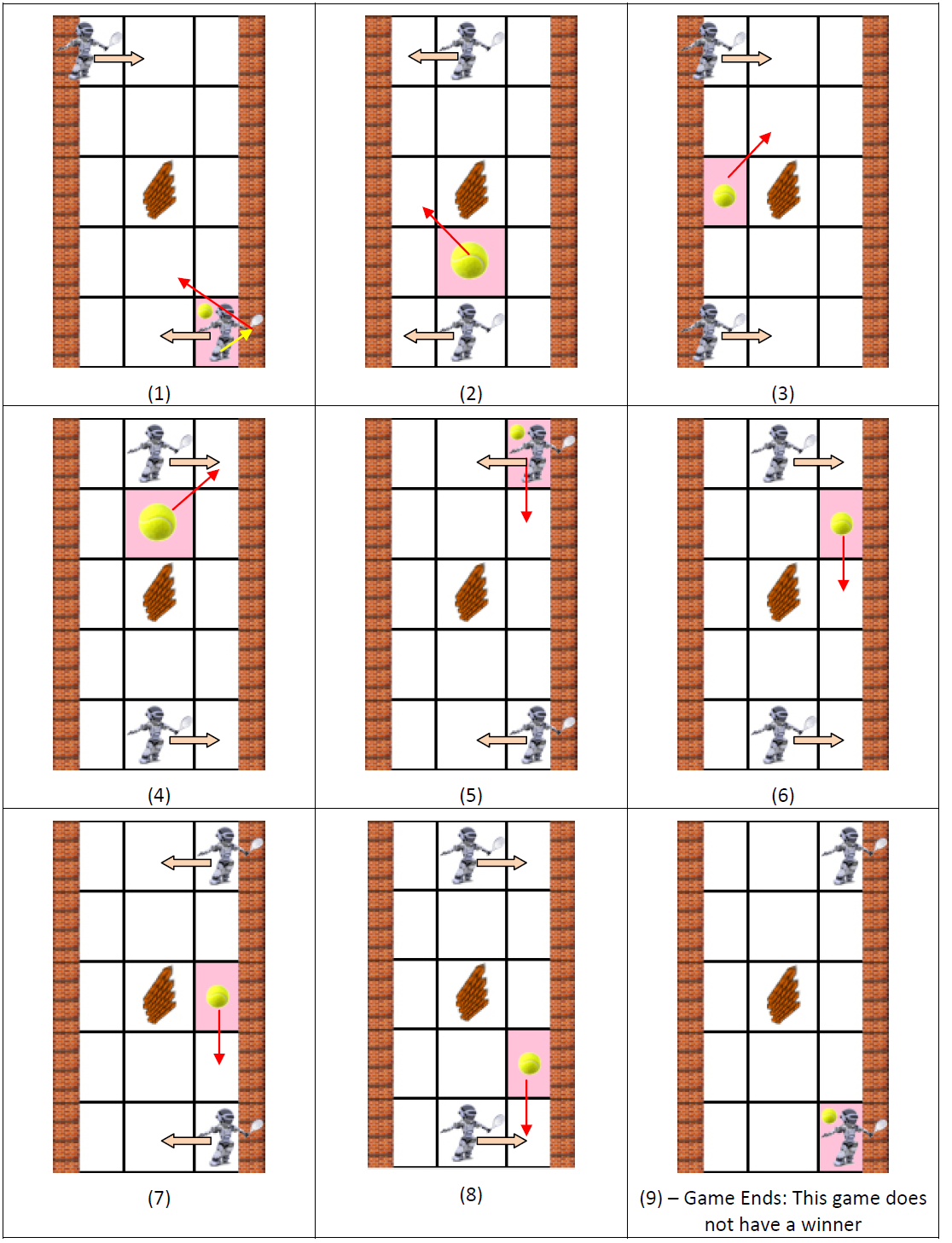


*In order to better explain the rules and the flow of the game, please also consider the following visualizations of a virtual tennis match between the two Robots.*

***Visualization I****: For the Input Sequence {****SL****} and Robot1 having the Ball at the beginning of the match.  
(Red arrows signify the next position of the ball, whereas pink arrows show where the robot should move at in order to follow the ball’s course according to Rule 7)*



***Visualization II****: For the Input Sequence {****RS****} and Robot2 having the Ball at the beginning of the match.*



*Please note that at Step 1 of Visualization II, although Robot2 sends the ball to the R, since it instantly bounces on the wall its direction changes and thus moves to the L. Moreover, in every step the robots move towards the ball, or they move by default to the right if they are already positioned on the same column with the ball (e.g. Step 1 of Visualization I). In case the robots cannot follow the default right movement rule, then they moved to the left (e.g. Step 7 of Visualization II). Finally, at Step 9 of Visualization II, the game ends without a winner since there are no available moves for Robot2 to select.*

*Input*

*The program receives its input from the standard input stream. The parameters that should be provided as input are the following:*

1. *Two positive integer numbers****n1****and****n2****(where 1 ≤****n1****≤ 15 and 1 ≤****n2****≤ 15) representing the dimensions of the tennis field (i.e. the number of rows and columns respectively). These two numbers will be provided in one line and should be separated by a comma (,) character.*
2. *A positive integer number****r1\_pos****(0 ≤****r1\_pos****<****n2****) representing the initial position (in terms of column number) of Robot1 (the row number position does not need to be provided since Robot1 is always positioned at the****1st row****of the tennis field).*
3. *A positive integer number****r2\_pos****(0 ≤****r2\_pos****<****n2****) representing the initial position in terms of column number) of Robot2 (the row number position does not need to be provided since Robot2 is always positioned at the****last row****of the tennis field).*
4. *A positive integer number****ball\_pos****(1 ≤****ball\_pos****≤ 2) representing which robot initially has the ball. More specifically, if****ball\_pos****equals to 1 then Robot1 will initially have the ball, whereas Robot2 will start the game if****ball\_pos****equals to 2.*
5. *A positive integer number****num\_of\_obstacles****(0 ≤****num\_of\_obstacles****≤ {(****n1****\*****n2****)-2(****n1****+****n2****)+4}) representing the number of obstacles that will be placed at the tennis field.*
6. ***num\_of\_obstacles****lines should follow representing the position of each obstacle in the tennis field. These positions should be given in the form of pairs of positive integer numbers****ob\_row****,****ob\_col****where 1 ≤****ob\_row****<****n1******-1****and 1 ≤****ob\_col****<****n2******-1****. These pairs should be provided as one pair per line and the two integer values at each line should be separated by a comma (,).*
7. *A sequence of characters belonging to the set {L,S,R} which represents the available moves from which the robots will select at which direction to throw the ball during gameplay. This sequence should be provided in one single line and the characters should not be delimited to each other.*

*Output*

*The program should be able to simulate a match between the two competing robots and print to the standard output stream the result of the game, as well as the state of the game (i.e. robot positions, ball position and sequence of movements used) when the game ends. More specifically, the program should output:*

1. *At the first line:*
   * *The comment: “****Winner: Robot1****” if Robot1 wins the match*
   * *The comment: “****Winner: Robot2****” if Robot2 wins the match*
   * *The comment: “****This game does not have a Winner.****”  if no robot wins the match*
2. *At the second line:*
   * *The comment: “****Robot1 At [x,y]****”, where x and y correspond the row and the column at which Robot1 is positioned when the game ends*
3. *At the third line:*
   * *The comment: “****Robot2 At [x,y]****”, where x and y correspond the row and the column at which Robot2 is positioned when the game ends*
4. *At the fourth line:*
   * *The comment: “****Ball At [x,y]****”, where x and y correspond the row and the column at which the Ball is positioned when the game ends*
5. *At the fifth line:*
   * *The comment: “****Sequence: XXXX…****”, where each X stands for a letter belonging to the set {L, R, S}. In this line all moves that were used during the game should be printed at the standard output stream (Beware: only the directions of the ball until the game ended should be printed at this line).*

***Note: There is a newline character at the end of the last line of the output.***

*Sample Input 1*

*4,4*

*0*

*3*

*1*

*1*

*1,1*

*LLRSLRSSR*

*Sample Output 1*

*Winner: Robot1*

*Robot1 At [0,1]*

*Robot2 At [3,2]*

*Ball At [3,1]*

*Sequence: L*

*Sample Input 2*

*8,4*

*3*

*2*

*2*

*5*

*1,1*

*2,1*

*3,2*

*5,1*

*4,2*

*SRLLRSLLRSSLL*

*Sample Output 2*

*Winner: Robot2*

*Robot1 At [0,2]*

*Robot2 At [7,1]*

*Ball At [0,1]*

*Sequence: SR*