Objective: To use arrays and functions in problem solving and learn interactive programming.

1 Tic Tac Toe

Tic Tac Toe is a popular game for kids. Of course, big kids like it too. The board of the game has a grid of 3×3 cells. Two players place “X” and “O” alternately into the empty cells. The goal is to get three connected “X” or “O” in the horizontal, vertical or diagonal direction.

Let’s implement the game using Java. The game can be represented as a sequence of states of a 2D 3×3 int array. We name the array as `board`. `board` is initialized with all 0s, where 0 indicates that a cell is empty. We denote the two players of the games as 1 and 2 and assume that player 1 places “X” and player 2 places “O” in the game. We thus can represent the state of the game after each player’s move by setting a 1 or 2 to the corresponding element in the array `board`. After each move, your program should check the array to see whether

- player 1 wins, if there are 3 1s in a row in any 4 directions.
- player 2 wins, if there are 3 2s in a row in any 4 directions.
- it is a tie, if there are no empty cells and nobody wins.

We still use the StdDraw class for drawing. StdDraw provides some limited functions for interacting with a program using a mouse. These methods are

```java
    double mouseX() return x coordinate of mouse
    double mouseY() return y coordinate of mouse
    boolean mousePressed() is the mouse currently being pressed or not
```

In your main method, there is a while loop in your main function to check the user input until the game ends. Your main loop looks something like:

```java
    int playerId = 1;
    int[][] board = new int[3][3];
    StdDraw.setXscale(0, 3); // x from 0 to 3
    StdDraw.setYscale(0, 3); // y from 0 to 3

    while (true) {
        if (StdDraw.mousePressed()) {
            // convert mouse locations to cell indexes
            // the cells are arranged as follows
```
int c = (int)(StdDraw.mouseX()); // column
int r = (int)(StdDraw.mouseY()); // row
if (board[r][c] == 0)
{
    board[r][c] = playerId;
    // place "X" or "O" on the board
    drawMarker(r, c, playerId);
    // check whether this player wins
    boolean iWin = checkWinner(board, playerId);
    if (iWin)
    {
        System.out.println("Player " + playerId + " wins");
        break;
    }
    // flip the playerId between 1 and 2
    playerId = 3 - playerId;
}
if (isTie(board))
{
    System.out.println("It is a tie");
    break;
}
}

Do not assume that mousePressed() responds only once when you press the left mouse button. In fact, it gives a sequence of messages. The above code segment avoids placing symbols into a cell multiple times by enforcing that the symbol placement only occurs once if the cell you click is empty.

You should implement some functions yourselves, for example drawMarker and checkWinner in the above code. Be free to choose whatever function names you like. The above code segment only illustrates some possible framework.

2 Bonus Question: Large Scale Tic Tac Toe

Extend Tic Tac Toe into a program that can generate boards of any size, for example 6 by 6. Your program should also be able to change the setting about how many same symbols must be in a row
Figure 1: Tic Tac Toe. (a)-(d) Steps in a game; (e) Another player wins; (f) It is a tie.

(in different directions) to win. On a bigger board, the number can be 4 or 5.

Figure 2: Large Tic Tac Toe. A player wins when there 4 same “X” or “O” in a row.

3 What to Submit

You should submit your Java programs. Pay attention to good Java programming style. Upload your Java files to webCT before the submission deadline. There will be 3 days grace period. But late submission would involve 10% point deduction for each day. Submissions later than 3 days are not accepted.