import java.util.Scanner;  
  
public class TicTacToe {  
 static char[] *board* = { ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ' };  
 static char *human* = 'X';  
 static char *computer* = 'O';  
  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.*in*);  
  
 *printBoard*();  
  
 while (true) {  
 int humanMove;  
 do {  
 System.*out*.print("Enter your move (1-9): ");  
 humanMove = sc.nextInt();  
 } while (!*isValidMove*(humanMove) || !*isCellEmpty*(humanMove));  
  
 *makeMove*(humanMove, *human*);  
 *printBoard*();  
  
 if (*checkWin*(*human*)) {  
 System.*out*.println("Congratulations! You win!");  
 break;  
 }  
 if (*isBoardFull*()) {  
 System.*out*.println("It's a draw!");  
 break;  
 }  
  
 int computerMove = *getBestMove*();  
 *makeMove*(computerMove, *computer*);  
 System.*out*.println("Computer's move: " + computerMove);  
  
 *printBoard*();  
  
 if (*checkWin*(*computer*)) {  
 System.*out*.println("Computer wins!");  
 break;  
 }  
 }  
 sc.close();  
 }  
  
 static boolean isValidMove(int move) {  
 return move >= 1 && move <= 9;  
 }  
  
 static boolean isCellEmpty(int move) {  
 return *board*[move - 1] == ' ';  
 }  
  
 static void makeMove(int move, char player) {  
 *board*[move - 1] = player;  
 }  
  
 static boolean isBoardFull() {  
 for (char cell : *board*) {  
 if (cell == ' ') {  
 return false;  
 }  
 }  
 return true;  
 }  
  
 static boolean checkWin(char player) {  
 return (*board*[0] == player && *board*[1] == player && *board*[2] == player)// for rows  
 || (*board*[3] == player && *board*[4] == player && *board*[5] == player)  
 || (*board*[6] == player && *board*[7] == player && *board*[8] == player)  
 || (*board*[0] == player && *board*[3] == player && *board*[6] == player)// for colms  
 || (*board*[1] == player && *board*[4] == player && *board*[7] == player)  
 || (*board*[2] == player && *board*[5] == player && *board*[8] == player)  
 || (*board*[0] == player && *board*[4] == player && *board*[8] == player)// for diagonal  
 || (*board*[2] == player && *board*[4] == player && *board*[6] == player);  
 }  
  
 static int minimax(int depth, boolean isMaximizing) {  
 if (*checkWin*(*human*)) {  
 return -1;  
 } else if (*checkWin*(*computer*)) {  
 return 1;  
 } else if (*isBoardFull*()) {  
 return 0;  
 }  
  
 if (isMaximizing) {  
 int bestScore = Integer.*MIN\_VALUE*;  
 for (int i = 0; i < 9; i++) {  
 if (*board*[i] == ' ') {  
 *board*[i] = *computer*;  
 int score = *minimax*(depth + 1, false);  
 *board*[i] = ' ';  
 bestScore = Math.*max*(bestScore, score);  
 }  
 }  
 return bestScore;  
 } else {  
 int bestScore = Integer.*MAX\_VALUE*;  
 for (int i = 0; i < 9; i++) {  
 if (*board*[i] == ' ') {  
 *board*[i] = *human*;  
 int score = *minimax*(depth + 1, true);  
 *board*[i] = ' ';  
 bestScore = Math.*min*(bestScore, score);  
 }  
 }  
 return bestScore;  
 }  
 }  
  
 static int getBestMove() {  
 int bestMove = -1;  
 int bestScore = Integer.*MIN\_VALUE*;  
  
 for (int i = 0; i < 9; i++) {  
 if (*board*[i] == ' ') {  
 *board*[i] = *computer*;  
 int score = *minimax*(0, false);  
 *board*[i] = ' ';  
 if (score > bestScore) {  
 bestScore = score;  
 bestMove = i + 1;  
 }  
 }  
 }  
  
 return bestMove;  
 }  
  
  
 static void printBoard() {  
  
 for (int i = 0; i < 9; i += 3) {  
 System.*out*.printf("| %c | %c | %c |\n", *board*[i], *board*[i + 1], *board*[i + 2]);  
 System.*out*.println("-------------");  
 }  
 }  
}