```
/**
 * This is the skeleton of the Pig game. It is your job to fill in the empty
functions to create a functional game of pig.
 * @author   Adam Smith
 * @version   1.1
 */

class Pig {
        /**
         * The score needed to win a round.
         */
        public static final int WINNING_SCORE = 100;

        public static void main(String[] args) {
                // intro, initialize players
                System.out.println("Welcome to Pig!");
                PigPlayer human = new HumanPigPlayer("Human");
                PigPlayer opponent = new ComputerPigPlayer("Skynet"); // could
be human too
                int[] roundsWon = new int[2];

                // round 1
                System.out.println("Round 1!");
                if (playRound(human, opponent)) roundsWon[0]++;
                else roundsWon[1]++;

                System.out.println();

                // round 2
                System.out.println("Round 2!");
                if (playRound(opponent, human)) roundsWon[1]++;
                else roundsWon[0]++;

                // report the final results
                reportFinalTally(roundsWon, human, opponent);
        }

        /**
         * Do one round, crediting the winner.
         * @param player1 the first player
         * @param player2 the second player
         * @return true if player1 won, false if player2
         */
        // This function must do the following:
        // 1. Enter an infinite loop, with player 1 taking a turn, then player
2.
        // 2. Keep track of each player's score and the turn number.
        // 3. When a player wins, print the winner, and break out of the loop.
        // 4. Return
        private static boolean playRound(PigPlayer player1, PigPlayer player2)
{
                return false;
        }

        /**
         * Play a single turn, returning how many points the player got.
         * @param player the player whose turn it is
         * @param turnNum the turn number (0-indexed)
         * @param score the player's score
         * @param opponentsScore the player's adversary's score
         * @return the points that the player won
         */
        // This function must do the following:
        // 1. Call the player's beginTurn() method.
        // 2. Loop so long as the player wants to continue rolling.
        // 3. Roll a die:
        //     a. If a 1 is rolled, return 0.
        //     b. On any other roll, add it to the pool.
        // 4. If the loop ends, return the pool's value.
        // 5. Be sure to print events frequently, so the human player can see
what's
        //    happening!
        private static int playTurn(PigPlayer player, int turnNum, int score,
int opponentsScore) {
               return 0;
        }

        /**
         * Deliver a final report, indicating the overall winner after all
rounds
         * have been played.
         * @param roundsWon an array of <code>int</code>s indicating the
number of rounds each player won
         * @param player1 the first player
         * @param player2 the second player
         */
        // This function must do the following:
        // 1. Print out both player's scores.
        // 2. Indicate who the winner was (or if there was a tie).
        private static void reportFinalTally(int[] roundsWon, PigPlayer
player1, PigPlayer player2) {

        }
}
```