```
package lesson8;

import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.util.Arrays;
import java.util.Random;
import java.util.Scanner;

public class Test2 {
	private static final Scanner scanner = new Scanner(System.in);
	private static final int MAX_TURNS = 100;

	private static final char[][][] MAZE_MAPS = {
		{
			{ '#', '#', '#', '#', '#', '#', '#', '#', '#', '#' },
			{ '#', ' ', ' ', ' ', ' ', ' ', '#', ' ', ' ', '#' },
			{ '#', ' ', '#', '#', '#', ' ', '#', '#', ' ', '#' },
			{ '#', ' ', '#', ' ', '#', ' ', ' ', '#', ' ', '#' },
			{ '#', ' ', '#', ' ', '#', '#', ' ', '#', ' ', '#' },
			{ '#', ' ', ' ', ' ', ' ', '#', ' ', ' ', ' ', '#' },
			{ '#', '#', '#', '#', ' ', '#', '#', '#', ' ', '#' },
			{ '#', ' ', ' ', '#', ' ', ' ', ' ', '#', 'T', '#' },
			{ '#', ' ', '#', '#', '#', '#', ' ', ' ', 'E', '#' },
			{ '#', '#', '#', '#', '#', '#', '#', '#', '#', '#' }
		},
		{
			{ '#', '#', '#', '#', '#', '#', '#', '#', '#', '#' },
			{ '#', ' ', '#', ' ', ' ', ' ', '#', ' ', 'T', '#' },
			{ '#', ' ', '#', '#', '#', ' ', '#', ' ', '#', '#' },
			{ '#', ' ', ' ', ' ', '#', ' ', ' ', ' ', ' ', '#' },
			{ '#', ' ', '#', '#', '#', '#', '#', ' ', '#', '#' },
			{ '#', ' ', ' ', ' ', ' ', ' ', '#', ' ', ' ', '#' },
			{ '#', '#', '#', '#', ' ', '#', '#', ' ', '#', '#' },
			{ '#', ' ', ' ', '#', ' ', ' ', ' ', ' ', ' ', '#' },
			{ '#', 'T', '#', '#', '#', '#', ' ', '#', 'E', '#' },
			{ '#', '#', '#', '#', '#', '#', '#', '#', '#', '#' }
		},
		{
			{ '#', '#', '#', '#', '#', '#', '#', '#', '#', '#' },
			{ '#', ' ', ' ', ' ', '#', ' ', ' ', ' ', ' ', '#' },
			{ '#', ' ', '#', '#', '#', ' ', '#', '#', ' ', '#' },
			{ '#', ' ', '#', ' ', ' ', ' ', '#', '#', ' ', '#' },
			{ '#', ' ', '#', ' ', '#', '#', '#', ' ', ' ', '#' },
			{ '#', ' ', ' ', ' ', ' ', ' ', '#', ' ', 'T', '#' },
```

```
                { '#', '#', '#', '#', ' ', '#', '#', ' ', '#', '#' },
                { '#', ' ', ' ', '#', ' ', ' ', ' ', ' ', ' ', '#' },
                { '#', ' ', '#', '#', '#', '#', ' ', '#', 'E', '#' },
                { '#', '#', '#', '#', '#', '#', '#', '#', '#', '#' }
        },
        {
                { '#', '#', '#', '#', '#', '#', '#', '#', '#', '#' },
                { '#', ' ', '#', ' ', ' ', ' ', ' ', '#', ' ', '#' },
                { '#', ' ', 'T', ' ', '#', '#', ' ', '#', ' ', '#' },
                { '#', ' ', '#', ' ', '#', ' ', ' ', '#', ' ', '#' },
                { '#', ' ', '#', '#', '#', ' ', '#', '#', ' ', '#' },
                { '#', ' ', ' ', ' ', ' ', ' ', '#', 'T', ' ', '#' },
                { '#', '#', '#', '#', '#', ' ', '#', '#', '#', '#' },
                { '#', ' ', ' ', ' ', '#', ' ', ' ', ' ', ' ', '#' },
                { '#', 'T', '#', ' ', '#', '#', '#', ' ', 'E', '#' },
                { '#', '#', '#', '#', '#', '#', '#', '#', '#', '#' }
        },
        {
                { '#', '#', '#', '#', '#', '#', '#', '#', '#', '#' },
                { '#', ' ', ' ', '#', ' ', ' ', ' ', ' ', ' ', '#' },
                { '#', ' ', '#', '#', ' ', '#', '#', '#', ' ', '#' },
                { '#', ' ', '#', ' ', ' ', ' ', ' ', '#', ' ', '#' },
                { '#', ' ', '#', ' ', '#', '#', ' ', '#', ' ', '#' },
                { '#', ' ', ' ', ' ', '#', ' ', ' ', ' ', ' ', '#' },
                { '#', '#', '#', '#', '#', ' ', '#', '#', ' ', '#' },
                { '#', ' ', ' ', ' ', ' ', ' ', '#', 'T', ' ', '#' },
                { '#', ' ', '#', '#', '#', ' ', ' ', ' ', 'E', '#' },
                { '#', '#', '#', '#', '#', '#', '#', '#', '#', '#' }
        },
        {
                { '#', '#', '#', '#', '#', '#', '#', '#', '#', '#' },
                { '#', 'P', ' ', ' ', '#', ' ', ' ', ' ', ' ', '#' },
                { '#', '#', '#', ' ', '#', ' ', '#', '#', ' ', '#' },
                { '#', ' ', 'T', ' ', '#', ' ', ' ', '#', ' ', '#' },
                { '#', ' ', '#', '#', '#', '#', '#', '#', ' ', '#' },
                { '#', ' ', '#', ' ', ' ', 'T', ' ', '#', ' ', '#' },
                { '#', ' ', '#', '#', '#', '#', ' ', '#', '#', '#' },
                { '#', ' ', ' ', ' ', '#', ' ', ' ', ' ', ' ', 'E' },
                { '#', '#', '#', ' ', '#', '#', '#', '#', '#', '#' },
                { '#', '#', '#', '#', '#', '#', '#', '#', '#', '#' }
        }

};
```

```
public static void main(String[] args) throws Exception {
        System.out.println("\uD83D\uDC6B 複数人プレイ迷路ゲーム（ターン制）");
        System.out.println("\uD83C\uDFAF ゴール: 'E' に最速でたどり着け！");
        System.out.println("➡️ 操作方法: W = 上, S = 下, A = 左, D = 右\n");

        System.out.print("プレイヤー1の名前を入力してください: ");
        Player player1 = new Player(scanner.nextLine());
        System.out.print("プレイヤー2の名前を入力してください: ");
        Player player2 = new Player(scanner.nextLine());

        playGame(player1);
        playGame(player2);

        announceWinner(player1, player2);
}

private static void playGame(Player player) {
        System.out.println("\n" + player.name + " のターン:");
        char[][] maze = deepCopyMaze(getRandomMaze());
        int x = 1, y = 1;
        long startTime = System.currentTimeMillis();

        // Track moves as strings for example: "W", "A", "S", "D"
        StringBuilder movesHistory = new StringBuilder();

        for (int turn = 0; turn < MAX_TURNS; turn++) {
                printMaze(maze, x, y);
                System.out.print("移動（WASD）: ");
                char move = scanner.next().toLowerCase().charAt(0);

                int newX = x, newY = y;
                switch (move) {
                case 'w':
                        newX--;
                        movesHistory.append("W ");
                        break;
                case 's':
                        newX++;
                        movesHistory.append("S ");
                        break;
                case 'a':
                        newY--;
                        movesHistory.append("A ");
                        break;
```

```
                        case 'd':
                                newY++;
                                movesHistory.append("D ");
                                break;
                        default:
                                System.out.println("❌ 無効な入力です。W, A, S, D を使ってください。");
                                continue;
                        }

                        // Check bounds
                        if (newX < 0 || newX >= maze.length || newY < 0 || newY >= maze[0].length) {
                                System.out.println("❌ 迷路の外です。別の方向を試してください。");
                                continue;
                        }

                        if (maze[newX][newY] == '#') {
                                playSoundSafe("打撃4.mp3");
                                System.out.println("🚧 壁にぶつかった！");
                                continue;
                        }

                        x = newX;
                        y = newY;

                        if (maze[x][y] == 'T') {
                                playSoundSafe("trap.mp3");
                                System.out.println("💥 トラップにかかった！スタート地点に戻ります！");
                                x = 1;
                                y = 1;
                                continue;
                        }

                        if (maze[x][y] == 'E') {
                                playSoundSafe("「おめでとう」.mp3");
                                long endTime = System.currentTimeMillis();
                                player.timeTaken = (endTime - startTime) / 1000.0;
                                System.out.println("🎉 " + player.name + " は " + player.timeTaken + " 秒でゴール！");
                                System.out.println("🚶 移動履歴: " + movesHistory.toString());
                                return;
```

```
                }
            }
            System.out.println("⏰ タイムオーバー！ゴールできませんでした。");
            player.timeTaken = Double.MAX_VALUE;
            System.out.println("🚶 移動履歴: " + movesHistory.toString());
        }

        private static void printMaze(char[][] maze, int playerX, int playerY) {
            for (int i = 0; i < maze.length; i++) {
                for (int j = 0; j < maze[i].length; j++) {
                    if (i == playerX && j == playerY) {
                        // Highlight player position in cyan
                        System.out.print("\u001B[36mP\u001B[0m ");
                    } else if (maze[i][j] == 'T') {
                        // Highlight traps in yellow
                        System.out.print("\u001B[33mT\u001B[0m ");
                    } else if (maze[i][j] == 'E') {
                        // Highlight exit in red
                        System.out.print("\u001B[31mE\u001B[0m ");
                    } else {
                        System.out.print(maze[i][j] + " ");
                    }
                }
                System.out.println();
            }
        }

        private static char[][] getRandomMaze() {
            Random rand = new Random();
            return MAZE_MAPS[rand.nextInt(MAZE_MAPS.length)];
        }

        private static char[][] deepCopyMaze(char[][] original) {
            char[][] copy = new char[original.length][original[0].length];
            for (int i = 0; i < original.length; i++) {
                copy[i] = Arrays.copyOf(original[i], original[i].length);
            }
            return copy;
        }

        private static void announceWinner(Player p1, Player p2) {
            System.out.println("\n\uD83C\uDFC1 ゲーム終了！");
            System.out.println(p1.name + ": " + (p1.timeTaken == Double.MAX_VALUE ? "失
敗" : p1.timeTaken + "秒"));
```

```
                System.out.println(p2.name + ": " + (p2.timeTaken == Double.MAX_VALUE ? "失
敗" : p2.timeTaken + "秒"));
                if (p1.timeTaken < p2.timeTaken) {
                        System.out.println("\uD83C\uDFC6 勝者: " + p1.name);
                } else if (p2.timeTaken < p1.timeTaken) {
                        System.out.println("\uD83C\uDFC6 勝者: " + p2.name);
                } else {
                        System.out.println("\uD83E\uDD1D 引き分け！");
                }
        }

        // Unified method to safely play sound files without crashing on errors
        private static void playSoundSafe(String filename) {
                try (FileInputStream fileInputStream = new FileInputStream(filename)) {
                        javazoom.jl.player.Player player = new
javazoom.jl.player.Player(fileInputStream);
                        player.play();
                } catch (FileNotFoundException e) {
                        System.out.println("⚠️ サウンドファイルが見つかりません: " + filename);
                } catch (Exception e) {
                        System.out.println("⚠️ サウンド再生中にエラーが発生しました: " +
e.getMessage());
                }
        }

        static class Player {
                String name;
                double timeTaken;

                Player(String name) {
                        this.name = name;
                        timeTaken = Double.MAX_VALUE;
                }
        }
}
```