Dontae Lundy

June 15, 2016

C# Programming

Battleship AI

Battleship AI : Design Document

# Problem Definition:

This document will define a High-Level overview of an AI program for the game “Battleship”. The purpose of the Battleship AI is to strategically and coherently play the game with opponents without violating the rules of the game (cheating). The AI must follow the same processes as all users, which begins at having absolutely no knowledge of the opponent’s grid layout – from there a series of guesses must be made in order to find a target. Once the AI find’s a target, the program should utilize various algorithms in order to destroy the target.

**Key Points:**

* Find Target
* When target is found, determine it’s shipType
* “Circle” the target in order to determine it’s layout status (Vertical, Horizontal)
* Once layout status is determine, focus attacks (either Horizontal or Vertical)
* If target is missed, save the attack state and continue attack on next turn

AI must also be able to choose it’s ship placement for it’s own grid. In addition, AI must recognize when it is under attack and examine it’s grid each time an attack occurs and let the opponent know whether or not their attack was successful. If the attack was successful, AI must alert the user of a ‘HIT’ as well as the type of ship that was Hit.

enum direction { Horizontal, Vertical }

enum feedback { HIT, MISS }

enum Vehicle { Aircraft, Battleship, PatrolBoat, Destroyer, Submarine }

class Grid {

// Create a Series of Squares as a 'Grid'

int size;

Square[,] grid;

public Grid(); // Constructor

public void populateGridKey(); // GridKey

public void print(); // Prints Grid

class Square {

// Each Position is represented as a single Square.

bool hit = false;

Coordinate myLocation;

public string symb;

public Square(); // Constructor

}

class Player {

String Name;

Grid myGrid;

Coordinate myShips, strikePoints;

public Player(string x, int y);

public void printGrid();

private void setGrid() {

// Players Choose Where to Place Ships

}

public void giveFeedback() {

// Announce Hit or Miss + ShipType

}

}

class AI : Player {

// AI Class.

bool found = false;

Coordinate[] old, newer; // Old = Guessed | newer - Not Guessed Yet.

public AI(int x): base("CPU", x) {

// AI Constructor

}

public void findTarget() {

// Searches for Targets through random guesses.

}

public void circleTarget() {

// Circles Hit Location to find Target

}

public void resumeAttack() {

// Resumes Attack on Position

}

}