**שיעורי בית ביסודות MiniMax – אופיר הופמן י3**

**ארבע בשורה חד מימדי**

enum States { notStarted, On, winAndGameOver };

class AIRow

{

int[] row;

States sts;

int Cursor;

int MinMaxCnt = 0;

const int WIN\_LEN = 4;

const int MAX\_DEEP = 7;

const int WIN\_VALUE = 1000;

public AIRow(int N)

{

sts = States.notStarted;

row = new int[N];

}

public int GetLen()

{

return WIN\_LEN;

}

public int GetDepth()

{

return MAX\_DEEP;

}

public void NewGame()

{

clear();

sts = States.On;

Cursor = 0;

}

private void clear()

{

for (int i = 0; i < row.Length; i++)

{

row[i] = 0;

}

// row[0] = 1;

// row[4] = 1;

// row[3] = 2;

}

public void PrintRow()

{

Console.SetCursorPosition(0, 10);

for (int i = 0; i < row.Length; i++)

Console.Write(" ");

Console.SetCursorPosition(((Cursor \* 3)) % 80, 10);

Console.ForegroundColor = ConsoleColor.Red;

Console.Write("@");

Console.ForegroundColor = ConsoleColor.White;

Console.SetCursorPosition(0, 11);

for (int i = 0; i < row.Length; i++)

{

Console.Write(row[i] + ", ");

}

Console.WriteLine();

int score = GetScore();

Console.SetCursorPosition(0, 13);

Console.WriteLine("Score: " + score);

Console.WriteLine(" Game State = " + sts.ToString());

Console.WriteLine("------------------ minimax times = " + MinMaxCnt + " ");

Console.WriteLine();

}

public void PlayerMoveCursor(ConsoleKey k)

{

if (k == ConsoleKey.RightArrow)

Cursor = (Cursor +1 ) % row.Length;

if (k == ConsoleKey.LeftArrow )

Cursor = (row.Length + Cursor -1) % row.Length;

}

public bool DoPlayerMove()

{

if (sts != States.On)

return false;

MinMaxCnt = 0;

if (row[Cursor] == 0)

{

row[Cursor] = 1;

return true;

}

return false;

}

public void DoPCMove()

{

if (sts != States.On)

return;

MinMaxCnt = 0;

int move\_i = -1; ;

int max = int.MinValue;

for (int i = 0; i < row.Length; i++)

{

if (row[i] == 0)

{

row[i] = 2;

int score = MiniMax(true, MAX\_DEEP);

row[i] = 0;

if (score >= max)

{

max = score;

move\_i = i;

}

}

}

if (move\_i != -1)

{

row[move\_i] = 2;

Cursor = move\_i;

}

}

// PC is 2, Player is 1

private int MiniMax(bool playerTurn, int depth)

{

MinMaxCnt++; // only for count how many times calling to it

int currScore = (playerTurn) ? int.MaxValue : int.MinValue;

//TODO

if (depth == 0 || NoMoveLeft() || Math.Abs(GetScore()) == WIN\_VALUE)

return GetScore();

int score = 0;

for (int i = 0; i < row.Length; i++)

{

if (row[i] == 0)

{

row[i] = (playerTurn) ? 1 : 2;

score = MiniMax(!playerTurn, depth - 1);

row[i] = 0;

if (playerTurn && score < currScore) // minimum

currScore = score;

else if (!playerTurn && score > currScore) //maximum

currScore = score;

}

}

return currScore;

}

public bool CheckWin()

{

int score = GetScore();

if (score == -1 \*(WIN\_VALUE))

{

Console.SetCursorPosition(0, 22);

Console.Write("1 ----- Player Won ----- ");

sts = States.winAndGameOver;

return true;

}

if (score == WIN\_VALUE)

{

Console.SetCursorPosition(0, 22);

Console.Write("2 ---- Computer Won ---- " );

sts = States.winAndGameOver;

return true;

}

return false;

}

// Return positive or negative

// the return value will be high and positive

// as long as it better to computer

// or negetive if it is good to player

// in case of win it will return WIN\_VALUE or - WIN\_VALUE

// (if there are WIN\_LEN items)

private int GetScore()

{

int ret1 = 0;

int ret2 = 0;

int cnt1 = 1;

int cnt2 = 1;

int max1 = 0;

int max2 = 0;

for (int i = 0; i < row.Length-1; i++)

{

if (row[i] == row[i + 1] && row[i] == 1)

cnt1++;

else if (row[i] == row[i + 1] && row[i] == 2)

cnt2++;

else

{

max1 = Math.Max(cnt1, max1);

max2 = Math.Max(cnt2, max2);

cnt1 = 1;

cnt2 = 1;

}

}

if (max1 == 4)

return -WIN\_VALUE;

else if (max2 == 4)

return WIN\_VALUE;

return max2 - max1;

}

public bool NoMoveLeft()

{

for (int i = 0; i < row.Length; i++)

if (row[i] == 0)

return false;

return true;

}

}

static void Main(string[] args)

{

AIRow game = new AIRow(13);

game.NewGame();

Console.WriteLine("1 = Player ...You. ");

Console.WriteLine("2 = Computer");

Console.WriteLine("Winner = sequence of " + game.GetLen());

Console.WriteLine("MAx Minimax Depth = " + game.GetDepth() );

Console.WriteLine("Use Arrows for move.");

Console.WriteLine("Use Spcae to put 1 in location");

bool PlayerTurn = true;

ConsoleKey key = 0;

bool win = false;

game.PrintRow();

bool inGame = true;

while (inGame)

{

if (PlayerTurn && !game.NoMoveLeft())

{

if (Console.KeyAvailable)

{

key = Console.ReadKey().Key;

if (key == ConsoleKey.LeftArrow || key == ConsoleKey.RightArrow)

game.PlayerMoveCursor(key);

else if (key == ConsoleKey.Q)

inGame = false;

else if (key == ConsoleKey.Spacebar)

{

bool Player\_moved = game.DoPlayerMove();

if (Player\_moved)

{

PlayerTurn = false;

win = game.CheckWin();

if (win)

inGame = false;

}

}

}

}

else if (!PlayerTurn && !game.NoMoveLeft())

{

Console.ForegroundColor = ConsoleColor.Yellow;

Console.Write("PC Think ...");

Console.ForegroundColor = ConsoleColor.White;

game.DoPCMove();

Console.WriteLine("\r PC Done !");

win = game.CheckWin();

if (win)

inGame = false;

PlayerTurn = true;

}

if (game.NoMoveLeft() && !win)

{

Console.WriteLine(" -- Tie -- ");

Console.WriteLine();

inGame = false;

}

game.PrintRow();

Thread.Sleep(50);

if (game.NoMoveLeft())

inGame = false;

}

Console.WriteLine("\nEND. Press any key");

Console.ReadKey();

}