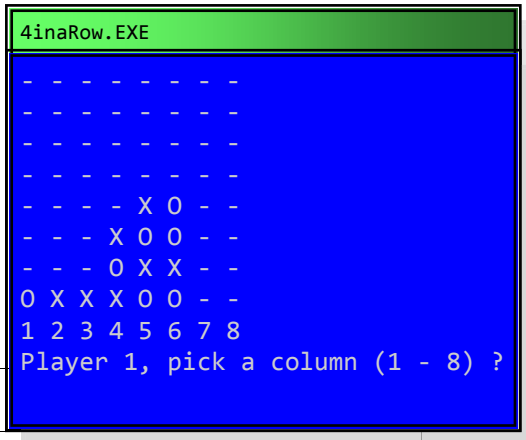
4INAROW.C

4inarow is a fairly well known board game, a sort of tic-tac-toe with gravity. Tokens of different colors are dropped into chutes, the winner is the one who gets 4 in a row, as the name implies. Keeping one step ahead of your opponent can be challenging and this computer opponent is no slouch. But it can be beaten and will prepare you well to beat a human opponent in real life.

4inarow is written by Joseph Larson based on a BASIC game by James L. Murphy as found in 'More BASIC Computer Games' by David H. Ahl © 1979.



4INAROW.C	You will need: a C/C++ complier .

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <string.h>

#define SHOWBD for (c=9; c >= 0; c--) puts (bd[c])
#define _X 'X'
#define _O 'O'

unsigned long v[16] = {1, 75, 500, 1e9, 1, 800, 4000, 1e9
                      ,1,100, 900, 1e7, 1, 450, 3000, 1e7};
int inrow[4], open[4], h[8];
char bd[9][20];

void intro (void) {
  puts ("Four in a Row\n---- -- - ---\n"
  "Stack X's and O's in order to make 4 in a row either vertically,\n"
  "horizontally diagonally before your opponent does.\n");
}

void init (void) {
  int c;

  for (c = 0; c < 8;) {
    h[c] = 0;
    strcpy (bd[++c], "- - - - - - - - ");
  }
  strcpy (bd[0], "1 2 3 4 5 6 7 8 ");
}

int count (int x, int y, char token) {
  int w, k, dx, dy, cx, cy, c, t;
  char op;

  x *= 2; op = (token - _X) ? _X : _O;
  for (c = 0; c < 4; c++) {
    inrow[c] = 1; open[c] = 0;
    dx = 2 * (c - 1 - (c > 2));
    dy = (c != 3);
    for (w = 0; w < 2; w++) {
      t = 1;
      for (k = 1; k < 4 && bd [cy = y + dy * k][cx = x + dx * k] != op; k++)
        if (cx <= 15 && cx >= 0 && cy <= 8 && cy > 0) {
```

Listing continued on page 2...

```
          if (t && bd[cy][cx] == token) inrow[c]++;
          else {open[c]++; t = 0;}
        }
      dx = -dx; dy = -dy;
    }
    if (inrow[c] > 3) return 1;
  }
  k = 0;
  for (c = 0; c < 8; c++) if (h[c] < 8) k++;
  if (!k) return 2;
  return 0;
}

int domove (int m, char token) {
  bd [++h[m]][2 * m] = token;
  return count (m, h[m], token);
}

int getmove (int pl) {
  int input = 0;

  do {
    if (input) puts ("Illegal move, try again.");
    printf ("Player %d, pick a column (1 - 8) ? ", pl); scanf ("%d", &input);
    if (input < 1 || input > 8 || h[input - 1] > 7) input = -1;
  } while (input < 0);
  return --input;
}

int compmove (void) {
  unsigned long rank, bestrank;
  int bestmove, w, x, y, c, n[4], numsame;
  char token;

  bestmove = bestrank = 0; numsame = 1;
  for (x = 0; x < 8; x++) {
    y = h[x] + 1;
    if (y < 9) {
      rank = 1; token = _O;
      for (w = 0; w < 2; w++) {
        if (count (x, y, token)) {
          printf ("Computer picks column %d\n", x + 1); return x;
        }
        for (c = 0; c < 4; c++) n[c] = 0;
        for (c = 0; c < 4; c++) {
          open[c] += inrow[c];
          if (open[c] > 3) {rank += 4; n[inrow[c] - 1]++;}
        }
        for (c = 0; c < 4; c++) if (n[c]--)
          rank += v[8 * w + 4 * (n[c] ? 1 : 0) + c] + n[c] * v[8 * w + c];
        token = _X;
      }
      if (y < 8) if (count(x, y + 1, token)) rank = 2;
      if (rank == bestrank) if (rand() < RAND_MAX / ++numsame) {
        bestrank = rank; bestmove = x;
      }
    }
```

Listing continued on page 3...

4INAROW.C	Listing Continued from page 2….

```
    if (rank > bestrank) {bestrank = rank; bestmove = x; numsame = 1;}
  }
  printf ("Computer picks column %d\n", bestmove + 1);
  return bestmove;
}

int main (void) {
  int c, numpl, w = 0;

  intro ();
  srand (time (NULL));
  init ();
  printf ("One or two human players? (1/2) "); scanf ("%d", &numpl);
  while (numpl > 2 || numpl < 1) {
    printf ("Please type the number 1 or 2 ? "); scanf ("%d", &numpl);
  }
  if (!--numpl) puts ("The Computer will be Player 2.");
  if (rand () % 2) {
    puts ("Player 1 goes first.");
    SHOWBD;
    domove (getmove (1), _X);
  } else puts ("Player 2 goes first.");
  while (!w) {
    SHOWBD;
    if (!(w = domove ((numpl) ? getmove (2) : compmove (), _O))) {
      SHOWBD;
      w = domove (getmove (1), _X);
    } else if (w == 1) w = 3;
  }
  SHOWBD;
  switch (w) {
    case 1 : puts ("Player 1 wins!"); break;
    case 2 : puts ("Tie game."); break;
    case 3 : puts ("Player 2 wins!");
  }
  exit (0);
}
```

Author's Notes:

This version recreates as accurately as possible the AI present in James L. Murphy's original BASIC game which was to assigns weights to the various the various "in a rows" and whether those are his or yours. In other words the computer opponent will assigns a different value if a move will give you a two or three in a row than if it will give itself two or three in a row. Some values end up being somewhat redundant because it will always take or block 4 in a row, so the "weight" in that case really doesn't matter. Still, it plays well enough and is guaranteed to prepare you for that rematch with your brother that you've been putting off for years.