Mugwump

Mugwump hunting has come a long way. While Hurkle hunting is still done one Hurkle at a time the old fashioned way Mugwump hunting uses a state of the art radar system that gives you the distance you are to multiple Mugwumps at once. Sure it can be confusing trying to triangulate multiple Mugwumps locations, but a little practice and it's not impossible.

This is the sort of game that is best played with a piece of graph paper and a compass near by to help you plan your next move.

A good mugwump hunter can find all the mugwumps in less than 15 turns. An excellent one, less than 10.

Mugwump was written by Joe Larson inspired by a BASIC game of the same name as found in 'BASIC Computer Games' edited by David H. Ahl © 1978.

mugwump.c linsting begins:

```
#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <time.h>

#define dist(a,b) sqrt((b.x - a.x) * (b.x - a.x) + (b.y - a.y) * (b.y - a.y))
#define NUM 5
#define SIZE 20

typedef struct {
  int x, y;
} COORD;

int greater (double *a, double *b) {return (*a > *b);}

void intro (void) {
  printf ("Mugwump Hunt\n-----------\n"
  "You are hunting Mugwumps using a state of the art radar system. There are\n"
  "%d mugwumps hidden hiding in a %d,%d grid. After every guess you will be\n"
  "told how close the remaining Mugwumps are to your guess.\n"
  "Input the your guess by typing the x and y location separated by a comma\n"
  "(IE \"5,5\" or \"3,10\")\n"
  "(Mugwumps only hide whole number locations.)\n\n",
  NUM, SIZE, SIZE);
}

void hunt (void) {
  int c, turns, left, dsize;
  COORD h[NUM], input;
  double d[NUM];
  time_t st, cur, dit;

  for (c = 0; c < NUM; c++) {
    h[c].x = rand() % SIZE + 1;
    h[c].y = rand() % SIZE + 1;
  }
  printf ("The Mugwumps are hiding.");
  turns = 0;
  left = NUM;
  while (left > 0) {
    turns ++;
    printf ("\n%d Mugwumps left.\n", left);
   input.x = input.y = -1;
    do {
      if (input.x != -1 && input.y != -1)
        printf ("X and Y values must be between 1 and %d", SIZE);
        printf ("\nWhere do you want to search? X,Y : ");
        scanf ("%d %*c %d", &input.x, &input.y);
    } while (input.x < 1 || input.x > SIZE || input.y < 1 || input.y > SIZE);
    dsize = 0;
    printf ("Mugwump Radar Searching from %d,%d...", input.x, input.y);
    for (c = 0; c < NUM; c++)
      if (h[c].x == input.x && h[c].y == input.y) {
        printf ("\nMugwump found!");
        h[c].x = h[c].y = -1;
        left--;
      } else if (h[c].x > 0) d[dsize++] = dist(h[c], input);
    if (dsize) {
      qsort (d, dsize, sizeof(double), greater);
      time (&st); dit = st;
```

Listing continued on next page...

Listing continued from previous page

```
      for (c = 0; c < dsize; c++) {
        do {
          time (&cur);
          if (difftime (cur, dit) > 0.2) {putchar ('.'); time (&dit);}
        } while (difftime (cur, st) < d[c] / 2);
        printf ("\nMugwump at distance %.2f", d[c]);
      }
    }
  }
  printf ("Congratulation! All Mugwumps found in %d turns!", turns);
}

int again (void) {
  char input;

  printf ("\nDo you want to play again? (y/n) ");
  while (!isalpha (input = getchar()));
  if (tolower (input) != 'n') return 1;
  return 0;
}

int main (void) {
  intro();
  srand (time (NULL));
  do {hunt();} while (again ());
  exit          (0);
}
```