3DMAZE.COM

Mazes are hard enough, but imagine being the mouse running one. Well now you can experience it in 3DMaze! Try to navigate the maze but be careful, it's easy to get lost.

Use the following keys to navigate:

F or W ..... Step Forward
B or S ...... Step Backward
L or A ...... Turn Left
R or D...... Turn Right
Z .... Zip forward to next turn or wall
M .... Show Map

3DMaze is written by Joseph Larson using a 3D drawing routine by Sean Barrett 1991, found at the International Obfuscated C Code Contest (http://www.ioccc.org).

3DMAZE.C	You will need: a C/C++ complier .

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

#define XMAX 39
#define YMAX 11
#define MID 26
#define 0 0
#define 1 1
#define 2 2
#define TRAVL 3
#define EXIT -1
#define D(a) (2 * (a + 1))
#define NP(a, b) (a * (b - 1) + (a - 1) * b)
#define SWAP(a,b) a ^= b; b ^= a; a ^= b
#define X(a) xytable[(a) % 4]
#define Y(a) xytable[(a + 1) % 4]

int scale[6] = {11, 10, 6, 3, 1, 0};
int xytable[4] = {1, 0, -1, 0};
int maze[D(XMAX)][D(YMAX)], xsize, ysize, x, y, face, hpos;

void drawmap (void) {
  int c, d;

  for (d = 2 * ysize; d >= 0; d--) {
    putchar ('\n');
    for (c = 1; c < D(xsize); c++)
      putchar ((c == x && d == y) ? '*' : "## ."[maze[c][d]]);
  }
}

void snr (int n1, int n2) { /* search and replace */
  int c, d;

  if (n1)
```

Listing continued on next page...

```
    for (c = 2; c < D(xsize); c++) for (d = 1; d < D(ysize) - 1; d++)
      if (maze[c][d] == n1) maze[c][d] = n2;
}

void generate (void) {
  int px[NP(XMAX, YMAX)], py[NP(XMAX, YMAX)], b, c, d;

  for (c = 0; c < D(XMAX); c++) for (d = 0; d < D(YMAX); d++)
    maze[c][d] = (c) ? 0 : EXIT;
  b = 0;
  /* populate potentials */
  for (c = 3; c < 2 * xsize; c += 2) for (d = 1; d < 2 * ysize; d += 2) {
    px[b] = c; py[b++] = d;
  }
  for (c = 2; c <= 2 * xsize; c += 2) for (d = 2; d < 2 * ysize; d += 2) {
    px[b] = c; py[b++] = d;
  }
  /* randomize potentials */
  for (b = 0; b < NP(xsize, ysize); b++) {
    c = rand () % NP(xsize, ysize);
    if (c - b) {SWAP (px[c], px[b]); SWAP (py[c], py[b]);}
  }
  /* make the maze */
  for (b = 0; b < NP(xsize, ysize); b++) {
    c = d = 0;
    if (py[b] % 2) c = 1; else d = 1;
    if (!maze[px[b] + c][py[b] + d] || !maze[px[b] - c][py[b] - d]
      || maze[px[b] + c][py[b] + d] != maze[px[b] - c][py[b] - d]) {
      snr (maze[px[b] + c][py[b] + d], b + 1);
      snr (maze[px[b] - c][py[b] - d], b + 1);
      maze[px[b]][py[b]] = maze[px[b] + c][py[b] + d]
      = maze[px[b] - c][py[b] - d] = b + 1;
    }
  }
  snr (maze[2][1], 1);
  /* now make an exit */
  maze[1][2 * (rand () % ysize) + 1] = 1;
  x =  2 * xsize; y =  2 * (rand () % ysize) + 1; face = 2;
}

int look (int f, int s) {
  int c, d, r;

  c = x + X(face) * f + X(face - 1) * s;
  d = y + Y(face) * f + Y(face - 1) * s;
  r = maze[c][d]; if (r == 1) maze[c][d] = 2;
  return r;
}

void draw (int col, char *s) {
  while (hpos < col) {putchar (' '); hpos++;}
  printf ("%s", s); hpos += strlen (s);
}

void draw3d(void) {
  int line, side, depth, p, q, i;
```

Listing continued on next page...

```
  putchar ('\n');
  for (line = -11; line <= 11; ++line) {
    for(side = 1, depth = 0; side + 3; side-=2) {
      for (; depth != 2 + 3 * side; depth += side) {
        if(look (depth, 0) < 1) {
          p = scale[depth];
          if (abs(line) < p && look(depth, 0) == 0 || abs (line) > p) break;
          for (i = -p; i < p ; ++i) draw (MID + i * 2, "--");
          draw (0, "-"); break;
        }
        if (depth == 5) continue;
        p = scale[depth + 1]; q = scale[depth];
        if (abs (line) > q) continue;
        if (abs (line) < p) draw (MID - side * (2 * p + 1), "|");
        else if (look (depth, side)) {
          if (abs (line) <= p)
            for(i = (side == 1 ? -q : p); i != (side == 1 ? -p : q);
              (abs (line)), ++i)
              draw (MID + 2 * i + (side == -1), "--");
        } else if (abs(line) == p)
          draw (MID - side * (2 * p + 1), "|");
        else draw (MID - (abs (line) * side * 2),
          (side == 1) ^ (line > 0) ? "\\" : "/");
      }
      depth -= side;
    }
    putchar ('\n'); hpos=0;
  }
  if (maze[x][y] != EXIT) maze[x][y] = TRAVL;
}

int move (void) {
  char in;

  printf ("\n%c (F)orward, (L)eft or (R)ight? ", "WSEN"[face]);
  do in = getche (); while (!isalnum (in)); in = tolower (in);
  if (in == 'm') drawmap ();
  else {
    if ((in == 'w' || in == 'f') && look (1, 0) != 0)
      {x += X(face); y += Y(face);}
    if ((in == 's' || in == 'b') && look (-1, 0) != 0)
      {x -= X(face); y -= Y(face);}
    face += (in == 'd' || in == 'r') + 3 * (in == 'a' || in == 'l');
    if (in == 'z'&& look (1, 0) != 0) do {x += X(face); y += Y(face); draw3d ();}
      while (look (1, 0) > 0 && look (0, 1) == 0 && look (0, -1) == 0);
    draw3d (); face %= 4;
  }
  return (in != 'q');
}

void setsize (void) {
  char in;

  printf ("(E)asy, (M)edium or (H)ard? ");
  do {
    do in = getche (); while (!isalnum (in)); in = tolower (in);
```

Listing continued on next page...

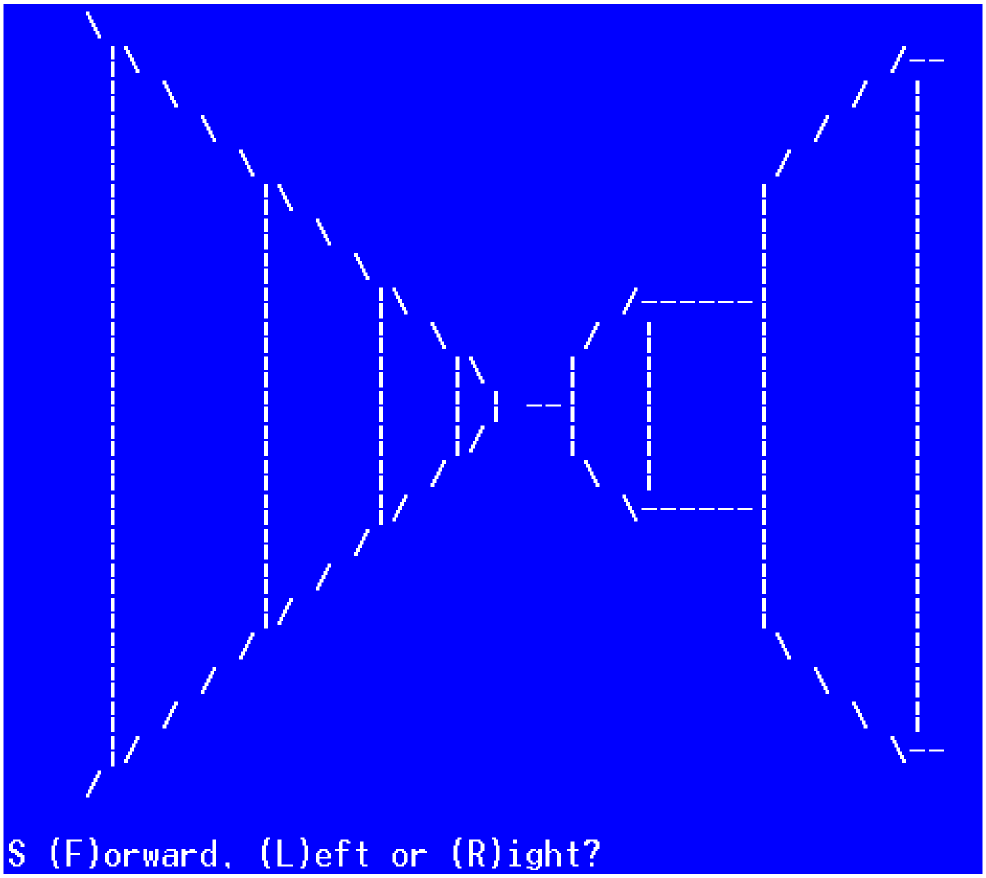
3DMAZE.C

Listing Continued from previous page…

```
    switch (in) {
      case 'e' : xsize = ysize = 5; break;
      case 'm' : xsize = 15; ysize = 11; break;
      case 'h' : xsize = XMAX; ysize = YMAX; break;
      default : printf ("\nInvalid Option. Please choose E, M, or H? ");
        xsize = ysize = 0;
    }
  } while (!xsize);
}

int main (void) {
  clock_t st, et;
  double secs;

  srand (time (NULL));
  setsize ();
  generate ();
  draw3d ();
  st = clock ();
  while (move () && look (0,0) != EXIT);
  if (look (0, 0) == EXIT) {
    et = clock ();
    secs = (double)(et - st) / CLOCKS_PER_SEC;
    snr (1, 2);
    drawmap ();
    printf ("\nYou escaped the Labyrinth in %2.3f seconds!\n", secs);
  }
  exit (0);
}
```



```
S (F)orward, (L)eft or (R)ight?
```