Buttons

Buttons is a fun puzzle game, not to difficult. Whenever you push a button down buttons around it will change. You can only push a button down if it is up. If all the buttons are down at one time you'll lose, unless that is your goal. If you complete all the goals you win the game.

On a personal note, I have a long history with this game. I first seem to recall playing it on the Commodore 64, although if I remember it was actually a written for the PET computer. I remade it on the C64 in BASIC using a sprite pointer that you controlled with the joystick and snazzy 3Dish graphics created with the C64's tile set and color control. This was also the first version that had multiple goals, where as the original only asked you to make an "O" to win. Imagine my disappointment when years later I went back and found it painfully slow. I made a new version in assembly years later, sans the multiple goals, and snazzy graphics.

For an extra challenge, look for the line in the function puzzle() that reads:

`bd = rand () % 0x01FE + 1;`

...and change it to...

`bd = rand () % 0x01FE + 1; goal[lv] = rand () % 0x01FE + 1;`

...and every time you play you'll have completely random goals.

Buttons was written by Joseph Larson

BUTTONS.C	You will need: a C/C++ complier .

```
/* Buttons 2007-Nov-17
 * by Joseph Larson (c) 2008
 */

#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <ctype.h>
#include <conio.h>

#define L(a,b) for (c = a; c < a + 3; c++) printf(gfx[b * !!(bd & 1 << c)], c+1)
#define R(a) printf ("\t");L(a,1);printf ("\n\t");L(a,2);printf ("\n\t");L(a,1)
#define D(a) ".O"[!!(goal[lv] & 1 << a)]
#define NUMGOAL 8

char *gfx[] = { "  ... ", " +===+", " | %d |"};
int bd, lv, won, won_mask,
move[] = {0x001b, 0x0007, 0x0036, 0x0049, 0x00ba, 0x0124, 0x00d8, 0x01c0, 0x01b0}
,goal[] = {0x01ef, 0x0010, 0x0054, 0x01d2, 0x004F, 0x001b, 0x0145, 0x0000};
/* O, ., \, T, L, square, corners, blank */

void show_buttons (void) {
  int c;

  printf ("\n\n");
  R(6); printf ("\tGoal:\n\n\t");
  L(3, 1); printf ("\t %c%c%c\n\t", D(6), D(7), D(8));
  L(3, 2); printf ("\t %c%c%c\n\t", D(3), D(4), D(5));
  L(3, 1); printf ("\t %c%c%c\n\n", D(0), D(1), D(2));
  R(0);
}

void puzzle (void) {
  char in;

  do lv = rand() % NUMGOAL; while (won_mask & 1 << lv);
  bd = rand () % 0x01FE + 1;
  do {
    show_buttons ();
```

Listing continued on next page...

BUTTONS.C	Listing Continued from previous page….

```
    printf ("\n\n\t\tmove ? ");
    do in = getche (); while (in < '0' || in > '9');
    if (in -= '0') {
      if (bd & (1 << (in - 1))) bd ^= move [in - 1];
      else puts ("\nInvalad. That button is not up.");
    }
  } while (bd && bd != goal[lv] && in);
  if (bd == goal[lv]) {
      show_buttons ();
    puts ("\n\n\t\tGoal!!!");
    won++;
    won_mask |= 1 << lv;
  } else if (!bd) {
    show_buttons ();
    puts ("\n\n\tYou have no more moves.");
  }
}

int want_more (void) {
  char yn;

  if (won < NUMGOAL) {
    printf ("\nDo you want to try another one? (y/n)");
    do yn = getche (); while (tolower (yn) != 'y' && tolower (yn) != 'n');
    return tolower (yn) == 'y';
  } else puts ("Congratulations! You've solved all the puzzles!\n\n"
    "You are a master button pusher.\n\nThanks for playing.");
  return 0;
}

int main (void) {
  won_mask = won = 0;
  srand (time (NULL));
  puts ("Buttons\n-------\n"
  "Try to make the field of buttons look like the goal. You can press any\n"
  "button that is up by typing the number on the button, but when you press\n"
  "a button other buttons around it will change. Each button always changes\n"
  "the same buttons around it.\n"
  "Type zero ('0') as your move to quit\n"
  "There are several goals to accomplish. Try to get them all!\n\n"
  "Press ENTER to start...");
  getche ();
  do puzzle (); while (want_more ());
  puts ("Good bye!");
  exit (0);
}
```