

Textul si imaginile din acest document sunt licentiate

Attribution-NonCommercial-NoDerivs  
CC BY-NC-ND



Codul sursa din acest document este licentiat

Public-Domain

Esti liber sa distribui acest document prin orice mijloace consideri (email, publicare pe website / blog, printare, sau orice alt mijloc), atat timp cat nu aduci nici un fel de modificari acestuia. Codul sursa din acest document poate fi utilizat in orice fel de scop, de natura comerciala sau nu, fara nici un fel de limitari.

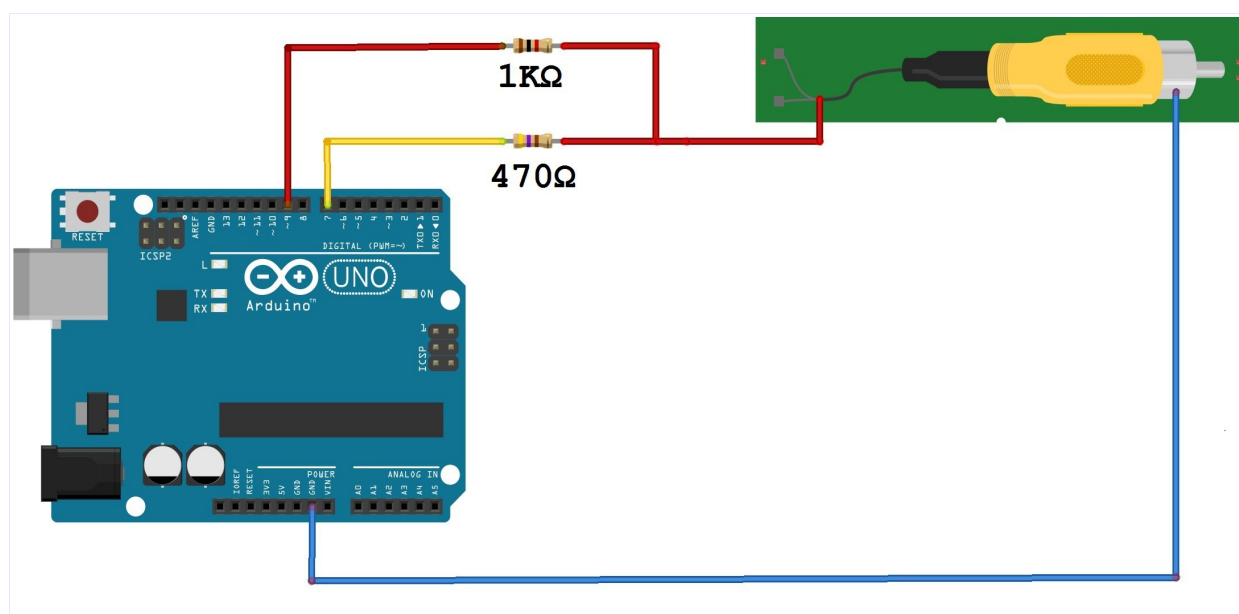
## Arduino si libraria TVOut

In acest tutorial vei descoperi cum se poate programa o placa Arduino pentru a genera semnal video format PAL, pentru televizoarele obisnuite. In prima parte a tutorialului vei instala libraria, iar in cea de-a doua parte vei incarca 2 sketch-uri. Primul program afiseaza pe ecranul televizorului un desen, cateva propozitii cu font-uri diferite, forme geometrice si un cub animat in 3D. Al doilea program afiseaza pe ecran jocul Game of Life.

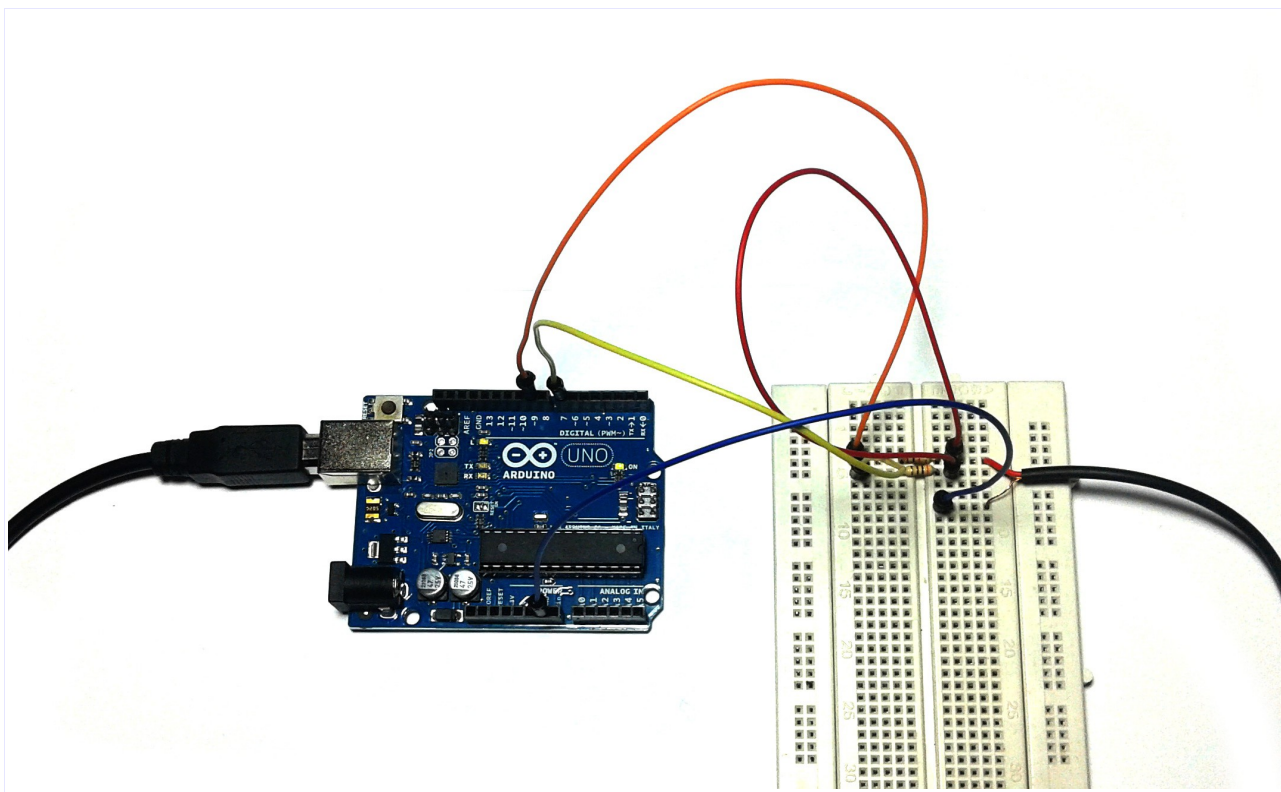
Vei avea nevoie de urmatoarele componente:

- O placa Arduino - <http://www.robofun.ro/arduino>
- Breadboard - <http://www.robofun.ro/breadboard>
- Fire de conexiuni tata - tata  
[http://www.robofun.ro/cabluri/fire\\_conexiune\\_tata\\_tata-110mm](http://www.robofun.ro/cabluri/fire_conexiune_tata_tata-110mm)
- Un rezistor de  $1k\Omega$  - <http://www.robofun.ro/electronice/rezistoare>
- Un rezistor de  $330\Omega$  - <http://www.robofun.ro/electronice/rezistoare>
- Un cablu RCA pentru televizor.

Conecteaza placa Arduino la televizor urmand diagrama de mai jos.



Foloseste imaginea urmatoare ca referinta pentru a conecta corect placa Arduino:



## Libraria TVOut.

Aceasta librerie este capabila de a genera diverse tipuri de semnale. In cazul de fata, te vei concentra doar pe un singur tip, mai exact pe format-ul PAL. Libraria se descarca de la urmatoarea adresa si se instaleaza ca orice alta librerie in Arduino:

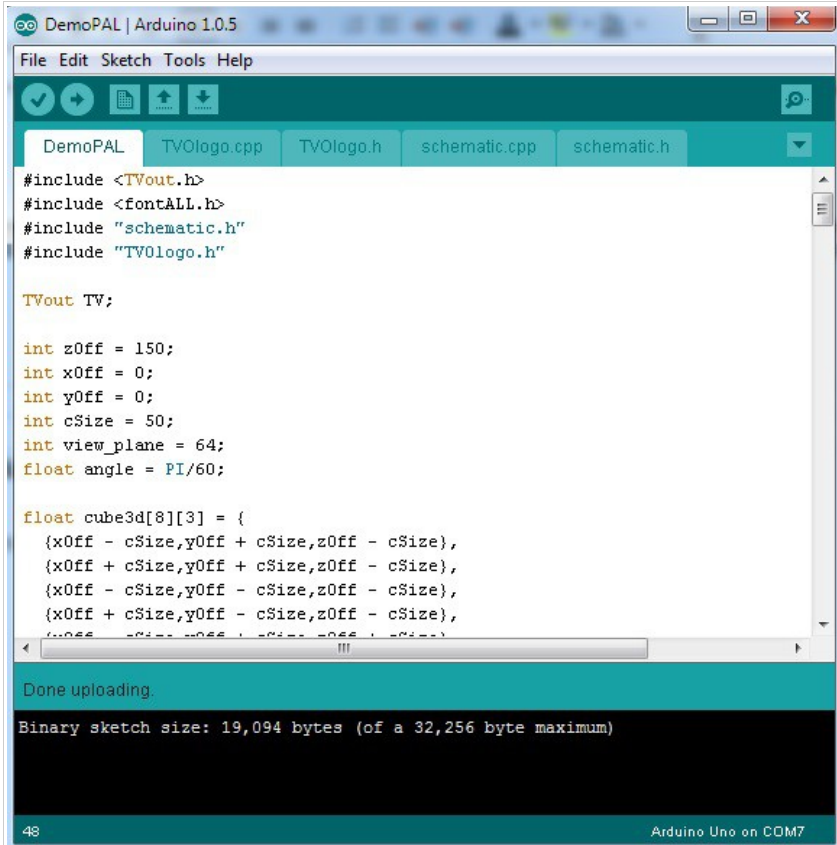
<https://code.google.com/p/arduino-tvout/downloads/list>

Name	Date modified	Type	Size
examples	11/16/2013 6:34 PM	File folder	
spec	11/16/2013 6:34 PM	File folder	
keywords	12/2/2010 9:22 PM	Text Document	1 KB
TVout	11/28/2010 10:56 ...	C++ Source	20 KB
TVout	11/28/2010 10:52 ...	C/C++ Header	6 KB
TVoutPrint	11/28/2010 9:36 PM	C++ Source	8 KB
version history	12/2/2010 10:03 PM	Text Document	5 KB
video_gen	11/25/2010 9:22 PM	C++ Source	11 KB
video_gen	11/11/2010 10:05 ...	C/C++ Header	2 KB

<http://www.robofun.ro/forum>

## Exemplul DemoPAL.

Deschide exemplul DemoPAL din biblioteca TVOut si incarca-l in placa Arduino. Comuta televizorul pe modul A/V sau EXT1. Vei observa cum vor aparea diverse formate de text, forme geometrice si un cub 3D.



```
Arduino IDE - DemoPAL | Arduino 1.0.5
File Edit Sketch Tools Help
[Icons]
DemoPAL TVOutlogo.cpp TVOutlogo.h schematic.cpp schematic.h
#include <TVOut.h>
#include <fontALL.h>
#include "schematic.h"
#include "TVOutlogo.h"

TVout TV;

int zOff = 150;
int xOff = 0;
int yOff = 0;
int cSize = 50;
int view_plane = 64;
float angle = PI/60;

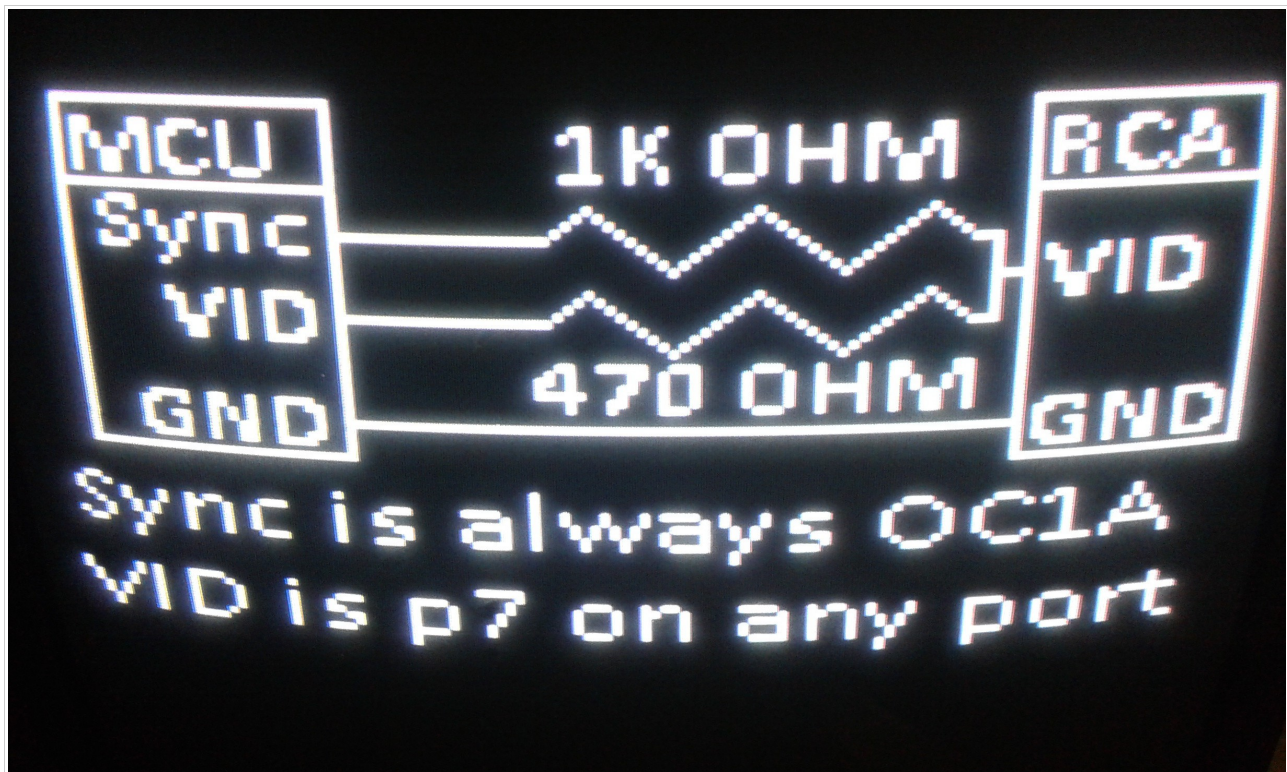
float cube3d[8][3] = {
  {xOff - cSize, yOff + cSize, zOff - cSize},
  {xOff + cSize, yOff + cSize, zOff - cSize},
  {xOff - cSize, yOff - cSize, zOff - cSize},
  {xOff + cSize, yOff - cSize, zOff - cSize},
  {xOff - cSize, yOff + cSize, zOff + cSize},
  {xOff + cSize, yOff + cSize, zOff + cSize},
  {xOff - cSize, yOff - cSize, zOff + cSize},
  {xOff + cSize, yOff - cSize, zOff + cSize}
};

Done uploading.
Binary sketch size: 19,094 bytes (of a 32,256 byte maximum)
48 Arduino Uno on COM7
```

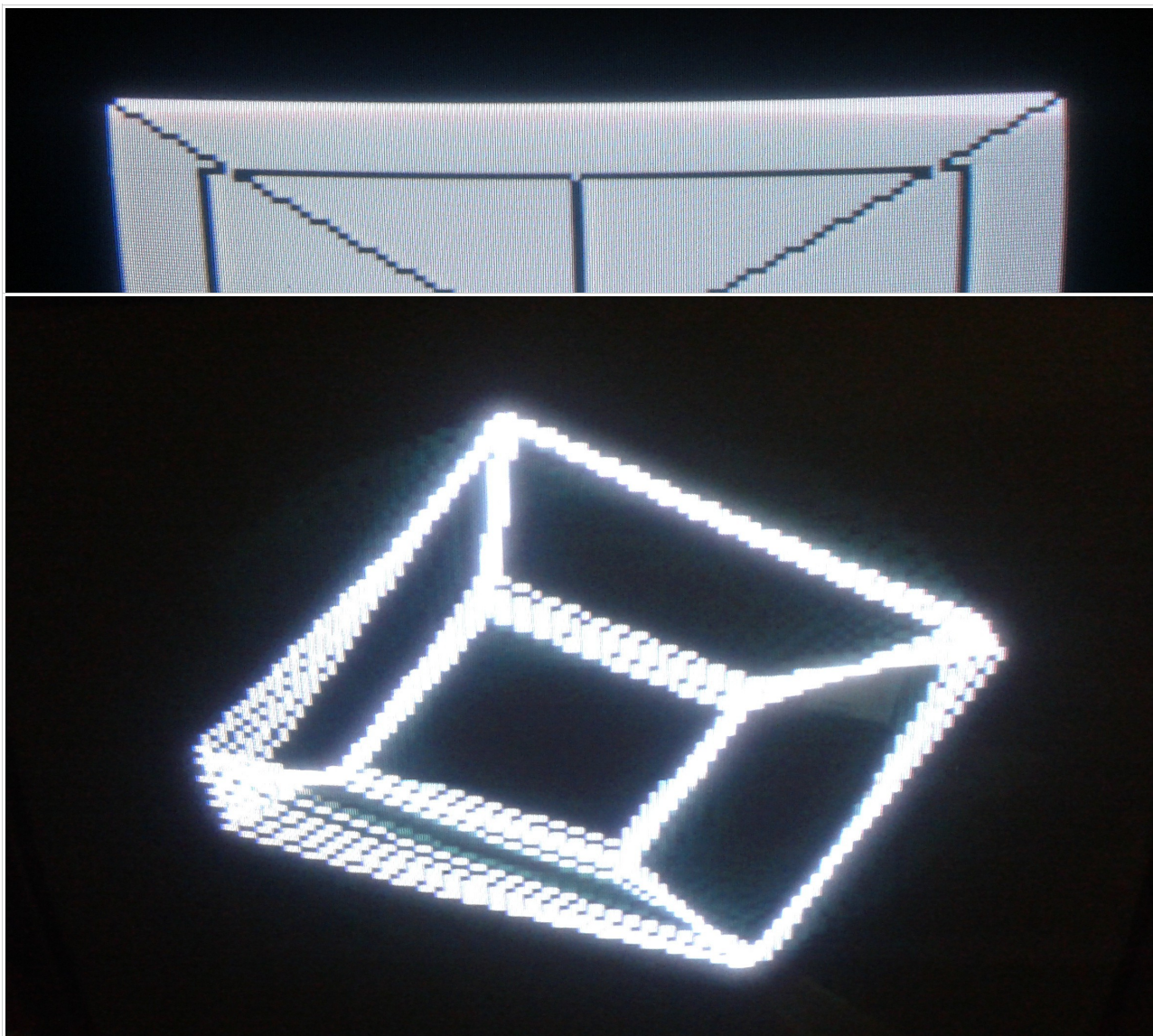
Acest exemplu te poate familiariza cu modul de functionare al placii Arduino. Imaginile urmatoare iti vor arata cum ar trebui sa arate ecranul televizorului tau.







Draw Basic Shapes



## Jocul Game of Life.

Game of Life este un „automat celular” creat de un matematician britanic pe nume John Horton Conway. Jocul se desfășoară pe o matrice bidimensională alcătuită din celule, fiecare luând una din cele 2 stări: on sau off. Jocul simulează evoluția celulelor pe baza a 4 reguli. Pe parcursul evoluției multe celule vor muri, iar altele vor trăi mai multe generații.

Programul de mai jos simulează acest joc pe placa Arduino. Tot ce trebuie să faci este să copiezi codul în Arduino și să încarci sketch-ul.

După ce ai încărcat sketch-ul, la scurt timp îți vor apărea celulele pe ecran.









```

        alive[i] = 0;
    }

    /**
     * Writes output to the console.
     */
    void do_output()
    {
        TV.clear_screen();
        TV.print("Board: ");
        TV.print(boardnum);
        TV.print(" Iteration: ");
        TV.println(iteration);

        for(int i = 0; i < ROWS; i++)
        {

            for(int j = 0; j < COLS; j++)
            {
                // WIDTH, HEIGHT
                if(isAlive(i,j))
                    TV.print("0");
                else
                    TV.print(" ");
            }
            if(i != ROWS -1)
                TV.print("\n");
        }

    }

    /**
     * Randomly fills the grid with alive cells after blanking.
     */
    void random_fill()
    {
        blank_alive();
        randomSeed(analogRead(0));
    }

```

```

// Fill 30% of the cells
int numToFill = (ROWS * COLS) * 30 / 100 ;

for(int r = 0; r < numToFill; r ++)
{
    int row = rand() % ROWS;
    int col = rand() % COLS;

    setAlive(row,col);
}

/**
 * Returns the index of the row below the current one.
 */
int rowBelow(int row)
{
    return (row + 1 < ROWS) ? row + 1 : 0;
}

/**
 * Returns the index of the row above the given one
 */
int rowAbove(int row)
{
    return (row > 0) ? row - 1 : ROWS - 1;
}

/** Returns the index of the col to the right of this one */
int colRight(int col)
{
    return (col + 1 < COLS) ? col + 1 : 0;
}

/** Returns the index of the col to the left of this one */
int colLeft(int col)
{
    return (col > 0) ? col - 1 : COLS - 1;
}

/** true if the cell to the left is alive*/
bool left(int row, int col)
{
    col = colLeft(col);
    return isAlive(row,col);
}

/** true if the cell to the right is alive*/

```



```

bool right(int row, int col)
{
    col = colRight(col);
    return isAlive(row,col);
}

/** true if the cell above is alive*/
bool above(int row, int col)
{
    row = rowAbove(row);
    return isAlive(row,col);
}

/** true if the cell below is alive*/
bool below(int row, int col)
{
    row = rowBelow(row);
    return isAlive(row,col);
}

/** true if the cell NE is alive*/
bool aboveright(int row, int col)
{
    row = rowAbove(row);
    col = colRight(col);
    return isAlive(row,col);
}

/** true if the cell SE is alive*/
bool belowright(int row, int col)
{
    row = rowBelow(row);
    col = colRight(col);
    return isAlive(row,col);
}

/** true if the cell NW is alive*/
bool aboveleft(int row, int col)
{
    row = rowAbove(row);
    col = colLeft(col);
    return isAlive(row,col);
}

/** true if the cell SW is alive*/
bool belowleft(int row, int col)
{
    row = rowBelow(row);

```

```
    col = colLeft(col);  
    return isAlive(row,col);  
}  
  
/**Returns the number of living cells sorrounding this one.*/  
int numberAround(int row, int col)  
{  
    int around = 0;  
    if(left(row,col))  
        around++;  
  
    if(right(row,col))  
        around++;  
  
    if(above(row,col))  
        around++;  
  
    if(below(row,col))  
        around++;  
  
    if(aboveright(row,col))  
        around++;  
}
```

```

    if(aboveleft(row,col))
        around++;

    if(belowright(row,col))
        around++;

    if(belowleft(row,col))
        around++;

    return around;
}

/**
 * Moves all of the cells
 */
void move()
{
    uint32_t nextRows[ROWS] = {
        0,0,0,0,0,0,0,0,0,0,0,0,0,0,0    };

    for(int i = 0; i < ROWS; i++)
    {
        for(int j = 0; j < COLS; j++)
        {
            int na = numberAround(i,j);
            if((na == 2 && isAlive(i,j)) || na == 3)
                nextRows[i] |= 1 << j;
        }
    }

    for(int i = 0; i < ROWS; i++)
        alive[i] = nextRows[i];
}

void setup()
{
    TV.begin(NTSC,120,96);
    TV.select_font(font4x6);
}

void loop() {
    boardnum++;
}

```



```
TV.println("Conways game of life for Arduino, Copyright 2012
Joseph Lewis <joeHms22@gmail.com>");
TV.delay(2000);

random_fill();

TV.print("Doing iterations");

for(iteration = 0; iteration < 50; iteration++)
{
    do_output();
    move();
    TV.delay(500);
}
}
```