```
/*
 * snake3.c
 *
 * Created: 2015-05-15 17:52:56
 *  Author: digpi14
 */


#include <avr/io.h>
#include <stdlib.h>
#include <stdio.h>
#include <util/delay.h>
#include <math.h>

int snake [2][15];
int apple [2];
char length;
char direction;
char knappar;


int main(void)
{
    config();
    startGame();
}

void startGame()
{
    resetLCD();
    createSnake();
    createApple();
    createBoard();
    pause();
    while(1)
    {
        moveSnake();
        checkCrash();
        drawSnake();;
        checkButton();
        _delay_ms(150);
        LED_OFF();
    }
}

void checkButton()
{
    knappar = PINA & 0b00000111;
    switch (knappar)
    {
        case 0:
        break;
        case 2:
        direction = (direction+1)%4;
        _delay_ms(80);
        break;
        case 1: direction = (direction +3)%4;
        _delay_ms(80);
        break;
        case 4:
        pause();
    }

}

void checkCrash()
{
    if(snake[0][length] == 0 || snake[0][length] == 63 || snake[1][length] == 0 || snake[1][length] == 63)
    {
        crash();
    }
    for(int i = 0; i< length; i++)
    if (snake[0][length] == snake[0][i] && snake[1][length] == snake[1][i])
```

```
    {
        crash();
    }
}

void crash()
{
    while(1)
    {
        knappar = PINA & 0b00000111;
        if(knappar == 4)
        {
            startGame();
        }
    }
}

void moveSnake()
{
    erasePixel(snake[0][0],snake[1][0]);
    int tempX;
    int tempY;
    switch (direction)
    {
        case 0:
        tempX = snake[0][length];
        tempY = snake[1][length]+1;
        for(int i = 0; i<length; i++)
        {
            snake[0][i] = snake[0][i+1];
            snake[1][i] = snake[1][i+1];
        }
        snake[0][length] = tempX;
        snake[1][length] = tempY;
        break;
        case 1:
        tempX = snake[0][length]+1;
        tempY = snake[1][length];
        for(int i = 0; i<length; i++)
        {
            snake[0][i] = snake[0][i+1];
            snake[1][i] = snake[1][i+1];
        }
        snake[0][length] = tempX;
        snake[1][length] = tempY;
        break;
        case 2:
        tempX = snake[0][length];
        tempY = snake[1][length]-1;
        for(int i = 0; i<length; i++)
        {
            snake[0][i] = snake[0][i+1];
            snake[1][i] = snake[1][i+1];
        }
        snake[0][length] = tempX;
        snake[1][length] = tempY;
        break;
        case 3:
        tempX = snake[0][length]-1;
        tempY = snake[1][length];
        for(int i = 0; i<length; i++)
        {
            snake[0][i] = snake[0][i+1];
            snake[1][i] = snake[1][i+1];
        }
        snake[0][length] = tempX;
        snake[1][length] = tempY;
        break;
    }
    if(snake[0][length] == apple[0] && snake[1][length] == apple[1])
    {
        grow();
    }
```

```
}

void grow()
{
    int tempX = snake[0][0];
    int tempY = snake[1][0];
    _delay_ms(60);
    checkButton();
    moveSnake();
    for(int i = length; i>=0; i--)
    {
        snake[0][i+1] = snake[0][i] ;
        snake[1][i+1] = snake[1][i] ;
    }
    snake[0][0] = tempX;
    snake[1][0] = tempY;
    length++;
    createApple();
    drawScore(length-2);
    LED_ON();
}

void createBoard()
{
    for (int i = 0; i<64; i++)
    {
        setPixel(0,i);
        setPixel(63,i);
        setPixel(i,0);
        setPixel(i,63);
    }
    drawS(10,77);
    drawC(10,85);
    drawO(10,93);
    drawR(10,101);
    drawE(10,109);
    draw0(30,90);
    draw0(30,96);
}

void createApple()
{
    apple[0] = rand()%61 +1;
    apple[1] = rand()%61 +1;
    for (int i = 0; i<=length; i++)
    {
        if(apple[0] == snake[0][i] && apple[1] == snake[1][i])
        createApple();
    }
    setPixel(apple[0],apple[1]);
}

void createSnake()
{
    direction = 0;
    length = 2;
    snake[0][0] = 30;
    snake[1][0] = 15;
    snake[0][1] = 30;
    snake[1][1] = 16;
    snake[0][2] = 30;
    snake[1][2] = 17;
    drawSnake();
}

void drawSnake()
{
    for (int i = 0; i<=length; i++)
    {
        setPixel(snake[0][i], snake[1][i]);
    }
}
```

```
void pause()
{
    _delay_ms(400);
    while(1)
    {
        knappar = PINA & 0b00000111;
        if(knappar == 4)
        {
            PINA = PINA & 0b11111000;
            _delay_ms(200);
            break;
        }
    }
}

void LED_ON()
{
    PORTD |= 0b01000000;
}

void LED_OFF()
{
    PORTD &= 0b10111111;
}



//Konfiguerar portarna, slår på skärmen samt släcker alla pixlar
void config()
{
    DDRA = 0x00;
    DDRB = 0xFF;
    DDRD = 0xFF;

    PORTB = 0x3F;
    res(1);
    rs(0);
    rw(0);
    cs(3);
    switchE();
    cs(0);
    resetLCD();
}

//En snabb metod för att rensa hela displayen
void resetLCD()
{
    for(char a = 0; a<=7; a++)
    {
        PORTB = 0xB8 + a;
        PORTD = 0x26;
        switchE();
        PORTD = 0x24;

        PORTB = 0b01000000;
        PORTD = 0x26;
        switchE();
        PORTD = 0x24;

        PORTB = 0x00;
        PORTD = 0x36;

        for(char b = 0; b<=63; b++)
        {
            switchE();
        }
    }
    for(char a = 0; a<=7; a++)
    {
        PORTB = 0xB8 + a;
        PORTD = 0x25;
        switchE();
        PORTD = 0x24;
```

```
        PORTB = 0b01000000;
        PORTD = 0x25;
        switchE();
        PORTD = 0x24;

        PORTB = 0x00;
        PORTD = 0x35;

        for(char b = 0; b<=63; b++)
        {
            switchE();
        }
    }

}



//Metoder för att tända respektiva släcka en pixel
void setPixel(int x, int y)
{
    int temp = getChar(x,y);
    int xChar = toChar(x%8);
    if(y>63)
    {
        cs(2);
    }
    if(y<64)
    {
        cs(1);
    }
    setX(x);
    setY(y);
    PORTB = temp | xChar;
    rs(1);
    rw(0);
    switchE();
    cs(0);

}
void erasePixel(int x, int y)
{
    int temp = getChar(x,y);
    int xChar = toChar(x%8);
    int xReverse = 0b11111111 - xChar;
    if(y>63)
    {
        cs(2);
    }
    if(y<64)
    {
        cs(1);
    }
    setX(x);
    setY(y);
    PORTB = temp & xReverse;
    rs(1);
    rw(0);
    switchE();
    cs(0);

}



//Hjälpmetoder för att skicka data till och från skärmen
int getChar(int x, int y)
{
    if(y>63)
    {
        cs(2);
```

```
    }
    if(y<64)
    {
        cs(1);
    }
    setY(y);
    setX(x);
    e(1);
    rs(1);
    rw(1);
    DDRB = 0x00;
    e(0);
    e(1);
    int temp = PINB;
    e(0);
    cs(0);
    DDRB = 0xFF;
    return temp;
}
int toChar(int x)
{
    switch (x)
    {
        case 0:
        return 1;
        case 1:
        return 2;
        case 2:
        return 4;
        case 3:
        return 8;
        case 4:
        return 16;
        case 5:
        return 32;
        case 6:
        return 64;
        case 7:
        return 128;

    }
}
void setY(int y)
{
    int newY = y;
    if(y>63)
    {
        newY = y-64;
    }
    rs(0);
    rw(0);
    PORTB = 0b01000000 + newY;
    switchE();
}
void setX(int x)
{
    rs(0);
    rw(0);
    int xPage = (x -(x%8))/8;
    PORTB = 0b10111000 + xPage;
    switchE();
}



//Hjälpmetoder för att hantera D-Porten
void e(char a)
{
    switch (a)
    {
        case 0:
        PORTD = PORTD & 0b11011111;
```

```
        break;
        case 1:
        PORTD = PORTD | 0b00100000;
        break;
    }

}
void res(char a)
{
    switch (a)
    {
        case 0:
        PORTD = PORTD & 0b11111011;
        break;
        case 1:
        PORTD = PORTD | 0b00000100;
        break;
    }

}
void rs(char a)
{
    switch (a)
    {
        case 0:
        PORTD = PORTD & 0b11101111;
        break;
        case 1:
        PORTD = PORTD | 0b00010000;
        break;
    }
}
void rw(char a)
{
    switch (a)
    {
        case 0:
        PORTD = PORTD & 0b11110111;
        break;
        case 1:
        PORTD = PORTD | 0b00001000;
        break;
    }
}
void cs(char a)
{
    switch (a)
    {
    case 0:
    PORTD = PORTD & 0b11111100;
    break;

    case 2:
    PORTD = PORTD & 0b11111100;
    PORTD = PORTD | 0b00000010;
    break;

    case 1:
    PORTD = PORTD & 0b11111100;
    PORTD = PORTD | 0b00000001;
    break;

    case 3:
    PORTD = PORTD | 0b00000011;
    break;
    }
}
void switchE()
{
    e(1);
    e(0);
}
```

```
// Små klasser för att skriva ut bokstäver och poäng på skärmen
void drawS(int x, int y)
{
    setPixel(x, y);
    setPixel(x, y+1);
    setPixel(x, y+2);
    setPixel(x, y+3);
    setPixel(x+1, y);
    setPixel(x+2, y+1);
    setPixel(x+2, y+2);
    setPixel(x+2, y+3);
    setPixel(x+2, y);
    setPixel(x+3, y+3);
    setPixel(x+4, y);
    setPixel(x+4, y+1);
    setPixel(x+4, y+2);
    setPixel(x+4, y+3);
}
void drawC(int x, int y)
{
    setPixel(x,y+1);
    setPixel(x,y);
    setPixel(x,y+2);
    setPixel(x,y+3);
    setPixel(x+1,y);
    setPixel(x+2,y);
    setPixel(x+3,y);
    setPixel(x+4,y+1);
    setPixel(x+4,y+2);
    setPixel(x+4,y+3);
    setPixel(x+4,y);
}
void drawO(int x, int y)
{
    setPixel(x,y+1);
    setPixel(x,y+2);
    setPixel(x,y+3);
    setPixel(x,y);
    setPixel(x+1,y);
    setPixel(x+1,y+3);
    setPixel(x+2,y);
    setPixel(x+2,y+3);
    setPixel(x+3,y);
    setPixel(x+3,y+3);
    setPixel(x+4,y+1);
    setPixel(x+4,y+3);
    setPixel(x+4,y);
    setPixel(x+4,y+2);
}
void drawR(int x, int y)
{
    setPixel(x,y);
    setPixel(x,y+1);
    setPixel(x,y+2);
    setPixel(x+1,y+3);
    setPixel(x+1,y);
    setPixel(x+2,y);
    setPixel(x+2,y+1);
    setPixel(x+2,y+2);
    setPixel(x+3,y);
    setPixel(x+3,y+3);
    setPixel(x+4,y);
    setPixel(x+4,y+3);
}
void drawE(int x, int y)
{
    setPixel(x,y);
    setPixel(x,y+1);
    setPixel(x,y+2);
    setPixel(x,y+3);
    setPixel(x+1,y);
    setPixel(x+2,y);
```

```
    setPixel(x+2,y+1);
    setPixel(x+2,y+2);
    setPixel(x+2,y+3);
    setPixel(x+3,y);
    setPixel(x+4,y);
    setPixel(x+4,y+1);
    setPixel(x+4,y+2);
    setPixel(x+4,y+3);
}
void drawNBR(int x, int y, int NBR)
{
    switch (NBR)
    {
        case 0:
        draw0(x,y);
        break;
        case 1:
        draw1(x,y);
        break;
        case 2:
        draw2(x,y);
        break;
        case 3:
        draw3(x,y);
        break;
        case 4:
        draw4(x,y);
        break;
        case 5:
        draw5(x,y);
        break;
        case 6:
        draw6(x,y);
        break;
        case 7:
        draw7(x,y);
        break;
        case 8:
        draw8(x,y);
        break;
        case 9:
        draw9(x,y);
        break;
    }
}
void draw0(int x, int y)
{
    setPixel(x,y);
    setPixel(x,y+1);
    setPixel(x,y+2);
    setPixel(x,y+3);
    setPixel(x+1,y+3);
    setPixel(x+1,y);
    setPixel(x+2,y+3);
    setPixel(x+2,y);
    setPixel(x+3,y+3);
    setPixel(x+3,y);
    setPixel(x+4,y);
    setPixel(x+4,y+1);
    setPixel(x+4,y+2);
    setPixel(x+4,y+3);
}
void draw1(int x, int y)
{
    setPixel(x, y+1);
    setPixel(x, y+2);
    setPixel(x+1, y+2);
    setPixel(x+2, y+2);
    setPixel(x+3, y+2);
    setPixel(x+4, y+2);
}
void draw2(int x, int y)
{
```

```
    setPixel(x,y);
    setPixel(x,y+1);
    setPixel(x,y+2);
    setPixel(x,y+3);
    setPixel(x+1,y+3);
    setPixel(x+2,y);
    setPixel(x+2,y+1);
    setPixel(x+2,y+2);
    setPixel(x+2,y+3);
    setPixel(x+3,y);
    setPixel(x+4,y);
    setPixel(x+4,y+1);
    setPixel(x+4,y+2);
    setPixel(x+4,y+3);
}
void draw3(int x, int y)
{
    setPixel(x,y);
    setPixel(x,y+1);
    setPixel(x,y+2);
    setPixel(x,y+3);
    setPixel(x+1,y+3);
    setPixel(x+2,y);
    setPixel(x+2,y+1);
    setPixel(x+2,y+2);
    setPixel(x+2,y+3);
    setPixel(x+3,y+3);
    setPixel(x+4,y);
    setPixel(x+4,y+1);
    setPixel(x+4,y+2);
    setPixel(x+4,y+3);
}
void draw4(int x, int y)
{
    setPixel(x,y);
    setPixel(x,y+3);
    setPixel(x+1,y);
    setPixel(x+1,y+3);
    setPixel(x+2,y);
    setPixel(x+2,y+1);
    setPixel(x+2,y+2);
    setPixel(x+2,y+3);
    setPixel(x+3,y+3);
    setPixel(x+4,y+3);
}
void draw5(int x, int y)
{
    setPixel(x,y);
    setPixel(x,y+1);
    setPixel(x,y+2);
    setPixel(x,y+3);
    setPixel(x+1,y);
    setPixel(x+2,y);
    setPixel(x+2,y+1);
    setPixel(x+2,y+2);
    setPixel(x+2,y+3);
    setPixel(x+3,y+3);
    setPixel(x+4,y);
    setPixel(x+4,y+1);
    setPixel(x+4,y+2);
    setPixel(x+4,y+3);
}
void draw6(int x, int y)
{
    setPixel(x,y);
    setPixel(x,y+1);
    setPixel(x,y+2);
    setPixel(x,y+3);
    setPixel(x+1,y);
    setPixel(x+2,y);
    setPixel(x+2,y+1);
    setPixel(x+2,y+2);
    setPixel(x+2,y+3);
```

```
    setPixel(x+3,y);
    setPixel(x+3,y+3);
    setPixel(x+4,y);
    setPixel(x+4,y+1);
    setPixel(x+4,y+2);
    setPixel(x+4,y+3);

}
void draw7(int x, int y)
{
    setPixel(x,y);
    setPixel(x,y+1);
    setPixel(x,y+2);
    setPixel(x,y+3);
    setPixel(x+1,y+3);
    setPixel(x+2,y+3);
    setPixel(x+3,y+3);
    setPixel(x+4,y+3);

}
void draw8(int x, int y)
{
    setPixel(x,y);
    setPixel(x,y+1);
    setPixel(x,y+2);
    setPixel(x,y+3);
    setPixel(x+1,y);
    setPixel(x+1,y+3);
    setPixel(x+2,y);
    setPixel(x+2,y+1);
    setPixel(x+2,y+2);
    setPixel(x+2,y+3);
    setPixel(x+3,y);
    setPixel(x+3,y+3);
    setPixel(x+4,y);
    setPixel(x+4,y+1);
    setPixel(x+4,y+2);
    setPixel(x+4,y+3);

}
void draw9(int x, int y)
{
    setPixel(x,y);
    setPixel(x,y+1);
    setPixel(x,y+2);
    setPixel(x,y+3);
    setPixel(x+1,y);
    setPixel(x+1,y+3);
    setPixel(x+2,y);
    setPixel(x+2,y+1);
    setPixel(x+2,y+2);
    setPixel(x+2,y+3);
    setPixel(x+3,y+3);
    setPixel(x+4,y+3);
}
void drawScore(int score)
{
    for (int i = 30; i<40; i++)
    {
        for (int j = 80; j<100;j++)
        erasePixel(i,j);
    }
    if (score<10)
    {
        drawNBR(30,90,0);
    }
    if (score>9)
    {
        drawNBR(30,90,1);
    }
    drawNBR(30,96,score%10);
}
```