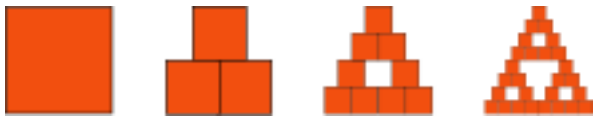
1. Week 9: Sierpinski and SDL

1.1 Sierpinski Squares

See also: en.wikipedia.org/wiki/Sierpinski_triangle

The Sierpinski triangle has the overall shape of an equilateral triangle, recursively subdivded into four smaller triangles :

However, we can approximate it by recursively drawing a square as three smaller squares, as show below :



The recursion should terminate when the squares are too small to draw with any more detail (e.g. one pixel, or one character in size).

1.2 Sierpinski Carpet

en.wikipedia.org/wiki/Sierpinski_carpet

The square is cut into 9 congruent subsquares in a 3-by-3 grid, and the central subsquare is removed. The same procedure is then applied recursively to the remaining 8 subsquares, ad infinitum.

www http://www.evilmadscientist.com/2008/sierpinski-cookies/

Exercise 1.1 Write a program that:
- Draws the Sierpinski carpet, recursively, using plain text.
- Draws the Sierpinski triangle, recursively, using plain text.

■

1.3 SDL - Intro

Many programming languages have no inherent graphics capabilities. To get windows to appear on the screen, or to draw lines and shapes, you need to make use of an external library. Here we use SDL[1], a cross-platform library providing the user with (amongst other things) such graphical capabilities.

www `https://www.libsdl.org/`

The use of SDL is, unsurprisingly, non-trival, so some simple wrapper files have been created (`neillsdl2.c` and `neillsdl2.h`). These give you some simple functions to initialise a window, draw rectangles, wait for the user to press a key etc. and are somewhat similar to `neillncurses.*`.

An example program using this functionality is provided in a file `blocks.c`, shown in Figure 1.1.

This program initialises a window, then sits in a loop, drawing randomly positioned and coloured squares, until the user presses the mouse or a key.

Using the `makefile` provided, compile and run this program.

SDL is already installed on lab machines. At home, if you're using a ubuntu-style linux machine, use: `sudo apt-get install libsdl2-dev`.

Exercise 1.2 Write a program that:
- Draws the Sierpinski carpet, recursively, using SDL.
- Draws the Sierpinski triangle, recursively, using SDL.

■

[1] actually, we are using the most recent version SDL2, which is installed on all the lab machines

```c
#include <stdio.h>
#include <stdlib.h>
#include "neillsdl2.h"

#define RECTSIZE 20
#define MILLISECONDDELAY 10

int main(void)
{

    SDL_Simplewin sw;
    SDL_Rect rectangle;
    rectangle.w = RECTSIZE;
    rectangle.h = RECTSIZE;

    Neill_SDL_Init(&sw);

    do{

        /* Sleep for a short time */
        SDL_Delay(MILLISECONDDELAY);

        /* Choose a random colour, a mixture of red, green and blue. */
        Neill_SDL_SetDrawColour(&sw,
                                rand()%SDL_8BITCOLOUR, rand()%SDL_8BITCOLOUR,
                                rand()%SDL_8BITCOLOUR);

        /* Filled Rectangle, fixed size, random position */
        rectangle.x = rand()%(WWIDTH-RECTSIZE);
        rectangle.y = rand()%(WHEIGHT-RECTSIZE);
        SDL_RenderFillRect(sw.renderer, &rectangle);

        /* Unfilled Rectangle, fixed size, random position */
        rectangle.x = rand()%(WWIDTH-RECTSIZE);
        rectangle.y = rand()%(WHEIGHT-RECTSIZE);
        SDL_RenderDrawRect(sw.renderer, &rectangle);

        /* Update window - no graphics appear on some devices until this is finished */
        SDL_RenderPresent(sw.renderer);
        SDL_UpdateWindowSurface(sw.win);

        /* Has anyone pressed ESC or killed the SDL window ?
           Must be called frequently - it's the only way of escaping */
        Neill_SDL_Events(&sw);

    }while(!sw.finished);


    /* Clear up graphics subsystems */
    atexit(SDL_Quit);

    return 0;

}
```


Figure 1.1: The program *blocks.c* used to demonstrate some SDL2 wrapper functions.