```c
/**
 * Copyright (c) 2020 Raspberry Pi (Trading) Ltd.
 *
 * SPDX-License-Identifier: BSD-3-Clause
 */

#include <stdio.h>
#include <stdlib.h>

#include "pico/stdlib.h"
#include "hardware/pio.h"
#include "hardware/clocks.h"
#include "ws2812.pio.h"

#define IS_RGBW true
#define NUM_PIXELS 150

#ifdef PICO_DEFAULT_WS2812_PIN
#define WS2812_PIN PICO_DEFAULT_WS2812_PIN
#else
// default to pin 2 if the board doesn't have a default WS2812 pin defined
#define WS2812_PIN 12
#endif
#define WS2812_POWER_PIN 11

static inline void put_pixel(uint32_t pixel_grb) {
    pio_sm_put_blocking(pio0, 0, pixel_grb << 8u); ㉙ Left shift the number by 8 bits to TX FIFO of Sy
}

static inline uint32_t urgb_u32(uint8_t r, uint8_t g, uint8_t b) {
    return
            ((uint32_t) (r) << 8) |
            ((uint32_t) (g) << 16) |
            (uint32_t) (b);
}
// void turn_on_LED_power(){
//     const uint LED_PIN = WS2812_POWER_PIN;
//     gpio_init(LED_PIN);
//     gpio_set_dir(LED_PIN,GPIO_OUT);
//     gpio_put(LED_PIN,1);
// }

void pattern_snakes(uint len, uint t) {
    for (uint i = 0; i < len; ++i) {
        uint x = (i + (t >> 1)) % 64;
        if (x < 10)
            put_pixel(urgb_u32(0xff, 0, 0));
        else if (x >= 15 && x < 25)
            put_pixel(urgb_u32(0, 0xff, 0));
        else if (x >= 30 && x < 40)
            put_pixel(urgb_u32(0, 0, 0xff));
        else
            put_pixel(0);
    }
}

void pattern_random(uint len, uint t) {
    if (t % 8)
        return;            ㉗ Return if t isn't a multiple of 8.
    for (int i = 0; i < len; ++i)
        put_pixel(rand()); ㉘ Set the pixel randomly
                              if t is a multiple of 8
}

void pattern_sparkle(uint len, uint t) {
    if (t % 8)
        return;
    for (int i = 0; i < len; ++i)
        put_pixel(rand() % 16 ? 0 : 0xffffffff);
}

void pattern_greys(uint len, uint t) {
    int max = 100; // let's not draw too much current!
    t %= max;
    for (int i = 0; i < len; ++i) {
        put_pixel(t * 0x10101);
        if (++t >= max) t = 0;
    }
}

typedef void (*pattern)(uint len, uint t);
const struct {
    pattern pat;
    const char *name;
} pattern_table[] = {
        {pattern_snakes,  "Snakes!"},
        {pattern_random,  "Random data"},
        {pattern_sparkle, "Sparkles"},
        {pattern_greys,   "Greys"},
};

int main() {
    //set_sys_clock_48();
    stdio_init_all(); ① Initialize std
    printf("WS2812 Smoke Test, using pin %d", WS2812_PIN); ② Print the operating PIN number
    //turn_on_LED_power();

    // todo get free sm
    PIO pio = pio0; ③ Set pio0 as PIO
    int sm = 0;      ④ Initialize the state machine
    uint offset = pio_add_program(pio, &ws2812_program); ⑤ Get the current state of SM

    ws2812_program_init(pio, sm, offset, WS2812_PIN, 800000, IS_RGBW); ⑥ Initialize the program
                                                                         with pio instance, SM, current
    int t = 0; ㉑ Initialize t.                                          state, operating pin, frequency
    while (1) {                                                          and choice of RGB
        int pat = rand() % count_of(pattern_table); ㉒ Initialize pat indicating LED pattern randomly
        int dir = (rand() >> 30) & 1 ? 1 : -1; ㉓ Initialize dir to 1/-1 randomly
        puts(pattern_table[pat].name);  ㉔ Print pattern name
        puts(dir == 1 ? "(forward)" : "(backward)"); ㉕ Print the direction
        for (int i = 0; i < 1000; ++i) {
            pattern_table[pat].pat(NUM_PIXELS, t); ㉖ Set the LED by chose pattern
            sleep_ms(10); ㉚ sleep 10 ms
            t += dir;     ㉛ Add t by dir
        }
    }
}

```