```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 2
#endif

static inline void put_pixel(uint32_t pixel_grb) {
    pio_sm_put_blocking(pio0, 0, pixel_grb << 8u);
}

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 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;
    for (int i = 0; i < len; ++i)
        put_pixel(rand());
}

void pattern_sparkle(uint len, uint t) {
    if (t % 8)
        return;
```

```
60      for (int i = 0; i < len; ++i)
61          put_pixel(rand() % 16 ? 0 : 0xffffffff);
62  }
63
64  void pattern_greys(uint len, uint t) {
65      int max = 100; // let's not draw too much current!
66      t %= max;
67      for (int i = 0; i < len; ++i) {
68          put_pixel(t * 0x10101);
69          if (++t >= max) t = 0;
70      }
71  }
72
73  typedef void (*pattern)(uint len, uint t);
74  const struct {
75      pattern pat;
76      const char *name;
77  } pattern_table[] = {
78          {pattern_snakes,  "Snakes!"},
79          {pattern_random,  "Random data"},
80          {pattern_sparkle, "Sparkles"},
81          {pattern_greys,   "Greys"},
82  };
83
84  int main() {
85      //set_sys_clock_48();
86      stdio_init_all();
87      printf("WS2812 Smoke Test, using pin %d", WS2812_PIN);
88
89      // todo get free sm
90      PIO pio = pio0;
91      int sm = 0;
92      uint offset = pio_add_program(pio, &ws2812_program);
93
94      ws2812_program_init(pio, sm, offset, WS2812_PIN, 800000, IS_RGBW);
95
96      int t = 0;
97      while (1) {
98          int pat = rand() % count_of(pattern_table);
99          int dir = (rand() >> 30) & 1 ? 1 : -1;
100         puts(pattern_table[pat].name);
101         puts(dir == 1 ? "(forward)" : "(backward)");
102         for (int i = 0; i < 1000; ++i) {
103             pattern_table[pat].pat(NUM_PIXELS, t);
104             sleep_ms(10);
105             t += dir;
106         }
107     }
108 }
109
```

① initialize all standard I/O libraries

②

③ Choose which PIO instance to use

④ define integer sm

⑤ load assembled program into PIO's instruction memory

⑥ instantiate an instance of the Neopixel driver program

⑦ define time integer t. →(go to ws2812.pio.h)

⑱ using an infinite while loop

⑲ Create a random pattern and direction based on it for 1 or -1

⑳ put patterns' name and direction name based on value of "dir"

㉑ give neopixel light intensities at "t" and sleep for 10 ms.

㉒ update "t" based on "dir" (1 or -1)