```python
# 10 Mesurador de Distància amb PIR, LED i Multiprocessing
# Utilitzar tres processos paral·lels per activar LEDs segons la distància detectada per l'HC
# SR04 i el moviment detectat pel PIR, on cada procés controla una resposta específica.
# Solució en codi:
# Solució en codi Python
import RPi.GPIO as GPIO
import time
from multiprocessing import Process

GPIO.setmode(GPIO.BCM)

PIR_PIN = 4
TRIG = 23
ECHO = 24
LED_PINS = [17, 27, 22]

GPIO.setup(TRIG, GPIO.OUT)
GPIO.setup(ECHO, GPIO.IN)
GPIO.setup(PIR_PIN, GPIO.IN)

for pin in LED_PINS:
    GPIO.setup(pin, GPIO.OUT)

def distance():
    GPIO.output(TRIG, True)
    time.sleep(0.00001)
    GPIO.output(TRIG, False)
    start, stop = time.time(), time.time()
    while GPIO.input(ECHO) == 0:
        start = time.time()

    while GPIO.input(ECHO) == 1:
        stop = time.time()
    return (stop - start) * 34300 / 2

def control_leds_by_distance(led_pin, min_dist, max_dist):
    while True:
        dist = distance()
        GPIO.output(led_pin, min_dist <= dist < max_dist)
        time.sleep(0.5)

def control_leds_by_pir():
    while True:
        if GPIO.input(PIR_PIN):
            for pin in LED_PINS:
                GPIO.output(pin, True)
            time.sleep(1)
            for pin in LED_PINS:
                GPIO.output(pin, False)
            time.sleep(1)

    if __name__ == '__main__':
        processes = [
            Process(target=control_leds_by_distance, args=(LED_PINS[0], 0, 10)),
            Process(target=control_leds_by_distance, args=(LED_PINS[1], 10, 20)),
            Process(target=control_leds_by_distance, args=(LED_PINS[2], 20, 100)),
            Process(target=control_leds_by_pir)
        ]

    for p in processes:
        p.start()

    for p in processes:
        p.join()
```