```
# Write your code here :-)
from random import uniform
# randint, uniform, choice
import pwmio
import board, busio, digitalio, adafruit_lis3dh
# analogio, board, busio, digitalio, pulseio, adafruit_lis3dh
# analogio, board, busio, digitalio, pulseio, adafruit_lis3dh
from time import sleep



MIN_PULSE, MAX_PULSE = 0.5, 2.4
PULSE_RANGE = MAX_PULSE - MIN_PULSE
FREQUENCY = 50
PERIOD_MS = 1.0 / FREQUENCY * 1000.0
SCALE = PERIOD_MS / 65535.0
PINS = (board.A1, board.A2, board.A3)

class Accel:
    def __init__(self):
        self._i2c = busio.I2C(board.ACCELEROMETER_SCL, board.ACCELEROMETER_SDA)
        self._int1 = digitalio.DigitalInOut(board.ACCELEROMETER_INTERRUPT)
        self._lis3dh = adafruit_lis3dh.LIS3DH_I2C(self._i2c, address=0x19, int1=self._int1)
        self._lis3dh.range = adafruit_lis3dh.RANGE_8_G

    def value(self):
        return self._lis3dh.acceleration
```

```
class Servo:
    def __init__(self, pin = 1):
        self.servo = pwmio.PWMOut(PINS[pin - 1], frequency=FREQUENCY)

    def turn_to(self, angle):
        pulse_ms = MIN_PULSE + (angle / 180) * PULSE_RANGE
        self.servo.duty_cycle = int(pulse_ms / SCALE)

servo = Servo(1)
accel = Accel()

def map_values(i, i_min, i_max, o_min, o_max):
    i_range = i_max - i_min
    o_range = o_max - o_min
    s = i - i_min
    return min(o_max, max(o_min, o_min + s / i_range * o_range))


# APPLICATION CODE STARTS HERE


mode = 1  # Change this to try the different modes

while True:
    if mode == 1:  # 0 to 180 by 26 (plays a neat rhythm)
        for turn in range(0, 181, 26):
            print(turn)
            servo.turn_to(turn)
            sleep(0.5 if turn == 0 else 0.2)
```

```
elif mode == 2:  # Turns randomly and sleeps randomly
    servo.turn_to(uniform(0, 180))
    sleep(uniform(0.1, 0.6))

  elif mode == 3:  # Turns based on x acceleration
    angle = map_values(accel.value().x, -9.81, 9.81, 0, 180)
    servo.turn_to(angle)
    sleep(0.05)
```