Code Explanation(Rack and Pinion with Servo Motor – Vertical Sliding Mechanism with Limit Switches)

```
import time
from adafruit_crickit import crickit
```

- `time` is for adding delays (`time.sleep()`).
- `crickit` gives access to the Crickit board features (motors, servos, sensors).

```
ss = crickit.seesaw
LimitS_1 = crickit.SIGNAL1
LimitS_2 = crickit.SIGNAL2
```

- `ss` is the Seesaw chip inside Crickit (handles I/O pins).
- Two **limit switches** are connected to **Signal1** and **Signal2** pins.

```
ss.pin_mode(LimitS_1, ss.INPUT_PULLUP)
ss.pin_mode(LimitS_2, ss.INPUT_PULLUP)
```

- Configures both pins as **input with pull-up resistors**.
- Means: when switch is pressed → pin reads **LOW (0)**,
  when released → pin reads **HIGH (1)**.

```
print("1 Continuous Servo demo!")
motor_dir = 0.5
```

- Prints a startup message.
- `motor_dir = 0.5` → initial motor direction **forward half-speed**.

```
while True:
```

- Infinite loop → keeps checking limit switches & controlling motor.

```
    limit1 = ss.digital_read(LimitS_1)
    limit2 = ss.digital_read(LimitS_2)
    print ("Limit1:", limit1, "Limit2:", limit2)
```

- Reads both limit switches (values `1` or `0`).
- Prints their states for debugging.

```
    if limit1:
```

```
    crickit.continuous_servo_1.throttle = -0.1   # Stop
    motor_dir= 0.5
    crickit.continuous_servo_1.throttle = motor_dir
```

- If **limit switch 1 is active**:
    - Stop motor briefly (`throttle = -0.1` ≈ stop).
    - Set direction forward (`motor_dir = 0.5`).
    - Restart servo forward.

```
elif limit2:
    crickit.continuous_servo_1.throttle = -0.1   # Stop
    motor_dir= -0.5
    crickit.continuous_servo_1.throttle = motor_dir
```

- If **limit switch 2 is active**:
    - Stop motor briefly.
    - Set direction reverse (`motor_dir = -0.5`).
    - Restart servo backward.

```
else:
    crickit.continuous_servo_1.throttle = motor_dir # Forwards
```

- If **no switch is pressed**, motor continues in the last set direction.

```
time.sleep(0.2)
```

- Small delay (200ms) to avoid too many reads and reduce switch bouncing.

Summary of Working

- Motor runs **forward** until it hits **limit switch 1** → reverses.
- Motor runs **backward** until it hits **limit switch 2** → forwards again.
- This creates a **back-and-forth motion** between two limit switches (like an elevator or sliding door).