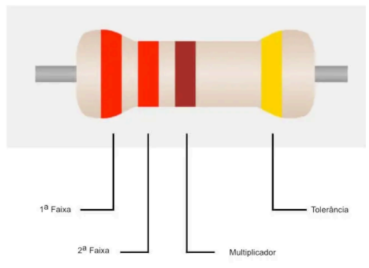
**AULA 4 e 4.1 – Raspberry Pi Pico**

**REDES e IoT**

**Resistores e PWM**

**O que são? (Resistores)**

Resistores são componentes que limitam a passagem de corrente elétrica em um circuito. Servem para proteger componentes, ajustar sinais e controlar tensão e corrente. Sua resistência é medida em ohms (Ω) e cada valor é indicado por faixas coloridas no corpo do resistor.



**Código das cores**

O código de cores geralmente tem 4 faixas de cores no resistor:

* Primeira faixa: A primeira cor indica o primeiro dígito do valor da resistência.
* Segunda faixa: A segunda cor indica o segundo dígito.
* Terceira faixa: A terceira cor indica o multiplicador (como você deve multiplicar os dois primeiros dígitos).
* Quarta faixa: A quarta cor indica a tolerância do resistor, ou seja, o quanto a resistência pode variar em relação ao valor nominal.

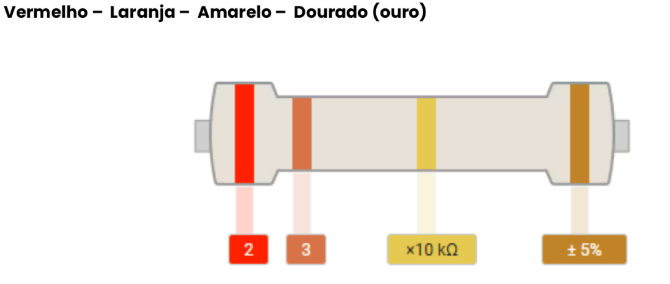
**Tabela de Cores dos Resistores:**

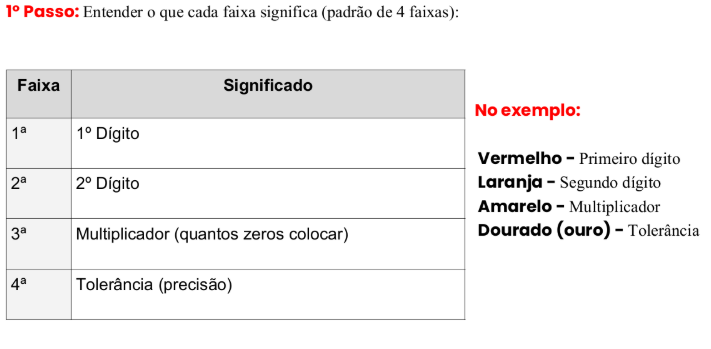
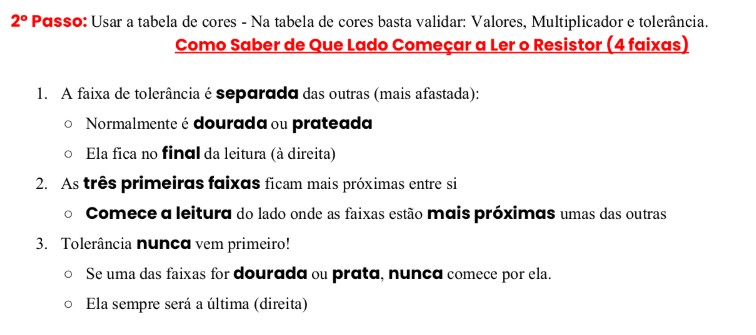
****

**Exemplo Prático para Calcular Resistor**

**(4 faixas)**

**Suponha um resistor com as seguintes cores:**

****

**** ****

**4º Passo: Aplicar leitura da tabela para achar valores:**

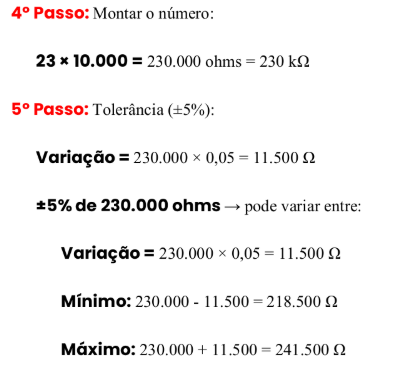
**Valores:**

**Vermelho – 2**

**Laranja – 3**

**Amarelo – x 10.000**

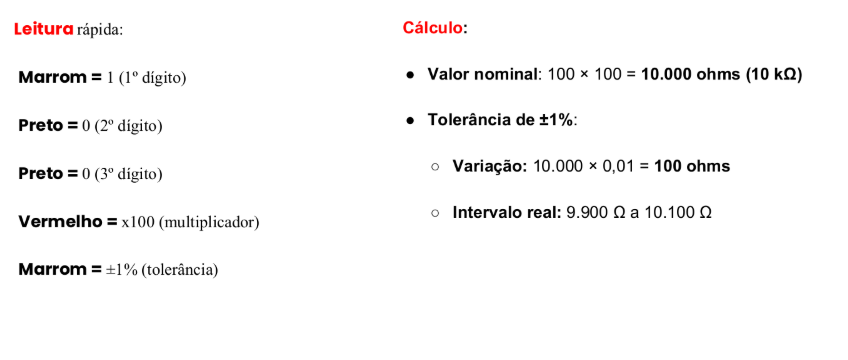
**Dourado – ±5%**

****

**Exemplo Prático para Calcular Resistor**

**(5 faixas)**

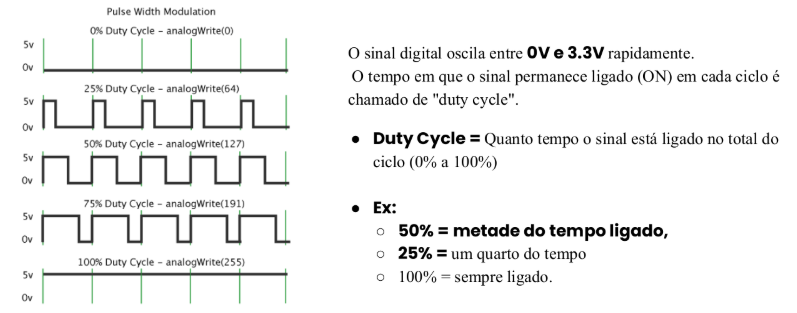
****

****

**PWM**

**O que é?**

**PWM** significa **Pulse Width Modulation** (Modulação por Largura de Pulso). É uma técnica usada para **simular sinais analógicos** usando sinais digitais.



**Para que serve?**

PWM permite controlar potência de forma digital, sendo útil para:

* Brilho de LEDs
* Velocidade de motores
* Controle de servos
* Volume de buzzers passivos
* Simular comportamento de valores analógicos com precisão digital.

**Como usar PWM no MicroPython?**

MicroPython tem uma biblioteca nativa para isso: **machine.PWM**.

from machine import Pin, PWM

led = PWM(Pin(15)) # Cria PWM no pino 15

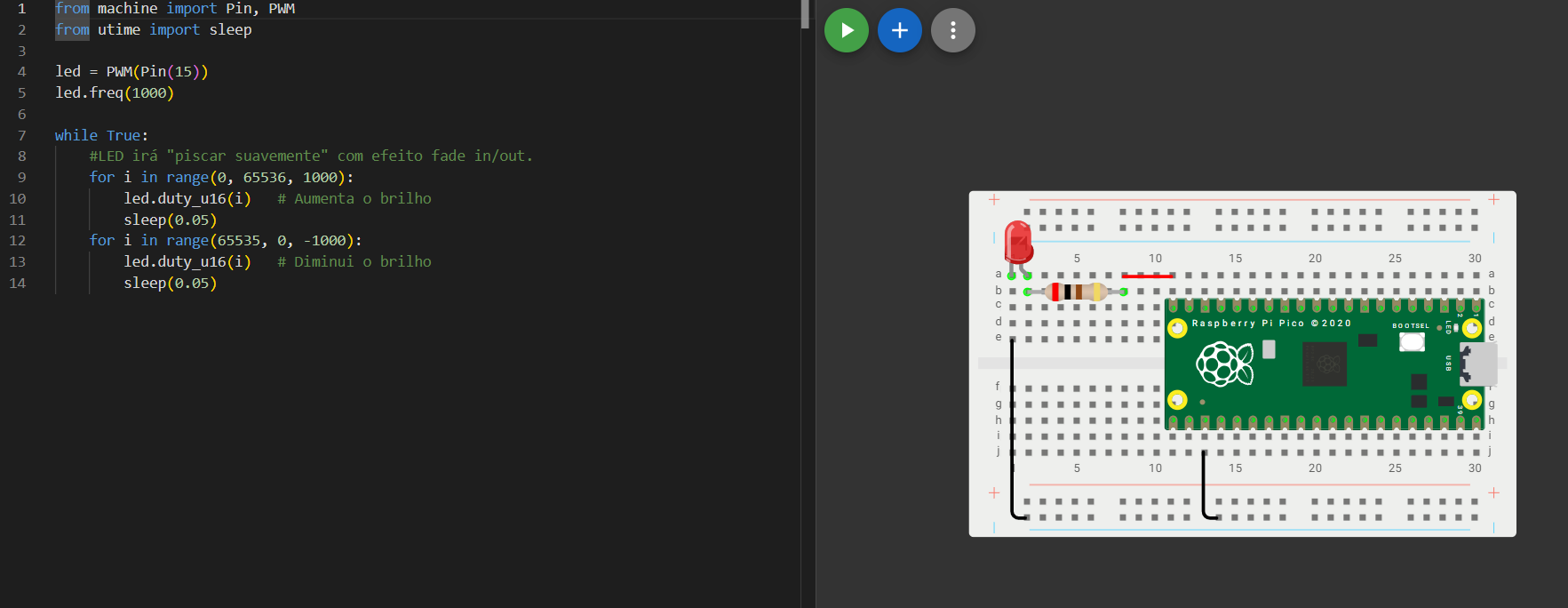
led.freq(1000) # Define a frequência (Hz)

led.duty\_u16(32768) # Define o duty cycle (0 a 65535) - 50%

Observações:

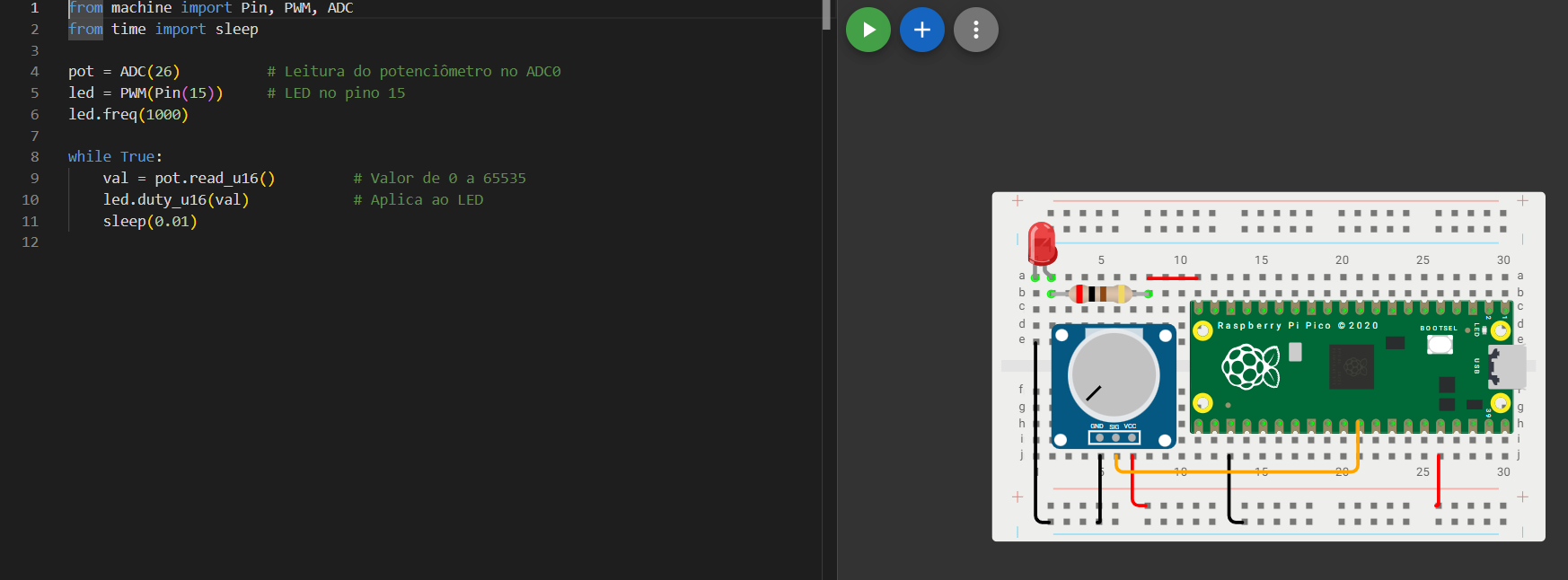
* Frequência pode ser entre 1Hz e 1MHz.
* Duty cycle em 16 bits (0 a 65535), não de 0 a 100.

**Exemplo:**

****

**Exemplo 2:**

**Potenciômetro e PWM**

****

**REALIZAÇÃO DE EXERCÍCIOS**