Aula 11 - Cores no terminal

Dificuldade	⭐
Tags	bold, cor do background, estilos, negative, underline

Cores no terminal

https://youtu.be/0hBlhkcA8O8

Há inúmeras formas de fazer a mesma coisa, para cores há o módulo colorize, iremos usar o **padrão ANSI**, que usa o escape sequence, um padrão que funciona para **vários ambientes**, agora usaremos ele no terminal.

O código **ANSI** começa com um \ e depois vem o código, para cores, iremos usar \033[m + código da cor.

Dentro da chave e **antes** do m vem os códigos, o **primeiro é para o estilo** (**negrito**, sublinhado, normal), dividindo com um ; há o código para a cor do texto, dividindo com um ; há o código para a cor do background.

O número zero significa que o elemento não será colocado.

Estilos

0 → none

1 → **bold** (**negrito**)

4 → underline (sublinhado)

7 → negative (inverte a cor do background para a cor de texto e a cor de texto para o background)

Cor do texto

30 → branco

31 → vermelho

32 → verde

33 → amarelo

34 → azul

35 → magenta (roxo)

36 → ciano

37 → cinza

Há como adicionarmos outras cores, mas para isso seria necessário **importar bibliotecas**.

Cores do background

40 → branco

41 → vermelho

42 → verde

43 → amarelo

44 → azul

45 → magenta(roxo)

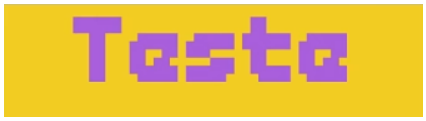
46 → ciano

47 → cinza

Os códigos de estilo são aqueles de unidade, os de cor de letra tem a dezena de 30, os códigos de background tem a dezena do 40.

`/033[ 0 ; 30 ; 41 m`

`/033[ 4 ; 33 ; 43 m`



`/033[ 1 ; 35 ; 43 m`

`/033[ 0 ; 30 ; 41 m`

`/033[ m` Não colocamos nada aqui pois o padrão do terminal é **preto no backgroud** e cinza na letra.

`/033[7; 30 m` Aqui, pelo uso do 7, as cores vão ser **trocadas**, o 30 do branco da letra vai para o background, e como o padrão do background é preto, o preto vai para a letra.

Padrão ASIN em código

```
print('\033[1;31;43mOlá mundo!\033[m')
```

Usamos o \33[m para acabar com a formatação na frase.

Variáveis de outras cores

```
n1 = 3
n2 = 5
print('Os valores são \033[32m{}\033[m e \033[33m{}\033[m!!' .format(n1, n2))
```

Um jeito mais organizado:

```
n = 'Ian'
print('Prazer, {}{}{}!!' .format('\033[1;31m', n, '\033[m')
```

Usando uma variável dicionário

```
n = 'Ian'
cores = {'limpa':'\033[m',
'azul':'\033[34m',
'amarelo':'\033[33m',
'pretoebranco':'\033[7;30m'}'
print('Prazer, {}{}{}!!!' .format(cores['pretoebranco'], n, cores['limpa']))
```

Se o seu programa for complexo, o melhor é **organizar as cores**, se for apenas uma configuração simples, a mudança de cores no print já é o suficiente.

Você tem que saber qual tipo de organização usar, **dependendo** da sua demanda.