



<b>PYTHON</b> Object Oriented Programming 	 <b>INSTITUTO FEDERAL</b> Paraíba Campus João Pessoa <b>Programação e Estrutura de Dados</b> <b>Professor:</b> Alex Sandro da Cunha Rêgo <b>Última atualização:</b> 28/02/2023	<b>Lista</b> <b>5</b>
<p align="center"><b>CLASSES E OBJETOS</b></p> <p align="center"><b>Modelagem e Implementação de um Semáforo</b></p>		

ORIENTAÇÕES
<p><b>Pré-requisitos:</b></p> <ul style="list-style-type: none"> <li>Lógica de programação, conhecimento na criação de classes e objetos em Python: construtor, métodos e propriedades públicas e privadas, de classe e de instância.</li> </ul> <p><b>Instruções</b></p> <ul style="list-style-type: none"> <li>Leia o enunciado com atenção e faça o que se pede</li> </ul>

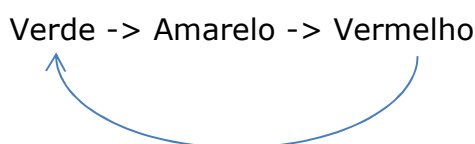
## 1. Fundamentação

O **semáforo** (ou sinal de trânsito) é um instrumento elétrico/eletrônico luminoso utilizado para controlar o tráfego de veículos, pedestres e ciclistas nas grandes cidades. Estes são muito utilizados em cruzamentos, onde é necessário alternar a permissão de passagem entre veículos. Mas também podem ser instalados apenas para interromper temporariamente a passagem dos veículos dando permissão para a travessia de pedestres.

Normalmente, é composto por três círculos de luzes luminosos que disciplinam a passagem de veículos:

- Verde:** passagem de veículos liberada;
- Laranja:** diminua a velocidade – proximidade do término de permissão da passagem;
- Vermelho:** o condutor não pode avançar.

O cálculo dos tempos no controle é gerado a partir das limitações físicas das vias que se interceptam e dos tempos estabelecidos para controle. A passagem de um sinal luminoso para outro estabelece uma transição, de acordo com a sequência a seguir:



Para cada transição, é estabelecido um tempo de permanência que pode ser livremente configurado.

Há diferentes variantes de semáforos que podem ser encontrados. Em particular consideremos o semáforo com visor temporizado ilustrado na Figura 1. Neste exemplo, o condutor do veículo tem uma noção do tempo que ainda resta para acontecer uma transição, haja vista que o tempo é exibido no visor de forma regressiva.



**Figura1.** Semáforo com visor temporizador.

Precisamos desse código para poder criar um baralho específico para o jogo **Sueca**. Não vamos entrar em detalhes em “como implementar um programa principal para simular o jogo”; desejemos apenas preparar o baralho para jogar **sueca**.

Esse jogo tem uma especificidade em relação ao baralho tradicional, pois as cartas 8,9 e 10 de qualquer naipe são retiradas do baralho. Porém, observemos que todo o comportamento da classe **Baralho** é básico para o baralho da **Sueca**. Além disso, para jogar **sueca**, há o conceito de **trunfo**: uma carta que é escolhida aleatoriamente do baralho (em cada partida) para determinar o naipe de maior “força”. Portanto, o trunfo deve ser registrado internamente pelo baralho. A carta que foi retirada como **trunfo** volta para a base do baralho.

As cartas do baralho para sueca são pontuadas da seguinte forma, independente do naipe: Ás=11 pontos, 7=10 pontos, Rei=4 pontos, Valete=3 pontos e Dama= 2 pontos. O restante das cartas tem pontuação igual a 0 (zero).

## 2. Tarefa

Neste exercício, vamos realizar a abstração do problema do **semáforo com visor temporizado**, identificar os objetos participantes da lógica de negócio e extrair a abstração de dados e procedimentos. Sendo assim, faça o que se pede:

1. Planeje e implemente a(s) classe(s) necessária(s) para atender ao domínio do problema.
2. Crie um programa principal à parte para testar o semáforo idealizado, de maneira que possa simular o semáforo funcionando. O programa principal deve permitir a configuração do tempo de permanência em cada indicador luminoso. Fique à vontade para decidir a lógica que vai determinar o ciclo de funcionamento do semáforo (exemplo: se vai funcionar por um tempo total **t** definido pelo usuário ou se vai cumprir um número **y** de ciclos de transição. Por ciclo de transição entende-se: passagem do sinal vermelho, para o verde e depois para o laranja, até retornar ao vermelho)