Protocolos em Swift (I)

Antes de implementar qualquer exercício, desenvolva o diagrama de classes correspondente.

Exercício 1:

Crie um protocolo chamado Imprimível.
Defina o método Imprimir neste protocolo, com a seguinte assinatura:

```
func imprimir() -> Void
```

Depois, crie as seguintes classes: Contrato, Foto e Documento.
Faça cada uma implementar o protocolo Imprimível e defina o comportamento do método Imprimir em cada uma delas:

- Contrato: imprima na tela *"Sou um contrato muito legal"*
- Foto: imprima na tela *"Sou uma selfie"*
- Documento: *"Sou um documento do Word".*

Crie uma classe Impressora que contenha uma lista de imprimíveis inicializada sem elementos.
Depois, crie dois métodos na classe impressora:

- `public func imprimirTudo() -> Void`
  Esse método será encarregado de percorrer a lista de imprimíveis e fazer a impressão.
- `public func agregarImprimível(umImprimível: Imprimible) -> Void`
  Esse método será encarregado de adicionar um imprimível à lista da impressora.

Crie um programa (Playground) e use-o para criar:

- um contrato
- uma foto
- um documento
- uma impressora

Adicione o contrato, a foto e o documento à impressora.
Peça para a impressora imprimir tudo.
Observe o comportamento.

Exercício 2:

Crie um protocolo chamado Voador.
Defina o método Voar nesse protocolo, com a seguinte assinatura:

```
func voar() -> Void
```

Depois, crie as seguintes classes: Pato, Avião, Super-Homem.
Faça cada uma implementar o protocolo Voador e definir o comportamento do método Voar em cada caso:

- Pato: o pato tem uma energia. Cada vez que voa, ele perde 5 unidades de energia e é exibida na tela a frase *"Estou voando como um pato"*
- Avião: o avião tem horas de voo. Cada vez que o avião voa, soma 13h de voo e é exibida na tela a frase *"Estou voando como um avião"*
- Super-Homem: o Super-Homem tem experiência. Cada vez que o Super-Homem voa, a experiência dele aumenta 3 unidades, e é exibida na tela a frase *"Estou voando como um campeão"*

Crie uma classe TorreDeControle que contenha uma lista de voadores inicializada sem elementos.
Depois, crie dois métodos na classe TorreDeControle:

- `public func voamTodos() -> Void`
  Esse método será encarregado de percorrer a lista de voadores e fazer com que eles voem.
- `public func adicionarVoador(umVoador: Voaador) -> Void`
  Esse método será encarregado de adicionar um voador à lista de voadores.

Crie um programa (Playground) e use-o para criar:

- um pato
- um avião
- o Super-Homem
- uma torre de controle

Adicione o pato, o avião e o Super-Homem à torre de controle.
Peça para a torre de controle fazer todos os voadores voarem.
Observe o comportamento.

Exercício 3:

Modele a seguinte situação.

1. Tenho um leitor de DVD que, ao reproduzir um DVD, reproduz seu conteúdo. O conteúdo do DVD pode ser representado por um Texto como forma de mostrar o conteúdo.

2. Como surgiram os BlueRay, quero adicionar uma função ao meu leitor de DVD para poder reproduzir um BlueRay. Para isso eu levo o aparelho para um fabricante, que adiciona essa função para poder reproduzi-los.

3. A moda retrô voltou, por isso voltaram a fabricar os famosos VHS de filmes com conteúdo inédito e reproduzíveis apenas nesse formato. Portanto, o filme não começa com um texto de filme, e sim exibe na tela quantos metros de fita ele tem.

Como estou cansado de levar meu aparelho ao técnico cada vez que surge um novo formato, começo a desenvolver, como programador, uma solução para este problema.