CURSO: PROGRAMAÇÃO PYTHON

MÓDULO 3. UNIDADES 1, 2, 3 e 4

Exercícios propostos de Classes e Objetos

Tokio.

DIRETRIZES GERAIS

- Guardar o documento de soluções com o seguinte formato para entrega:

  M2_01_nome_apelido1_apelido2
- Software recomendado: **Anaconda** – Jupyter. Junto a este documento está um padrão de Jupyter com os enunciados

- **Comentar o código**
- Utilizar nomes de variáveis apropriados. Se vamos guardar uma nota, chamamos a essa variável nota, não n ou x.

EXERCÍCIOS DE CLASSES E OBJETOS. EXERCÍCIO 1

Criação

- Criar uma classe chamada Aluno que tenha os atributos nome e nota.
- Fazer o construtor da classe com dois parâmetros. Adicionar no construtor um print para informar que o aluno foi criado com sucesso.
- Adicionar o método string (_str_), para que ao imprimir, se veja a sua informação.
- Criar um método chamado **qualificacao** que imprima se o aluno aprovou ou reprovou com base na sua nota

Experimentar

- Criar alguns objectos alunos
- Imprimir os objetos alunos criados (o objecto e seus atributos) e o método de qualificação

Tokio.

EXERCÍCIOS DE CLASSES E OBJETOS. EXERCÍCIO 2

Criação

Sobre o exercício anterior fazer a seguinte modificação:

No exercício anterior, ao fazer um print as notas e nomes, vamos obter a nota e nome do aluno. Como pode ser considerado um problema de segurança, vamos encapsular o código para que não se possa aceder diretamente às variáveis e o acesso tenha de ser através de métodos.

- Fazer com que as variáveis da classe sejam **privadas.** Lembrete: adicionar duplo traço à frente do nome da variável
- Criar métodos **getter e setter** para poder aceder e modificar as variáveis da classe

Experimentar

- Criar alguns objectos alunos
- Comprovar que apenas podemos aceder às variáveis da classe através dos seus métodos get e set (Podemos aceder diretamente ao __nome e __nota? Podemos modifica-los?)

Tokio.

EXERCÍCIOS DE CLASSES E OBJETOS. EXERCÍCIO 3

Criação

- Fazer uma classe chamada **Produto** que tenha os atributos codigo, nome, preço e tipo.
- Criar o **construtor** da classe com 4 parâmetros para receber codigo, nome, preço e tipo. Adicionar no construtor um print para informar de que o produto foi criado com sucesso
- Criar métodos **getter e setter** para aceder e modificar todos os atributos da classe

Experimentar

- Fazer alguns objectos produtos
- Experimentar estes novos métodos que se criaram (getter e setter)

EXERCÍCIOS DE CLASSES E OBJETOS. EXERCÍCIO 4

Criação

Ampliar o exercício anterior:

- copiar todo o código que já possui e adicionar o seguinte:

- Adicionar o método string (``__str__``), para imprimir um produto para o ecrã e exibir todos os dados desse produto.

- Criar um método chamado __calcular_total__ que recebe um número de unidades para esse produto, e com base nisso, calcular o preço total para esse conjunto de produtos. Exemplo: 5 unidades do produto 1

Experiência

- Experimentar o novo método que foi desenvolvido

EXERCÍCIOS DE CLASSES E OBJETOS. EXERCÍCIO 5

Criação

Continuar a ampliar o exercício anterior: Copiar todo o código que tiver e adicionar o seguinte:

- Desenvolver uma nova classe chamada **Pedido** que tenha os seguintes atributos: uma lista de produtos e uma lista de quantidades.
- Criar o **construtor** da classe. Adicionar um print ao construtor para informar que o pedido foi criado com sucesso. Deverá receber na contrução por parametro a lista de produtos e quatidades.
- Fazer o método **total_pedido** o qual devolva o **montante** total de um pedido.
- Criar o método **mostrar_pedido** o qual mostre todos os produtos e quantidades que compõem o pedido.

Experimentar

- Criar alguns pedidos.
- Experimentar os novos métodos que se criaram.

Tokio.

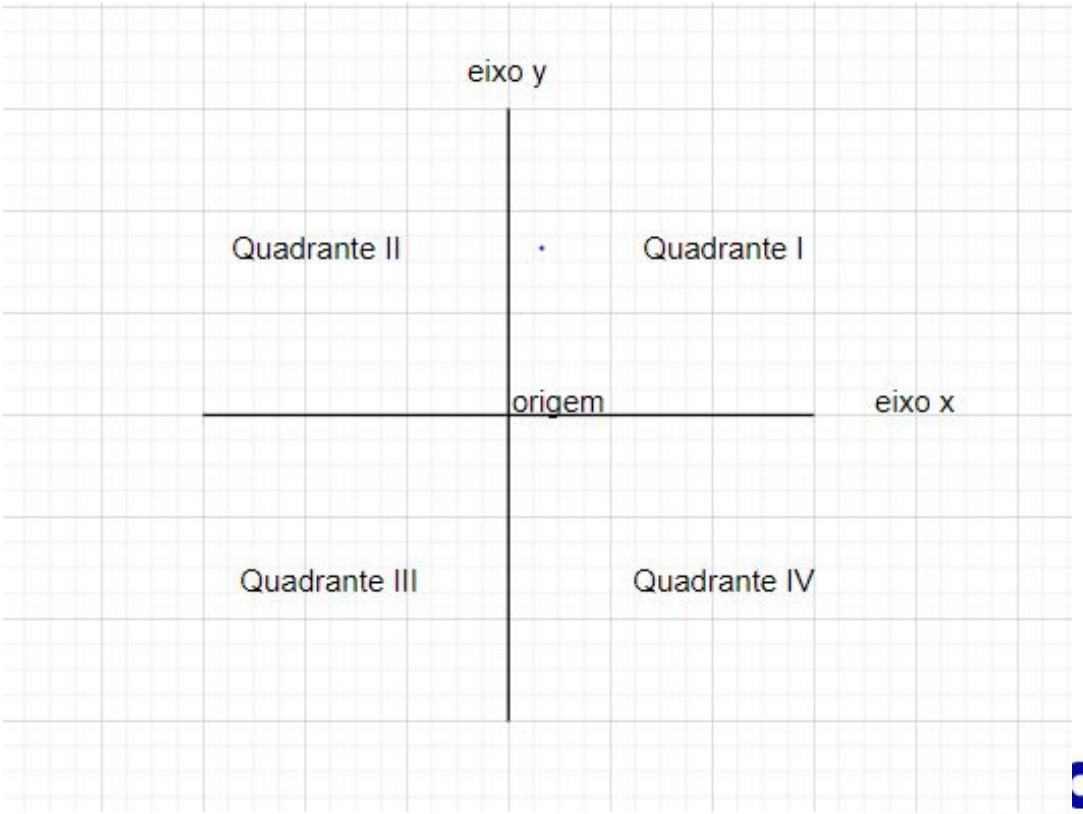
EXERCÍCIOS DE CLASSES E OBJETOS. EXERCÍCIO 6

Exercício de pontos, coordenadas e vetores sobre o plano cartesiano. A seguir os conceitos básicos.

O plano cartesiano

Representa um espaço bidimensional (em 2 dimensões), formado por duas retas perpendiculares, uma horizontal e outra vertical que se cortam num ponto. A reta horizontal denomina-se eixo das abscissas ou **eixo X**, embora que a vertical recebe o nome de eixo das ordenadas ou, simplesmente, **eixo Y**.

Quanto ao ponto onde se cortam, conhece-se como o **ponto de origem O**. O **plano divide-se em 4 quadrantes.**

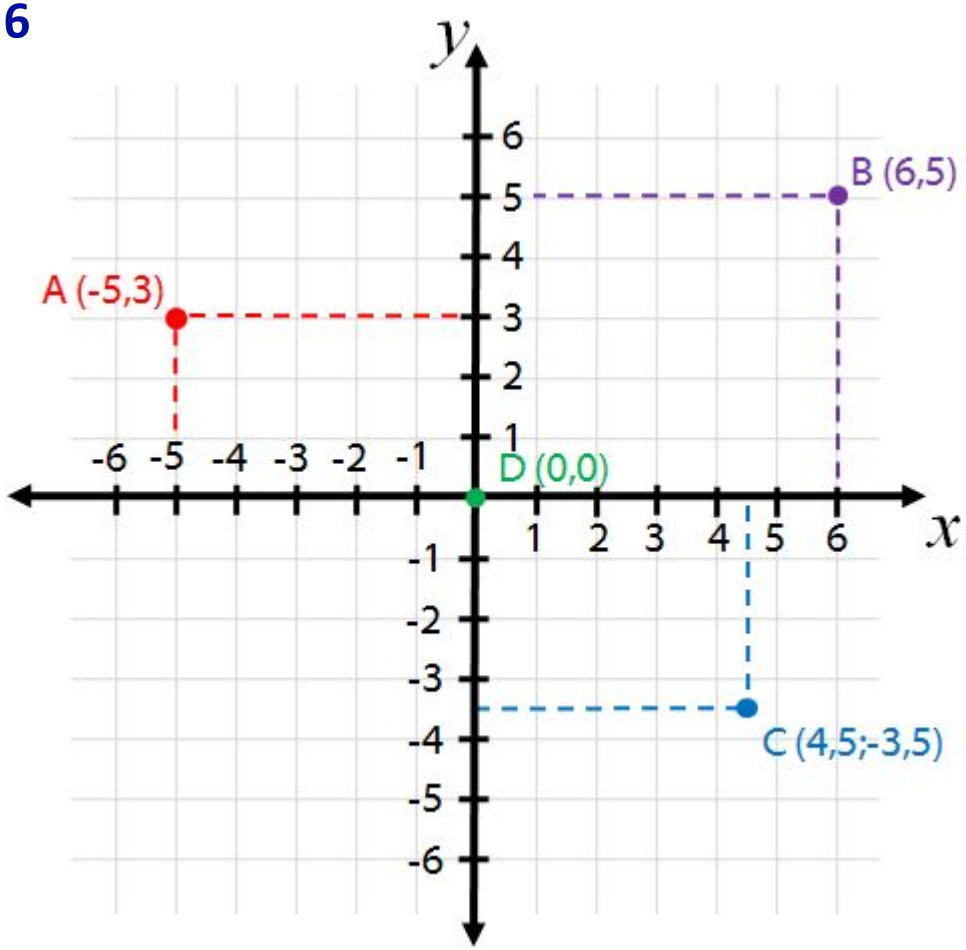


EXERCÍCIOS DE CLASSES E OBJETOS. EXERCÍCIO 6

Pontos e coordenadas

O objetivo é descrever a posição de **pontos** sobre o plano em forma de **coordenadas**, que se formam associando o valor do eixo X (horizontal) com o valor do eixo Y (vertical).

A representação de um ponto é simples: **P(X,Y)** onde X e Y são a distância horizontal (esquerda ou direita) e vertical (acima ou abaixo) respetivamente, utilizando como referência o ponto de origem (0,0), no centro do plano.



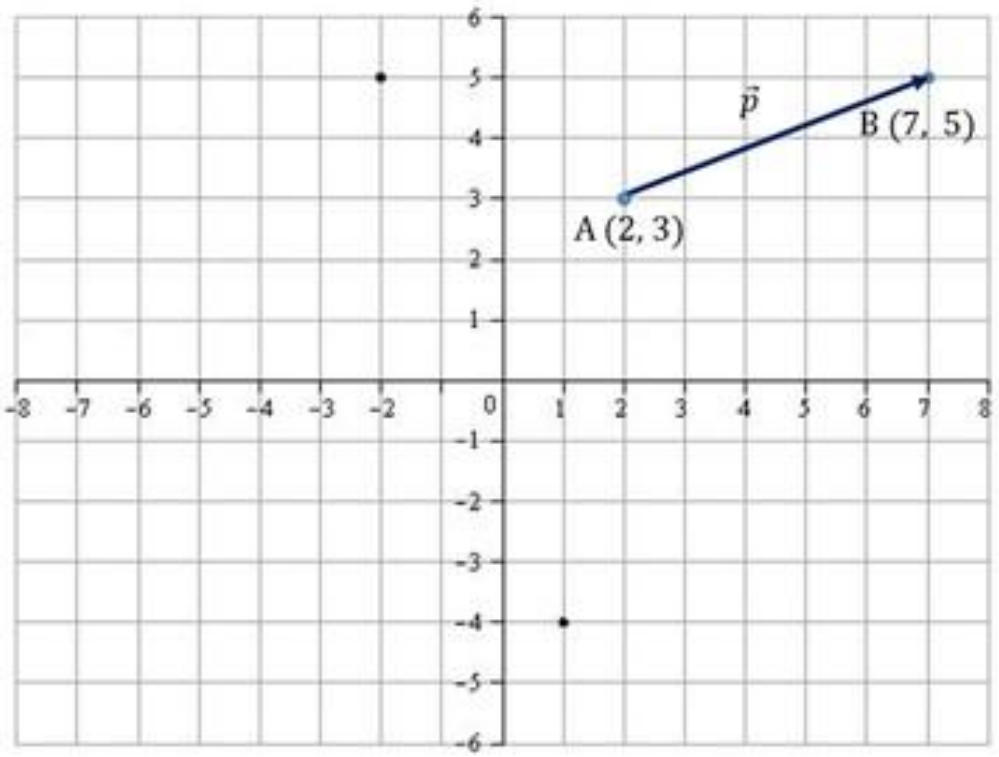
Tokio.

EXERCÍCIOS DE CLASSES E OBJETOS. EXERCÍCIO 6

Vetores no plano

Finalmente, um vetor no plano faz referência a um segmento orientado, gerado a partir de dois pontos distintos.

A efeitos práticos não deixa de ser uma linha feita desde um ponto inicial em direção a outro ponto final, pelo que se entende que um vetor tem comprimento e direção/sentido.



Nesta figura, podemos observar dois pontos A e B que poderíamos definir da seguinte forma:

- **A(x1, y1) => A(2, 3)**
- **B(x2, y2) => B(7, 5)**

E o vetor seria representado como a diferença entre as coordenadas do segundo ponto relativamente ao primeiro (o segundo menos o primeiro):

- **AB = (x2-x1, y2-y1) => (7-2, 5-3) => (5,2)**

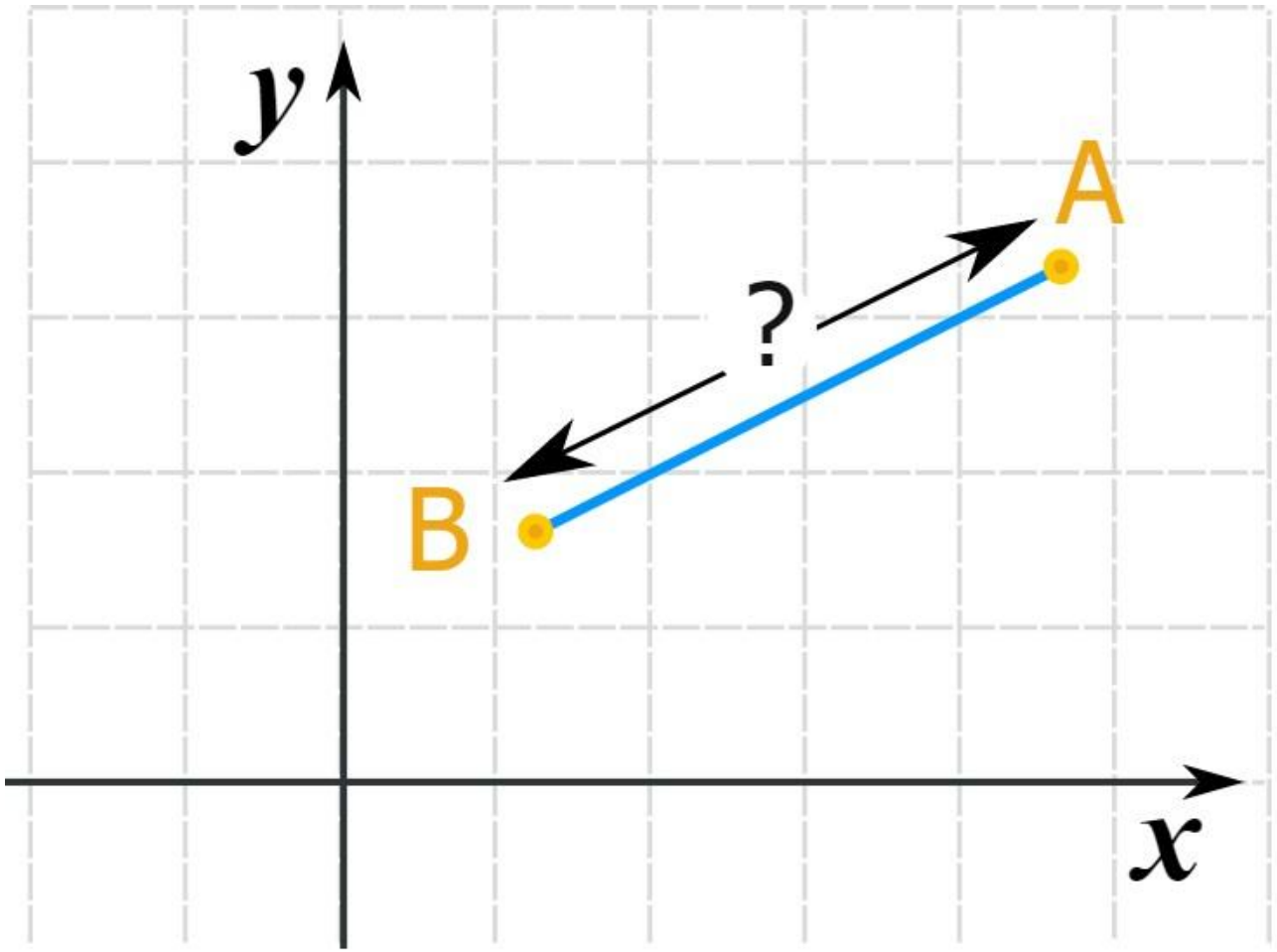
O que em definitiva não deixa de ser: 5 à direita e 2 acima.

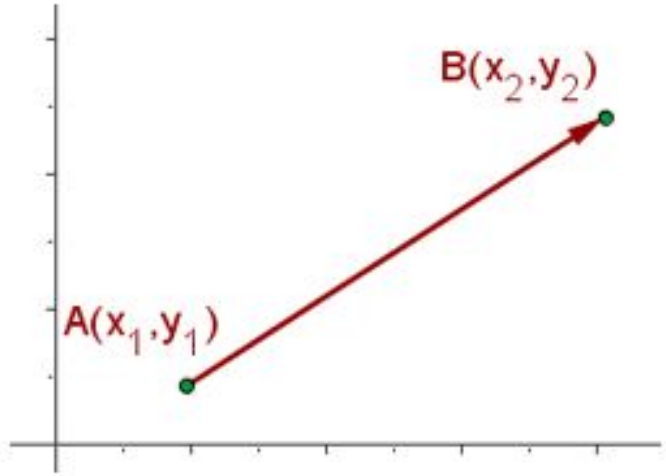
EXERCÍCIOS DE CLASSES E OBJETOS. EXERCÍCIO 6

Criação

- Criar uma classe chamada **Ponto** com as suas duas coordenadas X e Y.
- Adicionar um método construtor para criar pontos facilmente. Colocar o seu valor default a zero.
- Substituir o método string (_str_), para que ao imprimir um ponto, apareça em formato "(X,Y)"
- Adicionar um método chamado **quadrante** que indique a que quadrante pertence o ponto, ou se é a origem.
- Agregar um método chamado **vetor**, que receba outro ponto e calcule o vetor resultante entre os dois pontos.
- Incorporar um método chamado **distância**, que receba outro ponto, calcule a distância entre os dois pontos e imprime. A fórmula é a seguinte:

EXERCÍCIOS DE CLASSES E OBJETOS. EXERCÍCIO 6





- $A(x_1, y_1)$ $B(x_2, y_2)$
- $d(A, B) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$

Tokio.

EXERCÍCIOS DE CLASSES E OBJETOS. EXERCÍCIO 6

Nota: A função raiz quadrada em Python sqrt() deve-se importar do módulo math e utilizar da seguinte forma:

```
import math
math.sqrt(9)
```

- Criar uma classe chamada **Retângulo** com dois pontos (inicial e final) que fazem a diagonal do retângulo.
- Incorporar um método construtor para criar ambos os pontos, definindo por defeito dois pontos na origem.
- Adicionar ao retângulo um método chamado **base** que calcule e imprime a base.
- Agregar ao retângulo um método chamado **altura** que calcule e imprime a altura.
- Adicionar ao retângulo um método chamado **área** que calcule e imprime a área.
- Podemos identificar facilmente estes valores se tentar desenhar o quadrado a partir da sua diagonal. Se andar perdido, experimentar a desenhar num papel, de certeza que se vê muito mais claramente! Podemos utilizar a função abs() para saber o valor absoluto de um número.

EXERCÍCIOS DE CLASSES E OBJETOS. EXERCÍCIO 6

Experimentação

- Criar os pontos A(2, 3), B(5,5), C(-3, -1) e D(0,0) e imprimir.
- Averiguar a que quadrante pertence o ponto A, C e D.
- Calcular os vetores AB e BA (usar o método).
- Calcular a distância entre os pontos 'A e B' e 'B e A' (usar o método).
- Determinar qual dos 3 pontos A, B ou C, se encontra mais longe da origem, ponto (0,0).
- Fazer um retângulo utilizando os pontos A e B.
- Calcular a base, altura e área do retângulo (usar o método).

EXERCÍCIOS DE CLASSES E OBJETOS. EXERCÍCIO 7

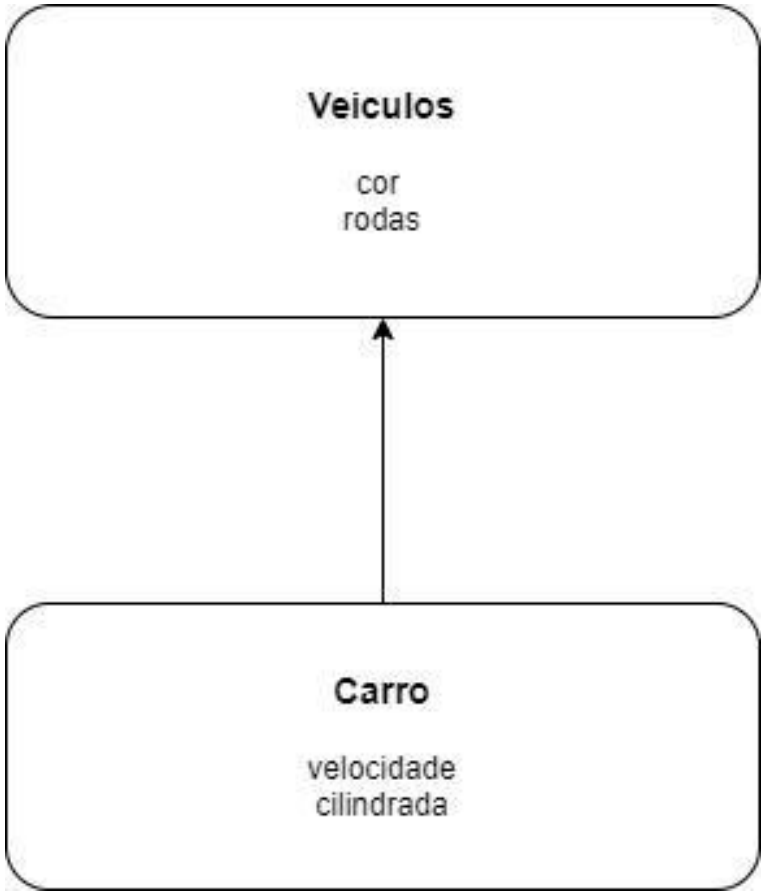
Criação
Trabalhar com heranças.

- Criar uma classe chamada **Veículo** que tenha os seguintes atributos: cor e rodas
- Fazer o construtor da classe.
- Substituir o método string (_str_), para que ao imprimir um veículo se possa ver a sua informação.
- Fazer uma classe que se chame **Carro** que herde de Veículo
- Criar o construtor da classe, que inclua tudo proveniente do construtor do veículo , adicionando a velocidade e a cilindrada (métodos de Carro).
- Substituir o método string (_str_), para que ao imprimir um carro para se possa ver a sua informação

Experimentar
Criar alguns objectos de Carros e mostrar a sua informação

Tokio.

EXERCÍCIOS DE CLASSES E OBJETOS. EXERCÍCIO 7



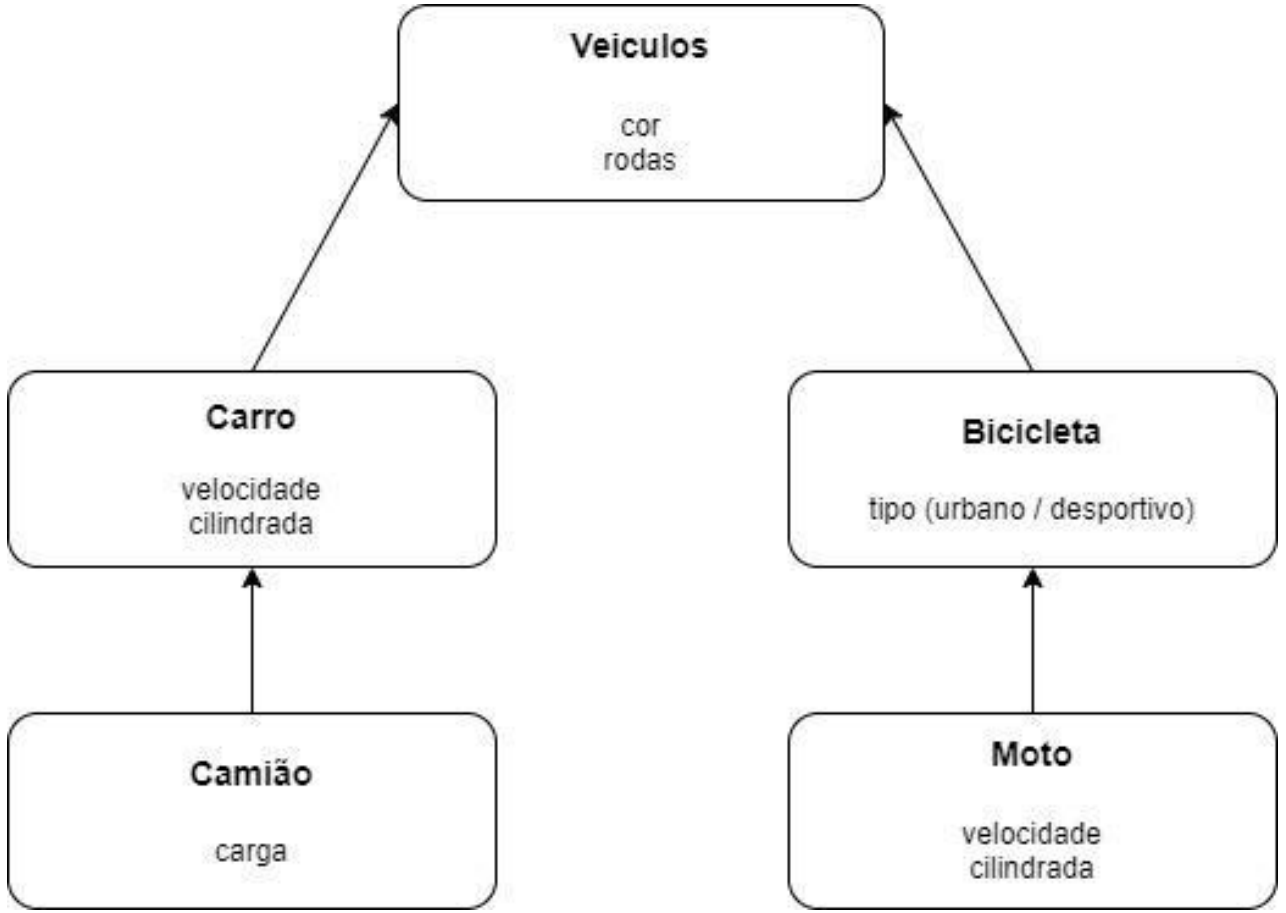
EXERCÍCIOS DE CLASSES E OBJETOS. EXERCÍCIO 8

Trabalhar com herança. Continuar a ampliar o exercício anterior. Copiar todo o código que já tem e adicionar o seguinte:

- Criar as classes herdadas de Veículo que aparecem na imagem.
- Fazer os construtores para essas classes.
- Substituir os métodos string (str), para que ao imprimir por ecrã se veja a informação correspondente.
- Criar uma lista e dentro dela introduzir pelo menos um objeto de cada subclasse de Veículo
- Fazer uma função chamada catalogar() que receba a lista de veículos e os percorra mostrando o nome da sua classe e os seus atributos.
- Modificar a função catalogar() para que receba um argumento opcional de rodas, fazendo com que mostre apenas o número de rodas que concorde com o valor do argumento. Também deve mostrar uma mensagem "Encontrou-se {} veículos com {} rodas:“apenas se enviar o argumento de rodas. Colocar à prova com 0, 2 e 4 rodas como valor.

Pista. Investigar para que serve esta instrução e se nos vai ser útil type(objeto). name

EXERCÍCIOS DE CLASSES E OBJETOS. EXERCÍCIO 8



EXERCÍCIOS DE CLASSES E OBJETOS. EXERCÍCIO 9

Criação

Trabalhar com herança. Continuar a ampliar o exercício anterior. Copiar todo o código que já tem e adicionar o seguinte:

Um método que se chame **filtrar,** que receba 2 parâmetros, a lista de veículos anteriomente construida e o nome de um dos tipos de veículos (Carro, Camião, etc..). O método deve percorrer a lista, procurar o nome do tipo da classe de cada objecto e encontrar os iguais ao argumento nome, e guardá-las numa nova lista.

Esta nova lista deve-se devolver ao programa principal. Por fim, deve imprimir os objectos da lista. A chamada no programa principal deverá ser assim: listaCarros = veiculo.filtrar(lista, "Carro")

Experimentar

Criar alguns veículos (mais de que um por tipo)
Filtrar pelos diferentes tipos de Veículo e mostrar os resultados