

Programação Orientada a Objetos

Exercícios

Prof. Emerson Paduan

Exercício 1)

Desenvolva uma classe **Carro**, com os seguintes atributos:

Marca (não pode ficar vazio);

Modelo (não pode ficar vazio);

Quantidade de combustível, em litros (não pode ser negativo);

Consumo médio, em quilômetros por litro (não pode ser negativo);

Quilometragem (não pode ser negativo);

Encapsule todos os dados (private), e não crie métodos set.

Crie os seguintes métodos:

- **dois construtores**: um que recebe todos os dados, e outro que recebe todos exceto a quilometragem e o nível de combustível (indicando um carro zero km e sem combustível).

- **andar**: Recebe a quilometragem a ser percorrida, que deve ser maior que zero. Caso haja combustível suficiente para rodar a quilometragem pedida (de acordo com o consumo do carro), faz as seguintes ações:

- incremente a quilometragem do carro;

- diminua a quantidade de combustível (de acordo com o consumo do seu carro);

- **abastecer**: Recebe uma quantidade de combustível (valor positivo) e adicione ao tanque do carro. Não pode receber um valor negativo;

- **exibir (toString)**: retorna uma String com todos os dados do carro;

Crie um objeto na **main** e faça o teste dos métodos desenvolvidos.

Programação Orientada a Objetos

Exercícios

Prof. Emerson Paduan

Exercício 2)

Crie uma classe *CelularPrePago*, que tenha os atributos:

Nome do cliente (pode ser vazio);
Telefone (não pode ser vazio);
Saldo (não pode ser negativo);

Todas as variáveis devem ser encapsuladas, porém o **saldo não deve possuir um método de set.**

O saldo será alterado pelos seguintes métodos:

- **fazerLigacao**: caso o usuário tenha saldo suficiente, desconta o valor da ligação do saldo; caso contrário, não faz a ligação. Considere que cada ligação custa R\$ 0,75.
- **recarregar**: recebe um valor a ser adicionado no saldo. **Não aceita valores negativos.**

Escreva também os seguintes métodos:

- **construtor** que receba **apenas o nome**, e **outro** que receba **todos** os atributos.
- **imprimir**, que retorna uma String com todos os dados.

Crie **dois objetos** no **main**, testando os métodos criados.

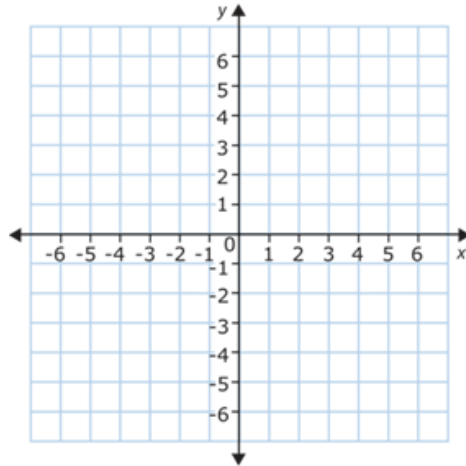
Programação Orientada a Objetos

Exercícios

Prof. Emerson Paduan

Exercício 3)

Implemente o problema do robô andando em uma sala, conforme o diagrama abaixo:



Deve existir um método para definir a quantidade máxima de linhas e colunas da sala (tamanho do ambiente) e inicializar o ambiente.

O método que inicia o **Robo** deve ler a posição inicial do robô.

Os métodos da classe **Robo** tem as seguintes funcionalidades:

- **norte()** → faz com que o robô se desloque +1 no eixo y
- **sul()** → faz com que o robô se desloque -1 no eixo y
- **leste()** → faz com que o robô se desloque +1 no eixo x
- **oeste()** → faz com que o robô se desloque -1 no eixo x

O ato de andar modifica as variáveis de posição do robô (x e y).

Você deve verificar se é possível o comando pedido, pois ele deve respeitar o tamanho máximo definido para o ambiente

Escreva um **App** contendo um menu com as seguintes opções:

- 1 - Andar para Frente
- 2 - Andar para Trás
- 3 - Parar
- 4 - Virar para Direita
- 5 - Virar para Esquerda
- 6 - Sair