

Programação Orientada a Objetos

EXERCÍCIO 23

Leia os capítulos 4 e 5 da apostila e resolva os exercícios.

EXERCÍCIO 24

Para evitar erros de digitação de sequências de números tais como: número de conta bancária, número de CPF, CNPJ e afins, geralmente adiciona-se ao número dígito(s) verificador(es). Por exemplo, no número de cpf “741.392.463-85”, os números 8 e 5 são dígitos verificadores.

Modifique a classe `Funcionario` da apostila acrescentando um atributo `String cpf` e o método `boolean validaCPF()`. O método deverá validar CPF's nos formatos “12345678910” ou “123.456.789-10”.

Pesquise:

1. <http://www.clubedainformatica.com.br/site/2003/10/22/algorithm-de-cpf/>
2. O método `charAt` da classe `String` e o método `isDigit` da classe `Character`

EXERCÍCIO 25

Modifique o método `adiciona` da classe `Empresa` para adicionar ao vetor `empregados` apenas o objeto funcionário que tenha CPF válido.

EXERCÍCIO 26

Faça uma classe `Cartao` que tenha internamente um objeto do tipo `Conta`, uma senha numérica e a `relacaoDeDebitos`.

- a) `relacaoDeDebitos` é um vetor que armazena o nome do estabelecimento onde foi efetuado o débito e o valor do débito. Proponha uma classe para armazenar estas informações.
- b) Tenha um método `debitar`, que tem como parâmetros: o valor a ser debitado, o nome do estabelecimento e a senha. A operação só é realizada se a senha passada por parâmetro for igual à armazenada no cartão e o saldo da conta permitir o débito. Neste caso, o nome do estabelecimento e o valor são armazenados em `relacaoDeDebitos`. O método `debitar` devolve um valor booleano indicando se a operação foi realizada.
- c) Um método `gerarFatura`, que recebe uma senha por parâmetro e imprime os nomes dos estabelecimentos e os valores das compras do cartão. A operação só é realizada se a senha passada por parâmetro for igual à armazenada no cartão.

EXERCÍCIO 27

Implemente uma classe `Ponto` para pontos bidimensionais (x,y), um método `negivo()` para transformar o ponto em seu negativo, um método `distancia()` para retornar a distância do ponto a partir da origem (0,0) e um método `print()`.

EXERCÍCIO 28

Implemente uma classe `Circulo`. Cada objeto dessa classe representará um círculo, armazenando seu `raio` e as coordenadas `x` e `y` de seu centro (objeto `Ponto`). Defina métodos para cálculo de `area()` e um método `perimetro()`.

EXERCÍCIO 29

Implemente uma classe `Tempo`. Cada objeto dessa classe representará uma hora específica do dia, armazenando `horas`, `minutos` e `segundos` como inteiros. Inclua um método `avance(int by, int m, int s)` para avançar a hora atual de um objeto existente, um método `reset(int h, int m, int s)` para redefinir a hora atual de um objeto existente e um método `print()`.

EXERCÍCIO 30

Implemente uma classe `Str`. Cada objeto dessa classe representará uma string de caracteres. Os atributos são: o comprimento da string e o vetor de caracteres. Faça um método `print()` e um método `substring(int inicial, int final)` e `acrescente(Str string)`.

EXERCÍCIO 31

Implemente uma classe `Matrix`. Inclua um método para `adicionar(int linha, int coluna, int valor)`, um método `inversa()` que retorna a inversa da matriz, um método `determinante()` que retorna o determinante da matriz, um método `boolean isSingular()` que retorna `true` ou `false` de acordo com a possibilidade de determinante ser zero e um método `print()`.

EXERCÍCIO 32

Através da manipulação genética, biólogos criaram uma bactéria que come lixo. Esta consome o dobro de seu peso em lixo por hora. O peso da bactéria é variável (entre 10 e 20 picogramas), o lixo é totalmente metabolizado. De 3 em 3 horas cada bactéria se divide em duas. Uma bactéria vive entre 22 e 28 horas e depois morre.

Modele a classe `Bacteria`. Ela deve representar adequadamente o estado de uma bactéria (tempo de vida, lixo metabolizado, etc). Deve ter métodos para retornar quanto lixo ela metabolizou, simular a passagem de uma hora (reduzindo o seu tempo de vida) e retornar o tempo de vida restante.

Crie uma classe `Colonia` que representa uma colônia de bactérias (vetor). A classe `Colonia` deve ter métodos que: simula a passagem de horas (das bactérias), retorna a quantidade de bactérias existentes e o total de lixo consumido. Faça um programa de teste que simula a passagem de N horas.

OBS: As bactérias que forem morrendo devem ser retiradas do vetor de bactérias da classe `Colônia`. Observe que será necessário gerenciar o espaço vazio que ficará no início do vetor.

EXERCÍCIO 33

Crie uma classe para representar uma pessoa, com os atributos privados de nome, idade e altura. Crie os métodos necessários e também um método para imprimir os dados de uma pessoa.

EXERCÍCIO 34

Crie uma classe `Agenda` que pode armazenar 10 pessoas e seja capaz de realizar as seguintes operações:

- `void armazenaPessoa(String nome, int idade, float altura);`
- `void removePessoa(String nome);`
- `int buscaPessoa(String nome);` // informa em que posição da agenda está a pessoa
- `void imprimeAgenda();` // imprime os dados de todas as pessoas da agenda
- `void imprimePessoa(int index);` // imprime os dados da pessoa que está na posição "i" da agenda.

OBS: o método `void armazenaPessoa(String nome, int idade, float altura)` não é uma boa prática de programação, pois aumenta o acoplamento entre classes. Pesquise e discuta com seus colegas!

EXERCÍCIO 35

A empresa MMF LTDA, do ramo de automóveis, possui um cadastro de funcionários em um armário de aço e um cadastro de empresas de recursos humanos (RH) em uma pequena caixa. Estas empresas são utilizadas pela MMF para selecionar seus novos funcionários. O cadastro de funcionários está em constante mudança, pois a MMF está passando por uma fase excelente no mercado e está contratando novos funcionários.

A empresa MMF Ltda. deseja automatizar todo o processo de admissão de funcionários, assim como o controle sobre as empresas de RH parceiras. Baseado nas informações acima e nas fichas abaixo, para os dois cadastros, desenvolver as questões que seguem:

- a) Definir classes para representar as fichas de funcionários e das empresas de RH;
- b) Sabendo que há atualmente 25 funcionários e 10 empresas parceiras, cria uma classe `EmpresaMMF` que tenha duas variáveis (vetor) que sejam capazes de armazenar estas quantidades de informações. Observe que esta classe `EmpresaMMF` terá um único objeto de instância que representará a MMF;
- c) A classe `EmpresaMMF` deve ter dois métodos para adicionar (objeto) funcionário e empresa de RH;
- d) Faça um método de `EmpresaMMF` que imprima os dados dos funcionários que possuem mais que N (lido) dependentes.

Nro da Empresa:											
No Func.:	Nome:										
Data Nasc.:	Est. Civil:										
Endereço:											
RG:	CIC:										
Fone:	No. de dependentes:										
<div> <div>Nome dos Dependentes</div> <div>Grau de Parentesco</div> </div> <table border="1"> <tr><td> </td><td> </td></tr> <tr><td> </td><td> </td></tr> <tr><td> </td><td> </td></tr> <tr><td> </td><td> </td></tr> <tr><td> </td><td> </td></tr> </table>											
Nome do Pai:											
Nome da Mãe:											

Fig 1. Ficha do Cadastro de Funcionários

Nro:
Nome da Empresa:
Endereço:
Pessoa para Contato:
e-mail:
Telefones

Fig 2. Ficha do Cadastro de Empresas de RH

EXERCÍCIO 36

Uma empresa transportadora deseja calcular a distância percorrida pelos caminhões. Para isso, foi desenvolvida uma ficha com o percurso de cada caminhão (uma ficha para cada um) com os seguintes dados:

- Número do caminhão;
- Número de cidades percorridas;
- Códigos de todas as cidades percorridas;

Assim, uma ficha com:

103, 06, 01, 05, 07, 03, 09, 03

indica que o caminhão nº 103 percorreu 6 cidades na seguinte ordem:

- da cidade 1 para a cidade 5;
- da cidade 5 para a cidade 7;
- da cidade 7 para a cidade 3;
- da cidade 3 para a cidade 9;
- da cidade 9 para a cidade 3;

cada caminhão percorre no máximo 6 cidades.

Para calcular a distância entre as cidades, a empresa possui uma tabela de distâncias:

	0	1	2	3	4	...	10
0							
1		0	15	10	18	...	90
2		15	0	25	42		115
3		10	25	0	12		75
4		18	42	12	0		87
:		:				:	
10		90	115	75	87		0

Faça um programa que:

- a) Definir uma classe Transportadora com a matriz de distâncias e o vetor com as fichas de caminhões; Proponha uma classe para representar a ficha.
- b) Crie um método para gerar a matriz de distâncias entre cidades automaticamente (aleatório);
- c) Leia ficha de percurso dos caminhões; Crie um método em Transportadora para adicionar a ficha;
- d) Leia o número de um caminhão; Crie um método que receba o número do caminhão e calcule/retorne a distância percorrida por ele.

(baseado no exercício 2.5.3.5 do livro Algoritmos Estruturados do Harry Farrer)

EXERCÍCIO 37

Modele e implemente uma classe chamada `ParDeDados`, composta por dois dados de seis lados e um método `rolar`. Crie uma classe `TestaParDeDados` com um método `main` que irá "lançar" (rolar) um objeto `ParDeDados` múltiplas vezes contando o número de vezes que aparece o número 6 em ambos os dados.

Pesquise pela classe `Random`.

EXERCÍCIO 38

Usando a classe `ParDeDados` da questão anterior, modele e implemente uma classe que representa um jogo chamado Pig, cujo objetivo é atingir 100 pontos antes do seu adversário. O jogo ocorre da seguinte maneira:

- a) Um jogador começa jogando um par de dados.
- b) Se os dois dados obtiverem um número diferente de 1, os pontos são acumulados no total de pontos da rodada do jogador:
 1. Se o total de pontos da rodada for menor que 20, volte para o passo (a);
 2. Se o total de pontos da rodada for maior ou igual a 20, acumule o total de pontos da rodada no total de pontos do jogador e passe o controle dos dados para o outro jogador, que começa no passo (a)
- c) Se o jogador obtiver 1 em um dos dados e NÃO estiver arriscando, este pode optar:
 1. em perder todos os pontos da rodada e passar o controle dos dados ao adversário, que começa no passo (a);
 2. em arriscar a ser o Pig. Neste caso o jogador recebe o status de ARRISCANDO, e volta para o passo (a) sem acumular os pontos do lançamento.
- d) Se o jogador obtiver 1 em um dos dados e JÁ estiver arriscando, perde TODOS os pontos acumulados durante o jogo (PIG) e passa o controle dos dados para o outro jogador, que começa no passo (a).

Crie as classes para representar o jogo Pig, o jogador e o programa principal (main). O jogo Pig é jogado por duas pessoas.

BOM ESTUDO!