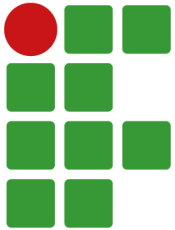
Programação Orientada a Objetos - POOS3

Tecnologia em Análise e Desenvolvimento de Sistemas

Exercícios Avaliativos
Caderno de exercícios avaliativos da disciplina.

2º semestre de 2018



INSTITUTO FEDERAL
São Paulo
Câmpus Araraquara

Prof. Ednilson G. Rossi (ednilsonrossi.ifsp@gmail.com)

Exercícios 1

- Implemente um sistema em Java que leia 5 conjuntos de três valores inteiros e informe qual o tipo de triângulo esses valores formam ou se não formam um triângulo.

  - O programa deve ser implementado em um editor de texto simples como bloco de notas e compilado no terminal.
  - Deve-se enviar o arquivo .java e o .class gerado.

Exercício 2

- Para o exemplo MinhaData implementado em aula, implemento um método que retorne a quantidade de dias entre a data instanciada e uma data passada como argumento.

  - O programa deve ser implementado em um editor de texto simples como bloco de notas e compilado no terminal.
  - Deve-se enviar os arquivos .java e os .class gerados.

Exercício 3

- Implemente um sistema que simule um jogo de dados, o qual o jogador lança 2 dados de seis lados.
- Deve-se construir uma classe Dado que representa um dado.
- O programa principal deve lançar os dados 3 vezes e apresentar a soma dos números dos dados a cada um dos lances.
- **Material de pesquisa:**
  - Números aleatórios em java: https://www.devmedia.com.br/numeros-aleatorios-em-java-a-classe-java-util-random/26355

Exercício 4

- Crie uma classe **USMoney** com dois atributos inteiros: **dollars** e **cents**.
- Adicione um construtor com dois parâmetros para a inicialização do objeto USMoney. O construtor deve verificar se o valor de cents está entre 0 e 99 e, se não estiver, transferir alguns dos cents para o atributo dollars para que ela passe a ter entre 0 e 99.
- Implemente um método **plus** que recebe um objeto USMoney como argumento. Esse método deve criar e retornar um novo objeto USMoney representando a soma dos objeto cujo método plus() está sendo chamado mais o argumento, sem modificar os valores dos dois objetos já existentes.
- Deve-se assegurar que o valor do atributo cents do novo objeto esteja entre 0 e 99. Por exemplo, se x for um objeto USMoney com 5 dollars e 80 cents e se y for um objeto USMoney de 1 dollar e 90 cents, **x.plus(y)** retornará um novo objeto USMoney com 7 dollars e 70 cents.
- Implemente um programa principal que teste vários casos de teste para o método plus da classe USMoney.

Exercício 5

- Implemente a classe caneta:
  - Atributos
    - cor
    - carga
    - tampa
    - tipo
  - Método
    - escreverPalavra()
  - Observação:
    - Para cada palavra escrita uma quantidade da carga é gasta.
- Implemente também um programa principal para executar as operações de uma caneta.

Exercício 6 (1 de 3)

- Baixe do repositório da disciplina no Github o projeto Exemplo14_Conjunto.
  - No projeto foram disponibilizadas a implementação de duas interfaces: ICoisa e IConjunto.
  - O projeto possui o diretório “doc” que contém a documentação das duas interfaces. Abra o arquivo “index.html” para acessar o comportamento desejado de cada método das interfaces.
- Implemente:
  - Classe concreta denominada “ArrayConjunto”. Esta classe deve implementar “IConjunto”. Além disso, ArrayConjunto possui um array de “ICoisa” e permite armazenar vários objetos do tipo “ICoisa”.
  - São métodos de ArrayConjunto:
    - public ArrayConjunto(); → Construtor que cria um ArrayConjunto com tamanho padrão.
    - public ArrayConjunto(int initialSize); → Construtor que cria um ArrayConjunto com o tamanho inicial informado por argumento, desde que seja um valor válido, caso contrário é criado com tamanho padrão.
    - Outros métodos necessários para correta atuação da classe.

Exercício 6 (2 de 3)

- Atenção para o método add() de IConjunto.
  - Esse método sempre deve inserir a coisa no Conjunto. O array interno o ArrayConjunto não deve possuir buracos nem valores nulos. Além disso, se uma nova coisa for inserida e o array estiver cheio, deve-se, de alguma forma, dobrar o tamanho atual do array e depois armazenar a coisa.
- Atenção: não é permitido utilizar objetos do pacote Collection.
- Implemente:
  - Classe “Telefone” que possui:
    - Atributos:
      - Int ddd
      - Int prefixo
      - Int sufixo
    - Métodos:
      - toString(); gets(); sets(); construtores()

Exercício 6 (3 de 3)

- Implemente:
  - Classe “Contato” que possui:
    - Atributos:
      - String nome
      - String e-mail
      - ArrayConjunto telefones
    - Métodos:
      - gets(); sets(); construtores; outros.
  - Classe “Agenda” que possui:
    - Atributos:
      - ArrayConjunto contatos
    - Métodos:
      - gets(); sets(); construtores; outros.
  - Main
    - Implementar um sistema que utilizando as classes criadas faça uma agenda de contatos. Deve-se implementar um sistema de fácil utilização e intuitivo.
- Para entregar, compacte o projeto desenvolvido no Eclipse e carregue no moodle. Casos de cópias terão atribuídas a nota zero aos envolvidos.

Exercício 9

- No Exemplo16_Fila disponibilizado no repositório da disciplina foi implementada a interface IFila.
- Implementar duas propostas distintas de fila que implemente a interface.
- Implemente testes de unidade (mínimo de 10 por implementação) para as duas implementações propostas.
- Deve-se gerar o javadoc do projeto.
- Entregar link do projeto carregado no github. Não carregar um arquivo zipado e sim o diretório do projeto.
- Não é permitido o uso de Collection.