

DADOS DO COMPONENTE CURRICULAR

Nome: Programação Orientada a Objetos

Série/Período: 2º semestre

Carga Horária: 100 h/r (120 aulas)

Docente Responsável:

EMENTA

Apresentar o paradigma de orientação a objetos como uma técnica para elaboração de projetos e implementação de sistemas de software de qualidade. Abordará o conceito de abstração de dados, modelagem de sistemas utilizando-se objetos, herança, composição, polimorfismo e a aplicação destes conceitos em situações práticas com uma linguagem de programação orientada a objeto.

OBJETIVOS

Geral

Conhecer a metodologia de desenvolvimento orientada a objetos, assim como uma linguagem de programação que utilize este paradigma.

Específicos

- Instalar e configurar o ambiente de desenvolvimento;
- Importar bibliotecas para uso em projetos de programação;
- Explorar documentação de API da linguagem utilizada;
- Escrever programas utilizando os recursos disponíveis para tratamento de erros e exceções;
- Conhecer fundamentos sobre o desenvolvimento de aplicações cliente/servidor;
- Conhecer uma linguagem de programação para desenvolvimento de aplicações voltadas para servidores;
- Conhecer uma linguagem de programação, voltada para realizar a interface entre o usuário e aplicação servidora;
- Apresentar a metodologia de desenvolvimento orientada a objetos, mostrando as técnicas e ferramentas para desenvolvimento de sistemas;
- Entender as principais diferenças entre programas desenvolvidos utilizando a tradicional metodologia de programação estruturada e orientada a objetos;
- Familiarizar-se com os principais conceitos que determinam o paradigma orientado a objeto;
- Valorizar a importância da utilização de boas práticas de programação na elaboração de código fonte.

CONTEÚDO PROGRAMÁTICO

UNIDADE I

- Apresentação da disciplina e dos recursos disponíveis (2 h/a)
- Fundamentos da Linguagem Java (10 h/a)
 - ❑ Histórico e evolução da linguagem Java
 - ❑ Arquitetura da tecnologia Java
 - ❑ Características da linguagem
 - ❑ Produtos e API's Java
 - ❑ Escrevendo, compilando e executando aplicações Java
 - ❑ Estado da arte em ambientes de desenvolvimento e execução
 - ❑ Abstração, objetos e visão geral de conceitos de POO
 - ❑ Utilização de suporte ferramental adequado e configuração do ambiente de trabalho
 - ❑ Automação de tarefas rotineiras com ANT e noções de *refactoring*
- Programação Orientada a Objetos com Java (20 h/a)
 - ❑ Classes e criação de objetos
 - ❑ Membros de classe: atributos e métodos (classe e instância)
 - ❑ Abstração de dados e encapsulamento
 - ❑ Construtores e suas características
 - ❑ Definindo mensagens e interface de objetos
 - ❑ Sobrecarga e sobreposição de métodos
 - ❑ Ciclo de vida dos objetos (instanciação à destruição)
 - ❑ Classes Wrappers (*Boolean*, *Character*, *Short*, *Integer*, etc.)
 - ❑ Estruturação e Manipulação de Objetos em Java
 - ❑ Herança e noções de Polimorfismo
 - ❑ Modelagem de Objetos usando a linguagem UML
- Entrada e Saída Padrão de Dados em Java (4 h/a)
 - ❑ Entrada padrão de dados (classe *Console*)
 - ❑ Saída padrão de dados (*System.out*)
 - ❑ Entrada/Saída de dados GUI (classe *JOptionPane*)
- Tipos, Literais, Operadores e Controle de Fluxo (8 h/a)
 - ❑ Palavras reservadas da linguagem
 - ❑ Constantes e variáveis
 - ❑ Tipos primitivos e de referência
 - ❑ Expressões
 - ❑ Coerção, conversão e promoção de tipos
 - ❑ Operadores: atribuição, aritméticos, relacionais, lógicos e bits
 - ❑ Estruturas de controle de fluxo
 - ❑ Operador ‘==’ versus método *equals(Object o)*.
 - ❑ Enumerações versus Variáveis de Classe;

UNIDADE II

- Encapsulamento e Visibilidade (8 h/a)
- Definindo e refinando encapsulamento
- Modificadores de visibilidade: *public, protected, default* e *private*
- Criação de pacotes em Java
- Importação de classes
- *Arrays* e *Strings* (4 h/a)
- Arrays simples e multidimensionais
- Ordenação de arrays (classe *Arrays*)
- Características e manipulação de *Strings* e caracteres
- Classes *String*, *StringBuilder* e *StringBuffer*
- Arquivos e Fluxos de Dados em Java (10 h/a)
- Manipulação de dados em arquivos (pacote *java.io*)
- Arquivos (classe *File*), fluxos de entrada e saída em Java
- Leitura e gravação de Objetos e Textos em Java
- Tratamento de Erros e Exceções (12 h/a)
- Fundamentos acerca de tratamentos de erros e seus tipos
- Mecanismos *Try-Catch* e *Finally*
- Capturando e lançando exceções, finalizando exceções
- Exceções padrão em Java
- Criando novas exceções
- Exceções *Runnable*

UNIDADE III

- Reutilização com Herança e Composição de Objetos (16 h/a)
- Quando usar Herança ou Composição
- Técnicas de composição e associação de objetos
- Herança: vantagens e desvantagens sobre composição
- Polimorfismo com herança e com composição
- *Upcasting* e *Downcasting*.
- Boas práticas de programação
- Padrões de Projeto (essenciais) e boas práticas de programação
- Interfaces e Polimorfismo (10 h/a)
- Fundamentos sobre polimorfismo
- Aplicando polimorfismo com Interfaces
- Classes abstratas e métodos abstratos
- Mecanismo *Late binding* (vinculação dinâmica)
- Interfaces e Herança múltipla em Java
- Conectividade e Aplicações em Rede com Java (6 h/a)
- Classes *Socket* e *ServerSocket*
- Objetos Distribuídos com RMI
- Coleções (10 h/a)
- Coleções e API de estruturas de dados fundamentais
- Tipos Genéricos
- Listas, Mapas, Pilhas, Conjuntos e Filas
- Métodos Genéricos

METODOLOGIA DE ENSINO

- Aulas teóricas expositivas, aulas práticas, pesquisas individuais e em grupo, seminários, discussões e listas de exercícios.

AVALIAÇÃO DO PROCESSO DE ENSINO E APRENDIZAGEM

- Provas escritas, trabalhos práticos e teóricos, seminários e listas de exercícios;
- Serão realizadas ao menos três avaliações formais.

RECURSOS NECESSÁRIOS

- Quadro branco e pincel atômico; Projetor multimídia; Softwares específicos para edição, compilação e execução de programas.

PRÉ-REQUISITOS

16 – Algoritmos e Lógica de Programação.

BIBLIOGRAFIA

Básica

DEITEL, H.; DEITEL, P. **Java: Como Programar**. 8^a edição. Pearson Brasil, 2010.
SIERRA, K. **Use a cabeça! Java**. 2.ed. Alta Books, 2009.

Complementar

ECKEL, B. **Thinking in Java**. Prentice Hall, 2008. (<http://www.bruceeckel.com>).