

# Programa do Minicurso

## 1 Identificação

Disciplina: Minicurso De Java Moderno

Carga Horária: 20 horas-aula    Teóricas: 10    Práticas: 10

Período: Março a Maio de 2018.

## 2 Cursos (público alvo)

Ciência da Computação;

Sistemas de Informação;

Interessados na linguagem Java.

## 3 Requisitos

Nenhum.

## 4 Ementa

Características principais da linguagem; *Bytecode* e Processo de compilação; Introdução ao paradigma imperativo; Tipos primitivos; Inferência de tipo; Estrutura condicional *if*. Laço de repetição *for*; *Raw-Arrays*; Tempo de vida de variáveis: Escopo; *For-each*. Introdução a funções e procedimentos; Sobre-carga (*overload*) de função; Busca linear; Complexidade de Algoritmos; Notação assintótica; *Bubble-Sort*. Classe, Objeto e Atributo; Alocação de memória na JVM; *Garbage-Collection*; Tempo de vida de objetos; Cópia e Referência. Método Construtor; Boas práticas: API; Documentação de código. Pilha; Padrão de projeto: Composição; Exceções. Programação defensiva; *Early-Return*; Tipagem Forte; Asserções. Interfaces; Métodos *default*; Classes abstratas; Herança e problema do diamante; *Generics*. *Callback*; Tipos de dados para funções; Classes anônimas; Funções *lambda*. Sistema de pacotes Java; *Classpath*; Arquivo JAR; Arquivo JAR executável. Processamento de texto; Laço de repetição: *while*; Formato JSON; Operações em arquivos.

Passos de otimização. *Tail-call recursion*. *JIT Compilation*. Otimizações da JVM. Lista com vetor. Complexidade amortizada.

## 5 Objetivos

**Geral:** Auxiliar na aprendizagem das disciplinas de Programação Orientada a Objetos 1 e 2, demonstrando os recursos da linguagem, boas práticas, reusabilidade, *tradeoffs* entre soluções (performance, uso de memória, manutenção...) e as novidades nas versões mais recentes..

**Específico:** São objetivos do minicurso:

- Apresentar os recursos básicos da linguagem Java, dentre eles:
  - Classes e método `main`;
  - Declaração/instanciação de variáveis;
  - Ciclo de vida de variáveis;
  - Referência e cópia;
  - Métodos e construtores.

Além das características e funcionamento interno deles;

- Criar pequenos programas simples para mostrar o uso das estruturas básicas da programação imperativa (comparação, seleção, repetição, funções...);
- Melhorar o desempenho e entendimento dos alunos das disciplinas de Programação Orientada a Objetos 1 e 2;
- Instruir os participantes a respeito das boas práticas de programação, a fim de que façam códigos bem estruturados, reutilizáveis e legíveis, aplicando conceitos de código-limpo e modularidade;
- Introduzir noções de algoritmos e resolução de problemas computacionalmente;
- Expor formas de melhorar a manutenibilidade de software através de metodologias da programação moderna;
- Demonstrar o uso dos recursos recentes e progressão da linguagem Java;
- Demonstrar o funcionamento interno da JVM (*Java Virtual Machine*).

## 6 Conteúdo Programático

1. Introdução a Java (2 horas):
  - (a) Características principais da linguagem;
  - (b) *Bytecode* e Processo de compilação;
  - (c) Introdução ao paradigma imperativo;
  - (d) Tipos primitivos;
  - (e) Inferência de tipo;
  - (f) Estrutura condicional `if`.
2. Coleções (2 horas):
  - (a) Laço de repetição *for*;
  - (b) *Raw-Arrays*;
  - (c) Tempo de vida de variáveis: Escopo;
  - (d) *For-each*.
3. Introdução a Algoritmos (2 horas):
  - (a) Introdução a funções e procedimentos;
  - (b) Sobrecarga (*overload*) de função;
  - (c) Busca linear;
  - (d) Complexidade de Algoritmos;
  - (e) Notação assintótica;
  - (f) *Bubble-Sort*.
4. Tipos definidos por usuário (2 horas):
  - (a) Classe, Objeto e Atributo;
  - (b) Alocação de memória na JVM;
  - (c) *Garbage-Collection*;
  - (d) Tempo de vida de objetos;
  - (e) Cópia e Referência.
5. Métodos (2 horas):
  - (a) Método Construtor;
  - (b) Boas práticas: API;
  - (c) Documentação de código.

6. Introdução a Estruturas de Dados (2 horas):
  - (a) Pilha;
  - (b) Padrão de projeto: Composição;
  - (c) Exceções.
7. Técnicas de programação (2 horas):
  - (a) Programação defensiva;
  - (b) *Early-Return*;
  - (c) Tipagem Forte;
  - (d) Asserções.
8. Programação genérica e polimorfismo (2 horas):
  - (a) Interfaces;
  - (b) Métodos *default*;
  - (c) Classes abstratas;
  - (d) Herança e problema do diamante;
  - (e) *Generics*.
9. API funcional (2 horas):
  - (a) *Callback*;
  - (b) Tipos de dados para funções;
  - (c) Classes anônimas;
  - (d) Funções *lambda*.
10. Pacotes e bibliotecas (2 horas):
  - (a) Sistema de pacotes Java;
  - (b) *Classpath*;
  - (c) Arquivo JAR;
  - (d) Arquivo JAR executável.
11. Persistência (2 horas):
  - (a) Processamento de texto;
  - (b) Laço de repetição: *while*;
  - (c) Formato JSON;
  - (d) Operações em arquivos.

12. (Extra) Otimização (2 horas):
- (a) Passos de otimização;
  - (b) *Tail-call recursion*;
  - (c) *JIT Compilation*;
  - (d) Otimizações da JVM;
  - (e) Lista com vetor;
  - (f) Complexidade amortizada.

## 7 Cronograma

O cronograma obedece a mesma ordem do Conteúdo Programático, sendo cada tópico uma aula.

## Referências

- [1] Oracle. Java Platform SE 10 Documentation. <https://docs.oracle.com/javase/10/docs/api/index.html>.
- [2] Thomas H. Cormen, Clifford Stein, Ronald L. Rivest, and Charles E. Leiserson. *Introduction to Algorithms*. McGraw-Hill Higher Education, 2nd edition, 2001.
- [3] Raul Sidnei Wazlawick. *Engenharia de Software para Sistemas de Informação: Conceitos e práticas que fazem sentido*. 2012.