MINISTÉRIO DA EDUCAÇÃO
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA BAHIA-IFBA
PRÓ-REITORIA DE ENSINO

Plano de Ensino Cursos Técnicos

IDENTIFICAÇÃO		
EIXO TECNOLÓGICO: INFORMAÇÃO E COMUNICAÇÃO		
CURSO / FORMA ou GRAU / MODALIDADE: TÉCNICO EM INFORMÁTICA INTEGRADO AO ENSINO MÉDIO **CURSO: FORMA:**(x) integrado, ( ) subsequente		
COMPONENTE CURRÌCULAR: LINGUAGEM DE PROGRAMAÇÃO I		
ANO / SEMESTRE: 2025	**SEMESTRE ou ANO DA TURMA:** 2º ANO	**CARGA HORÀRIA:** 90H / 120 Aulas
TURNO: DIURNO	**TURMAS:** A e B	
DIRETOR(A) GERAL DO CAMPUS: JOSINALDO MANOEL CARDOSO		
DIRETOR (A) DE ENSINO: EVERALDO COSTA SILVA NETO		
DOCENTE(A): RAFAEL TOSTA SANTOS		
EMENTA		
Princípios da orientação a objetos: objetos, classes, atributos, métodos, mensagens, encapsulamento, herança, composição, polimorfismo, interfaces, acoplamento. Tratamento de exceções. Testes e depuração de código. Paradigmas de Programação		

OBJETIVOS
OBJETIVO GERAL DO CURSO
Modelar sistemas utilizando o paradigma de programação orientada a objetos, e demais conceitos relacionados. Escrever, compilar, depurar e testar programas na linguagem de programação Java; Aplicar estruturas de dados orientadas a objetos para modelar dados. Tratar erros e exceções em programas na linguagem Java.
OBJETIVOS DO COMPONENTE CURRICULAR
Objetivos estabelecidos para a Unidade I:
Familiarizar-se com o ambiente de desenvolvimento, metodologia e ferramentas utilizadas na disciplina. Interpretar corretamente enunciados de problemas e traduzi-los em algoritmos. Revisar e aplicar conceitos de lógica de programação (variáveis, operadores, expressões, estruturas condicionais e de repetição).

Conhecer os principais paradigmas de programação e situar o paradigma imperativo no contexto da linguagem Java.

Implementar programas simples em Java, explorando sintaxe básica, tipos de dados, operadores e estruturas de controle.

Desenvolver raciocínio lógico por meio da resolução de problemas práticos.

Objetivos estabelecidos para a Unidade II:

Identificar os conceitos fundamentais da POO (classes, objetos, atributos, métodos, encapsulamento).

Aplicar corretamente os modificadores de acesso e compreender sua importância no controle de visibilidade.

Implementar herança, polimorfismo e outros mecanismos de reutilização de código.

Compreender e aplicar relacionamentos entre classes (associação, agregação, composição) e os princípios de acoplamento e coesão.

Explorar classes abstratas, interfaces, atributos e métodos estáticos.

Desenvolver pequenos sistemas orientados a objetos integrando os conceitos estudados.

Consolidar práticas de programação limpa e organizada.

Objetivos estabelecidos para a Unidade III:

Utilizar adequadamente o tratamento de exceções para garantir robustez e segurança no código.

Trabalhar com estruturas de dados (List, Set, Map) e compreender o uso de iteradores e coleções genéricas.

Compreender e aplicar conceitos de programação orientada a eventos e interface gráfica com Swing ou JavaFX.

Manipular arquivos e implementar persistência de dados em sistemas simples.

Integrar POO, exceções e coleções em projetos práticos.

Desenvolver uma aplicação com interface gráfica e persistência em arquivos, aplicando boas práticas de programação, testes e depuração.

Apresentar o projeto final, demonstrando domínio dos conceitos e habilidades adquiridas.

METODOLOGIA

Propostas metodológicas para a Unidade I:

As aulas combinarão exposição dialogada e práticas em laboratório, com uso de situações-problema e exercícios graduais para desenvolver o raciocínio lógico e a compreensão da linguagem Java.
Serão utilizados materiais multimídia, ambientes de desenvolvimento (IDE) e exemplos práticos para consolidar os conceitos estudados.
A aprendizagem será acompanhada de forma contínua por meio de atividades práticas, exercícios de fixação e avaliações progressivas.

Propostas metodológicas para a Unidade II:

A metodologia integrará momentos teóricos e práticos, com análise de código, estudos de caso e desenvolvimento de projetos em grupo.
Os conteúdos serão apresentados de modo contextualizado e aplicado, permitindo o uso dos conceitos de classes, herança, polimorfismo e interfaces em situações reais.

A consolidação do aprendizado ocorrerá por meio de mini-projetos, exercícios aplicados e revisões de código, com foco na qualidade, organização e clareza das soluções desenvolvidas.

Propostas metodológicas para a Unidade III:

O processo de aprendizagem será orientado por projetos integradores, envolvendo o uso de POO, tratamento de exceções, coleções, manipulação de arquivos e interfaces gráficas.
As aulas incluirão demonstrações técnicas, implementação orientada e testes iterativos, estimulando o desenvolvimento de soluções completas e funcionais.
A consolidação dos conteúdos será feita por meio de projetos práticos, apresentações e análises de código, considerando a integração dos conceitos e a aplicação das boas práticas de programação.

CONTEÚDO PROGRAMÁTICO

Conteúdo programático para a Unidade I:

Apresentação da disciplina, metodologia, critérios de avaliação e ferramentas de desenvolvimento. (semana 1)

Interpretação de problemas e leitura de enunciados em programação. (semana 2)

Resolução de problemas: construção de algoritmos e pensamento computacional. (semana 3)

Revisão da lógica de programação: variáveis, operadores e expressões. (semana 4)

Avaliação 1: resolução de problemas e aplicação prática. (semana 5)

Paradigmas de programação: imperativo, funcional e orientado a objetos. (semana 6)

Introdução à linguagem Java: sintaxe básica e primeiro programa. (semana 7)

Tipos de dados primitivos, entrada e saída em Java. (semana 8)

Operadores e expressões em Java. (semana 9)

Estruturas condicionais em Java. (semana 10)

Estruturas de repetição em Java. (semana 11)

Projeto prático: implementação de algoritmos básicos em Java. (semana 12)

Revisão e sistematização dos conteúdos da unidade. (semana 13)

Conteúdo programático para a Unidade II:

Introdução à Programação Orientada a Objetos: conceitos de objetos e classes. (semana 14)

Atributos, estado e comportamento; métodos e mensagens. (semana 15)

Encapsulamento e modificadores de acesso. (semana 16)

Avaliação 2: exercícios de modelagem e implementação. (semana 17)

Herança e hierarquia de classes. (semana 18)

Polimorfismo e ligação dinâmica. (semana 19)

Relacionamentos entre classes: associação, agregação e composição; acoplamento e coesão. (semana 20)

Classes abstratas e interfaces. (semana 21)

Atributos e métodos estáticos. (semana 22)

Exercícios práticos de aplicação dos conceitos de POO. (semana 23)

Mini-projeto: desenvolvimento de um sistema simples orientado a objetos. (semana 24)

Revisão e sistematização dos conteúdos da unidade. (semana 25)

Avaliação 3: aplicação prática dos conceitos de POO. (semana 26)

Conteúdo programático para a Unidade III:

Tratamento de exceções: uso das estruturas try, catch, finally e throws. (semana 27)

Classes e bibliotecas de exceções em Java. (semana 28)

Estruturas de dados básicas: List, Set e Map. (semana 29)

Avaliação 4: exercícios de aplicação de estruturas e exceções. (semana 30)

Iteradores e uso prático de coleções. (semana 31)

Programação orientada a eventos (event listeners). (semana 32)

Introdução à construção de interfaces gráficas com Java (Swing/JavaFX). (semana 33)

Manipulação de arquivos em Java (entrada e saída). (semana 34)

Exercícios práticos sobre arquivos e coleções. (semana 35)

Integração de conceitos: POO, exceções e coleções. (semana 36)

Projeto prático: desenvolvimento de aplicação com interface gráfica e persistência em arquivos. (semana 37)

Testes e depuração de código; boas práticas de programação. (semana 38)

Apresentação dos projetos finais. (semana 39)

Avaliação 5: encerramento e síntese dos conteúdos da unidade. (semana 40)

AVALIAÇÃO

PROPOSTA AVALIATIVA PARA A UNIDADE I

ATIVIDADE AVALIATIVA I

Instrumento avaliativo: Lista de exercícios	**Realização em equipe:** Não	**Perspectiva pedagógica:** Diagnóstica

Objetivos para a atividade:

Identificar o nível de compreensão dos conceitos básicos de lógica e estruturação de algoritmos.

Desenvolver a habilidade de traduzir problemas em representações lógicas e estruturadas.

Estimular a aplicação dos primeiros conceitos de variáveis, operadores e expressões.

Parâmetros avaliativos considerados na apreciação da atividade:	**Domínio cognitivo:**	Relevante
	Cumprimento das tarefas:	Relevante
	Qualidade das tarefas:	Relevante
	Capacidade de produzir em equipe:	Não se aplica
	Autonomia:	Relevante

Critérios de Avaliação:

Clareza e correção lógica das soluções (40%)

Aplicação adequada dos conceitos estudados (30%)

Organização e legibilidade das respostas (20%)

Cumprimento dos prazos estabelecidos (10%)

Previsão de avaliação (em semana letiva): 5º semana	**Previsão da revisão dos resultados da atividade avaliativa:** 6º semana

ATIVIDADE AVALIATIVA II

Instrumento avaliativo: Atividade prática	**Realização em equipe:** Não	**Perspectiva pedagógica:** Formativa

Objetivos para a atividade:

Aplicar os conceitos de lógica na implementação de algoritmos em Java.

Compreender o funcionamento de variáveis, operadores e estruturas de controle.

Desenvolver o pensamento computacional e a autonomia na resolução de problemas.

Parâmetros avaliativos considerados na apreciação da atividade:	**Domínio cognitivo:**	Relevante
	Cumprimento das tarefas:	Relevante
	Qualidade das tarefas:	Relevante
	Capacidade de produzir em equipe:	Não se aplica
	Autonomia:	Relevante

Critérios de Avaliação:

Funcionamento correto dos programas (40%)

Clareza e organização do código (30%)

Aplicação dos conceitos estudados (20%)

Cumprimento do prazo e formatação adequada (10%)

Previsão de avaliação (em semana letiva): 10º semana	**Previsão da revisão dos resultados da atividade avaliativa:** 11º semana

ATIVIDADE AVALIATIVA III

Instrumento avaliativo: Projeto	**Realização em equipe:** Não	**Perspectiva pedagógica:** Cumulativa

Objetivos para a atividade:

Integrar os conceitos estudados em um projeto prático.

Desenvolver um programa funcional aplicando estruturas de decisão e repetição.

Demonstrar domínio técnico e compreensão lógica dos processos de execução do código.

Parâmetros avaliativos considerados na apreciação da atividade:	**Domínio cognitivo:**	Relevante
	Cumprimento das tarefas:	Relevante
	Qualidade das tarefas:	Relevante
	Capacidade de produzir em equipe:	Não se aplica
	Autonomia:	Relevante

Critérios de Avaliação:

Funcionamento e aderência ao problema proposto (40%)

Clareza e organização do código (25%)

Aplicação correta dos conceitos e boas práticas (25%)

Documentação e apresentação da solução (10%)

Previsão de avaliação (em semana letiva): 13º semana	**Previsão da revisão dos resultados da atividade avaliativa:** 14º semana

PROPOSTA AVALIATIVA PARA A UNIDADE II

ATIVIDADE AVALIATIVA I

Instrumento avaliativo: Atividade prática	**Realização em equipe:** Não	**Perspectiva pedagógica:** Formativa

Objetivos para a atividade:

Aplicar os princípios de abstração e encapsulamento em situações práticas de modelagem e implementação em Java.

Parâmetros avaliativos considerados na apreciação da atividade:	**Domínio cognitivo:**	Relevante
	Cumprimento das tarefas:	Relevante
	Qualidade das tarefas:	Relevante
	Capacidade de produzir em equipe:	Não se aplica
	Autonomia:	Relevante

Critérios de Avaliação:

1. Correção sintática e lógica do código (0 a 4,0 pontos)4,0 pontos – Código sem erros de sintaxe, executa corretamente e resolve integralmente o problema proposto; lógica clara e consistente.3,0 pontos – Pequenos erros ou inconsistências que não comprometem significativamente a execução ou a solução.2,0 pontos – Apresenta erros de lógica ou execução parcial do problema.1,0 ponto – Código com falhas graves de lógica ou que não resolve o problema adequadamente.0,0 ponto – Código não executa ou não atende ao problema proposto.

2. Uso adequado de classes e métodos (0 a 3,0 pontos)3,0 pontos – Estrutura bem organizada, com classes e métodos corretamente definidos e responsabilidades bem distribuídas.2,0 pontos – Organização razoável, com alguns problemas na definição ou uso de classes e métodos.1,0 ponto – Estrutura inadequada ou uso limitado de classes e métodos.0,0 ponto – Não utiliza adequadamente classes e métodos.

3. Encapsulamento e modularização (0 a 3,0 pontos)3,0 pontos – Uso correto de modificadores de acesso, boa separação de responsabilidades, baixo acoplamento e alta coesão.2,0 pontos – Aplica parcialmente os princípios de encapsulamento e modularização.1,0 ponto – Uso inadequado ou superficial dos princípios.0,0 ponto – Não evidencia encapsulamento nem modularização.

Síntese da composição da notaNota final =(Correção sintática e lógica) + (Uso de classes e métodos) + (Encapsulamento e modularização)Total máximo: 10,0 pontos

Previsão de avaliação (em semana letiva): 17º semana	**Previsão da revisão dos resultados da atividade avaliativa:** 18º semana

ATIVIDADE AVALIATIVA II

Instrumento avaliativo: Projeto	**Realização em equipe:** Sim	**Perspectiva pedagógica:** Cumulativa

Objetivos para a atividade:

Implementar uma estrutura de classes inter-relacionadas, utilizando herança e sobrescrita de métodos.

Parâmetros avaliativos considerados na apreciação da atividade:	**Domínio cognitivo:**	Relevante
	Cumprimento das tarefas:	Relevante
	Qualidade das tarefas:	Relevante

	Capacidade de produzir em equipe:	Relevante
	Autonomia:	Relevante

Critérios de Avaliação:

1. Correção do funcionamento do programa (0 a 3,0 pontos)3,0 pontos – O programa executa corretamente, sem erros, atendendo integralmente aos requisitos propostos.2,0 pontos – O programa funciona parcialmente, com pequenos erros ou limitações.1,0 ponto – O programa apresenta falhas significativas de execução ou lógica.0,0 ponto – O programa não executa ou não atende ao problema proposto.

2. Adequação à proposta (0 a 2,0 pontos)2,0 pontos – Atende completamente aos requisitos e objetivos definidos na atividade.1,0 ponto – Atende parcialmente à proposta, com lacunas ou desvios.0,0 ponto – Não atende à proposta estabelecida.

3. Clareza do código e documentação (0 a 2,0 pontos)2,0 pontos – Código bem organizado, legível, com nomes significativos e documentação/comentários adequados.1,0 ponto – Código compreensível, porém com problemas de organização ou documentação insuficiente.0,0 ponto – Código confuso, desorganizado ou sem documentação.

4. Aplicação correta dos conceitos de POO (0 a 3,0 pontos)3,0 pontos – Uso adequado de classes, métodos, encapsulamento e organização orientada a objetos.2,0 pontos – Aplica parcialmente os conceitos de POO, com algumas inconsistências.1,0 ponto – Uso limitado ou inadequado dos conceitos de POO.0,0 ponto – Não evidencia aplicação dos conceitos de POO.

Síntese da composição da notaNota final =(Correção do funcionamento) + (Adequação à proposta) + (Clareza e documentação) + (Aplicação de POO)Total máximo: 10,0 pontos

Previsão de avaliação (em semana letiva): 20º semana	**Previsão da revisão dos resultados da atividade avaliativa:** 21º semana

ATIVIDADE AVALIATIVA III

Instrumento avaliativo: Estudo de caso	**Realização em equipe:** Não	**Perspectiva pedagógica:** Cumulativa

Objetivos para a atividade:

Identificar falhas de design em um código fornecido e aplicar melhorias com base em boas práticas de POO.

Parâmetros avaliativos considerados na apreciação da atividade:	**Domínio cognitivo:**	Relevante
	Cumprimento das tarefas:	Relevante
	Qualidade das tarefas:	Relevante
	Capacidade de produzir em equipe:	Não se aplica
	Autonomia:	Relevante

Critérios de Avaliação:

1. Efetividade das melhorias propostas (0 a 4,0 pontos)4,0 pontos – As melhorias propostas resolvem de forma completa os problemas identificados, com ganhos claros de desempenho, legibilidade ou organização do código.3,0 pontos – As melhorias são adequadas e resolvem a maior parte dos problemas, com pequenos pontos a aprimorar.2,0 pontos – As melhorias são parcialmente efetivas, resolvendo apenas parte dos problemas identificados.1,0 ponto – As melhorias são superficiais ou pouco impactantes.0,0 ponto – Não apresenta melhorias relevantes ou não resolve os problemas propostos.

2. Clareza na justificativa das alterações (0 a 3,0 pontos)3,0 pontos – Justificativas claras, bem fundamentadas e coerentes com os princípios de programação e boas práticas.2,0 pontos – Justificativas compreensíveis, porém com limitações na fundamentação ou profundidade.1,0 ponto – Justificativas vagas, pouco claras ou pouco relacionadas às alterações realizadas.0,0 ponto – Não apresenta justificativas ou estas são incoerentes.

3. Qualidade do código final (0 a 3,0 pontos)3,0 pontos – Código bem estruturado, legível, organizado, seguindo boas práticas (nomes significativos, indentação adequada, modularização).2,0 pontos – Código funcional e relativamente organizado, com alguns problemas de legibilidade ou padronização.1,0 ponto – Código pouco organizado, difícil de compreender ou com más práticas evidentes.0,0 ponto – Código inadequado, desorganizado ou não funcional.

Síntese da composição da notaNota final =(Efetividade das melhorias) + (Clareza da justificativa) + (Qualidade do código final)Total máximo: 10,0 pontos

Previsão de avaliação (em semana letiva):	**Previsão da revisão dos resultados da atividade avaliativa:**
22º semana	23º semana

PROPOSTA AVALIATIVA PARA A UNIDADE III

ATIVIDADE AVALIATIVA I

Instrumento avaliativo: Lista de exercícios	**Realização em equipe:** Não	**Perspectiva pedagógica:** Formativa

Objetivos para a atividade:

Utilizar coleções e tratamento de erros para manipular dados de forma segura e eficiente em programas Java.

Parâmetros avaliativos considerados na apreciação da atividade:	**Domínio cognitivo:**	Relevante
	Cumprimento das tarefas:	Relevante
	Qualidade das tarefas:	Relevante
	Capacidade de produzir em equipe:	Não se aplica
	Autonomia:	Relevante

Critérios de Avaliação:

1. Correção e funcionamento do código (0 a 3,0 pontos)3,0 pontos – Código executa corretamente, sem erros, atendendo integralmente aos requisitos propostos.2,0 pontos – Código funcional, com pequenos erros ou limitações que não comprometem significativamente o resultado.1,0 ponto – Código com falhas relevantes de lógica ou execução parcial.0,0 ponto – Código não executa ou não atende à proposta.

2. Uso adequado das estruturas de dados (0 a 3,0 pontos)3,0 pontos – Seleção e uso corretos de estruturas (List, Set, Map), com boa adequação ao problema e manipulação eficiente.2,0 pontos – Uso adequado, porém com escolhas subótimas ou limitações na aplicação.1,0 ponto – Uso inadequado ou pouco eficiente das estruturas de dados.0,0 ponto – Não utiliza estruturas de dados adequadamente ou de forma incorreta.

3. Implementação correta de blocos try/catch (0 a 2,0 pontos)2,0 pontos – Tratamento de exceções adequado, com uso correto de try, catch (e quando necessário finally/throws), prevenindo falhas de execução.1,0 ponto – Uso parcial ou com limitações no tratamento de exceções.0,0 ponto – Não utiliza tratamento de exceções ou utiliza de forma incorreta.

4. Clareza e organização do código (0 a 2,0 pontos)2,0 pontos – Código limpo, bem estruturado, com nomes significativos, boa indentação e organização geral.1,0 ponto – Código compreensível, porém com problemas de organização ou padronização.0,0 ponto – Código confuso, desorganizado ou de difícil compreensão.

Síntese da composição da notaNota final =(Correção e funcionamento) + (Estruturas de dados) + (Tratamento de exceções) + (Clareza e organização)Total máximo: 10,0 pontos

Previsão de avaliação (em semana letiva): 27º semana	**Previsão da revisão dos resultados da atividade avaliativa:** 28º semana

ATIVIDADE AVALIATIVA II

Instrumento avaliativo: Projeto	**Realização em equipe:** Sim	**Perspectiva pedagógica:** Cumulativa

Objetivos para a atividade:

Desenvolver um projeto que combine POO, coleções e persistência de dados, com interface simples para interação do usuário.

Parâmetros avaliativos considerados na apreciação da atividade:	**Domínio cognitivo:**	Relevante
	Cumprimento das tarefas:	Relevante
	Qualidade das tarefas:	Relevante
	Capacidade de produzir em equipe:	Relevante
	Autonomia:	Relevante

Critérios de Avaliação:

1. Funcionalidade geral do sistema (0 a 3,0 pontos)3,0 pontos – O sistema funciona corretamente, sem falhas, atendendo integralmente aos requisitos propostos.2,0 pontos – O sistema apresenta funcionamento geral adequado, com pequenas falhas ou limitações.1,0 ponto – O sistema possui falhas significativas que comprometem parte das funcionalidades.0,0 ponto – O sistema não funciona ou não atende aos requisitos.

2. Organização do código-fonte (0 a 2,0 pontos)2,0 pontos – Código bem estruturado, modularizado, com separação adequada de responsabilidades e padronização consistente.1,0 ponto – Código compreensível, porém com problemas de organização ou modularização.0,0 ponto – Código desorganizado, sem estrutura clara.

3. Documentação (0 a 2,0 pontos)2,0 pontos – Documentação clara e completa (comentários, descrição de classes/métodos, instruções de uso).1,0 ponto – Documentação parcial ou superficial.0,0 ponto – Ausência

de documentação ou documentação inadequada.

4. Aplicação dos princípios de POO e boas práticas de programação (0 a 3,0 pontos)3,0 pontos – Uso adequado de encapsulamento, abstração, organização orientada a objetos e boas práticas (nomes significativos, baixo acoplamento, alta coesão).2,0 pontos – Aplicação parcial dos princípios e boas práticas, com algumas inconsistências.1,0 ponto – Uso limitado ou inadequado dos conceitos de POO e boas práticas.0,0 ponto – Não evidencia aplicação dos princípios de POO nem boas práticas.

Síntese da composição da notaNota final =(Funcionalidade) + (Organização do código) + (Documentação) + (POO e boas práticas)Total máximo: 10,0 pontos

Previsão de avaliação (em semana letiva): 31º semana	**Previsão da revisão dos resultados da atividade avaliativa:** 32º semana

ATIVIDADE AVALIATIVA III

Instrumento avaliativo: Produção oral	**Realização em equipe:** Sim	**Perspectiva pedagógica:** Diagnóstica

Objetivos para a atividade:

Apresentar e justificar as escolhas técnicas do projeto final, explicando o funcionamento do sistema e a aplicação dos conceitos aprendidos.

Parâmetros avaliativos considerados na apreciação da atividade:	**Domínio cognitivo:**	Relevante
	Cumprimento das tarefas:	Relevante
	Qualidade das tarefas:	Relevante
	Capacidade de produzir em equipe:	Relevante
	Autonomia:	Relevante

Critérios de Avaliação:

1. Clareza na exposição oral (0 a 2,5 pontos)2,5 pontos – Apresentação clara, objetiva e bem estruturada; domínio do conteúdo e boa comunicação.2,0 pontos – Apresentação compreensível, com pequenas falhas de organização ou fluidez.1,0 ponto – Apresentação com dificuldades de comunicação ou organização das ideias.0,0 ponto – Não apresenta ou apresentação incompreensível.

2. Coerência entre o código e o relatório (0 a 2,5 pontos)2,5 pontos – Total alinhamento entre o código desenvolvido e o que é descrito no relatório.2,0 pontos – Pequenas inconsistências entre código e documentação.1,0 ponto – Divergências relevantes entre código e relatório.0,0 ponto – Não há coerência entre código e relatório.

3. Capacidade de argumentação técnica (0 a 2,5 pontos)2,5 pontos – Justifica decisões de forma consistente, utilizando conceitos técnicos adequados.2,0 pontos – Apresenta justificativas, porém com limitações na fundamentação.1,0 ponto – Argumentação superficial ou pouco consistente.0,0 ponto – Não consegue justificar as decisões técnicas.

4. Estética e funcionalidade da aplicação apresentada (0 a 2,5 pontos)2,5 pontos – Aplicação funcional, estável e com boa apresentação visual/interface.2,0 pontos – Funcional, porém com limitações estéticas ou

pequenos problemas de usabilidade.1,0 ponto – Apresenta falhas de funcionamento ou interface pouco adequada.0,0 ponto – Aplicação não funcional ou inadequada.

Síntese da composição da notaNota final =(Exposição oral) + (Coerência código-relatório) + (Argumentação técnica) + (Estética e funcionalidade)Total máximo: 10,0 pontos

Previsão de avaliação (em semana letiva):	**Previsão da revisão dos resultados da atividade avaliativa:**
34º semana	35º semana

RECUPERAÇÃO DA APRENDIZAGEM:

Proposta de recuperação da aprendizagem para a Unidade I:

Objetivo: Consolidar lógica de programação, estruturas de controle e sintaxe em Java.
Instrumento: Lista de exercícios e pequeno projeto integrador em Java.
Forma de realização: Individual, com acompanhamento e feedback.
Critérios: Funcionamento correto, clareza e organização do código, aplicação correta dos conceitos.
Previsão: Semana seguinte à divulgação das avaliações.
Nota: Substitui a menor nota da unidade se for maior.

Previsão de avaliação (em semana letiva): 15º semana

Proposta de recuperação da aprendizagem para a Unidade II:

Objetivo: Compreender e aplicar conceitos de POO: classes, objetos, métodos, herança e polimorfismo.
Instrumento: Mini-projeto de POO e exercícios de modelagem UML.
Forma de realização: Individual ou em dupla, com feedback sobre implementação e design.
Critérios: Aplicação correta da POO, clareza e organização do código e modelagem, cumprimento das funcionalidades.
Previsão: Semana seguinte à divulgação das avaliações.
Nota: Substitui a menor nota da unidade se for maior.

Previsão de avaliação (em semana letiva): 26º semana

Proposta de recuperação da aprendizagem para a Unidade III:

Objetivo: Integrar POO, coleções, exceções, arquivos e interface gráfica em aplicações completas.
Instrumento: Projeto prático individual com implementação de conceitos avançados e melhorias em código existente.
Forma de realização: Individual, em laboratório ou remoto, com orientação pontual.
Critérios: Funcionalidade completa, organização e boas práticas no código, aplicação integrada dos conceitos.
Previsão: Semana seguinte à divulgação das avaliações.
Nota: Substitui a menor nota da unidade se for maior.

Previsão de avaliação (em semana letiva): 39º semana

RECURSOS E MATERIAIS DIDÁTICOS

Recursos didáticos e materiais necessários à Unidade I:

Computadores com ambiente de desenvolvimento configurado (Java JDK e IDE — VS Code, Eclipse ou IntelliJ);

Projetor multimídia e quadro branco para exposição e demonstrações de código;

Materiais digitais complementares, como slides, apostilas, vídeos curtos e tutoriais introdutórios;

Ambientes virtuais de aprendizagem ou repositórios on-line (GitHub, Google Classroom, etc.) para compartilhamento de atividades e códigos;

Compiladores e interpretadores on-line para testes rápidos de lógica e sintaxe.

Recursos didáticos e materiais necessários à Unidade II:

Ambiente de desenvolvimento Java (IDE e JDK atualizados);

Projetor multimídia e quadro branco digital ou físico para ilustração de diagramas e fluxos de classes;

Ferramentas de modelagem UML (como draw.io, PlantUML ou Lucidchart) para representação de classes e relações;

Repositórios de código (Git e GitHub) para controle de versões e colaboração entre equipes;

Material de apoio digital, incluindo exemplos de código comentado, exercícios orientados e guias de boas práticas de POO;

Plataformas on-line de codificação para reforçar a aprendizagem autônoma.

Recursos didáticos e materiais necessários à Unidade III:

Ambiente de desenvolvimento Java (IDE com suporte a Swing/JavaFX);

Bibliotecas e APIs Java padrão (java.io, java.util, javax.swing, javafx, etc.);

Ferramentas de versionamento e colaboração (Git, GitHub);

Ferramentas de depuração e teste de código integradas à IDE;

Materiais digitais de apoio, como tutoriais de interface gráfica, guias de tratamento de exceções e exemplos de manipulação de arquivos;

Projetor multimídia, laboratório de informática e acesso à internet para pesquisa e desenvolvimento dos projetos finais.

BIBLIOGRAFIA

BIBLIOGRAFIA BÁSICA:

DEITEL, P. DEITEL, H.; Java como Programar. 8. ed. Prentice Hall, 2010. ISBN 9788576055631.
CORNELL, G.; HORSTMANN, C. S.; Core Java. 8. ed. Pearson Education, 2010. ISBN 9788576053576.
SANTOS, R.; Introdução à Programação Orientada a Objetos Usando Java. 2. ed. Campus, 2013. ISBN 9788535274332.
MARTINS, F. M.; Projetos de POO em JAVA. FCA, 2014. ISBN 9789727227921.
FUGERI, S.; Java 8 – Ensino Didático – Desenvolvimento e Implementação de Aplicações. Érica, 2015. ISBN 9788536513041.
MANZANO, J. A. N. G.; JÚNIOR, R. A. C.; Java 8 Programação de Computadores. Érica, 2014. ISBN 9788536509266.

BIBLIOGRAFIA COMPLEMENTAR:

ROSSI, R.; Programação de Computadores em Java. 2. ed. Nova Terra, 2014. ISBN 9788561893361.
SCHILDT, H.; SKIEN, D.; Programação com Java, uma introdução abrangente.
HARBOUR, J. S.; Programação de Games com Java. Cengage Learning, 2009. ISBN 9788522107315;

MELO, A. A.; LUCKOW, D. H.; Programação Java para Web; 2. ed. NOVATEC, 2015. ISBN 9788575224458.
KATHY, S.; Use a Cabeça Java. Alfa Books, 2005. ISBN 9788576081739.
TUCKER, A. B.; NOONAN, R. E.; Linguagens de programação – Princípios e Práticas. 2. ed. McGraw Hill, 2009. ISBN 9788577260447.
FURGERI, S.; Modelagem de Sistemas Orientados a Objetos. Érica, 2013. ISBN 9788577260447.

BIBLIOGRAFIAS PARA APROFUNDAMENTO

Não informada.

OBSERVAÇÃO

Não existem observações registradas sobre este Plano de Ensino.

Obs: O Plano de Ensino só deverá ser arquivado no setor pedagógico após ser assinado pelas partes responsáveis desse documento.

Em 17 de junho de 2026.

Documento assinado eletronicamente por **VÂNIA SANTOS DA SILVA**, **Membro(a) da Comissão**, em 18/06/2026, às 10:52, conforme decreto nº 8.539/2015.

Documento assinado eletronicamente por **Jorge Souza Azevedo Moniz Barreto**, **Membro(a) da Comissão**, em 19/06/2026, às 08:38, conforme decreto nº 8.539/2015.

Documento assinado eletronicamente por **Everaldo Costa Silva Neto**, **Presidente da Comissão**, em 29/06/2026, às 09:11, conforme decreto nº 8.539/2015.

A autenticidade do documento pode ser conferida no site http://sei.ifba.edu.br/sei/controlador_externo.php?acao=documento_conferir&acao_origem=documento_conferir&id_orgao_acesso_externo=0 informando o código verificador **4934458** e o código CRC **834901EC**.