|  |  |  |  |
| --- | --- | --- | --- |
| **PLANO DE ENSINO** | | | |
| **CURSO** | | **MÓDULO** | **Componente Curricular Sigla** |
| Técnico em Desenvolvimento de Sistemas | | 1º | LOP |
| **COMPONENTE CURRICULAR** | **AULAS PREVISTAS** | **DOCENTE** | **COMPONENTE CURRICULAR** |
| Lógica de Programação e Algoritmos | 50 | Lucas e Wellington | Lógica de Programação e Algoritmos |
| **UNIDADE DE COMPETÊNCIA** | **OBJETIVO** | | |
|  | Proporcionar capacidades básicas e socioemocionais que permitem desenvolver algoritmos, por meio de lógica de programação e versionamento, para resolução de problemas. | | |

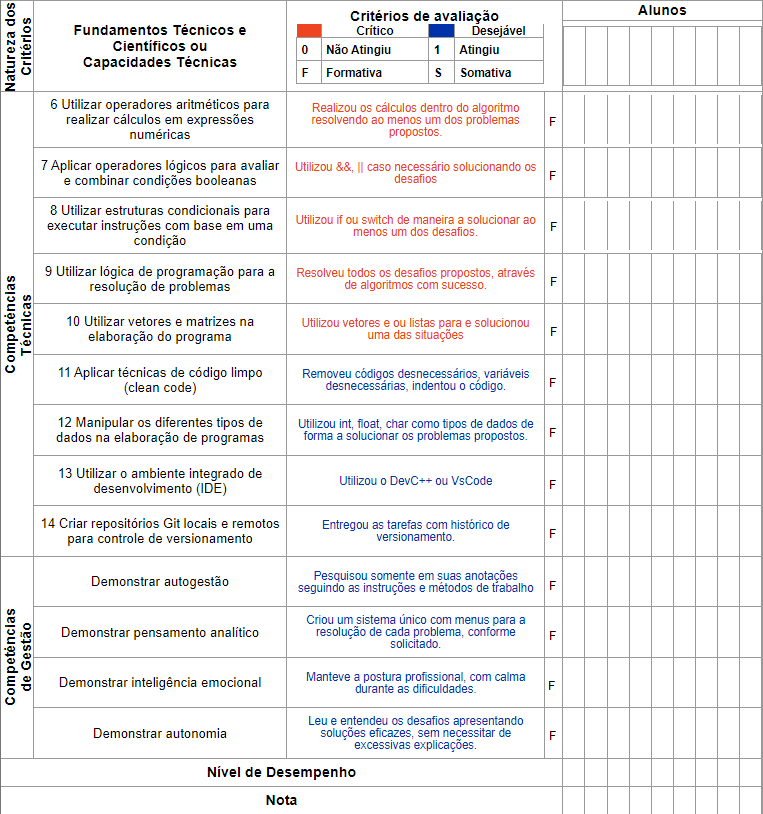
|  |
| --- |
| **Competências Específicas e Socioemocionais:**  **Capacidades Básicas** |
| 6 Utilizar operadores aritméticos para realizar cálculos em expressões numéricas  7 Aplicar operadores lógicos para avaliar e combinar condições booleanas  8 Utilizar estruturas condicionais para executar instruções com base em uma condição  9 Utilizar lógica de programação para a resolução de problemas  10 Utilizar vetores e matrizes na elaboração do programa  11 Aplicar técnicas de código limpo (clean code)  12 Manipular os diferentes tipos de dados na elaboração de programas  13 Utilizar o ambiente integrado de desenvolvimento (IDE)  14 Criar repositórios Git locais e remotos para controle de versionamento |

|  |
| --- |
| **Capacidades Socioemocionais** |
| 1 Demonstrar autogestão  2 Demonstrar pensamento analítico  3 Demonstrar inteligência emocional  4 Demonstrar autonomia |

|  |
| --- |
| **CONHECIMENTOS** |
| 3.6. Vetores  3.7. Matrizes  3.8. Técnicas de código limpo (clean code)  4 Ambiente de desenvolvimento  4.1. Instalação e configuração  4.2. Gerenciamento de dependências  4.3. Recursos e interfaces  5 Manipulação de arquivos  5.1. Escrita  5.2. Leitura  6 Git - Sistema de controle de versões distribuído  6.1. Evolução  6.2. Comandos iniciais  6.2.1.Init  6.2.2.Add  6.2.3.Status  6.2.4.Config  6.2.5.Commit  6.2.6.Log  7 Versionamento em nuvem  7.1. Serviços  7.1.1.Github  7.1.2.BitBucket  7.1.3.Azure Repository  7.2. Pull Requests  7.3. Resolução de conflitos |

|  |  |
| --- | --- |
| **ATIVIDADE** | **SITUAÇÃO DE APRENDIZAGEM - Formativa** |
| **TEXTO** | |
| **Obs:** Além da situação-problema a seguir, exercícios com outras temáticas, serão propostos a fim de desenvolver habilidades de abstração.  **Contextualização**: A escola SENAI de Jaguariúna necessita de um sistema de controle de acesso as suas dependências. Este sistema precisa classificar o mais detalhadamente possível todas as pessoas que entram e saem diariamente deste prédio.  **Desafio**: Como técnico em desenvolvimento de sistemas, você foi contratado para fazer parte de uma equipe que criará o sistema e o instalará em um computador localizado na portaria.  Os seguintes requisitos do sistema foram levantados pelo analista de sistemas:  - Classificar todos os visitantes de acordo com sua função;  - Cadastrar cada um conforme sua classificação;  - Registrar qual o motivo da visita a escola;  - Quantificar cada tipo de pessoa e calcular o total.   * Exemplos de tipos de pessoas que acessam o prédio conforme suas funções: Professores, Alunos, Visitantes, Representantes de Empresas, Pais de alunos. * Exemplo de motivos de visita: Ministrar aulas, realizar treinamento, Fazer matrícula, Realizar trabalhos administrativos, Fazer Visitas técnicas. * O analista escolheu a linguagem JavaScript e solicita que como técnico, membro da equipe, você crie as classes que serão necessárias para iniciar o desenvolvimento do sistema, conforme os requisitos apresentados e aplique as convenções corretas, já que outros integrantes da equipe desenvolverão outras partes do sistema.   **Entregas**:   * Apresente um diagrama de classes representando apenas os relacionamentos de herança. * Através da plataforma IDE Eclipse apresente o pacote contendo as classes criadas e seus respectivos atributos e métodos, colocando em prática as técnicas de abstração estudadas. * O código deverá apresentar a correta aplicação das técnicas de herança, agregação e/ou composição e poliformismo. | |

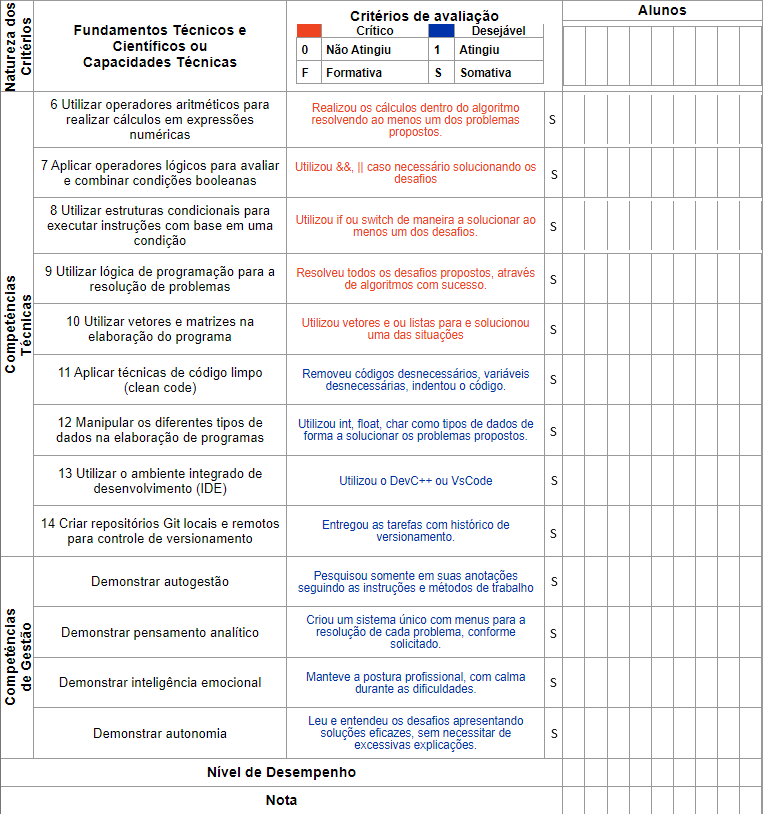
**INSTRUMENTO DE REGISTRO DE AVALIAÇÃO FORMATIVA**



|  |  |  |
| --- | --- | --- |
| **PLANO DE AULA** | | |
| **CONHECIMENTOS** | **ESTRATÉGIAS DE ENSINO** | **INTERVENÇÕES MEDIADORAS** |
| 3.6. Vetores  3.7. Matrizes  3.8. Técnicas de código limpo (clean code) | Exposição dialogada com auxílio de quadro branco e projetor multimídia.  Lista de exercícios. | O que é vetor?  Qual a diferença entre vetor e matriz?  Como realizar a limpeza de um código?  Por que limpar um código? |
| 4 Ambiente de desenvolvimento  4.1. Instalação e configuração | Exposição dialogada e Demonstrações. | O que é um ambiente de desenvolvimento?  Qual a diferença entre dev e prod? |
| 4.2. Gerenciamento de dependências  4.3. Recursos e interfaces | Demonstrações e Atividades Práticas | O que são dependências?  Por que utilizar um gerenciador de dependências? |
| 5 Manipulação de arquivos  5.1. Escrita  5.2. Leitura | Exposição dialogada, exemplos de casos de uso e Atividades Práticas. | O que é um arquivo?  Qual a diferença entre arquivo e pasta?  O que é um arquivo binário?  O que é um arquivo de texto? |
| 6 Git - Sistema de controle de versões distribuído  6.1. Evolução  6.2. Comandos iniciais  6.2.1.Init  6.2.2.Add  6.2.3.Status | Exposição dialogada, exemplos de casos de uso e Atividades Práticas. | Quem criou o git?  O que é versionamento?  Por que controlar as versões de um software?  Qual a diferença entre github e git? |
| 6.2.4.Config  6.2.5.Commit  6.2.6.Log | Demonstrações e Atividades Práticas | O que é um commit?  Como ver os históricos das alterações do software? |
| 7 Versionamento em nuvem  7.1. Serviços  7.1.1.Github  7.1.2.BitBucket  7.1.3.Azure Repository | Exposição dialogada e Demonstrações. | O que é github?  O que é nuvem? |
| 7.2. Pull Requests  7.3. Resolução de conflitos | Desenvolvimento de Projeto e Desafios em grupo. | O que é pull request?  Como contribuir em projetos open source? |

|  |  |
| --- | --- |
| **ATIVIDADE** | **SITUAÇÃO DE APRENDIZAGEM – Somativa** |
| **TEXTO** | |
| **Obs:** Aplicada como forma de avaliação Somativa.  **Contextualização**: A empresa LAMBDEV, produtora de bebidas, inaugurou um novo depósito e solicitou ao departamento de tecnologia da informação um sistema que classifique o estoque de produtos finais que serão armazenados neste novo ambiente.  **Desafio**: Como técnico em desenvolvimento de sistemas, você foi contratado para fazer parte de uma equipe que criará o sistema e o instalará em um computador localizado no estoque.  Os seguintes requisitos do sistema foram levantados pelo analista de sistemas:  - Classificar todas as bebidas de acordo com seu tipo;  - Cadastrar cada uma conforme sua classificação;  - Registrar as datas de entrada e saída de cada produto;  - Quantificar cada tipo de produto e calcular o total.   * Exemplos de tipos de bebidas fabricadas pela LAMBDEV: Alcoólica, não alcoólica, cerveja, refrigerante, cachaça e suco. * Basicamente as ações que envolvem estes produtos são entradas e saídas do estoque. * O analista escolheu a linguagem JavaScript e solicita que como técnico, membro da equipe, você crie as classes que serão necessárias para iniciar o desenvolvimento do sistema, conforme os requisitos apresentados e aplique as convenções corretas, já que outros integrantes da equipe desenvolverão outras partes do sistema.   **Entregas**:   * Apresente um diagrama de classes representando apenas os relacionamentos de herança. * Através da plataforma IDE Eclipse apresente o pacote contendo as classes criadas e seus respectivos atributos e métodos, colocando em prática as técnicas de abstração estudadas. * O código deverá apresentar a correta aplicação das técnicas de herança e poliformismo. | |

**INSTRUMENTO DE REGISTRO DE AVALIAÇÃO FORMATIVA**



Instrumento de avaliação / exercícios / Atividades.

|  |  |  |  |
| --- | --- | --- | --- |
| **Componente curricular:** Lógica de Programação e Algoritmos | **Atividade:** Lista de Exercícios 4 | | **Nota:** |
| **Unidade:** SENAI Jaguariúna | **Data: / /** | |  |
| **Aluno:** | **N°:** | **Turma:** |
|  | | | |

Tabela

Descrição gerada automaticamente



|  |  |  |  |
| --- | --- | --- | --- |
| ELABORAÇÃO | DATA | APROVAÇÃO | DATA |
| **Júlio** | **/ /** |  | **/ /** |

***ANEXOS:***

Exercícios ou qualquer material utilizado no dia a dia.

Atividades\_6\_a\_8\_Orientado\_Objeto;

CRONOGRAMA (deixar por último) O cronograma deve ser atualizado a cada turma nova.

**Cronograma e Acompanhamento de Distribuição de Aulas**

**Curso**: Técnico em Desenvolvimento de Sistemas

**Componente Curricular:** Fundamentos De Programação Orientada A Objeto

**Turma:** 1DES

**Professor**: Reenye e Wellington Lima 2º Sem. 2023

|  |  |  |
| --- | --- | --- |
| Programa Analítico | Dia/Mês/Ano  (Previsto) | Dia/Mês/Ano  (Realizado) |
| 3.6. Vetores  3.7. Matrizes | 08/04/2024 |  |
| 3.8. Técnicas de código limpo (clean code) | 15/04/2024 |  |
| 4 Ambiente de desenvolvimento  4.1. Instalação e configuração  4.2. Gerenciamento de dependências  4.3. Recursos e interfaces | 22/04/2024 |  |
| 5 Manipulação de arquivos  5.1. Escrita  5.2. Leitura | 29/04/2024 |  |
| 6 Git - Sistema de controle de versões distribuído  6.1. Evolução  6.2. Comandos iniciais  6.2.1.Init | 06/05/2024 |  |
| 6.2.2.Add  6.2.3.Status  6.2.4.Config  6.2.5.Commit  6.2.6.Log | 13/05/2024 |  |
| 7 Versionamento em nuvem  7.1. Serviços  7.1.1.Github  7.1.2.BitBucket  7.1.3.Azure Repository | 20/05/2024 |  |
| 7.2. Pull Requests  7.3. Resolução de conflitos | 27/05/2024 |  |
| Avaliação Formativa | 03/06/2024 |  |
| Avaliação Somativa | 10/06/2024 |  |
| Atividades de recuperação | 17/06/2024 |  |
| Elaborado por: Lucas e Wellington  Data: 23/01/2024 | | |