|  |
| --- |
| Análise e Desenvolvimento de Sistemas  **ALGORITMOS E ESTRUTURA DE DADOS EM PYTHON**  RELATÓRIO DE AULAS PRÁTICAS  Nome: Igor Pereira da Silva    RA: 2537577    Polo de matrícula: São Paulo – Anchieta    Local da realização da Aula Prática: UNIP - Anchieta    Ano da postagem:2025  PROFESSOR DA AULA PRÁTICA: Fábio Assis |

**ATIVIDADE (ROTEIRO OU AULA): IMPLEMENTANDO ALGORITMOS EM PYTHON**

**ORIENTAÇÕES**:

Cada aluno (ou equipe) deve produzir um relatório curto (1 a 2 páginas) contendo:

**Resumo Teórico:**

* Explicar, com palavras próprias, o que é lógica de programação e por que ela é importante.

Lógica de programação é basicamente a forma de pensar que você usa para resolver problemas passo a passo, como se estivesse dando instruções para um amigo fazer alguma coisa. Quando a gente programa, a gente precisa organizar essas instruções de um jeito que o computador entenda e consiga executar sem erro. Ela é importante porque, sem essa forma de pensar clara e organizada, o programa pode não funcionar direito, ficar confuso ou travar. Além disso, a lógica ajuda a gente a pensar melhor na vida também, porque ensina a dividir problemas grandes em partes menores, encontrar soluções e ser mais organizado.

* Mencionar brevemente o que é pseudocódigo e fluxograma e como ajudam na organização de ideias.

O **pseudocódigo** é uma maneira de descrever a lógica de um algoritmo usando uma linguagem simples e próxima do português, sem se preocupar com a sintaxe específica de uma linguagem de programação. Ele serve para **planejar passo a passo** como um problema será resolvido, tornando mais fácil entender e organizar as ideias antes de programar. O **fluxograma** é uma representação gráfica do algoritmo, usando símbolos padronizados para indicar ações, decisões e fluxos do processo. Ele ajuda a visualizar a sequência de passos e as decisões envolvidas na resolução de um problema. Ambos são ferramentas que **organizam ideias,** permitindo que o raciocínio seja estruturado de forma clara e lógica, facilitando a implementação de algoritmos e a comunicação entre pessoas envolvidas no desenvolvimento.

* Citar as vantagens de usar Python para aprender programação.  Definição de algoritmo, variável, condicional, laço;

**Vantagens de usar Python para aprender programação:**  
Python é uma linguagem de programação amplamente recomendada para iniciantes porque possui uma sintaxe simples e clara, parecida com a linguagem natural. Isso permite que os estudantes se concentrem na lógica do programa sem se perder em detalhes complexos de sintaxe. Além disso, Python possui uma grande quantidade de bibliotecas e recursos que facilitam desde cálculos matemáticos até desenvolvimento web e manipulação de dados, tornando o aprendizado mais prático e motivador.

**Definição de conceitos fundamentais:**

* **Algoritmo:** É uma sequência de passos lógicos e bem definidos que devem ser seguidos para resolver um problema ou realizar uma tarefa. Por exemplo, uma receita de bolo é um algoritmo, pois indica cada passo necessário para o resultado final.
* **Variável:** É um espaço na memória do computador que armazena um valor que pode ser alterado durante a execução do programa. Por exemplo, idade = 20 define uma variável chamada idade com o valor 20.
* **Condicional:** Estrutura que permite que o programa tome decisões com base em condições. Exemplo em Python:

**Vantagens de usar Python para aprender programação:**  
Python é uma linguagem de programação amplamente recomendada para iniciantes porque possui uma sintaxe simples e clara, parecida com a linguagem natural. Isso permite que os estudantes se concentrem na lógica do programa sem se perder em detalhes complexos de sintaxe. Além disso, Python possui uma grande quantidade de bibliotecas e recursos que facilitam desde cálculos matemáticos até desenvolvimento web e manipulação de dados, tornando o aprendizado mais prático e motivador.

**Definição de conceitos fundamentais:**

* **Algoritmo:** É uma sequência de passos lógicos e bem definidos que devem ser seguidos para resolver um problema ou realizar uma tarefa. Por exemplo, uma receita de bolo é um algoritmo, pois indica cada passo necessário para o resultado final.
* **Variável:** É um espaço na memória do computador que armazena um valor que pode ser alterado durante a execução do programa. Por exemplo, idade = 20 define uma variável chamada idade com o valor 20.
* **Condicional:** Estrutura que permite que o programa tome decisões com base em condições. Exemplo em Python:
* **Laço (ou loop):** Estrutura que repete um bloco de código enquanto uma condição for verdadeira, economizando esforço e evitando a repetição manual de comandos.

**Código-Fonte Comentado:**

* Inserir o *código*-*fonte* *completo* da atividade proposta.
* Comentar as principais linhas, ressaltando o uso de print(), input(), variáveis etc.

**REFERÊNCIAS**: O aluno deverá colocar o nome dos livros e *sites* utilizados para a realização da atividade. As regras para fazer referência ao material utilizado deverão ser de acordo com a ABNT.

# CRITÉRIOS PARA AVALIAÇÃO

|  |  |  |
| --- | --- | --- |
| **Critério** | **Peso** | **Descrição** |
| **Qualidade do Resumo Teórico** | 2,0 | Clareza e correlação correta entre conceitos. |
| **Estrutura e**  **Organização do Código**  **Funcionamento da Solução** | 3,0 | O código deve estar indentado corretamente, usar nomes de variáveis adequados e conter comentários informativos (quando necessários). |
| 3,0 | Execução sem erros, entrada e saída corretas, uso de condições e laços. |
| **Criatividade e**  **Aprimoramentos** | 2,0 | Adição de perguntas extras, uso de strings multilinha, personalização das mensagens e outras melhorias que demonstrem domínio do conteúdo. |

**Nota** **Final:** Será a soma dos valores obtidos em cada critério. Alunos ou equipes que não cumprirem os requisitos mínimos de funcionamento do código ou não entregarem o relatório dentro do prazo terão sua nota diminuída proporcionalmente.

**DA ATIVIDADE (ROTEIRO OU AULA): ESTRUTURAS DE DADOS**

**LINEARES EM PYTHON: LISTAS, PILHAS, FILAS E EFICIÊNCIA. NOTAÇÃO BIG-O**

**ORIENTAÇÕES**:

Cada aluno deve produzir um relatório sucinto (2 a 3 páginas) contendo:

**Resumo Teórico**

* Definição de listas, pilhas, filas e introdução à notação Big-O.
* Código-fonte comentado das soluções desenvolvidas.
* Reflexão sobre desafios encontrados e critérios para selecionar cada estrutura;

**Códigos-Fontes Comentados**

* Inserir o *código*-*fonte* *completo* da atividade proposta.
* Comentar as principais linhas, ressaltando o uso das estruturas de dados.

**REFERÊNCIAS**: O aluno deverá colocar o nome dos livros e *sites* utilizados para a realização da atividade. As regras para fazer referência ao material utilizado deverão ser de acordo com a ABNT.

# CRITÉRIOS PARA AVALIAÇÃO

|  |  |  |
| --- | --- | --- |
| **Critério** | **Peso** | **Descrição** |
| **Clareza do**  **Resumo**  **Teórico** | 2,0 | Clareza e correlação correta entre conceitos. |
| **Organização e**  **Comentários do**  **Código** | 3,0 | O código deve estar indentado corretamente, usar nomes de variáveis adequados e conter comentários informativos (quando necessários). |
| **Funcionalidade do Código** | 2,0 | Execução sem erros, entrada e saída corretas. |
| **Criatividade e**  **Aprimoramentos** | 3,0 | Funcionalidades extras, simulações realistas |

**Nota** **Final:** Será a soma dos valores obtidos em cada critério. Alunos ou equipes que não cumprirem os requisitos mínimos de funcionamento do código ou não entregarem o relatório dentro do prazo terão sua nota diminuída proporcionalmente.

**ESTRUTURAS DE DADOS NÃO LINEARES - ÁRVORES E GRAFOS EM PYTHON**

**ORIENTAÇÕES**:

Cada aluno deve produzir um relatório curto (2 a 3 páginas), contendo:

**Resumo Teórico:**

* Definição de árvore e grafo, vantagens de cada estrutura.
* Comentário sobre a escolha de Python para ilustrar algoritmos de percursos;

**Códigos Desenvolvidos:**

* Inserir o *código*-*fonte* *completo* da atividade proposta.
* Comentar as principais linhas, ressaltando criação de vértices, filas, laços e condições de parada.

**Conclusão:**

* Descrever se a aula ajudou no entendimento de árvores e grafos.

**REFERÊNCIAS**: O aluno deverá colocar o nome dos livros e *sites* utilizados para a realização da atividade. As regras para fazer referência ao material utilizado deverão ser de acordo com a ABNT.

# CRITÉRIOS PARA AVALIAÇÃO

|  |  |  |  |
| --- | --- | --- | --- |
| **Critério** | **Peso** | | **Descrição** |
| **Clareza do**  **Resumo**  **Teórico** |  | 2,0 | Clareza e correlação correta entre conceitos. |
| **Organização e**  **Comentários do Código** |  | 3,0 | O código deve estar indentado corretamente, usar nomes de variáveis adequados e conter comentários informativos (quando necessários). |
| **Funcionalidade do Código** |  | 2,0 | Execução sem erros, entrada e saída corretas. |
| **Criatividade e**  **Aprimoramentos** |  | 3,0 | Inclusão de pesos, tratamento de exceções ou visualização gráfica simples. |

**Nota** **Final:** Será a soma dos valores obtidos em cada critério. Alunos ou equipes que não cumprirem os requisitos mínimos de funcionamento do código ou não entregarem o relatório dentro do prazo terão sua nota diminuída proporcionalmente.

**ALGORITMOS DE ORDENAÇÃO**

**EM PYTHON (Bubble Sort, Selection Sort, Insertion Sort, Merge Sort e Quick Sort)**

**ORIENTAÇÕES**:

Cada aluno deve produzir um relatório sintético (2 a 3 páginas) contendo:

**Resumo Teórico:**

* Definir ordenação e justificar sua relevância em ciência da computação.
* Explicar diferenças conceituais entre algoritmos quadráticos e loglineares.
* Comentar vantagens e limitações de cada método; **Códigos Desenvolvidos:**
* Inserir as implementações completas, destacando linhas decisivas (trocas, partições, fusões).
* Incluir tabela dos tempos obtidos nas medições.

**REFERÊNCIAS**: O aluno deverá colocar o nome dos livros e *sites* utilizados para a realização da atividade. As regras para fazer referência ao material utilizado deverão ser de acordo com a ABNT.

# CRITÉRIOS PARA AVALIAÇÃO

|  |  |  |
| --- | --- | --- |
| **Critério** | **Peso** | **Descrição** |
| **Clareza do**  **Resumo**  **Teórico** | 2,0 | Clareza e correlação correta entre conceitos. |
| **Organização e**  **Comentários do**  **Código** | 3,0 | O código deve estar indentado corretamente, usar nomes de variáveis adequados e conter comentários informativos (quando necessários). |
| **Funcionalidade do Código** | 3,0 | Execução correta e apresentação dos resultados de tempo. |
| **Criatividade e**  **Aprimoramentos** | 2,0 | Introdução de visualização gráfica simples, pivô aleatório ou otimizações. |

**Nota** **Final:** Será a soma dos valores obtidos em cada critério. Alunos ou equipes que não cumprirem os requisitos mínimos de funcionamento do código ou não entregarem o relatório dentro do prazo terão sua nota diminuída proporcionalmente.

**ALGORITMOS DE PESQUISA - BUSCA LINEAR E BUSCA BINÁRIA EM PYTHON**

**ORIENTAÇÕES**:

Cada aluno deve produzir um relatório (2 a 3 páginas) contendo:

**Resumo Teórico:**

* Explicar a diferença entre pesquisa exaustiva e pesquisa por divisão.
* Comparar custos de busca linear e binária em termos de complexidade e de requisitos de ordenação; **Códigos Desenvolvidos:**
* Inserir implementações completas de ambos os métodos.
* Apresentar tabela com tempos coletados para três tamanhos distintos de listas.

**REFERÊNCIAS**: O aluno deverá colocar o nome dos livros e *sites* utilizados para a realização da atividade. As regras para fazer referência ao material utilizado deverão ser de acordo com a ABNT.

# CRITÉRIOS PARA AVALIAÇÃO

|  |  |  |
| --- | --- | --- |
| **Critério** | **Peso** | **Descrição** |
| **Clareza do**  **Resumo**  **Teórico** | 2,0 | Clareza conceitual e uso correto de terminologia. |
| **Organização e**  **Comentários do**  **Código** | 3,0 | O código deve estar indentado corretamente, usar nomes de variáveis adequados e conter comentários informativos (quando necessários). |
| **Funcionalidade do Código** | 3,0 | Execução correta e apresentação dos resultados de tempo. |
| **Criatividade e**  **Aprimoramentos** | 2,0 | Execução correta e apresentação dos resultados de tempo. |

**Nota** **Final:** Será a soma dos valores obtidos em cada critério. Alunos ou equipes que não cumprirem os requisitos mínimos de funcionamento do código ou não entregarem o relatório dentro do prazo terão sua nota diminuída proporcionalmente.

**ATIVIDADE (ROTEIRO OU AULA): TABELAS DE DISPERSÃO (HASH TABLES) E OS HEAPS EM PYTHON**

**ORIENTAÇÕES**:

Cada aluno deve produzir um relatório (2 a 3 páginas) contendo:

**Resumo Teórico:**

* Definir tabelas de dispersão, explicar colisões e tratamentos.
* Descrever heaps binários e justificar eficiência em filas de prioridade;

**Códigos Desenvolvidos:**

* Inserir implementações das medições solicitadas, com observações sobre linhas-chave (cálculo de hash, heappush, heappop).
* Apresentar tabela dos tempos obtidos em cada experimento.

**REFERÊNCIAS**: O aluno deverá colocar o nome dos livros e *sites* utilizados para a realização da atividade. As regras para fazer referência ao material utilizado deverão ser de acordo com a ABNT.

# CRITÉRIOS PARA AVALIAÇÃO

|  |  |  |
| --- | --- | --- |
| **Critério** | **Peso** | **Descrição** |
| **Clareza do**  **Resumo**  **Teórico** | 2,0 | Precisão conceitual e clareza de exposição. |
| **Organização e**  **Comentários do**  **Código** | 3,0 | O código deve estar indentado corretamente, usar nomes de variáveis adequados e conter comentários informativos (quando necessários). |
| **Funcionalidade do Código** | 3,0 | Execução sem erros, coleta e exibição confiável dos tempos. |
| **Criatividade e**  **Aprimoramentos** | 2,0 | Implementação de heap de máx-prioridade, visualizações simples ou análise de fator de carga. |

**Nota** **Final:** Será a soma dos valores obtidos em cada critério. Alunos ou equipes que não cumprirem os requisitos mínimos de funcionamento do código ou não entregarem o relatório dentro do prazo terão sua nota diminuída proporcionalmente.

**ATIVIDADE (ROTEIRO OU AULA): ALGORITMOS DE GRAFOS - DIJKSTRA, BELLMAN-FORD, KRUSKAL E PRIM EM PYTHON**

**ORIENTAÇÕES**:

Cada aluno deve produzir um relatório (2 a 3 páginas) contendo:

**Resumo Teórico:**

* Explicar diferenças entre caminhos mínimos de fonte única e árvores geradoras mínimas.
* Apontar condições de aplicabilidade (pesos negativos, denso × esparso);

**Códigos Desenvolvidos:**

* Incluir implementações completas, indicando linhas de relaxamento e

união-busca.

* Apresentar tabela de tempos e pesos totais das árvores.

**REFERÊNCIAS**: O aluno deverá colocar o nome dos livros e *sites* utilizados para a realização da atividade. As regras para fazer referência ao material utilizado deverão ser de acordo com a ABNT.

# CRITÉRIOS PARA AVALIAÇÃO

|  |  |  |
| --- | --- | --- |
| **Critério** | **Peso** | **Descrição** |
| **Clareza do**  **Resumo**  **Teórico** | 2,0 | Precisão conceitual e clareza de exposição. |
| **Organização e**  **Comentários do**  **Código** | 3,0 | O código deve estar indentado corretamente, usar nomes de variáveis adequados e conter comentários informativos (quando necessários). |
| **Funcionalidade do Código** | 3,0 | Execução sem erros, resultados coerentes. |
| **Criatividade e**  **Aprimoramentos** | 2,0 | Uso de visualizações, análise de ciclos negativos ou comparação com bibliotecas externas. |

**Nota** **Final:** Será a soma dos valores obtidos em cada critério. Alunos ou equipes que não cumprirem os requisitos mínimos de funcionamento do código ou não entregarem o relatório dentro do prazo terão sua nota diminuída proporcionalmente.

**TÍTULO DA ATIVIDADE (ROTEIRO OU AULA): TÉCNICAS DE DIVISÃO E CONQUISTA E DE PROGRAMAÇÃO DINÂMICA EM PYTHON**

**ORIENTAÇÕES**:

Cada aluno deve produzir um relatório (2 a 3 páginas) contendo:

**Resumo Teórico:**

* A explanação das diferenças estruturais entre divisão e conquista e programação dinâmica.
* A justificativa dos ganhos obtidos com memoização ou tabulação nos problemas escolhidos; **Descrição das Classes Criadas:**
* A inclusão das três versões do algoritmo escolhido (recursiva simples, memoizada, bottom-up).
* A apresentação dos tempos medidos em tabela.

**REFERÊNCIAS**: O aluno deverá colocar o nome dos livros e *sites* utilizados para a realização da atividade. As regras para fazer referência ao material utilizado deverão ser de acordo com a ABNT.

# CRITÉRIOS PARA AVALIAÇÃO

|  |  |  |
| --- | --- | --- |
| **Critério** | **Peso** | **Descrição** |
| **Clareza do**  **Resumo**  **Teórico** | 2,0 | A inclusão das três versões do algoritmo escolhido (recursiva simples, memoizada, bottom-up). |
| **Organização e**  **Comentários do**  **Código** | 3,0 | A apresentação dos tempos medidos em tabela. |
| **Funcionalidade do Código** | 3,0 | A inclusão das três versões do algoritmo escolhido (recursiva simples, memoizada, bottom-up). |
| **Criatividade e**  **Aprimoramentos** | 2,0 | Análise gráfica, discussão sobre consumo de memória ou casos extremos.. |

**Nota** **Final:** Será a soma dos valores obtidos em cada critério. Alunos ou equipes que não cumprirem os requisitos mínimos de funcionamento do código ou não entregarem o relatório dentro do prazo terão sua nota diminuída proporcionalmente.