Trabalho em Grupo

1 Atividade em Grupo

A realização de um trabalho em grupo tem como principal objetivo desenvolver a capacidade de colaboração entre os estudantes. Ao trabalhar em equipe, os alunos têm a oportunidade de trocar ideias, dividir responsabilidades e aprender a valorizar diferentes perspectivas. Esse processo não só enriquece o aprendizado coletivo, mas também aprimora habilidades essenciais para o ambiente profissional, como comunicação, negociação e resolução de conflitos.

Além disso, o trabalho em grupo estimula o desenvolvimento de competências organizacionais e de liderança. Os estudantes precisam coordenar suas atividades, definir metas comuns e garantir que todos os membros contribuam de maneira equitativa para o sucesso do projeto. Essa dinâmica fortalece o senso de responsabilidade e comprometimento de cada aluno, preparando-os para desafios que exigem trabalho em equipe e gestão eficiente de recursos.

Nesse contexto, a atividade em grupo será desenvolvida em grupos de 4 integrantes. Cada integrante tem um *papel* específico a ser avaliado no trabalho em geral

O conceito de papel aqui apresentado refere-se à responsabilidade que cada integrante do grupo deve assumir durante a realização do trabalho. Cada papel é essencial para o sucesso do projeto e contribui para o desenvolvimento de habilidades específicas. Os papéis são: GitMaster, Codificador, Apresentador e Redator.

1.1 GitMaster (A)

O GitMaster é responsável por gerenciar o repositório do projeto no GitHub. Ele deve criar o repositório, adicionar os demais integrantes como colaboradores e garantir que todas as alterações sejam documentadas adequadamente. O GitMaster também deve assegurar que o código esteja sempre atualizado e que os commits sejam feitos de forma clara e organizada.Além desta tarefas gerais outras necessárias são:

- Apresentar uma documentação no repositório do GitHub com a descrição do problema a ser resolvido;
- Instruções dos passos para compilar, executar e utilizar o programa;
- Apresentação sobre os problemas que surgiram durante o desenvolvimento do trabalho e como foram resolvidos;

O trabalho do GitMaster é fundamental para garantir que o projeto seja desenvolvido de forma colaborativa e eficiente. Ele deve ter um bom conhecimento sobre o uso do Git e do GitHub, além de habilidades de organização e comunicação para coordenar o trabalho em equipe. Por isso o GitMaster deve:

- Padronizar o uso do *Git* e do *GitHub* entre os integrantes do grupo;
- Criar *commits* claros e descritivos, explicando as alterações realizadas no código;
- Resolver conflitos de *merge* quando necessário, garantindo que o código esteja sempre atualizado e funcionando corretamente;

- Criar *branches* para novas funcionalidades ou correções de *bugs*, permitindo que o trabalho seja feito de forma isolada e segura;
- Criar *tags* para marcar versões importantes do projeto, facilitando o acompanhamento do progresso e a identificação de problemas;
- Zelar pela integridade do repositório, garantindo que o código esteja sempre em um estado funcional e que as alterações sejam documentadas de forma adequada.

1.2 Codificador (B)

O Codificador é o responsável por implementar a solução do problema proposto. Ele deve escrever o código-fonte, garantindo que esteja bem estruturado, legível e eficiente. O Codificador deve seguir as boas práticas de programação, como a utilização de comentários explicativos e a organização do código em funções ou classes, quando necessário. Além disso, o Codificador deve realizar testes para garantir que o programa funcione corretamente e atenda aos requisitos especificados. Resumidamente:

- Implementar a solução do problema proposto;
- Garantir que o código esteja bem estruturado, legível e eficiente;
- Realizar testes para garantir que o programa funcione corretamente;

Outras tarefas que *podem* mas não são obrigatórias são:

- Criar uma rotina de execução do programa que permita a execução de testes automatizados;
- Comentar o código de forma clara e objetiva, explicando a lógica por trás de cada parte do programa;
- Separar o código em módulos ou arquivos, se necessário, para facilitar a manutenção e a compreensão do programa;
- Documentar as funções criadas, explicando seus propósitos e como elas se relacionam com o restante do código;

1.3 Apresentador (C)

O Apresentador é o responsável por preparar e realizar a apresentação do trabalho. Ele deve elaborar um material visual (como slides) que resuma o problema, a solução proposta e os resultados obtidos. O Apresentador deve ser capaz de explicar claramente o funcionamento do programa, destacando os principais pontos do código e as decisões tomadas durante o desenvolvimento. Além disso, o Apresentador deve estar preparado para responder a perguntas e esclarecer dúvidas. resumidamente:

- Preparar e realizar a apresentação do trabalho;
- Elaborar um material visual (como slides) que resuma o problema, a solução proposta e os resultados obtidos;
- Explicar claramente o funcionamento do programa, destacando os principais pontos do código e as decisões tomadas durante o desenvolvimento;

- Estar preparado para responder a perguntas e esclarecer dúvidas.

Para ter sucesso na apresentação, o Apresentador deve:

- Ensaiar a apresentação várias vezes, garantindo que o tempo seja respeitado e que todos os pontos importantes sejam abordados;
- Utilizar recursos visuais de forma eficaz, como gráficos, diagramas e imagens, para ilustrar os conceitos apresentados e tornar a apresentação mais envolvente;
- Estar preparado para lidar com perguntas difíceis ou críticas construtivas, respondendo de forma calma e fundamentada;
- Utilizar o *feedback* recebido durante a apresentação para aprimorar futuras apresentações e desenvolver habilidades de comunicação eficazes.

As apresentações serão realizadas em sala de aula, com duração de 10 minutos para cada grupo. O apresentador poderá realizar a apresentação de forma individual ou em conjunto com os demais integrantes do grupo, dependendo da dinâmica escolhida pelo grupo. É importante que o apresentador esteja confortável com o conteúdo e confiante na sua capacidade de transmitir as informações de forma clara e objetiva.

1.4 Redator (D)

Responsável documental do trabalho, o Redator deve elaborar um relatório que descreva o problema, a solução proposta e os resultados obtidos. O relatório deve ser claro, conciso e bem estruturado. O Redator deve incluir no relatório uma descrição das instruções do programa, explicando a lógica por trás de cada parte do programa e como elas se relacionam com o restante do código. Além disso, o Redator deve garantir que o relatório esteja livre de erros gramaticais e ortográficos. Resumidamente:

- Elaborar um relatório que descreva o problema, a solução proposta e os resultados obtidos;
- Incluir no relatório uma descrição das instruções do programa, explicando a lógica por trás de cada parte do programa e como elas se relacionam com o restante do código;
- Garantir que o relatório esteja livre de erros gramaticais e ortográficos.

Outras tarefas que *podem* ser desenvolvidas pelo Redator são:

- Utilizar uma formatação adequada para o relatório, como margens, espaçamento entre linhas e fontes, para garantir que o documento seja fácil de ler e visualmente agradável;
- Incluir referências bibliográficas, se necessário, para dar crédito às fontes utilizadas na pesquisa e desenvolvimento do trabalho;
- Revisar o relatório várias vezes, garantindo que todas as informações estejam corretas e que o texto esteja fluente e coerente;

2 Responsabilização e Avaliação

Cada integrante será avaliado pela atividade que que está descrito no papel que assumiu. Neste caso, por exemplo, se um integrante decidiu pela codificação do programa, ele será avaliado pela qualidade do código, pela clareza dos comentários, enfim, de tudo que foi descrito no papel de codificador.

Mas atenção, isto não é impeditivo para que os demais integrantes do grupo não possam contribuir com a codificação, mas o papel de codificador é o que será avaliado. O mesmo vale para os demais papéis. De fato a contribuição de todos em todos os papéis é desejável, mas a avaliação será feita de acordo com o papel assumido por cada integrante.

2.1 Formação dos grupos

Os grupos terão em sua formação padrão 4 integrantes, mas poderão ser formados grupos menores. Sua formação é livre escolha entre os próprios integrantes, podendo inclusive formar grupos com integrantes de outra turma. Na formação padrão, cada integrante irá assumir um papel diferente.

2.1.1 Grupos menores

Número de Integrantes	**Papéis Atribuídos**
1	$I_1 \rightarrow A, B, C, D$
2	$I_1 \rightarrow A, C$, $I_2 \rightarrow B, D$
3	$I_1 \rightarrow A$, $I_2 \rightarrow B, C$, $I_3 \rightarrow D$

Tabela 1: Distribuição de papéis para grupos menores

2.1.2 Desistência de integrantes

Caso um integrante desista do trabalho, o grupo deverá redistribuir os papéis entre os integrantes restantes. A tabela 1 pode ser utilizada como referência para a redistribuição dos papéis. É importante que o grupo se organize para que todos os papéis sejam preenchidos, garantindo que o trabalho seja realizado de forma eficiente e completa.

2.2 Avaliação

A avaliação final corresponderá a 70% da nota da terceira unidade, sendo o restante composto por uma prova individual no formato teste (semelhante ao que ocorreu na primeira e irá ocorrer na segunda unidade). Cada papel será avaliado de acordo com os critérios descritos anteriormente, considerando a qualidade do trabalho realizado e o cumprimento das responsabilidades atribuídas. Além disso, será levado em conta o desempenho do grupo como um todo, incluindo a colaboração entre os integrantes e a apresentação do trabalho. Os critérios de avaliação incluem:

- Qualidade técnica do trabalho realizado em cada papel;
- Clareza e organização na apresentação e no relatório;
- Cumprimento dos prazos estabelecidos;
- Capacidade de resolver problemas e superar desafios durante o desenvolvimento do trabalho;

- Colaboração e interação entre os integrantes do grupo.

A nota final será calculada individualmente, considerando o desempenho de cada integrante em seu respectivo papel.

3 Propostas de Trabalhos

A seguir estão algumas propostas de trabalhos que podem ser desenvolvidas pelos grupos. Cada grupo deve escolher uma proposta e desenvolver o trabalho de acordo com as diretrizes estabelecidas. As propostas são:

3.1 Implementação de Tabela Hash com Colisões

Objetivos de aprendizagem:

- Compreender o funcionamento de tabelas hash.
- Implementar funções de hash e mecanismos de tratamento de colisão.
- Analisar desempenho de diferentes técnicas.

Descrição da atividade: Implemente uma estrutura de tabela hash em C ou C++, com suporte a inserção, busca e remoção. Teste duas estratégias de colisão: encadeamento separado e sondagem linear. Compare o desempenho de ambas em termos de tempo e número de colisões.

Extras opcionais:

- Implementar sondagem quadrática e duplo hashing.
- Permitir redimensionamento automático da tabela.

Critérios de avaliação:

- Implementação correta das funções básicas (30%)
- Tratamento eficiente de colisões (20%)
- Relatório com testes e análise (30%)
- Clareza e organização do código (20%)

3.2 Dicionário com Hash

Objetivos de aprendizagem:

- Aplicar tabelas hash em uma aplicação real.
- Manipular dados textuais usando hashing.

Descrição da atividade: Crie um programa que funcione como um dicionário: o usuário pode adicionar palavras e seus significados, buscar e remover termos. Utilize uma tabela hash para organizar os dados.

Extras opcionais:

- Salvar e carregar o dicionário de um arquivo.
- Suportar múltiplos significados por palavra.

Critérios de avaliação:

- Funcionamento correto das operações (30%)
- Uso eficaz de hash (20%)
- Interface amigável no terminal (20%)
- Relatório com análise e exemplos (30%)

3.3 Verificador de Duplicatas

Objetivos de aprendizagem:

- Utilizar conjuntos hash para verificação rápida.
- Comparar estratégias de busca com e sem hash.

Descrição da atividade: Crie um programa que receba uma lista de nomes ou e-mails e identifique entradas duplicadas. Mostre o tempo de execução em comparação com um algoritmo tradicional (como busca linear ou ordenação + comparação).

Extras opcionais:

- Interface gráfica simples.
- Suporte à importação de listas de arquivos CSV.

Critérios de avaliação:

- Correção na detecção de duplicatas (40%)
- Comparação de desempenho (20%)
- Clareza do código e uso de estrutura adequada (20%)
- Relatório com gráficos (20%)

3.4 Mini Sistema de Autenticação com Hash de Senhas

Objetivos de aprendizagem:

- Entender aplicações de hashing em segurança.
- Utilizar funções de hash criptográficas (ex: SHA-256).

Descrição da atividade: Implemente um sistema simples de cadastro e login. As senhas devem ser armazenadas em hash e não em texto puro. Use bibliotecas prontas para SHA-256 ou similar.

Extras opcionais:

- Implementar salting manual.

- Criar interface com menus (terminal ou web).

Critérios de avaliação:

- Segurança na manipulação de senhas (40%)
- Clareza e robustez do código (20%)
- Testes funcionais apresentados (20%)
- Relatório sobre técnicas utilizadas (20%)

3.5 Encurtador de URLs com Hash

Objetivos de aprendizagem:

- Criar um mapeamento reversível entre chaves curtas e URLs completas.
- Usar hash para gerar identificadores únicos.

Descrição da atividade: Implemente um encurtador de URLs: o programa recebe uma URL e gera uma chave curta baseada em hash. Ao receber uma chave curta, ele retorna a URL original.

Extras opcionais:

- Interface web simples.
- Persistência em arquivo ou banco de dados.

Critérios de avaliação:

- Funcionalidade correta de encurtamento e recuperação (40%)
- Estrutura de dados bem implementada (20%)
- Relatório e testes (20%)
- Interface clara e usabilidade (20%)

3.6 Verificador de Integridade de Arquivos com Hash

Objetivos de aprendizagem:

- Aplicar funções de hash na verificação de arquivos.
- Trabalhar com leitura de arquivos binários.

Descrição da atividade: Implemente um programa que calcule o hash de arquivos e compare versões para verificar alterações. Suporte a múltiplos algoritmos (ex: MD5, SHA-256).

Extras opcionais:

- Criar um "manifesto"com todos os hashes de um diretório.
- Comparar diretórios inteiros.

Critérios de avaliação:

- Leitura e hash corretos de arquivos (30%)
- Interface clara e funcional (20%)
- Testes práticos (30%)
- Relatório com análise de casos (20%)

3.7 Índice Invertido de Palavras com Hash

Objetivos de aprendizagem:

- Construir estrutura de busca textual com hash.
- Criar mapas entre palavras e documentos.

Descrição da atividade: Leia um conjunto de documentos (textos curtos) e construa um índice invertido: uma estrutura que associa cada palavra às ocorrências nos documentos. Use tabelas hash para representar o índice.

Extras opcionais:

- Permitir buscas com múltiplas palavras.
- Calcular frequência das palavras.

Critérios de avaliação:

- Construção correta do índice (40%)
- Clareza na estrutura de dados (20%)
- Consulta e visualização dos resultados (20%)
- Relatório com análise de desempenho (20%)

3.8 Detecção de Plágio com Hash de Frases (k-grams)

Objetivos de aprendizagem:

- Detectar similaridade entre documentos.
- Trabalhar com hash de sequências textuais.

Descrição da atividade: Implemente um sistema que divide documentos em sequências de k palavras (k-grams), calcula o hash de cada k-gram e compara conjuntos de hash entre documentos. Calcule a taxa de similaridade.

Extras opcionais:

- Interface gráfica com visualização dos trechos semelhantes.
- Exportar relatório de comparação.

Critérios de avaliação:

- Implementação correta dos k-grams (30%)
- Cálculo eficiente da similaridade (30%)
- Testes com textos reais (20%)
- Relatório com análise de resultados (20%)

3.9 Tabela Hash Dinâmica com Redimensionamento

Objetivos de aprendizagem:

- Compreender e implementar redimensionamento de tabelas hash.
- Analisar impacto da carga na eficiência da estrutura.

Descrição da atividade: Implemente uma tabela hash que cresce dinamicamente quando a taxa de ocupação excede 70%. Re-hash todas as chaves ao redimensionar.

Extras opcionais:

- Permitir redução de tamanho quando a carga for muito baixa.
- Implementar diferentes funções de hash configuráveis.

Critérios de avaliação:

- Funcionamento correto da inserção e redimensionamento (40%)
- Desempenho antes e depois do redimensionamento (30%)
- Relatório com gráficos e análise (30%)

4 Considerações Finais

Datas, formalização da formação dos grupos e entrega do trabalho serão apresentados em um documento auxiliar. É importante que os grupos se organizem desde o início, definindo claramente os papéis e responsabilidades de cada integrante. A colaboração e a comunicação eficaz são essenciais para o sucesso do trabalho em grupo.