Projetos em Linguagem C

Introdução: A Importância do Desenvolvimento de Projetos

Desenvolver um projeto de software, especialmente em uma linguagem como C, representa uma experiência prática fundamental no processo de formação de futuros profissionais de tecnologia. A execução de projetos envolve a aplicação integrada de uma série de conhecimentos adquiridos ao longo do curso, permitindo aos alunos não apenas reforçar conceitos, mas também aprender novas habilidades que são essenciais para o sucesso na área de programação. Ao trabalhar em um projeto, os alunos têm a oportunidade de ver como conceitos teóricos — como estruturas de controle, ponteiros, modularidade e manipulação de arquivos — se manifestam em situações práticas, que são a base da resolução de problemas reais.

O desenvolvimento de um projeto também ensina a importância da organização e da divisão de responsabilidades, especialmente quando feito em equipe. Cada membro precisa entender sua função, colaborar com outros membros, lidar com possíveis conflitos e integrar diferentes partes do sistema para formar um produto coeso. Isso contribui significativamente para habilidades interpessoais, como comunicação, trabalho em equipe e gestão de tempo, que são altamente valorizadas no ambiente de trabalho.

Além disso, a criação de um projeto envolve etapas como o planejamento da arquitetura, definição de objetivos, implementação e documentação, proporcionando uma experiência que simula o ciclo de vida de desenvolvimento de software em ambientes reais. Esta abordagem faz com que os alunos compreendam a importância da modularidade, da reutilização de código e da documentação — habilidades que são fundamentais para o desenvolvimento de softwares de qualidade e que garantam a manutenção eficiente do código no futuro.

Outro ponto crucial é que, ao finalizar um projeto, os alunos têm a chance de ver o impacto do seu trabalho funcionando de maneira tangível, o que gera uma sensação de conquista e motiva o aprendizado contínuo. Projetos oferecem o contexto necessário para entender erros e falhas, criando uma cultura de depuração e melhoria contínua. Assim, o desenvolvimento de projetos se torna um poderoso catalisador para a consolidação de conceitos técnicos e o desenvolvimento pessoal dos alunos.

Apresentação dos Cases de Projeto

A seguir, são apresentados 10 diferentes cases de projeto, cada um focando em um problema prático do cotidiano. Cada equipe de alunos terá a responsabilidade de desenvolver a solução utilizando os recursos aprendidos em aula, aplicando conceitos como registros (`structs`), ponteiros, modularidade (arquivos `.h` e `.c`), e manipulação de arquivos para persistir dados. A estrutura modular e a documentação detalhada são requisitos obrigatórios para garantir a qualidade e a compreensão do código, além de possibilitar a manutenção futura.

Case 1: Sistema de Controle de Livros de Biblioteca**

Objetivo Geral: Desenvolver um sistema para gerenciar o acervo de uma biblioteca, permitindo adicionar, remover, buscar e listar livros.

- **Estrutura Base (`struct`)**: Informações sobre livros (ID, título, autor, ano de publicação, exemplares disponíveis).
- **Funcionalidades**: Cadastro de novos livros, remoção, busca por título ou autor, listagem de todos os livros, e persistência dos dados em arquivos.

Case 2: Sistema de Gerenciamento de Reservas de Hotel**

Objetivo Geral: Criar um sistema para gerenciar reservas de quartos em um hotel, com controle de check-in e check-out.

- **Estrutura Base (`struct`)**: Informações sobre reservas de quartos (número do quarto, nome do hóspede, datas de check-in e check-out).
- **Funcionalidades**: Registrar reserva, cancelar, check-in e check-out, listagem de quartos e salvar em arquivos.

Case 3: Sistema de Gerenciamento de Estoque de Loja**

Objetivo Geral: Desenvolver um sistema para gerenciar o estoque de uma loja, incluindo controle de entrada e saída de produtos.

- **Estrutura Base (`struct`)**: Informações sobre produtos (ID, nome, preço, quantidade em estoque).
- **Funcionalidades**: Adicionar e atualizar produtos, listar o estoque, persistência dos dados em arquivos.

Case 4: Sistema de Gerenciamento de Pacientes em Clínica**

Objetivo Geral: Implementar um sistema para gerenciar os registros dos pacientes de uma clínica médica.

- **Estrutura Base (`struct`)**: Dados dos pacientes (ID, nome, idade, histórico médico).
- **Funcionalidades**: Adicionar e remover pacientes, listar pacientes, salvar e carregar registros usando arquivos.

Case 5: Sistema de Cadastro de Alunos e Notas**

Objetivo Geral: Criar um sistema para cadastro de alunos e gerenciamento de notas.

- **Estrutura Base (`struct`)**: Informações dos alunos (ID, nome, notas e média).
- **Funcionalidades**: Registrar alunos e suas notas, calcular médias, listar alunos, persistência dos dados em arquivos.

Case 6: Sistema de Controle de Funcionários de Empresa**

Objetivo Geral: Criar um sistema para gerenciar funcionários de uma empresa, incluindo cadastro, demissão e pesquisa de informações.

- **Estrutura Base (`struct`)**: Informações dos funcionários (ID, nome, salário, cargo).
- **Funcionalidades**: Adicionar, remover e listar funcionários, salvar e carregar informações dos funcionários.

Case 7: Sistema de Reserva de Salas de Reunião**

Objetivo Geral: Implementar um sistema de reservas para salas de reuniões.

- **Estrutura Base (`struct`)**: Dados sobre salas e reservas (ID da sala, quem reservou, data e horário de início/fim).
- **Funcionalidades**: Registrar e cancelar reservas, listar reservas, salvar e carregar as informações em arquivos.

Case 8: Sistema de Gerenciamento de Clubes Esportivos**

Objetivo Geral: Desenvolver um sistema para gerenciar membros de um clube esportivo.

- **Estrutura Base (`struct`)**: Dados dos membros (ID, nome, idade, esporte praticado).
- **Funcionalidades**: Registrar novos membros, remover membros, listar todos os membros, salvar e carregar os dados.

Case 9: Sistema de Inventário de Ferramentas**

Objetivo Geral: Criar um sistema para gerenciar o inventário de ferramentas de uma oficina.

- **Estrutura Base (`struct`)**: Informações sobre ferramentas (ID, nome, quantidade disponível).
- **Funcionalidades**: Adicionar e atualizar quantidade, listar ferramentas, persistência dos dados utilizando arquivos.

Case 10: Sistema de Cadastro de Filmes**

Objetivo Geral: Desenvolver um sistema para cadastrar filmes, com informações como título, diretor e ano de lançamento.

- **Estrutura Base (`struct`)**: Informações sobre os filmes (ID, título, diretor, ano de lançamento).
- **Funcionalidades**: Registrar e remover filmes, listar todos os filmes, salvar e carregar as informações em arquivos.

Metodologia para Avaliação

Para garantir uma avaliação justa e criteriosa de cada projeto desenvolvido, a metodologia de avaliação será composta por diferentes aspectos, cada um com igual peso. A nota de cada critério será dada de 0 a 2,5, totalizando um máximo de 10 pontos.

1. **Funcionalidade** (Nota 0-2,5)
    - **Pergunta**: O sistema cumpre todos os requisitos e funcionalidades especificados?
    - **Critérios**:
        - Todos os requisitos do case foram implementados.
        - Todas as funcionalidades funcionam como esperado, sem falhas.
2. **Qualidade do Código** (Nota 0-2,5)

- **Pergunta**: O código segue boas práticas de programação?
- **Critérios**:
  - Uso correto de variáveis no estilo camelCase.
  - Código bem estruturado e modularizado.
  - Organização dos arquivos (`.h` e `.c`) que garanta a modularidade.
  - Funções bem definidas e sem excesso de complexidade.

3. **Uso dos Conceitos Aprendidos** (Nota 0-2,5)
   - **Pergunta**: O projeto utilizou adequadamente os conceitos estudados?
   - **Critérios**:
     - Uso correto de registros (`struct`), ponteiros, e arquivos.
     - Implementação da aritmética de ponteiros quando aplicável.
     - Aplicação de conceitos como modularidade e reutilização de código.
4. **Documentação do Software** (Nota 0-2,5)
   - **Pergunta**: A documentação do software está completa e clara?
   - **Critérios**:
     - Manual do usuário bem elaborado, explicando como compilar, rodar e usar o sistema.
     - Comentários no estilo Doxygen, descrevendo a função de cada parte do código.
     - Submissão do código no GitHub, com um `README.md` que descreva o projeto e como utilizá-lo.
     - Inclusão dos nomes de todos os integrantes da equipe.

Apresentação dos Alunos

Cada equipe deverá fazer uma apresentação ao final do desenvolvimento do projeto. Esta apresentação é uma parte crucial para que os alunos demonstrem seu entendimento sobre o que foi feito e como foi feito, além de refletir sobre o processo de desenvolvimento. Durante a apresentação, a equipe deverá focar nos seguintes pontos:

- **Arquitetura do Sistema**: Explicar como o sistema foi modularizado e como as funcionalidades foram distribuídas entre os arquivos `.c` e `.h` .
- **Desafios Enfrentados**: Compartilhar as maiores dificuldades encontradas durante o desenvolvimento e as estratégias usadas para superar esses desafios. Esta reflexão ajuda na compreensão dos processos de solução de problemas e melhoria contínua.

Com essa metodologia de avaliação e a apresentação final, busca-se não apenas avaliar o código, mas também desenvolver nos alunos uma perspectiva global do ciclo

de vida de desenvolvimento de software, enfatizando a importância da documentação, da modularidade, da colaboração e da comunicação clara. Esta experiência proporcionará uma base sólida para os desafios que os alunos enfrentarão no mercado de trabalho, ajudando-os a se tornarem profissionais preparados e competentes.