Aula 14: Projetos Finais

Objetivos

- Aplicar conhecimentos adquiridos em projetos práticos complexos
- Desenvolver sistemas completos integrando múltiplas estruturas de dados
- Demonstrar domínio em análise de algoritmos e otimização
- Apresentar soluções técnicas de forma profissional

Estrutura dos Projetos

Formato de Apresentação

- **Duração:** 15-20 minutos por equipe
- **Composição:** 2-3 alunos por equipe
- **Demonstração:** Sistema funcionando + código + análise
- **Documentação:** Relatório técnico completo

Critérios de Avaliação

1. **Complexidade Técnica (25%):** Uso adequado de estruturas de dados
2. **Implementação (25%):** Qualidade do código e funcionalidade
3. **Análise (20%):** Complexidade computacional e otimizações
4. **Apresentação (15%):** Clareza e profissionalismo
5. **Documentação (15%):** Relatório técnico e comentários

Projeto 1: Sistema de Roteamento de Rede

Descrição

Desenvolva um simulador de roteamento de pacotes em uma rede de computadores.

Requisitos Técnicos

- **Representação da Rede:** Grafo ponderado (latência/bandwidth)
- **Algoritmos de Roteamento:** Dijkstra, Bellman-Ford, Floyd-Warshall
- **Tabelas de Roteamento:** Hash tables para busca rápida
- **Simulação de Tráfego:** Filas de prioridade para pacotes

Estrutura Base

```
typedef struct Pacote {
    int id;
    int origem;
    int destino;
```

Projeto 2: Sistema de Gerenciamento de Banco de Dados Simples

Descrição

Implemente um SGBD básico com suporte a índices e consultas.

Requisitos Técnicos

- **Armazenamento:** Árvores B+ para índices primários
- **Índices Secundários:** Hash tables para busca por atributos
- **Buffer Pool:** LRU cache para páginas
- **Parser SQL:** Análise de comandos SELECT, INSERT, DELETE

Estrutura Base

```
typedef struct Registro {
    int id;
    char dados[256];
    int ativo;
```

Projeto 3: Compilador para Linguagem Simples

Descrição

Desenvolva um compilador completo para uma linguagem de programação simples.

Requisitos Técnicos

- **Análise Léxica:** Tokenização usando autômatos finitos
- **Análise Sintática:** Parser recursivo descendente
- **Tabela de Símbolos:** Hash table com escopo aninhado
- **Geração de Código:** Assembly ou bytecode

Estrutura Base

```
typedef enum {
    TOKEN_NUMERO, TOKEN_IDENTIFICADOR, TOKEN_OPERADOR,
    TOKEN_PALAVRA_CHAVE, TOKEN_DELIMITADOR, TOKEN_EOF
} TipoToken;
```

Projeto 4: Sistema de Recomendação Inteligente

Descrição

Desenvolva um sistema de recomendação usando algoritmos de filtragem colaborativa.

Requisitos Técnicos

- **Grafo de Usuários:** Similaridade baseada em preferências
- **Matriz de Avaliações:** Representação esparsa eficiente
- **Algoritmos ML:** k-NN, clustering, SVD
- **Índices:** Busca eficiente por similaridade

Estrutura Base

```
typedef struct Usuario {
    int id;
    char nome[100];
    HashMap *avaliacoes;  // item_id -> rating
```

Projeto 5: Simulador de Sistema Operacional

Descrição

Implemente um simulador de sistema operacional com escalonamento e gerência de memória.

Requisitos Técnicos

- **Escalonamento:** FIFO, SJF, Round Robin, Prioridade
- **Gerência de Memória:** Paginação, algoritmos de substituição
- **Sistema de Arquivos:** Estrutura hierárquica com índices
- **Sincronização:** Semáforos e monitores

Estrutura Base

```
typedef struct Processo {
    int pid;
```

Projeto 6: Engine de Jogos 2D com IA

Descrição

Desenvolva uma engine de jogos 2D com inteligência artificial para NPCs.

Requisitos Técnicos

- **Estruturas Espaciais:** Quadtree para detecção de colisão
- **Pathfinding:** A* para navegação
- **IA Comportamental:** Máquinas de estado finitas
- **Sistema de Entidades:** Component-based architecture

Estrutura Base

```
typedef struct Entidade {
    int id;
    float x, y;
    float velocidade_x, velocidade_y;
```

Cronograma de Apresentações

Semana 1: Propostas de Projeto

- Apresentação das ideias (5 min por equipe)
- Feedback e refinamento dos requisitos
- Formação definitiva das equipes

Semana 2-4: Desenvolvimento

- Checkpoint semanal de progresso
- Mentoria individual por equipe
- Resolução de dúvidas técnicas

Semana 5: Apresentações Finais

Formato da Apresentação:

Recursos e Suporte

Ferramentas Recomendadas

- **Desenvolvimento:** GCC, Make, Git
- **Debugging:** GDB, Valgrind
- **Profiling:** gprof, perf
- **Documentação:** Doxygen, Markdown
- **Apresentação:** PowerPoint, LaTeX Beamer

Bibliotecas Permitidas

- **Gráficas:** SDL2, SFML (apenas para interface)
- **Estruturas Básicas:** Implementação própria obrigatória
- **Utilitários:** matemática padrão (math.h)

Avaliação e Notas

Distribuição de Pontos

- **Projeto Final:** 60% da nota do bimestre
- **Laboratórios:** 30% da nota do bimestre
- **Participação:** 10% da nota do bimestre

Critérios Detalhados

Complexidade Técnica (25 pontos)

- **Excelente (23-25):** Uso avançado de múltiplas estruturas
- **Bom (20-22):** Uso adequado das estruturas necessárias
- **Regular (15-19):** Uso básico com algumas limitações
- **Insuficiente (0-14):** Estruturas inadequadas ou incorretas

Considerações Finais

Este projeto final representa a culminação do aprendizado em Algoritmos e Complexidade. É uma oportunidade de demonstrar não apenas conhecimento técnico, mas também capacidade de análise, síntese e apresentação profissional.

Lembre-se:

- A originalidade e criatividade são valorizadas
- A qualidade é mais importante que a quantidade de features
- A análise teórica deve estar alinhada com os resultados práticos
- A apresentação é parte fundamental da avaliação

Boa sorte e excelente trabalho a todos!

Prof. Vagner Cordeiro

Sistemas de Informação

Algoritmos e Complexidade - 2024