PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
Instituto de Ciências Exatas e de Informática

Linguagem C

Davi Manoel Bernardes
Felipe Costa Unsonst
Felipe Quites Lopes

Resumo

A linguagem C surgiu nos anos 1970, criada por Dennis Ritchie nos laboratórios Bell, no contexto da evolução das linguagens de programação para atender às necessidades de desenvolvimento de sistemas operacionais como o Unix. O problema identificado foi a limitação de linguagens anteriores, como BCPL e B, que não ofereciam flexibilidade e desempenho suficientes para aplicações de base. A justificativa para o estudo de C está em sua relevância histórica e prática, por ser considerada a mãe das linguagens modernas e ainda amplamente utilizada no meio acadêmico e profissional. O objetivo do seminário foi apresentar suas principais características, paradigmas e aplicações, além de discutir sua importância na formação de programadores. O desenvolvimento contemplou a análise do histórico, a exploração de paradigmas como o imperativo e a simulação de orientação a objetos, além do estudo de recursos como ponteiros, modularização e biblioteca padrão. Os resultados mostraram que a linguagem continua atual, por sua eficiência, simplicidade relativa e por servir como base para diversas linguagens mais modernas. Conclui-se que C mantém extrema relevância para a computação contemporânea, sendo fundamental tanto para o aprendizado de lógica e estruturas quanto para a construção de softwares de alto desempenho.

Palavras-chave: linguagem C; paradigmas de programação; histórico da computação.

Abstract

O estudo da linguagem C apresenta-se motivado por sua relevância histórica e por sua influência direta no desenvolvimento de sistemas operacionais, compiladores e diversas linguagens modernas. O objetivo do seminário foi analisar a evolução da linguagem desde sua criação, discutir seus principais paradigmas e características, e evidenciar sua importância tanto no meio acadêmico quanto profissional. A metodologia adotada consistiu em revisão bibliográfica e análise de conceitos fundamentais, abordando aspectos como tipos

de dados, uso de ponteiros, modularização, biblioteca padrão e portabilidade entre sistemas operacionais. Os resultados indicaram que, apesar de sua origem nos anos 1970, C mantém-se atual e necessária, por possibilitar compreensão aprofundada do funcionamento interno dos computadores, fornecer base sólida para o aprendizado de outras linguagens e ainda ser empregada em aplicações críticas de software e hardware. Conclui-se que a linguagem C continua sendo uma ferramenta indispensável na formação de programadores e na construção de sistemas de alto desempenho, destacando-se por sua simplicidade relativa, eficiência e ampla aplicabilidade.

Keywords: linguagem C; paradigmas de programação; histórico da computação.

1 INTRODUÇÃO

A linguagem C ocupa um papel central na história da computação, sendo considerada a base de muitas das linguagens modernas. Criada por Dennis Ritchie nos anos 1970, nos laboratórios Bell, surgiu da necessidade de substituir linguagens como BCPL e B, que apresentavam limitações em flexibilidade e desempenho. Desde então, C consolidou-se como ferramenta essencial no desenvolvimento de sistemas operacionais, compiladores e softwares de base, além de influenciar diretamente linguagens posteriores, como C++, Java,e Go. O estudo dessa linguagem justifica-se por sua relevância histórica e prática, fornecendo aos programadores um conhecimento sólido sobre o funcionamento interno dos computadores e os fundamentos da programação.

2 DESENVOLVIMENTO)

O seminário iniciou-se com a apresentação do contexto histórico que deu origem à linguagem C, ressaltando sua evolução desde a criação do Unix até os padrões definidos pela ANSI e ISO, bem como sua adaptação para diferentes plataformas. Destacou-se que a simplicidade relativa da linguagem, aliada ao seu desempenho, fez com que ela fosse amplamente utilizada em sistemas embarcados, softwares de alto desempenho e até mesmo em partes do núcleo de sistemas operacionais modernos, como Linux e Windows.

Em seguida, foram discutidos os paradigmas de programação. O foco principal recaiu sobre o paradigma imperativo ou procedural, no qual a lógica do programa é organizada em funções, instruções sequenciais e estruturas de controle, como condicionais e laços de repetição. Também foi abordada a forma como C permite simular conceitos da orientação a objetos por meio do uso de structs e funções associadas, ainda que sem suporte nativo a herança ou polimorfismo. Além disso, reconheceu-se que a linguagem possibilita, em menor grau, a aplicação de princípios funcionais, como a utilização de funções puras.

Outro ponto importante foi a análise das características técnicas da linguagem. Foram explorados os tipos de dados primitivos (int, char, float, double), modificadores como short, long e unsigned, além do uso de arrays e strings. A biblioteca padrão, composta por mais de 200 funções organizadas em cabeçalhos como <stdio.h>, <stdlib.h>, <string.h> e <math.h>, foi apresentada como recurso fundamental para a construção de programas robustos e reutilizáveis. A modularização foi destacada como prática essencial, possibilitando separar a interface da implementação e facilitando a manutenção do código.

Grande ênfase foi dada ao estudo dos ponteiros, considerados uma das ferramentas mais poderosas e desafiadoras da linguagem. Esses recursos permitem manipular diretamente endereços de memória, criando estruturas dinâmicas como listas encadeadas, árvores e grafos, além de possibilitar maior controle sobre o desempenho do programa. Embora proporcionem grande flexibilidade, exigem do programador atenção especial para evitar falhas, como vazamentos de

memória e acessos inválidos.

Foram também explorados os recursos de controle de fluxo (if/else, switch case) e de repetição (for, while e do while), que conferem à linguagem um conjunto versátil de ferramentas para estruturar algoritmos. Associados à possibilidade de modularização, esses elementos tornam o código mais organizado, reutilizável e legível.

Por fim, abordou-se a portabilidade da linguagem C, com exemplos de instalação e utilização do compilador GCC em sistemas Linux, Windows e MacOS. Essa característica reforça sua adoção como base para projetos multiplataforma, além de destacar seu papel como linguagem introdutória no ensino acadêmico. Cada integrante do grupo apresentou também reflexões pessoais, enfatizando a relevância da linguagem tanto no aspecto técnico quanto histórico, além de sua utilidade como base conceitual para a aprendizagem de outras tecnologias.

2.1 Exemplos de Algoritmos

Exemplos de Algoritmos feitos para o desenvolminento do trabalho:
Autor: Davi Manoel:

void funcao(int soma, int num) if((num / 10) == 0) //compara se o neumro foi somado por completo
soma += (num printf("return;
soma += (num funcao(soma, num/10); //chama a funcao com num/10
int main()
char string[20]; int num,soma = 0;
while(scanf("
if(strcmp(string, "FIM") == 0) //se String for FIM interrompe o programa return 0; num = atoi(string); //CONVERTE STRING PARA INT funcao(soma,num); //chama funcao recursiva
}
Autor: Felipe Quites:

int main()
Estudante alunos[20]; int quantidade = 0; int opcao;
do
printf("======================"); printf("1. Inserir aluno"); printf("2. Dar nota a um aluno"); printf("3. Listar alunos"); printf("4. Sair"); printf("Escolha uma opcao: "); scanf("getchar();
if (opcao == 1)
if (quantidade >= 20) printf("Limite de alunos atingido!"); continue;

int matricula, idade; char nome[50];

```
        printf("Digite a matrícula: "); scanf("getchar();
        printf("Digite o nome: "); fgets(nome, sizeof(nome), stdin); nome[strcspn(nome, "")] =
”;
        printf("Digite a idade: "); scanf("
        alunos[quantidade] = criarEstudante(matricula, nome, idade, 0.0); quantidade++; printf("Aluno
inserido com sucesso!");
        else if (opcao == 2)
if (quantidade == 0)  printf("Nenhum aluno cadastrado."); continue;

        int matricula; float nota; printf("Digite a matrícula do aluno: "); scanf("
        int encontrado = 0; for (int i = 0; i < quantidade; i++)
if (alunos[i].matricula == matricula)  printf("Digite a nota a adicionar: "); scanf("darNota(alunos[i],
nota); printf("Nota adicionada! Nova soma de notas: encontrado = 1; break;

if (!encontrado)
printf("Aluno com matrícula
        else if (opcao == 3)
if (quantidade == 0)
printf("Nenhum aluno cadastrado.");  else
printf("================== Lista de alunos ====================="); for (int i = 0;
i < quantidade; i++) imprimirEstudante(alunos[i]);
else if (opcao == 4)
printf("Saindo...");  else
printf("Opcao invalida!");
        while (opcao != 4);

        return 0;

        Autor: Felipe costa:

        int main()
char palavra[100];
        while (true)
fgets(palavra, sizeof(palavra), stdin); palavra[strcspn(palavra, "")] = ”;
        if (strcmp(palavra, "FIM") == 0)
break;

        int tam = strlen(palavra); bool pali = true;
        for (int i = 0; i < tam / 2; i++)
if (palavra[i] != palavra[tam - i - 1])
```

```
pali = false; break;


        if (pali)
printf("SIM");  else
printf("NAO");

return 0;
```

3 CONCLUSÃO

Conclui-se que a linguagem C mantém extrema relevância tanto no meio acadêmico quanto no profissional. Apesar de ter sido criada há mais de cinco décadas, ainda é utilizada como linguagem introdutória em cursos de programação, por sua simplicidade relativa e proximidade com o funcionamento da máquina. Além disso, continua a ser aplicada em projetos que exigem eficiência e controle detalhado de recursos, mantendo-se atual e indispensável. Dessa forma, o estudo da linguagem C não apenas contribui para a formação de uma base sólida em programação, mas também permite compreender a lógica e o funcionamento de diversas linguagens modernas que dela derivaram.

Pinho (s.d.) Wikipédia (2025) PUC Minas (2024) PhoenixNAP (2024) PUC Minas (2025) Kernighan e Ritchie (1988)

REFERÊNCIAS

KERNIGHAN, B. W.; RITCHIE, D. M. **The C Programming Language**. 2. ed. Englewood Cliffs: Prentice Hall, 1988.

PHOENIXNAP. **How to Install GCC on Windows**. 2024. Acesso em: 3 out. 2025. Disponível em: <https://phoenixnap.com/kb/install-gcc-windows>.

PINHO, A. L. **Histórico da Linguagem C**. s.d. Acesso em: 3 out. 2025. Disponível em: <https://www.inf.pucrs.br/~pinho/LaproI/Historico/Historico.htm>.

PUC MINAS. **Aulas de Algoritmo e Estrutura de Dados**. 2024. Material didático interno.

PUC MINAS. **Aulas de Linguagem de Programação**. 2025. Material didático interno.

WIKIPéDIA. **C (linguagem de programação)**. 2025. Acesso em: 3 out. 2025. Disponível em: <https://pt.wikipedia.org/wiki/C_(linguagem_de_programa%C3%A7%C3%A3o)>.