INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SANTA CATARINA
BACHARELADO EM ENGENHARIA DE TELECOMUNICAÇÕES

Anne Damaris Lopes Martins

RELATÓRIO PROJETO FINAL DE PROGRAMAÇÃO I - PRG1

SÃO JOSÉ, 2025

1 INTRODUÇÃO

Este relatório tem por objetivo apresentar o programa desenvolvido pela aluna Anne Damaris Lopes Martins, na disciplina de Programação I do curso de Bacharelado em Engenharia de Telecomunicações – IFSC, ministrada pelos professores Sergio Maurício Prolo Santos Junior e Ana Luiza Scharf.

O projeto foi elaborado na linguagem C, utilizando a plataforma online GitHub, que é baseada em nuvem e utiliza o sistema Git para hospedagem e colaboração em projetos de desenvolvimento de software. O GitHub é um serviço amplamente utilizado para controle de versão, permitindo que desenvolvedores armazenem, compartilhem e trabalhem colaborativamente no código, registrando todas as alterações realizadas. Isso facilita o trabalho em equipe e o rastreamento de mudanças ao longo do desenvolvimento.

1.1 OBJETIVO GERAL

O Objetivo da atividade foi aplicar o conteudo desenvolvido em sala ao longo do semestre letivo, no qual forma desenvolvidos os contudos basicos para aprender a linguagem de programação, focado na linguagem C, por ser uma linguagem de porpósito geral e muito utilizada em sistemas operacionais e embarcados, e utilizando os comandos e estruturas estudados ao longo o semestre.

1.2 OBJETIVOS ESPECÍFICOS

No projeto foi proposto o desenvolvimento de um sistema que integrasse todas as ferramentas estudadas, demonstrando suas funcionalidades. Entre os conceitos aplicados destacam-se: comandos if e if-else, operador condicional ternário, comando switch, funções e variáveis globais, introdução a ponteiros, passagem de parâmetros por referência, comando while, comando do-while, comando for, comandos break, continue e goto, estruturas (struct), passagem de estruturas para funções, arranjos (vetores), passagem de arranjos para funções, arranjos multidimensionais e manipulação de strings.

O sistema foi estruturado em torno de uma struct “pedido”, que foi definida para armazenar todas as informações referentes a cada pedido realizado por um cliente e concentra todas as informações necessárias para o controle eficiente dos pedidos realizados no sistema. Essa estrutura contém campos para o identificador do pedido (ID), o nome do cliente, o sabor da pizza, o tipo de bebida, o status atual do pedido, a quantidade de pizzas solicitadas, além dos preços unitários da pizza e da bebida.

2 MANUAL DO SISTEMA DESENVOLVIDO.

O programa simula o sistema de pedidos de uma pizzaria simples, oferecendo as seguintes funcionalidades:

- Cadastro de novos pedidos;
- Listagem de todos os pedidos realizados;
- Atualização da quantidade de pizzas em um pedido;
- Alteração do status do pedido (por exemplo: "em preparo", "entregue");
- Exibição do total acumulado no caixa;

Além da definição da estrutura, o programa conta com um conjunto de funções principais que organizam as operações do sistema. A função menu exibe ao usuário o menu principal com as opções disponíveis. A função novo pedido é responsável pelo cadastro de um novo pedido, incluindo o cálculo do valor total.

Para visualização, a função Listar Pedidos apresenta todos os pedidos realizados até o momento. Quando necessário alterar a quantidade de pizzas de um pedido existente, utiliza-se a função Atualizar Pedido. Já a função Alterar Status possibilita modificar o status do pedido, permitindo, por exemplo, marcar um pedido como “preparo” ou “entregue”. Por fim, a função Mostrar Caixa exibe o valor total acumulado em caixa, calculado com base nos pedidos registrados.

Ao escolher cadastrar um novo pedido, a função Novo Pedido é acionada. Nela, o usuário informa o nome do cliente, o sabor da pizza (entre queijo, bacon ou calabresa), a bebida desejada (água ou refrigerante) e a quantidade de pizzas. Com esses dados, o programa calcula o valor total do pedido e armazena as informações em um vetor de pedidos para controle.

A função Listar Pedidos exibe todos os pedidos realizados até o momento, mostrando detalhes como o identificador do pedido, o nome do cliente, o sabor da pizza, a bebida escolhida, a quantidade, o valor total e o status atual de cada pedido.

Para alterar a quantidade de pizzas de um pedido já cadastrado, o programa utiliza a função Atualizar Pedido, que localiza o pedido pelo seu ID e permite modificar apenas esse campo específico.

Já a função Alterar Status possibilita a mudança do status do pedido, que pode estar em diferentes etapas, como "preparo", "pendente" ou "entregue", facilitando o acompanhamento do andamento dos pedidos.

Por fim, a função Mostrar Caixa calcula e exibe o valor total em Reais de todos os pedidos realizados até o momento, considerando tanto as pizzas quanto as bebidas.

3 METODOLOGIA

A matodologia usada foi a pagina do Github, em paralelo com as aulas ministradas em sala e pelos professores Sergio Maurício Prolo Santos Junior e Ana Luiza Scharf.

O programa utiliza vetores de estruturas para armazenar até cinco pedidos, conforme definido pela constante Max_pedidos. Para garantir uma melhor organização e facilitar a compreensão do código, as funcionalidades foram divididas em funções específicas. O fluxo de execução do programa é controlado por estruturas de decisão do tipo switch e por um laço do...while, que gerenciam o funcionamento do menu principal. Além disso, o sistema é totalmente interativo, permitindo que os dados sejam inseridos pelo usuário através do teclado e exibidos diretamente no terminal.

O desenvolvimento do sistema foi conduzido em etapas bem definidas. Inicialmente, foi criada uma estrutura denominada Pedido, responsável por armazenar as informações de cada pedido realizado. Em seguida, foi implementado um vetor capaz de armazenar até cinco pedidos simultaneamente. Para organizar as funcionalidades, foram elaboradas funções modulares que realizam as principais ações do sistema, como cadastrar novos pedidos, listar os existentes, atualizar informações, alterar o status e exibir o total acumulado no caixa.

O controle do fluxo do programa ocorre por meio de menus interativos, gerenciados por um laço do...while combinado com a estrutura switch. A entrada dos dados é feita utilizando as funções scanf() e fgets(), garantindo uma interação eficiente com o usuário. Por fim, todo o sistema foi programado e testado em ambiente de terminal, utilizando o compilador GCC nos sistemas operacionais Linux e Windows.

4 RESULTADOS E DISCUSSÃO

Este projeto proporcionou a aplicação prática dos principais conceitos da linguagem C aprendidos durante o primeiro semestre, reforçando a importância da modularização, da manipulação de estruturas e do uso de menus interativos. Além disso, reforçou a importância da modularização do código através de funções, a manipulação de estruturas e o uso de menus interativos.

O sistema implementado é funcional, interativo e atende aos requisitos estabelecidos, demonstrando que conceitos básicos de programação podem ser integrados para criar soluções úteis e escaláveis. Este projeto pode ser expandido no futuro para incluir funcionalidades mais avançadas, tais como gerenciamento de estoque, interface gráfica, integração com banco de dados e sistema de pedidos online.

5 ANEXO I – CÓDIGO-FONTE COMPLETO

```c
#include <stdio.h>
#include <string.h>

#define MAX 5

typedef struct {
    int id;
    char cliente[10];
    char sabor[10];
    char bebida[10];
    int quantidade;
    float total;
} Pedido;

void menu();
void novo(Pedido pedidos[], int *qtd, float *caixa);
void listar(Pedido pedidos[], int qtd);
void totalCaixa(float caixa);

int main()
{
    Pedido pedidos[MAX];
    int qtd = 0;
    float caixa = 0;
    int opcao;

    do {
        menu();
        printf("Opcao: ");
        scanf("%d", &opcao);
        getchar();

        if (opcao == 1) {
            novo(pedidos, &qtd, &caixa);
        } else if (opcao == 2) {
            listar(pedidos, qtd);
        } else if (opcao == 3) {
            totalCaixa(caixa);
        } else if (opcao != 0) {
            printf("Opcao invalida!\n");
        }

    } while (opcao != 0);

    return 0;
}

void menu()
{
    printf("\n Boa Tarde :) \n");
    printf("\n Bem vindo ao Gerenciador de Pizzaria :v \n");
```

```
    printf("\n --- MENU ---\n");
    printf("1 - Anotar novo pedido\n");
    printf("2 - Listar pedidos\n");
    printf("3 - Mostrar total em caixa\n");
    printf("0 - Sair\n");
}

void novo(Pedido pedidos[], int *qtd, float *caixa)
{
    if (*qtd >= MAX) {
        printf("Limite de pedidos atingido!\n");
        return;
    }

    printf("Nome do cliente: ");
    fgets(pedidos[*qtd].cliente, 10, stdin);
    strtok(pedidos[*qtd].cliente, "\n");

    int sabor;
    printf("\n Sabores:\n");
    printf("1 - Queijo (39.90)\n");
    printf("2 - Bacon (45.90)\n");
    printf("3 - Calabresa (45.90)\n");
    printf("Escolha: ");
    scanf("%d", &sabor);
    getchar();

    if (sabor == 1) {
        strcpy(pedidos[*qtd].sabor, "Queijo");
        pedidos[*qtd].total = 39.90;
    } else if (sabor == 2) {
        strcpy(pedidos[*qtd].sabor, "Bacon");
        pedidos[*qtd].total = 45.90;
    } else {
        strcpy(pedidos[*qtd].sabor, "Calabresa");
        pedidos[*qtd].total = 45.90;
    }

    int bebida;
    printf("Bebidas:");
    printf("\n1- Agua (5.00)");
    printf("2- Refri (10.00)");
    printf("Escolha: ");
    scanf("%d", &bebida);
    getchar();

    if (bebida == 1) {
        strcpy(pedidos[*qtd].bebida, "Agua");
        pedidos[*qtd].total += 5.00;
    } else {
        strcpy(pedidos[*qtd].bebida, "Refri");
        pedidos[*qtd].total += 10.00;
    }
```

```
    printf("Quantidade: ");
    scanf("%d", &pedidos[*qtd].quantidade);

    getchar();

    pedidos[*qtd].id = *qtd + 1;
    pedidos[*qtd].total *= pedidos[*qtd].quantidade;

    *caixa += pedidos[*qtd].total;
    (*qtd)++;

    printf("Pedido feito com sucesso!");
}

void listar(Pedido pedidos[], int qtd)
{
    printf("\n--- LISTA DE PEDIDOS ---\n");
    for (int i = 0; i < qtd; i++) {
        printf("ID: %d | Cliente: %s | Sabor: %s | Bebida: %s | Qtde: %d | Total:
R$ %.2f\n",
               pedidos[i].id,
               pedidos[i].cliente,
               pedidos[i].sabor,
               pedidos[i].bebida,
               pedidos[i].quantidade,
               pedidos[i].total);
    }
    if (qtd == 0) {
        printf("Nenhum pedido ainda.\n");
    }
}

void totalCaixa(float caixa)
{
    printf("\nTotal no caixa: R$ %.2f\n", caixa);
}
```