

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA FARROUPILHA
CAMPUS SÃO BORJA
CURSO TÉCNICO EM INFORMÁTICA INTEGRADO



ABÍLIO GOULART, BRUNO PICON, DÉBORAH ROCHA, JOÃO VITOR
BELLADONA, KELI BATISTA, PEDRO KOCHHAN

SISTEMA DE GESTÃO DE MULTAS

São Borja/RS

2024/1

SUMÁRIO

1. INTRODUÇÃO	3
2. OBJETIVOS DO SISTEMA	2
3. METODOLOGIA	4
3.1. Escrita do Código	4
3.2. Compilação	4
3.3. Tarefas Automatizada	4
3.4. Execução	4
4. IMPLEMENTAÇÃO	4
4.1 Funções	5
4.2. Pré-requisitos Atendidos	6
4.3. Dependência	6
4.4 Problemas e soluções	6
4.5. Regras de Negócio	7
5. RESULTADOS	8
6. REFERÊNCIAS	8
7. ANEXOS	9

1 INTRODUÇÃO

No Brasil, o sistema de notificação de multas de trânsito ainda é amplamente baseado no envio de cartas físicas, um método que tem se mostrado obsoleto e ineficiente. Em 2023, o Conselho Nacional de Trânsito (Contran) revelou que cerca de 30% das notificações enviadas pelos Correios não chegaram aos destinatários, devido a motivos como endereços desatualizados ou extravios. Esse problema tem prejudicado tanto os motoristas quanto os órgãos responsáveis pela fiscalização, destacando a urgência de modernizar o processo e garantir uma comunicação mais eficaz.

Diante desse cenário, este trabalho propõe uma solução informatizada desenvolvida na linguagem C, com o objetivo de registrar, consultar e armazenar informações sobre infrações de trânsito de forma mais eficiente. O sistema é baseado em uma estrutura de dados eficiente, utilizando listas encadeadas para gerenciar dados essenciais como a placa do veículo, o nome do condutor, o número da CNH e o modelo do veículo. Dentre as principais funcionalidades, destacam-se o registro de multas com dados completos do veículo e do condutor, a consulta das infrações por CNH, a listagem de todas as multas registradas, o salvamento e a exportação de dados para um arquivo de texto e o cálculo do valor da multa com base no tipo de infração. Além de detalhar as funcionalidades e as regras de negócio do sistema, o trabalho aborda os benefícios da digitalização do processo de notificação de multas. A solução proposta visa superar as limitações do modelo tradicional, proporcionando uma gestão mais eficiente e transparente das infrações. Espera-se que essa abordagem contribua para a redução de custos operacionais, melhora na comunicação com os motoristas e aumento da precisão na gestão das multas de trânsito.

2 . Objetivos do Sistema

O objetivo principal do sistema é gerenciar multas de trânsito de forma eficiente e organizada, garantindo precisão no registro, cálculo e acompanhamento das infrações. Os objetivos específicos incluem:

- **Registro de Infrações:** Armazenar dados do veículo, condutor, tipo de infração e data de forma estruturada.
- **Cálculo Automático das Multas:** Calcular o valor das multas automaticamente com base na gravidade da infração.

- **Busca e Consulta Rápida:** Facilitar a busca por multas usando filtros como placa do veículo ou número da CNH.

3. METODOLOGIA

O desenvolvimento do sistema foi dividido em três etapas principais: **Escrita do Código, Compilação e Execução.**

3.1. Escrita do Código

O código foi baseado em referências do GitHub e nas técnicas aprendidas nas aulas. As funcionalidades principais, como **validação de CNH e placa, cálculo do valor da multa e persistência de dados em arquivos de texto**, foram implementadas com modularidade para facilitar testes e manutenção.

3.2. Compilação

A compilação foi realizada utilizando ferramentas integradas no editor de código Visual Studio Code, com configurações adequadas ao ambiente de desenvolvimento escolhido. A configuração foi realizada para garantir um processo de compilação fluido e sem erros.

3.3. Tarefas automatizadas

O uso de ferramentas como o "Code Runner" e a configuração de Makefiles ajudaram a facilitar o processo de compilação e minimizar a chance de erro humano. O código foi compilado de forma eficiente, com o comando adequado para a execução do processo.

3.4. Execução

Testamos o sistema no terminal integrado, validando o registro de multas, o armazenamento de dados em arquivos de texto e a navegação do menu. Entradas válidas e inválidas foram testadas para garantir o correto funcionamento do programa.

4 IMPLEMENTAÇÃO

O sistema de gestão de multas foi desenvolvido utilizando uma lista encadeada para armazenar dinamicamente os registros dos veículos multados. Cada elemento contém um ponteiro para um veículo, permitindo múltiplas inserções e consultas. Aqui está uma visão geral de como o projeto foi estruturado:

4.1 Funções

- **Funções de Registro:** Incluem `registra()` e `valorMulta()`, responsáveis pelo cadastro de veículos e cálculo automático do valor das infrações com base na gravidade.

```
1 float valorMulta(float a); //Protótipo da função de multa
2 int* cnhpbusca; //Declarei um ponteiro do tipo inteiro para a função de buscar multa por cnh
3 void registra(Veiculo* ptr, boasvindas(),alternativas(), linha(), aguarde(),enviar_arquivo(Elemento* lista, FILE* arquivo, float valormulta); //Protótipo das funções feitas
4 int main(){
```

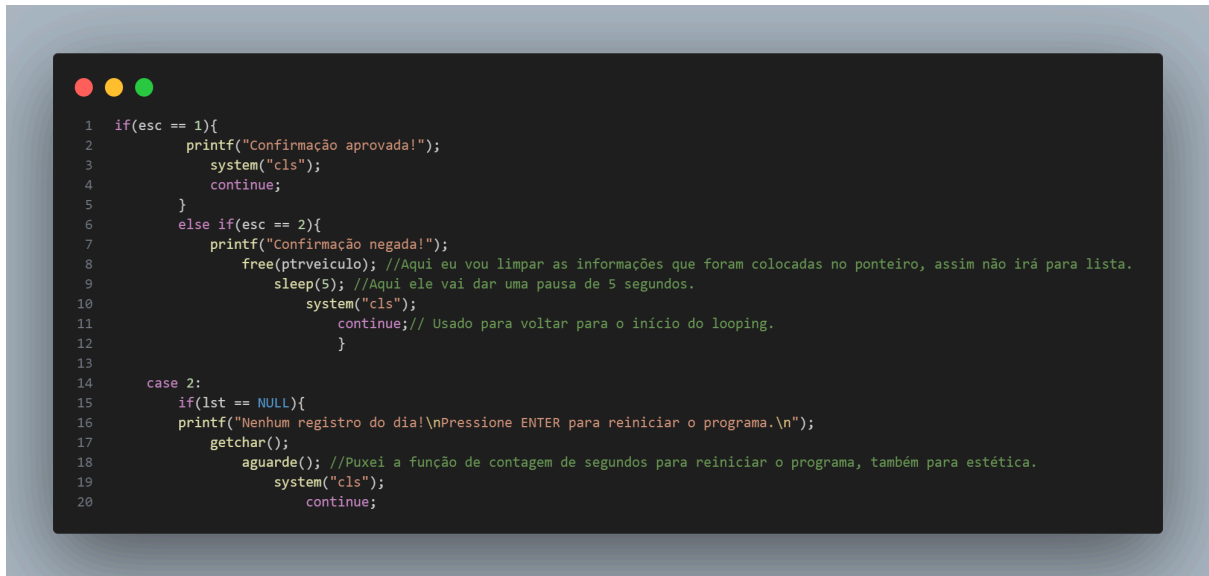
- **Funções de Exibição e Busca:** `mostra_lista()` e `busca_multa()` permitem visualizar e localizar registros com base no número de CNH

```
1 typedef struct elemento Elemento; //Criei um elemento do tipo elemento
2 Elemento* insere_elemento_final(Elemento* lista, Veiculo* veic); //Protótipo para a função de inserir o elemento no final da lista
3 void mostra_lista(Elemento* lista, float valormulta); //Protótipo da função para mostrar a lista
4 Elemento* criar_lista(); //Função para iniciar a lista
```

- **Funções de Inserção:** Funções como `insere_elemento_final` e `criar_lista` são responsáveis pela criação e inserção de elementos na lista, enquanto a função `mostra_lista` percorre a lista para exibir os registros.

```
1 typedef struct elemento Elemento; //Criei um elemento do tipo elemento
2 Elemento* insere_elemento_final(Elemento* lista, Veiculo* veic); //Protótipo para a função de inserir o elemento no final da lista
3 void mostra_lista(Elemento* lista, float valormulta); //Protótipo da função para mostrar a lista
4 Elemento* criar_lista(); //Função para iniciar a lista
```

- **Interação com o Usuário:** O programa utiliza um menu interativo, com estrutura de controle baseada em `switch`, para oferecer uma interface clara e organizada.



```

1  if(esc == 1){
2      printf("Confirmação aprovada!");
3      system("cls");
4      continue;
5  }
6  else if(esc == 2){
7      printf("Confirmação negada!");
8      free(ptrveiculo); //Aqui eu vou limpar as informações que foram colocadas no ponteiro, assim não irá para lista.
9      sleep(5); //Aqui ele vai dar uma pausa de 5 segundos.
10     system("cls");
11     continue; // Usado para voltar para o início do looping.
12 }
13
14 case 2:
15     if(lst == NULL){
16         printf("Nenhum registro do dia!\nPressione ENTER para reiniciar o programa.\n");
17         getchar();
18         aguarde(); //Puxei a função de contagem de segundos para reiniciar o programa, também para estética.
19         system("cls");
20         continue;

```

- **Controle de Fluxo**

O programa segue uma execução estruturada com menus simples e interação via opções digitadas. Utiliza uma estrutura `switch` para redirecionar ações, como o registro de multas com a função `registra()`, que solicita dados do veículo e do condutor. Funções auxiliares, como `valorMultas()`, automatizam cálculos, enquanto técnicas como limpeza de buffer (`getchar()`) e tela (`system("cls")`) melhoram a usabilidade. Informações são armazenadas em arquivos de texto no modo de anexação, garantindo integridade dos registros e permitindo revisão antes da gravação final. A estrutura proporciona operação eficiente e clara para gerenciar multas.

4.2 Pré-requisitos Atendidos

Antes de iniciar a instalação, certifique-se de que os seguintes requisitos estão atendidos:

Editor de Código: Utilize um editor de código que facilite o desenvolvimento em C.

Recomendamos:

- Visual Studio Code (VS Code): Permite configuração fácil do compilador e inclui suporte a extensões C/C++ para depuração e autocompletar.
- Code::Blocks: Um ambiente integrado que já inclui suporte ao compilador.

- Dev-C++: Uma IDE leve e simples para desenvolvimento em C.

O fluxo ideal é realizar o desenvolvimento diretamente no editor de código com o compilador já configurado.

4.3 Dependências

O projeto utiliza apenas bibliotecas padrão do C: `stdio.h`, `stdlib.h`, `string.h`, e `locale.h`.
E a biblioteca que foi usada para colocar as funções de interface: `func.h`.

4.4 Problemas e soluções

Ao decorrer do desenvolvimento do trabalho, nos deparamos com alguns problemas que tivemos que buscar meios de solucionar, dentre eles, elencamos os seguintes e suas respectivas soluções:

- Busca CNH:

Erro: O ponteiro `cnhpbusca` não foi inicializado corretamente, resultando em falha na busca por CNH.

Solução: Corrigimos a inicialização do ponteiro, garantindo que ele apontasse para um valor válido.

- Arquivo Infinito:

Erro: A função de salvar multas estava criando um loop infinito, gerando entradas repetidas no arquivo.

Solução: Ajustamos a lógica de fluxo para garantir que os dados fossem gravados corretamente e sem duplicação.

4.5 Regras de Negócio

Registro de Multas

- Campos obrigatórios: Placa, CNH, Infração, Valor, Data/Hora.
- Validação: Placa e CNH devem ser válidas.

Cálculo do Valor da Multa

- Baseado na infração: O valor varia conforme a gravidade e reincidência.

Persistência dos Dados

- Armazenamento: Multas são gravadas em arquivos de texto sem sobrescrever dados existentes.

Busca de Multas

- Por CNH ou Placa: Exibe as multas associadas.

Exclusão de Multas

- Identificação: Exclusão realizada por número único da multa.

Validação de Dados

- Formato correto: CNH e placa devem seguir formatos válidos, com mensagens de erro para dados inválidos.

Relatórios

- Multas registradas: Relatórios geram totais e valores das multas por CNH ou placa.

Segurança

- Acesso restrito: Apenas usuários autorizados podem manipular dados.

Testes

- Testes unitários e de integração: Funções são testadas e o sistema é validado como um todo.

5. RESULTADOS

O sistema informatizado desenvolvido demonstrou eficiência na digitalização do processo de notificação de multas de trânsito, superando as limitações do modelo tradicional. A utilização de listas encadeadas como estrutura de dados foi essencial para o registro estruturado e o

gerenciamento eficiente das informações, permitindo operações dinâmicas, como adições, remoções e consultas frequentes.

Nos testes realizados, o sistema registrou corretamente dados como placa, CNH, tipo de infração e valor da multa, garantindo integridade e persistência por meio de armazenamento em arquivos de texto. A validação de dados impediu entradas inválidas, aumentando a confiabilidade do sistema. Além disso, a interface simples e funcional facilitou a interação do usuário, assegurando uma experiência intuitiva e acessível.

6. REFERÊNCIAS

CURSO EM VÍDEO. *Linguagem C*. Disponível em: <https://www.cursoemvideo.com/>.

Acesso em: 15 jan. 2025.

DEVMEDIA. *Validação de CNH e Placas em C*. Disponível em:

<https://www.devmedia.com.br/forum>. Acesso em: 22 jan. 2025.

GEEKSFORGEES. *File Handling em C*. Disponível em: <https://www.geeksforgeeks.org/>.

Acesso em: 18 jan. 2025.

GITHUB. *Repositório BRUNO SIQUEIRA*. Disponível em: <https://github.com/>. Acesso em:

10 jan. 2025.

REDDIT. *Discussões de Projetos em C*. Disponível em:

https://www.reddit.com/r/C_Programming/. Acesso em: 12 jan. 2025.

STACK OVERFLOW. *Validação de Dados em C*. Disponível em:

<https://stackoverflow.com/>. Acesso em: 25 jan. 2025.

TREINAWEB. *Manipulação de Arquivos em C*. Disponível em:

<https://www.treinaweb.com.br/>. Acesso em: 17 jan. 2025.

W3SCHOOLS. *Conceitos Básicos de C*. Disponível em: <https://www.w3schools.com/c/>.

Acesso em: 28 jan. 2025.

YOUTUBE. *Filipe Deschamps: Validação no C*. Disponível em: <https://www.youtube.com/>.

Acesso em: 19 jan. 2025.

7 ANEXOS

Código-Fonte do Sistema

O código desenvolvido para o sistema de registro de multas está disponível no seguinte repositório:

<https://github.com/Deborahrs/Trabalho-Final--Estrutura-de-dados>