

VITÓRIA-ES

MINI SISTEMA BIBLIOTECÁRIO

2020

SUMÁRIO

1 INTRODUÇÃO	3
2 DESCRIÇÃO	4
1.1 FUNCIONALIDADES	4
1.2 ESTRUTURA DO PROGRAMA	4
1.2.1 MANIPULAÇÃO DE ARQUIVOS	5
1.2.2 INTERFACE GRÁFICA INPUT/OUTPUT.....	6
1.2.3 FUNÇÕES	7
3 INTEMPÉRIES NA PROGRAMAÇÃO	9
3.1 PONTEIROS NÃO INICIALIZADOS.....	9
3.2 ANTIVÍRUS	10
4 FORMATAÇÃO DOS ARQUIVOS	11
5 CONFIGURAÇÕES	12
6 REFERÊNCIAS	13

1 INTRODUÇÃO

O objetivo deste trabalho é criar um sistema simples de controle de estoque de uma livraria.

Este trabalho tem, para além do inerente interesse acadêmico, um forte interesse prático.

2 DESCRIÇÃO

1.1 FUNCIONALIDADES

Em termos de funcionalidades, o programa permite ao utilizador:

- **Cadastrar Livros**
- **Consultar Estoque**
- **Vender Livros**
- **Consultar Saldo da loja**
- **Salvar Dados**

1.2 ESTRUTURA DO PROGRAMA

O desenvolvimento do programa teve como base uma divisão estrutural de onde resultaram os seguintes módulos:

- **Manipulação de arquivos** - define a infraestrutura de armazenamento de dados no disco rígido;
- **Interface gráfica Input/Output** - Possibilita a interação do utilizador com o programa.
- **Funções** - Contém todas as estruturas pertinentes à aplicação.

1.2.1 MANIPULAÇÃO DE ARQUIVOS

A manipulação de arquivos se deu a partir da biblioteca **stdio.h**. Além de fornecer as funções de manipulação de arquivos, essa biblioteca também define várias macros, dentre elas NULL e EOF, que definem um ponteiro nulo e o fim de arquivo, respectivamente. Além disso, é nela que está definido o tipo FILE. Sua sigla significa “*Standard input/output header*”.

As funções utilizadas foram:

Função	O que faz?
fopen()	Abre um arquivo.
fclose()	Fecha o arquivo garantindo a transferência do <i>buffer</i> .
fflush()	Descarrega o <i>buffer</i> .
fscanf()	Leitura de entrada formatada (semelhante ao scanf()).
fprintf()	Escrita de saída formatada (semelhante ao printf()).
fputc()	Insere um caracter no arquivo.

Tabela 1: Principais funções de manipulação de arquivos da biblioteca stdio.h

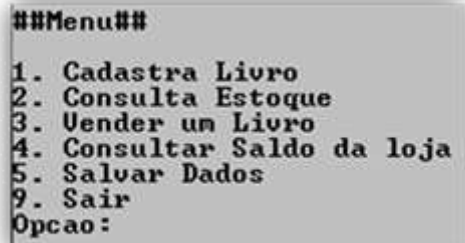
Na utilização da função *fopen()* foram utilizados os seguintes modos de abertura de arquivos:

Modo	O que faz?
”r+”	Abre o arquivo para leitura e escrita, a partir do início. O arquivo deve existir.
”w+”	Cria um arquivo vazio para leitura e escrita. Se já havia o arquivo, ele é perdido.
”a+”	Abre para adição ou leitura no final do arquivo. Se o arquivo não existir, a função o cria.

Tabela 2: Modos de abertura de arquivos

1.2.2 INTERFACE GRÁFICA INPUT/OUTPUT

A interação do usuário com a aplicação se deu a partir do próprio CMD. Com uma pequena modificação de cores usando o comando `system("color 70")`. Como podemos ver na figura abaixo:



```
##Menu##  
1. Cadastra Livro  
2. Consulta Estoque  
3. Vender um Livro  
4. Consultar Saldo da loja  
5. Salvar Dados  
9. Sair  
Opcao:
```

Figura 1: Aplicação rodando no CMD

Na parte de entrada de dados, foram aplicadas funções afim de validar a entrada, de modo que só a entrada válida é permitida. Essas funções indicam para o usuário quando a entrada é inválida, e aguardam uma entrada válida para prosseguir com a aplicação.

1.2.3 FUNÇÕES

FILE* Init_Estoque();

Essa função inicia o ponteiro do arquivo “estoque.txt” e o retorna se o arquivo for aberto com sucesso, caso contrário retorna NULL. Além de que se o arquivo “estoque.txt” não existir, o cria e em seguida o inicia com uma linha formatada nula.

FILE* Init_Saldo();

Essa função inicia o ponteiro do arquivo “saldo.txt” e o retorna se o arquivo for aberto com sucesso, caso contrário retorna NULL. Além de que se o arquivo “saldo.txt” não existir, o cria e em seguida o inicia com um saldo de 0.

int Escolha();

Essa função imprime o menu, e retorna uma opção inteira.

void Cadastra_Livro(FILE *file_estoque, Livro* EstoqueLivros, int* numero_obras_diferentes_pointer);

Essa função recebe os argumentos necessários para se cadastrar um livro, além de gravar o livro em Estoquelivros, grava o livro em “estoque.txt”.

void Consulta_Estoque(Livro* EstoqueLivros, int numero_obras_diferentes);

Essa função exibe um submenu para decisão entre *consulta por ISBN e consulta por título* . Em seguida efetua a consulta por meio das funções *consultaEstoqueTituloString()* e *consultaEstoqueISBN()* e imprime o resultado.

void vendeLivro(float* saldo_pointer, Livro* EstoqueLivros, int numero_obras_diferentes);

Essa função consulta o ISBN fornecido, utilizando *consultaEstoqueISBN()*; Se for um ISBN existente, pede pela quantidade à ser vendida. Se tiver estoque para a quantidade informada, a venda é realizada e *saldo_pointer* é modificada. Caso contrário é exibido que não tem a quantidade desejada em estoque.

void consultaSaldo(float saldo);

Essa função apenas imprime a variável saldo. Já que seu valor já foi acessado por *Puxa_saldo()*.

void salvaDados(float saldo, Livro* EstoqueLivros, int numero_obras_diferentes);

Essa função salva todos os livros de EstoqueLivros em “estoque.txt” e o saldo em “saldo.txt”.

int Puxa_Estoque(FILE* file_estoque, Livro* EstoqueLivros);

Essa função recebe todos os livros gravados em “estoque.txt” e os passa para EstoqueLivros. E retorna a quantidade de obras diferentes que existem em “estoque.txt”.

float Puxa_Saldo(FILE* file_saldo);

Essa função retorna o saldo gravado em “saldo.txt”.

void consultaEstoqueTituloString(char* string, Livro* EstoqueLivros, int numero_obras_diferentes)

Essa função “varre” a EstoqueLivros comparando a substring com o título. Utiliza *strstr()*. Exibe todas as obras com a substring informada pelo usuário.

int consultaEstoqueTitulo(char* string, Livro* EstoqueLivros, int numero_obras_diferentes);

Essa função “varre” a EstoqueLivros comparando a string com o título. Utiliza *strcmp()*. Retorna -1 se não houver nenhum livro com o título informado, retorna uma posição(=índice de EstoqueLivros[x]) válida se houver ocorrência do título informado.

int consultaEstoqueISBN(int ISBN, Livro* EstoqueLivros, int numero_obras_diferentes);

Essa função “varre” a EstoqueLivros comparando os ISBN’s. Retorna -1 se não houver nenhum livro com o ISBN informado, retorna uma posição(=índice de EstoqueLivros[x]) válida se houver ocorrência do ISBN informado.

int validadorNumeral(const char* texto);

Essa função utiliza da recursividade, para somente receber uma entrada inteira positiva e diferente de zero. Além de imprimir o texto informado.

int validadorISBN(const char* texto);

Essa função utiliza da recursividade, para somente receber uma entrada composta exclusivamente por números e com tamanho fixo de 9 números, assim como é o padrão ISBN usado como base. Além de imprimir o texto informado.

float validadorValor(const char* texto);

Essa função utiliza da recursividade, para somente receber uma entrada real válida. Além de imprimir o texto informado.

3 INTEMPÉRIES NA PROGRAMAÇÃO

3.1 PONTEIROS NÃO INICIALIZADOS

O programa “crashava” sempre que o ponteiro não inicializado tentava ser acessado. E este erro não gerava nenhuma Warning ou Error na compilação do arquivo fonte.

```
int main()
{

    int* pointer; // Declaração de um ponteiro inteiro
    int n = *pointer; // Erro o ponteiro não foi inicializado somente declarado
    printf("%d", n); // Violação de acesso

    return 0;

}
```

Tabela 3: Código com erro de ponteiro não inicializado

```
int main()
{

    int* pointer; // Declaração de um ponteiro inteiro
    int var = 10;
    pointer = &var; // Inicialização do ponteiro com um valor válido (endereço de var)
    int n = *pointer; // Sem erro na atribuição
    printf("%d", n); // Sem violação de acesso

    return 0;

}
```

Tabela 4: Código sem o erro de ponteiro não inicializado

3.2 ANTIVÍRUS

O antivírus interpretava a aplicação como um malware quando era utilizado o comando *fopen()*, juntamente com qualquer modo de abertura de arquivos para escrita. Somente leitura o mesmo problema não era observado. Isso ocorre pois o antivírus fica “atento” a qualquer modificação no disco rígido, por isso alguns vírus se utilizam exclusivamente da memória RAM, para gerar menos “suspeita”.

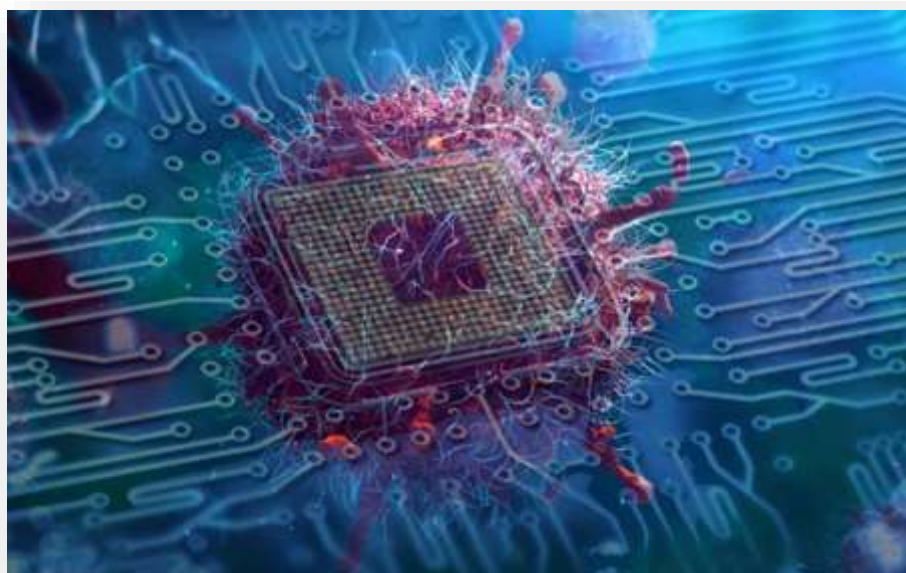


Figura 2: Vírus ilustrativo.

4 FORMATAÇÃO DOS ARQUIVOS

A formatação dos dados se deu como mostrado nas figuras abaixo:

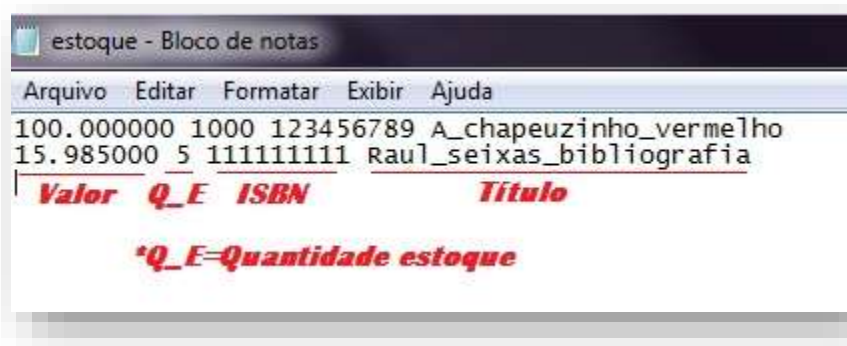


Figura 3: Print arquivo “estoque.txt”, com anotações indicativas.

Para impressão no arquivo o título foi alterado, do seguinte modo, onde havia “ ” na string foi substituído por “_”. Para leitura do arquivo o título foi alterado, do seguinte modo, onde havia “_” na string foi substituído por “ ”.

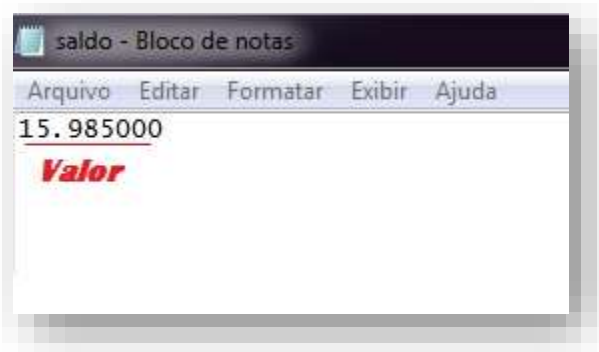



Figura 4: Print arquivo “saldo.txt”, com anotações indicativas.

5 CONFIGURAÇÕES

Versão do minGW utilizado para compilação do programa:

 i686-8.1.0-posix-dwarf-rt_v6-rev0	MinGW-W64	01/12/2020
---	-----------	------------

Sistema operacional:

Windows 7 Professional

Sistema Operacional de 64 Bits

6 REFERÊNCIAS

C Como Programar

H. M. Deitel, P. J. Deitel
Bookman. Porto Alegre, 2001

C Completo e Total

Herbert Schildt
Makron Books Editora Ltda. São Paulo, 2001

<http://www.cplusplus.com/reference/cstdio/>