

My Project

Generated by Doxygen 1.15.0

1 Banco Malvader – Sistema Bancário em C	1
1.1 Objetivos do Projeto	1
1.2 Funcionalidades	1
1.3 Estrutura de diretórios	2
1.4 Estrutura dos Módulos	2
1.4.1 banco.c	2
1.4.2 cliente.c/cliente_data.h	2
1.4.3 ordenacao.c	2
1.4.4 io.c	3
1.4.5 main.c	3
1.5 Requisitos	3
1.6 Executar no Terminal (Windows)	3
1.6.1 Passo a passo	3
1.7 Compilar manualmente com GCC	4
1.7.1 Linux	4
1.7.2 Windows (MinGW)	4
1.8 Compilando com CMake	4
1.8.1 Windows	4
1.9 Gerar documentação com Doxygen	4
1.9.1 1. Criar Doxyfile	4
1.9.2 2. Configurar	4
1.9.3 3. Gerar documentação	4
1.10 Visualizar HTML	4
1.11 Gerar PDF da Documentação com LaTeX	5
1.11.1 1. Instalar LaTeX	5
1.11.1.1 Windows (recomendado)	5
1.11.1.2 Linux (Ubuntu/Debian)	5
1.11.1.3 macOS	5
1.11.2 2. Compilar PDF	5
1.11.3 3. Abrir o PDF	5
1.11.3.1 Windows	5
1.11.3.2 Linux	5
1.11.3.3 macOS	5
1.12 Compilar pelo CLion	5
1.12.1 Comandos Úteis	6
1.12.2 Autores	6
1.12.3 Licença	6
Index	7

Chapter 1

Banco Malvader – Sistema Bancário em C

Sistema bancário modular em linguagem C, desenvolvido como projeto acadêmico.

O sistema implementa manipulação de clientes, operações bancárias, listagens, ordenação e persistência de dados em arquivos de texto.

Toda a documentação do código-fonte é gerada automaticamente com **Doxygen**.

1.1 Objetivos do Projeto

- Aplicar fundamentos de modularização em C.
 - Separação de responsabilidades entre módulos.
 - Persistência de dados utilizando arquivos texto.
 - Uso de ponteiros, estruturas, vetores dinâmicos, ordenação e busca.
 - Documentação de código profissional via Doxygen.
-

1.2 Funcionalidades

- Cadastro de clientes
 - Depósito
 - Saque
 - Transferência entre contas
 - Alteração de dados
 - Consulta de saldos
 - Exclusão lógica de clientes (marcando como inativo)
-

- Reativação de contas
 - Ordenação:
 - por nome
 - por número da conta
 - Registro de movimentações em arquivo
-

1.3 Estrutura de diretórios

```
Banco_Malvader/  
src/  
  main.c           # Função principal / loop do menu  
  banco.c          # Lógica de operações bancárias  
  banco.h  
  cliente.c        # Controle dos dados de clientes  
  cliente_data.h  
  ordenacao.c      # Ordenação (quicksort + comparadores)  
  ordenacao.h  
  io.c             # Funções de entrada e validação  
  io.h  
  ...  
data/  
  clientes.txt  
  movimentos.txt  
  ...  
README.md  
Doxyfile
```

1.4 Estrutura dos Módulos

1.4.1 banco.c

Responsável por:

- depósito
- saque
- transferência
- carregamento e gravação de dados
- busca de clientes

1.4.2 cliente.c / cliente_data.h

Define a estrutura `Cliente` e funções associadas.

1.4.3 ordenacao.c

Implementa `quicksort` e comparadores:

- por nome
- por número da conta

1.4.4 io.c

Funções de entrada e validação:

- leitura de `int`, `double` e strings
- tratamento de buffer
- leitura formatada para conta e valor

1.4.5 main.c

Controla:

- o menu
 - chamadas aos módulos
 - laço principal de execução
-

1.5 Requisitos

- GCC, Clang, MSVC ou outro compilador C.
 - CMake (opcional).
 - Doxygen (para documentação).
 - LaTeX (MiKTeX, TexLive ou MacTex – para gerar PDF).
-

1.6 Executar no Terminal (Windows)

Pré-requisitos:

- CMake instalado
- Compilador C (MinGW-w64, Clang, MSVC...)

1.6.1 Passo a passo

1. Pressione Win + R, digite:

```
cmd
```

2. Vá até o diretório do projeto, por exemplo:

```
cd "C:\Users\...\Banco_Malvader"
```

1. Para compilar:

```
compile.bat
```

1. Para executar:

```
run.bat
```

1. Ou compilação + execução automática:

```
start.bat
```

1.7 Compilar manualmente com GCC

1.7.1 Linux

```
gcc -o banco src/*.c
./banco
```

1.7.2 Windows (MinGW)

```
gcc -o banco.exe src/*.c
banco.exe
```

1.8 Compilando com CMake

```
mkdir build
cd build
cmake ..
make
./banco
```

1.8.1 Windows

```
mkdir build
cd build
cmake -G "MinGW Makefiles" ..
mingw32-make
banco.exe
```

1.9 Gerar documentação com Doxygen

Pré-requisitos:

- Doxygen instalado na máquina

1.9.1 1. Criar Doxyfile

Vá até o diretório do projeto, e digite:

```
doxygen -g
```

1.9.2 2. Configurar

No Doxyfile altere:

```
PROJECT_NAME      = "Banco Malvader"
INPUT              = ./src
RECURSIVE          = YES
GENERATE_HTML      = YES
GENERATE_LATEX     = YES
OUTPUT_DIRECTORY  = docs
```

Para usar o README como página inicial:

```
USE_MDFILE_AS_MAINPAGE = README.md
```

1.9.3 3. Gerar documentação

```
doxygen Doxyfile
```

1.10 Visualizar HTML

```
start html/index.html
```

ou

```
start docs/html/index.html
```

1.11 Gerar PDF da Documentação com LaTeX

1.11.1 1. Instalar LaTeX

1.11.1.1 Windows (recomendado)

Instalar MiKTeX:

<https://miktex.org/download>

Marque:

```
Install missing packages automatically
```

1.11.1.2 Linux (Ubuntu/Debian)

```
sudo apt update
sudo apt install texlive-full
```

1.11.1.3 macOS

```
brew install mactex
```

1.11.2 2. Compilar PDF

Depois de gerar o LaTeX com o Doxygen:

```
cd latex
```

Use o XeLaTeX (melhor para Unicode):

```
xelatex refman.tex
xelatex refman.tex
```

O PDF será criado em:

```
latex/refman.pdf
```

ou

```
docs/latex/refman.pdf
```

1.11.3 3. Abrir o PDF

1.11.3.1 Windows

```
start refman.pdf
```

1.11.3.2 Linux

```
xdg-open refman.pdf
```

1.11.3.3 macOS

```
open refman.pdf
```

1.12 Compilar pelo CLion

1. Abrir o projeto
 2. Esperar o CMake carregar
 3. Ctrl + F9 para compilar
 4. Shift + F10 para executar
-

1.12.1 Comandos Úteis

Apagar build do CMake:

```
rm -rf build
```

Gerar documentação novamente:

```
doxygen Doxyfile
```

Recompilar PDF:

```
cd latex  
xelatex refman.tex
```

1.12.2 Autores

- Carlos Eduardo Estrela
 - Cesar Amaru
 - Thyago Oliveira
 - Samuel Ribeiro Braga
 - Yan Gabriel Pereira
-

1.12.3 Licença

Projeto criado para fins acadêmicos, podendo ser utilizado como base para estudos, ensino e evolução de projetos C.

Index

Banco Malvader – Sistema Bancário em C, [1](#)