

**Instituto Politécnico de Viseu**

**Escola Superior de Tecnologia e Gestão de Viseu**

**Departamento de Informática**

**Unidade Curricular**: Estrutura de Dados

**Professor:** Francisco Morgado

**Relatório Relativo ao Trabalho Prático**

**Tema: Simulação de um Supermercado**

**Realizado por:**

Viseu, 2023

**Instituto Politécnico de Viseu**

**Escola Superior de Tecnologia e Gestão de Viseu**

**Departamento de Informática**

**Relatório relativo ao Trabalho Prático**

**Curso de Licenciatura em Engenharia Informática**

**Unidade Curricular de Estrutura de Dados**

**Simulação de um Supermercado**

**Ano Letivo 2022/23**

Índice

[1.Introdução 4](#_Toc136549501)

[2. Conceção da Arquitetura de Informação e da Aplicação 5](#_Toc136549502)

[2.1. Menu 5](#_Toc136549503)

[2.2. Estruturas de dados 6](#_Toc136549504)

[2.2.2 estruturação de dados dos clientes 6](#_Toc136549505)

[2.3. Simulação 7](#_Toc136549506)

[2.3.1. EntradaCliente 7](#_Toc136549507)

[2.3.2 ProcessarCorredor 8](#_Toc136549508)

[2.2.4. ProcessarCaixas 9](#_Toc136549509)

[2.2.4.1. GerirAberturaCaixa 9](#_Toc136549510)

[2.2.4.2. Redistribuir Clientes 10](#_Toc136549511)

[2.2.5. SairCliente 11](#_Toc136549512)

[2.2.5.1. OferecerProduto 12](#_Toc136549513)

[2.2.5.2. GerirFechoCaixa 12](#_Toc136549514)

[2.4. Gerar Estatística 13](#_Toc136549515)

[3. Bibliotecas e definições 14](#_Toc136549516)

[3.1. Biblioteca do ficheiro main.c 14](#_Toc136549517)

[3.2. Valores previamente definidos 14](#_Toc136549518)

[4. Conclusão 15](#_Toc136549519)

**INDICE DE FIGURAS**

* **Figura 1 e 2- Menu -------------------------------------------------------------------------------------------------;**
* **Figura 3- Estrutura para representar um cliente-----------------------------------------------------------;**
* **Figura 4- Estrutura para representar um produto----------------------------------------------------------;**
* **Figura 4- Estrutura para representar uma caixa------------------------------------------------------------;**
* **Figura 5- Estrutura para representar um funcionário -----------------------------------------------------;**
* **Figura 6- Estatística-------------------------------------------------------------------------------------------------;**
* **Figura 7- Estrutura para representar um supermercado -------------------------------------------------;**
* **Figura 8- Entrada de um cliente----------------------------------------------------------------------------------;**
* **Figura 9- Informações do cliente --------------------------------------------------------------------------------;**
* **Figura 10-Processo caixas -----------------------------------------------------------------------------------------;**
* **Figura 11-Gerir Abertura de caixa --------------------------------------------------------------------;**
* **Figura 12-Redistribuidor Clientes---------------------------------------------------------------------;**
* **Figura 13-Sair cliente ------------------------------------------------------------------------------------;**
* **Figura 14- Oferecer produto---------------------------------------------------------------------------;**
* **Figura 15- Gerir fecho caixa----------------------------------------------------------------------------;**
* **Figura 16- Gerar estatística----------------------------------------------------------------------------;**

# 1.Introdução

Este projeto foi desenvolvido para a unidade curricular de Estruturas de Dados da licenciatura de Engenharia Informática da Escola Superior de Tecnologia e Gestão de Viseu.

O código foi desenvolvido no software CodeBlocks e Visual Studio. Este é constituído por vários ficheiros e várias funções. Todo o código tem como base a matéria abordada nas aulas da unidade curricular e fazendo uso de algumas adaptações aos códigos desenvolvidos na mesma. Claro está, em grande parte foi necessário desenvolver funções específicas e otimizadas ao projeto em questão. Tivemos também a necessidade de fazer a nossa própria pesquisa e aprofundar alguns conhecimentos através de material disponibilizado pelos docentes e outros.

O objetivo deste projeto prático é implementar um programa em linguagem C que simule o funcionamento de um hipermercado. O programa permitirá analisar diversos parâmetros, como tempo médio de espera, número médio de clientes nas filas das caixas. Com base nesses parâmetros, o programa deverá decidir quando abrir ou fechar caixas, procurando minimizar o número de caixas abertas, mantendo o tempo de espera o mais baixo possível.

Além disso, o programa deverá implementar uma medida de reforço da confiança dos clientes. Isto é, caso um cliente exceda o tempo máximo de espera, um produto da sua lista de compras será oferecido como forma de compensação. Para controlar os custos decorrentes dessa medida, o programa deverá armazenar informações sobre o número de produtos oferecidos e seus respetivos custos.

Este projeto desafiou-nos a aplicar os conceitos e estruturas de dados estudados na unidade curricular para analisar o problema proposto e implementar uma solução correta e eficaz.

O programa desenvolvido poderá auxiliar o Conselho de Administração da cadeia de hipermercados a melhorar a qualidade de serviço, garantindo tempos de espera razoáveis, minimizando o número de caixas abertas e controlando os custos relacionados à medida de reforço da confiança dos clientes.

Neste relatório pretendemos fazer uma análise a algumas funções que considerámos mais importantes e nas quais desenvolvemos uma solução mais específica ao projeto proposto.

# 2. Conceção da Arquitetura de Informação e da Aplicação

O programa tem vários ficheiros “.c” (extensão de arquivo da linguagem de programação C) em que cada um tem a sua respetiva função e vários ficheiros “.h” onde implementamos as estruturas de dados que achamos convenientes. O ficheiro “main.c” é o ficheiro principal do programa, onde temos a função menu(), RUN() e ExecutarOpcao().

Tendo em conta que o programa vai ser executado tendo em conta alguns valores como o numero máximo de clientes que o utilizador quer que estejam, em media, em cada caixa, o tempo máximo de espera que o cliente pode esperar (oferecendo um produto caso esse tempo seja ultrapassado) e até o numero máximo de clientes permitidos dentro do supermercado. Esses valores foram são definidos no inicio do ficheiro main.c.

Durante a realização deste código tivemos sempre em atenção a prevenção de erros de compilação, fazendo uso de verificações de ponteiros nulos. Em algumas funções, onde consideramos necessário, implementamos o retorno de valores para verificar se a tarefa que cada função deveria fazer foi executada com sucesso (valor 1= ou não (valor 0).

# 2.1. Menu

A função menu() exibe opções para o usuário e solicita que ele faça uma escolha, ou seja, declara uma variável inteira chamada “opcao” para armazenar a escolha do usuário.

Mostrando assim um menu com várias opções usando a função print.

Entra em um loop do-while que solicita ao usuário que digite a sua escolha usando a função scanf() e armazena o valor na variável “opcao”.

O loop continua enquanto a escolha do usuário for menor que 0 ou maior que 9, o que garante que apenas sejam escolhidas opções válidas.

Quando o usuário faz uma escolha válida, a função retorna o valor armazenado na variável opção que será usado na função ExecutarOpcao(). A função ExecutarOpcao() irá executar a opção anteriormente digitada pelo utilizador.

Desta forma, criamos uma interface interativa num programa onde o usuário precisa fazer seleções específicas com base no menu apresentado.

**Uma imagem com texto, captura de ecrã, Tipo de letra

Descrição gerada automaticamenteUma imagem com texto, captura de ecrã, número, software

Descrição gerada automaticamente**

Figura 1 e 2 - Menu

A função RUN() tem o objetivo de chamar todas as funções de gestão do supermercado presente no ficheiro simulação.c. Será aqui que a simulação e gestão será toda feita chamando as respetivas funções.

# 2.2. Estruturas de dados

As estruturas de dados são formas de armazenar e organizar dados de maneira eficiente.

Para este projeto achamos conveniente a criação de estruturas para os clientes, as caixas, os produtos, os funcionários, as estatísticas e o supermercado. Complementando estas estruturas criamos estruturas para listas dos quatro primeiros (clientes, caixas, produtos e funcionários). E as respetivas estruturas ELEMENTO que serão os nossos nós das listas.

De seguida apresentamos as estruturas uma a uma, sendo que a parte que estrutura as listas apresentamos apenas um exemplo de clientes pois as restantes são nos mesmos moldes apenas alterando os nomes, convenientemente, e colocando como ponteiro \*info a estrutura em causa.

# 2.2.2 estruturação de dados dos clientes

Uma imagem com texto, captura de ecrã, Tipo de letra, número

Descrição gerada automaticamente

Figura 3-Estrutura para representar um cliente

Uma imagem com texto, captura de ecrã, número, Tipo de letra

Descrição gerada automaticamente

Figura -Estrutura para resentar um produto

Uma imagem com texto, captura de ecrã, Tipo de letra, número

Descrição gerada automaticamente

Figura -Estrutura para representar uma caixa

Uma imagem com texto, captura de ecrã, Tipo de letra

Descrição gerada automaticamente

Figura -Estrutura para representar um funcionario

Uma imagem com texto, captura de ecrã, Tipo de letra, número

Descrição gerada automaticamente

Figura -Estatísticas

Uma imagem com texto, captura de ecrã, Tipo de letra, número

Descrição gerada automaticamente

Figura -Estrutura para representar o supermercado

# 2.3. Simulação

O objetivo das funções presentes neste ficheiro é, com a ajuda de outras funções, simular o funcionamento do supermercado, incluindo a entrada de clientes, o processamento de corredores, o atendimento nas caixas e a saída dos clientes. Todas essas etapas são geridas por funções próprias que trabalham juntas para simular o funcionamento de um supermercado. E gerem, ainda, a abertura e o fecho das caixas. O código também inclui chamadas a outras funções que podem estar definidas em outros arquivos, como funções de manipulação de listas e funções auxiliares para cálculos e gravação de dados.

Neste ficheiro simulação.c temos as seguintes funções:

# Uma imagem com texto, captura de ecrã, Tipo de letra Descrição gerada automaticamente2.3.1. Entrada Cliente um cliente e os produtos que compra, atualiza ainda as informações do cliente e introduz o mesmo na lista de clientes dentro do supermercado.

Figura -Entrada de um Cliente

* Depois de alocar a memoria, a função EscolherCliente escolhe um cliente disponível da lista de clientes e armazena o resultado na variável C.
* Chama a função EscolherProdutosComprados para escolher os produtos que o cliente irá comprar, e armazena a lista resultante na variável C->produtos\_comprados.
* Chama a função AtualizaHorasCliente para atualizar as horas do cliente com base no relógio do supermercado, e armazena o resultado na variável C.
* Insere o cliente na lista de clientes do supermercado usando a função InserirInicioClienteOrdem, que insere o cliente por ordem de quem irá primeiro para a caixa.
* Grava a ação de entrada do cliente no registo do supermercado através da função Gravar e depois retorna o ponteiro para o cliente C, o cliente que entrou.

# 2.3.2 Processar Corredor

Uma imagem com texto, captura de ecrã, número, Tipo de letra

Descrição gerada automaticamente

Figura -Informações do cliente

Essa função processa o movimento dos clientes pelos corredores do supermercado. Ela verifica se existem clientes dentro do supermercado e quais estão prontos para irem para as caixas. Posteriormente seleciona uma caixa disponível e que tenha o menor tempo de espera. Move, então, o cliente para a caixa correspondente, atualizando as informações do cliente e da caixa.

* Obtém o tempo atual do relógio do supermercado usando a função VerTempoRelogio.
* Entra em um loop que percorre todos os clientes dentro do supermercado verificando se o horário em que o cliente pode entrar na fila de um caixa é menor que o tempo atual. Escolhendo assim uma caixa disponível e com menor tempo de espera, através da função EscolherCaixa.
* De seguida atualiza os dados da caixa usando a função AtualizaDadosCaixa.
* Adiciona o cliente à fila da caixa escolhido usando a função AddFila e nisso remove o cliente, pelo código, da lista de clientes nos corredores usando a função RemoverClienteCodigo.
* Atualiza o tempo de espera de todos os clientes usando a função TempoEsperaCliente.
* Grava a ação de processamento do corredor no registo de operações através a função Gravar e assim termina.

# 2.2.4. Processar Caixas

Essa função é responsável por processar as caixas do supermercado. Ela verifica a necessidade de abrir novas caixas, redistribui os clientes para a caixa aberta e atualiza as informações dos clientes em espera.

Uma imagem com texto, captura de ecrã, Tipo de letra, número

Descrição gerada automaticamente

Figura -Processar caixas

Verifica se é necessário abrir novas caixas com base na função GerirAberturaCaixa, presente noutro ficheiro, e se a função retornar 1, indica que foi necessário abrir uma nova caixa e que tudo foi executado com sucesso. Essa verificação é feita mediante o número medio de clientes por caixa.

# 2.2.4.1. Gerir Abertura Caixa

Uma imagem com texto, captura de ecrã, Tipo de letra, número

Descrição gerada automaticamente

Figura -Gerir Abertura de Caixa

* Aqui é calculada a média de clientes em cada caixa usando a função MediaClientesCaixa e em seguida o código verifica se a média de clientes é maior que o número máximo de clientes permitidos no supermercado (S->n\_max\_clientes) previamente definida no ficheiro main.c .
* Se a média de clientes for maior que o número máximo, a função chama a função AbrirCaixa para abrir uma nova caixa no supermercado (CX = AbrirCaixa(S)).
* Se o ponteiro CX não for nulo, isto é, uma caixa foi corretamente aberta, é mostrada uma mensagem dando a informação de que a caixa foi aberta e o seu respetivo número
* Posteriormente, chama a função Gravar para registrar a abertura da caixa no registro de operações. Retornando ainda o valor 1, o que significa que foi necessário abrir uma caixa e tudo foi realizado com sucesso.

Chama a função RedistribuirClientes, presente noutro ficheiro, para redistribuir os clientes entre as caixas disponíveis e a função EscolherCaixa é usada para escolher um caixa disponível e com o menor tempo de espera, atualizando também os tempos de espera dos clientes usando a função TempoEsperaCliente.

# 2.2.4.2. Redistribuir Clientes

Uma imagem com texto, captura de ecrã, Tipo de letra

Descrição gerada automaticamente

Figura 12-Redistribuir Clientes

* Obtém o número total de clientes em todos as caixas usando a função TotalClientesCaixas, e faz com que de seguida se inicie um loop que percorre todos as caixas do supermercado.
* No primeiro loop, inicializa o ponteiro NO\_cliente para percorrer a fila de clientes de cada caixa, depois inicia-se o segundo loop que percorre todos os clientes na fila de cada caixa.
* Dentro do segundo loop, verifica se o tempo de espera do cliente (NO\_cliente->info->tempo\_espera) é maior que o tempo de atendimento da caixa atual (CX->tempo\_atendimento).
* Se o tempo de espera do cliente for maior que o tempo de atendimento da caixa atual, o cliente é transferido para a nova caixa (CX) e se isso acontecer, o código atualiza os dados da nova caixa.
* Depois de percorrer todos os clientes da fila de um caixa, o código atualiza os dados da caixa atual (CX) e passa para a próxima caixa, continuando o loop até que todas as caixas sejam percorridas.
* No final, a função retorna NULL, indicando que a redistribuição dos clientes foi concluída.

Grava a ação de processamento das caixas no registo de operações usando a função Gravar.

Por fim imprime informações sobre a conclusão da função para fins de depuração.

# 2.2.5. Sair Cliente

**Uma imagem com texto, captura de ecrã, Tipo de letra

Descrição gerada automaticamente**Essa função lida com a saída dos clientes do supermercado. Ela verifica o tempo atual, processa a saída dos clientes que já concluíram suas compras, oferece produtos a clientes que esperaram muito tempo na fila, calcula o valor total da compra do cliente, atualiza as informações de faturação das caixas e dos funcionários e remove os clientes atendidos das filas. Coloca, ainda, os clientes atendidos na lista de clientes que foram atendidos pela caixa em questão e verifica a necessidade de encerrar caixas.

Figura -Sair cliente

* Obtém o tempo atual através da função VerTempoRelogio e armazena-o na variável tempo\_atual.
* Inicializa os ponteiros NO\_caixa e NO\_cliente para percorrer a lista de caixas e a fila de clientes em cada caixa, respectivamente e enquanto houver caixas na lista, a variável inicializa com um valor como zero para acumular o valor total da compra.
* Inicia um loop para percorrer os clientes na fila da caixa atual enquanto o tempo de saída do cliente for menor ou igual ao tempo atual e depois imprime a mensagem de que o cliente saiu. Desta forma o loop apenas é interrompido quando todos os clientes que têm de sair tiverem saído.

Se o cliente esperou muito tempo chama se a função OferecerProduto e oferece-se um produto se necessário.

# 2.2.5.1. Oferecer Produto

Uma imagem com texto, captura de ecrã, Tipo de letra, número

Descrição gerada automaticamente

Figura -Oferecer produto

É uma função que verifica se um cliente (C) esperou muito tempo na fila da caixa e, se for o caso, oferece o produto de menor valor que o cliente comprou.

* Verifica se o tempo total de espera do cliente (C->tempo\_espera\_total) é maior que o tempo máximo de espera permitido no supermercado (S->tempo\_max\_espera), definido previamente no ficheiro main.c.
* Se o tempo total de espera do cliente for maior que o tempo máximo de espera, a função chama a função ProcurarProdutoMenorValor para encontrar o produto de menor valor na lista de produtos comprados pelo cliente.
* Em seguida, a função chama a função InserirProduto para inserir o produto de menor valor na lista de produtos oferecidos pelo supermercado.

Calcula o valor de compra do cliente com a função ValorCompra, seguido de inserir o cliente na lista de clientes atendidos usando a função InserirInicioCliente e depois remove o mesmo usando a função RemoverFila.

Caso estejam abertas mais que duas caixas chama a função GerirFechoCaixa para verificar a necessidade de encerrar caixas.

# 2.2.5.2. Gerir Fecho Caixa

Uma imagem com texto, captura de ecrã, Tipo de letra, número

Descrição gerada automaticamente

Figura -Gerir fecho caixa

A função GerirFechoCaixa é a responsável por verificar a necessidade de fechar caixas.

* Calcula a média de clientes em cada caixa usando a função MediaClientesCaixa e armazena o resultado na variável media\_clientes.
* Calcula o número de caixas abertas usando a função TotalCaixasAbertas e armazena o resultado na variável caixas abertas.
* Inicializa o ponteiro NO\_caixa com o início da lista de caixas do supermercado (S->caixas->Inicio).
* Verifica se a média de clientes é menor que o número máximo de clientes permitidos no supermercado (S->n\_max\_clientes), previamente definido no ficheiro main.c e se o número de caixas abertas é maior que o número de caixas permanentes (S->caixas\_permanentes), definido pelo utilizador no inicio da execução do programa, se for verdadeira, entra em um loop para percorrer a lista de caixas e fecha a primeira que aparece.
* Se não fechar nenhuma caixa, a função termina sem realizar mais ações.

Ainda na função SairCliente, a mesma atualiza os tempos de espera dos clientes usando a função TempoEsperaCliente e grava a ação de saída dos clientes no registo de operações usando a função Gravar.

# 2.4. Gerar Estatística

Uma imagem com texto, captura de ecrã, Tipo de letra, número

Descrição gerada automaticamente

Figura 16-Gerar Estatística

A função GerarEstatisticas é responsável por calcular e gerar estatísticas relacionadas ao funcionamento do supermercado. Obtém o ponteiro para a estrutura ESTATISTICAS dentro da estrutura SUPERMERCADO e armazena-o na variável estatisticas.

Nesta função são calculadas/obtidas as seguintes estatísticas:

* Total de clientes atendidos
* Total de caixas abertas durante a execução ainda se encontrem fechadas aquando da chamada da função gerar estatísticas
* Número total de produtos vendidos
* Número total de produtos oferecidos
* Total de faturação
* Total de perdas (através dos produtos oferecidos)
* Total de tempo que o supermercado esteve aberto até à chamada da função
* Media de tempo de espera
* Media de faturação, isto é, media de gastos por cliente

Ao longo do código essas funções são sempre armazenadas, na estrutura do tipo estatísticas do supermercado.

A função encerra, não havendo valor de retorno.

# 3. Bibliotecas e definições

# 3.1. Biblioteca do ficheiro main.c

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <time.h>

#include <stdbool.h>

#include <unistd.h>

#include <windows.h>

#include <malloc.h>

#include <ctype.h>

#include <locale.h>

#include <math.h>

# 3.2. Valores previamente definidos

#define INDISPONIVEL 0

#define DISPONIVEL 1

Na linha "#define INDISPONIVEL 0" define uma constante chamada INDISPONIVEL com o valor 0.

Em "#define DISPONIVEL 1" define a constante DISPONIVEL com o valor 1.

#define TEMPO\_MAX\_ESPERA 120

#define N\_CLIENTES\_MAX 10

#define CLIENTES\_MAX 100

A constante "#define TEMPO\_MAX\_ESPERA 120" define o tempo máximo de espera em segundos para que após isso, produtos sejam oferecidos aos clientes. Neste caso 120 segundos.

A constante "#define N\_CLIENTES\_MAX 10" define em 10 o número máximo, em media, de clientes por caixa.

A constante “#define CLIENTES\_MAX 100” define o número máximo de clientes dentro do supermercado como 100.  
Todas estas constantes podem ser alteradas conforme necessário.

# 4. Conclusão

O desenvolvimento deste projeto prático permitiu explorar os nossos conhecimentos e desafios enfrentados na criação e desenvolvimento deste tipo de soluções para uma cadeia de hipermercados. O programa implementado demonstrou a capacidade de simular o funcionamento do hipermercado e analisar parâmetros importantes, como tempo médio de espera, número médio de clientes nas filas, faturação e perdas.

Ao tomar decisões estratégicas com base nos dados coletados, foi possível equilibrar o tempo de espera dos clientes e o tamanho das filas. A medida de reforço da confiança dos clientes, que oferece um produto do carrinho de compras em caso de espera excessiva, também foi implementada para garantir a qualidade de serviço e a satisfação dos clientes.

Ao finalizar este projeto, compreendemos a importância de aplicar os conceitos e estruturas de dados estudados na unidade curricular para resolver problemas reais. A capacidade de análise, tomada de decisões e implementação de soluções eficazes foram habilidades aprimoradas ao longo do processo.

No entanto, é importante ressaltar que este projeto tem margem para alterações, como por exemplo nas informações exibidas durante a sua execução. Optamos por imprimir a menor quantidade de informação possível para tornar a consola mais simples de entender no entanto dados como todas as informações dos clientes e produtos têm opção de serem mostrados. O código foi feito a pensar nas necessidades que possam surgir.

Em suma, o desenvolvimento deste projeto prático proporcionou-nos uma visão aprofundada dos desafios enfrentados pela administração de um hipermercado ao otimizar a afetação de empregados. A implementação de um programa capaz de simular o funcionamento do hipermercado e analisar dados permitiu-nos tomar decisões estratégicas.

Esperamos que este projeto tenha contribuído para o aprimoramento das nossas habilidades e que nos tenha preparado para enfrentar desafios similares no futuro.