**Engenharia Informática**

Estruturas de Dados

2019/2020

**Relatório**

**Ficheiros de Texto e Binarios**

**António Pereira Nº2019141051**

**David Ferreira Nº2019141281**

**Data: 05/04/2020**

Índice

[1 Introdução 3](#_Toc37525437)

[2 Trabalho realizado 4](#_Toc37525438)

[2.1 Escreva três subprogramas que executem as seguintes tarefas: 4](#_Toc37525439)

[2.2 Elabore subprogramas que contabilizem, o número de linhas, o número de palavras, o número de caracteres, o número de ocorrências de um caractere indicado pelo utilizador existentes num determinado ficheiro de texto indicado pelo utilizador. 5](#_Toc37525440)

[2.3 Altere os subprogramas anteriores de forma que seja escrito, no final do ficheiro, a informação de cada uma das alíneas, assim como a hora em que foram registadas. NOTA: procure na biblioteca <time.h> as funções time(...), ctime(...). 7](#_Toc37525441)

[2.4 Elabore os subprogramas que considerar necessários para criar ficheiros de *n* números naturais (100, 500, 1000) ordenados por ordem crescente, por ordem decrescente e aleatoriamente. 9](#_Toc37525442)

[2.5 Desenvolva um subprograma que escreva num ficheiro os primeiros 50 números do ficheiro *aleat100.txt*, os 100 primeiros números do ficheiro *aleat500.txt* e os primeiros 500 números do ficheiro *aleat1000.txt*. Analise o ficheiro obtido e separe em dois novos ficheiros os números pares (em *pares.txt*) dos números ímpares (em *impares.txt*). Averigue se há mais números pares ou números ímpares no primeiro ficheiro criado. 11](#_Toc37525443)

[2.6 Desenvolva um programa que faça a gestão dos alunos inscritos numa turma. Para cada aluno deve conhecer-se o nome e o número. 14](#_Toc37525444)

[2.7 Para averiguar as diferenças entre ficheiros de texto e ficheiros binários, crie subprogramas para cada uma das seguintes alíneas. Todas as alíneas devem ser testadas no main 18](#_Toc37525445)

[2.8 Refaça o exercício 6: usando ficheiros binários, desenvolva um programa que faça a gestão dos alunos inscritos numa turma. Para cada aluno deve conhecer-se o nome e o número 24](#_Toc37525446)

[2.9 Pretende-se guardar num ficheiro binário chamado *temperaturas.dat* os dados recolhidos por um termómetro digital automático que regista a temperatura de 15 em 15 minutos 25](#_Toc37525447)

[3 Conclusão 30](#_Toc37525448)

[4 Referências 31](#_Toc37525449)

[5 Anexos Erro! Marcador não definido.](#_Toc37525450)

# Introdução

Com este trabalho pretende-se criar programas que guardem um determinado conteúdo, introduzido pelo utilizador, em ficheiros texto. Também pretende-se conseguir ler de um ficheiro texto um determinado conteúdo seja ele já existente ou introduzido pelo o utilizador. Outro objetivo deste trabalho é conseguir guardar um determinado conteúdo num ficheiro binário e ler desse ficheiro esse conteúdo.

# Trabalho realizado

## Escreva três subprogramas que executem as seguintes tarefas:

criar um ficheiro com o nome e conteúdo indicados pelo utilizador;

acrescentar informação, fornecida por um utilizador, a um ficheiro;

escrever na consola o conteúdo de um ficheiro.

**Nome do Ficheiro:** exe1\_ED.c

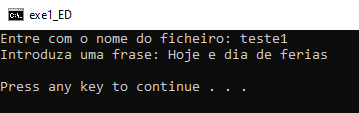
Neste primeiro exercício foi-nos pedido para fazer três tarefas. A primeira é criar um ficheiro com o nome e o conteúdo introduzido pelo utilizador. A segunda tarefa é para o utilizador conseguir acrescentar informação ao ficheiro, para isso, abríamos o ficheiro no modo de abertura “a”. Este modo de abertura deixa utilizador acrescentar informação e se o ficheiro não existir cria-o e se existir coloca o ponteiro no final e deste modo não apaga o conteúdo do ficheiro. A última tarefa é escrever na consola o conteúdo do ficheiro. Para isso o main tem que ter dois argumentos: o primeiro argumento é o argc e o segundo é o argv. O argumento argc indica o número de argumentos da linha de comando, já o segundo argumento argv é um array de ponteiros de caracteres que lista todos os argumentos. O programa deste exercício é o seguinte:



Figura 1- Programa para escrever conteúdos num ficheiro

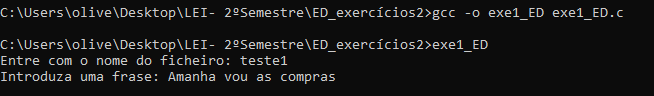
Output do exercício 1 no terminal do Atom:

Figura 2- Criação do ficheiro teste1 e escrever conteúdo nesse ficheiro



Output deste exercício na linha de comandos do Windows:

Figura 3- Acrescentar conteúdo ao ficheiro teste1



## Elabore subprogramas que contabilizem, o número de linhas, o número de palavras, o número de caracteres, o número de ocorrências de um caractere indicado pelo utilizador existentes num determinado ficheiro de texto indicado pelo utilizador.

**Nome do Ficheiro:** exe2\_ED.c

Para este exercício fizemos todo num único programa. Este exercício consistia em pedir ao utilizador o nome do ficheiro para isso, abrimos o ficheiro no modo de leitura(r), pois neste modo o programa não vai apagar o ficheiro, mas vai apontar o ponteiro para o início do ficheiro e depois o que programa faz é contar o número de linhas, palavras, caracteres e o número de ocorrências de um determinado caractere introduzido pelo o utilizador. Para contar o número de linhas o que o programa faz é percorrer o conteúdo do ficheiro e quando for igual ao “\n” incrementa uma variável contadora que é para contar o número de linhas. Já para contar as palavras, sempre que for igual a um espaço vazio ou igual ao “\n” é uma palavra, no entanto para contar o número de caracteres é o contrário de contar as palavras, ou seja, sempre que for diferente de um espaço vazio ou diferente do “\n” é um caractere. Por último, para contar o número de ocorrências de um determinado caractere pedimos ao utilizador que insira o caractere que quer procurar e depois o que o programa faz é ver se o caractere que o utilizador inseriu é igual aos caracteres que estão no ficheiro, sempre que for igual incrementa uma variável contador que é para saber o número de ocorrências desse caractere.

O programa deste exercício é o seguinte:

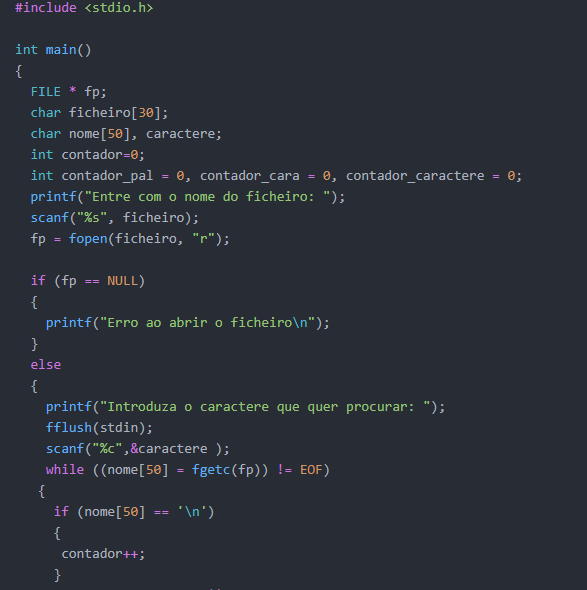


Figura 4- Programa para contar o número de linhas, palavras, caracteres e de um determinado caractere introduzido pelo o utilizador

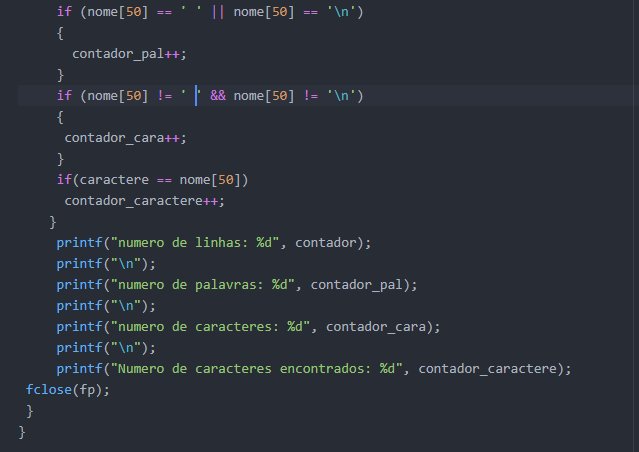
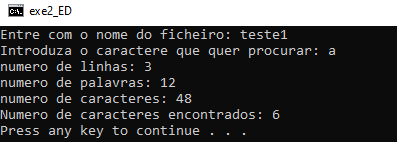


Figura 5- Programa para contar o número de linhas, palavras, caracteres e de um determinado caractere introduzido pelo o utilizador

Exemplo de output deste exercício:

Figura 6- Output do exercício 2



## Altere os subprogramas anteriores de forma que seja escrito, no final do ficheiro, a informação de cada uma das alíneas, assim como a hora em que foram registadas. NOTA: procure na biblioteca <time.h> as funções time(...), ctime(...).

**Nome do Ficheiro:** exe3\_ED.c

Neste exercício, é pedido para acrescentarmos aos ficheiros dos exercícios anteriores a informação de cada uma das alíneas e a hora em que foram registados. Para isso, utilizamos a variável time\_t que serve para guardar a hora. Já para guardar a data utilizamos a função time e para apresentar a data e a hora utilizamos a função ctime.

Para utilizar estas funções é preciso a libraria time.h. O programa é o seguinte:

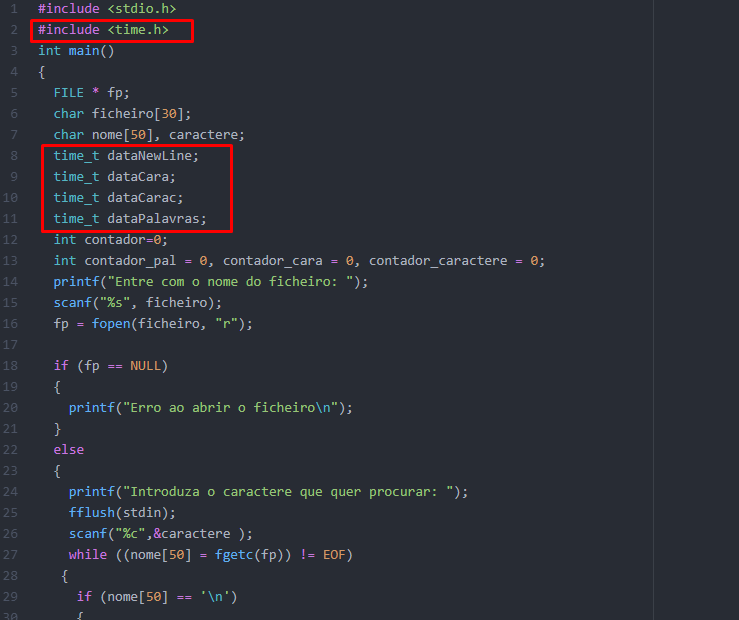


Figura 7- Programa para apresentar a hora e data de cada alínea

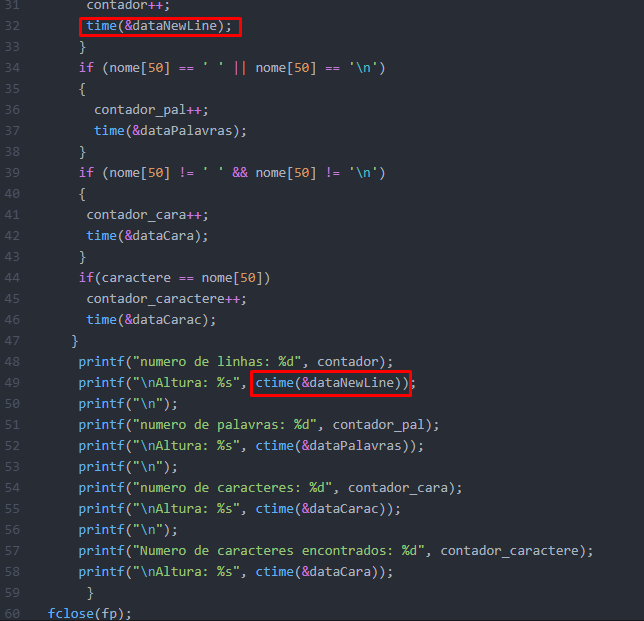
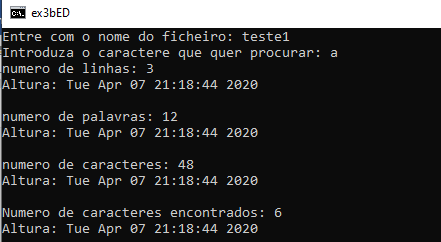


Figura 8- Programa para apresentar a hora e data de cada alínea

O output deste exercício é o seguinte:

Figura 9- Output do exercício 3



## Elabore os subprogramas que considerar necessários para criar ficheiros de *n* números naturais (100, 500, 1000) ordenados por ordem crescente, por ordem decrescente e aleatoriamente.

**Nome do Ficheiro:** exe4RandomED.c

Neste exercício a professora pediu-nos para criar ficheiros que estivessem ordenados por ordem decrescente, crescente e aleatoriamente. O nome dos ficheiros tem de indicar os números de elementos e o modo como estão ordenados, para isso o que fizemos foi pedir ao utilizador que introduzisse a quantidade de números(100, 500,1000) que quer ter nos ficheiros e depois dentro das funções esse número e o nome do ficheiro, que é diferente para cada função e já é indicado em cada função, vão ser concatenados através da função snprintf para indicar o nome de cada ficheiro, isto é, o ficheiro cresc500.txt contém os números entre 1 a 500 por ordem crescente. A primeira função é para ordenar os números por ordem crescente, a segunda função é a da ordenar os números por ordem decrescente e a última função é para ordenar por ordem aleatória. Na última função para introduzir os números por ordem aleatória utilizamos a função rand que vai escrever no ficheiro os números por ordem aleatória entre 1 e a variável quantidade (variável referente ao número de elementos que o utilizador quer introduzir), por exemplo, se o utilizador quiser inserir 100 números o que esta função vai fazer é escrever 100 números entre 1 a 100. As funções são as seguintes:

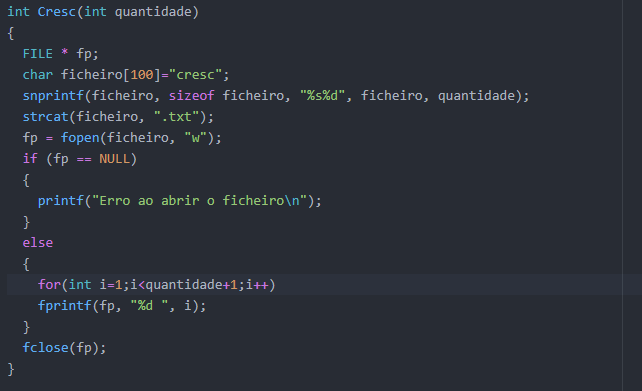


Figura 10- Função para escrever no ficheiro os números por ordem crescente

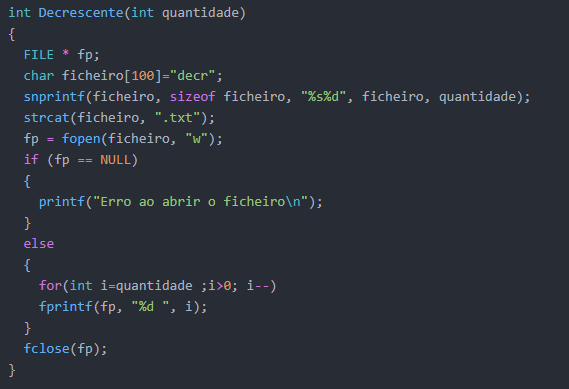


Figura 11- Função para escrever no ficheiro os números por ordem aleatória

Figura 12- Função para escrever no ficheiro os números por ordem decrescente

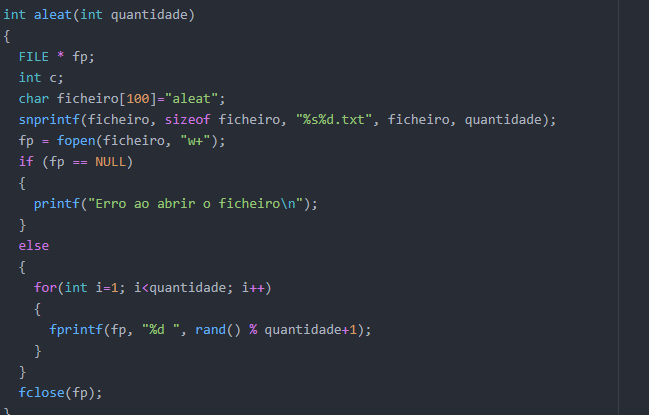
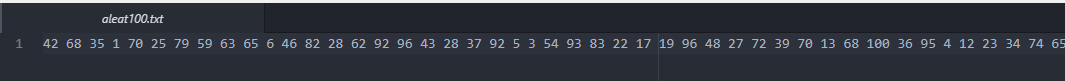
Os ficheiros são os seguintes:

Figura 13- Ficheiro onde estão os números por ordem aleatória



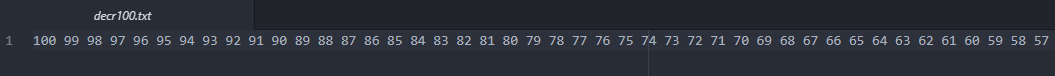
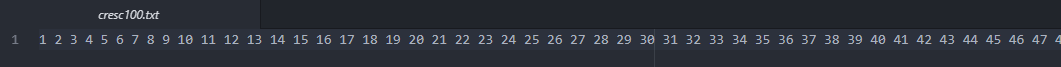


Figura 14- Ficheiro onde estão os números por ordem crescente

Figura 15- Ficheiro onde estão os números por ordem decrescente

## Desenvolva um subprograma que escreva num ficheiro os primeiros 50 números do ficheiro *aleat100.txt*, os 100 primeiros números do ficheiro *aleat500.txt* e os primeiros 500 números do ficheiro *aleat1000.txt*. Analise o ficheiro obtido e separe em dois novos ficheiros os números pares (em *pares.txt*) dos números ímpares (em *impares.txt*). Averigue se há mais números pares ou números ímpares no primeiro ficheiro criado.

**Nome do Ficheiro:** exe5\_ED.c

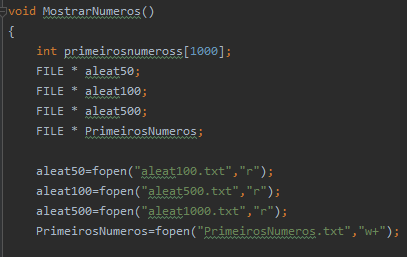
Para fazer este exercício criámos 2 funções, sendo a primeira esta:

Figura 16- Ficheiros onde vão ser escritos os números dos ficheiros do exercício 4

Aqui, criámos 3 ficheiros com conteúdos dos ficheiros aleat do exercício anterior. O 4º ficheiro, vai servir para armazenar os valores dos outros 3, que vai também ser útil para separar em 2 ficheiros os números pares e os números ímpares. Para armazenarmos no ficheiro “PrimeirosNumeros.txt”, fizemos o seguinte:



Figura 17- Ciclos para guardar os números no ficheiro PrimeirosNumeros.txt

O que fizemos aqui foi simplesmente guardar dentro de um vetor o conteúdo dos 3 ficheiros de acordo com o que a professora pretendia. Depois de guardarmos no vetor, simplesmente colocámos o que estava dentro do vetor, no ficheiro “PrimeirosNumeros.txt”.

Para separarmos o conteúdo do ficheiro em 2 ficheiros, de números pares e números ímpares, fizemos a seguinte função:

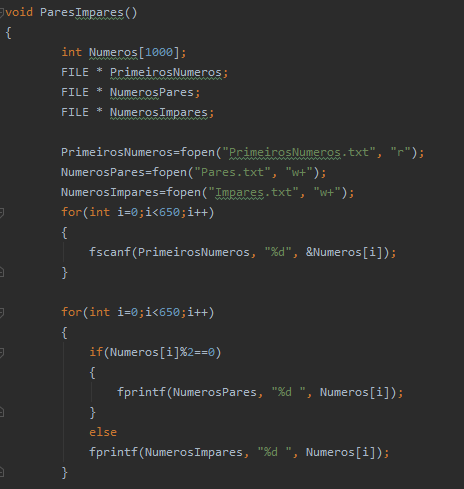


Figura 18- Função para escrever nos ficheiros os números pares e ímpares

O que fizemos nesta função foi simplesmente colocar os números do ficheiro “PrimeirosNumeros.txt” num vetor e de seguida, um ciclo que verificasse os números do vetor. Caso sejam pares, vão para o ficheiro “Pares.txt”, caso contrário, caso sejam ímpares, vão para o ficheiro “Ímpares.txt”.

## Desenvolva um programa que faça a gestão dos alunos inscritos numa turma. Para cada aluno deve conhecer-se o nome e o número.

**Nome do Ficheiro:** exe6\_ED.c

Neste exercício, foi-nos pretendido pela professora criar um programa que faça a gestão de alunos inscritos numa turma. O programa tem que permitir listar alunos, remover alunos, inscrever alunos e para cada execução apresentar os dados dos alunos inscritos. Para isso fizemos um menu de opções pois torna-se mais fácil para o utilizador usar e para nós. O menu de opções é o seguinte:

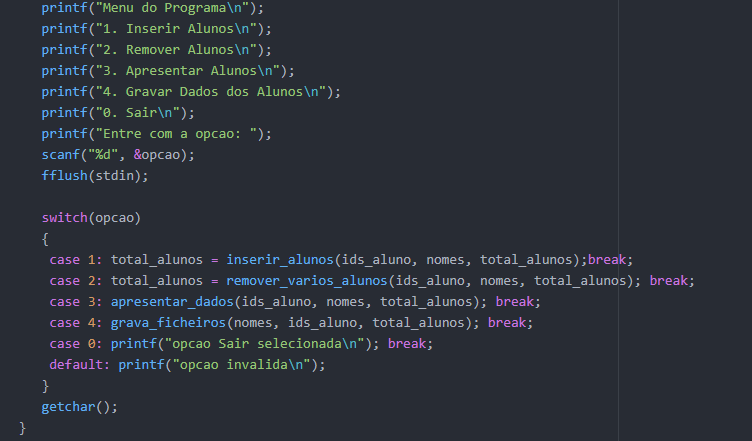


Figura 19- Menu de opções do exercício 6

A primeira opção do programa é de inserir alunos. Esta função permite que o utilizador consiga inserir os dados dos alunos, isto é, nome e número de aluno. A função é a seguinte:

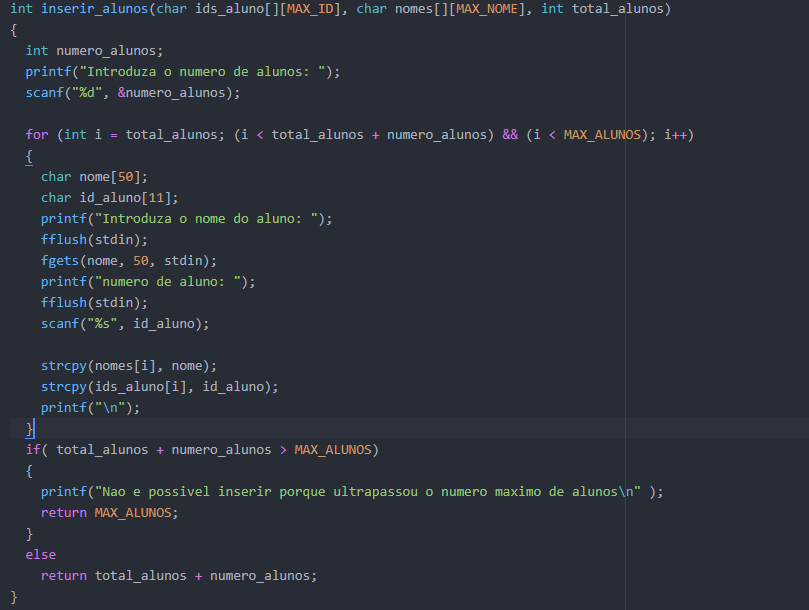


Figura 20- Função para inserir alunos

Output da função de inserir alunos:

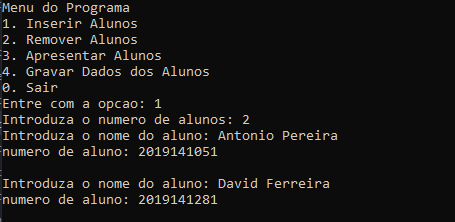


Figura 21- Interface da função inserir\_alunos

A segunda opção do nosso programa é a de remover alunos. Nesta função o utilizador tem que inserir o número do aluno que quer remover e o programa vai comparar o número que o utilizador inseriu com os números que se encontram na string ids\_alunos se for igual essa posição vai ficar com os dados dos alunos seguintes. A função de puxar os elementos atras e a função de remover alunos são as seguintes:

Figura 22- Função para puxar os dados dos alunos uma posição atrás

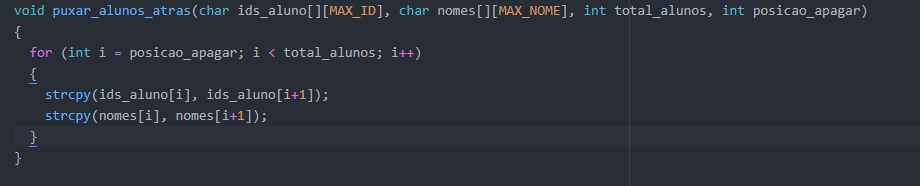
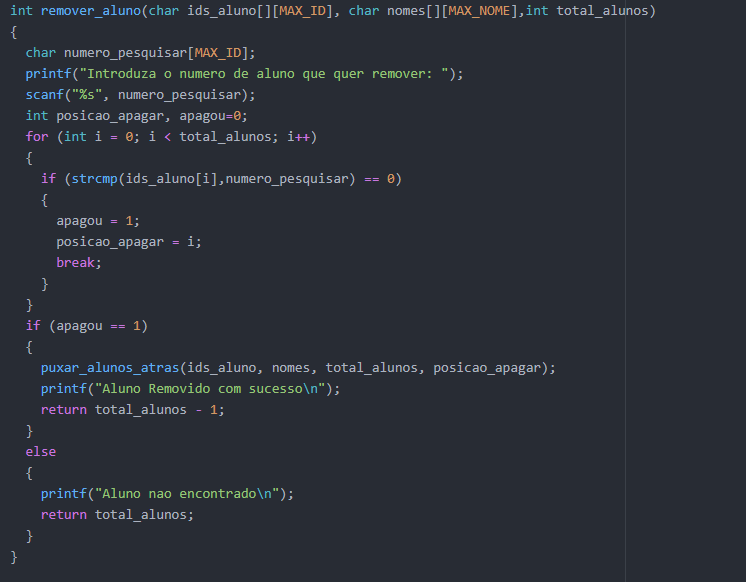


Figura 23- Função para remover alunos

Para o utilizador conseguir remover mais que um aluno fizemos uma função que pede ao utilizador que insira quantos alunos quer remover. É realizado um for que vai ate ao número de alunos que o utilizador quer remover e dentro deste for é chamada a função que remove os alunos. Esta função é a que é chamada no menu de opções. A função para remover mais que um aluno é a seguinte:

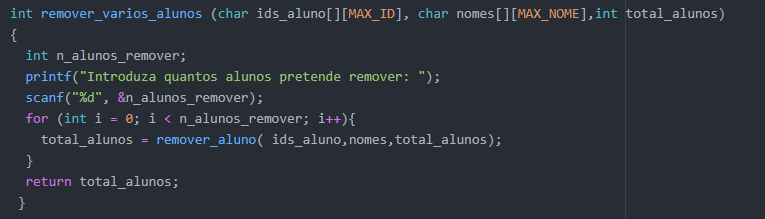
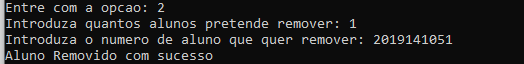


Figura 24- Função para remover mais que um aluno

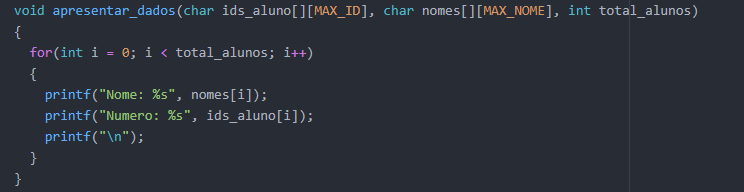
Exemplo de output da função de remover alunos:

Figura 25- Interface da função remover\_varios\_alunos



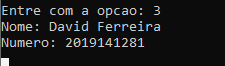
A terceira opção do menu é a de apresentar alunos. Esta função o que faz é imprimir o nome e número de aluno. Para isto fizemos um for a percorrer a lista dos alunos e dentro deste for são realizados dois printfs um para o nome dos alunos e outro para os números dos alunos. A função é a seguinte:

Figura 26- Função para apresentar alunos



O output desta função é o seguinte:

Figura 27- Interface da função apresentar\_dados



A última opção do nosso menu é para gravar os dados dos alunos. Nesta função o ficheiro é aberto no modo de escrita(w) e o que a função faz é escrever no ficheiro de texto o nome e número de alunos através de dois fputs. Para o programa conseguir ler os dados dos alunos do ficheiro fizemos uma função para ler os dados, para isso, abríamos o ficheiro anterior no modo de leitura. Depois disto, é realizado um while que percorre todo o ficheiro e dentro deste while é realizado um fgets, que serve para ler os nomes dos alunos, e a leitura dos números é feita no while através também de um fgets. As funções de gravar e ler os dados dos alunos são as seguintes:

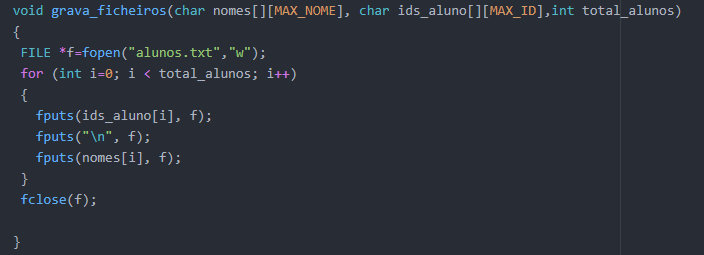


Figura 28- Função para gravar os dados dos alunos

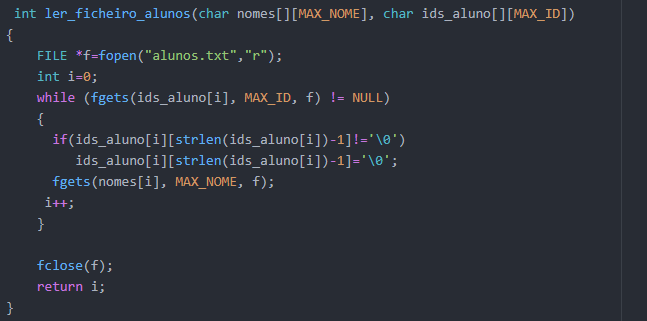


Figura 29- Função para ler os dados dos alunos

## Para averiguar as diferenças entre ficheiros de texto e ficheiros binários, crie subprogramas para cada uma das seguintes alíneas. Todas as alíneas devem ser testadas no main

**Nome do Ficheiro:** exe7\_ED.c

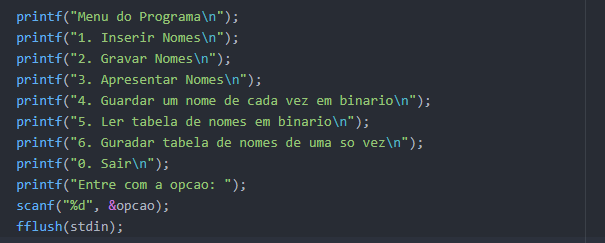
Neste exercício, foi-nos pretendido pela professora verificar as diferenças entre os ficheiros de texto e ficheiros binários. Para isso, a professora pediu que inseríssemos os nomes dos alunos e guardar esses nomes tanto em ficheiros de texto como em ficheiros binários e ler os nomes dos ficheiros de texto e dos ficheiros binários. Deste modo, para se tornar mais simples e fácil de compreender para nós e para o utilizador decidimos fazer um menu de opções como fizemos para o exercício anterior. O menu é o seguinte:

Figura 30- Menu de opções do exercício 7

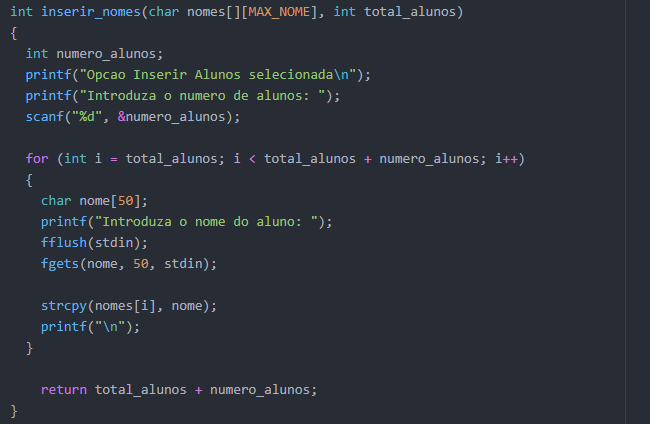
Para a alínea a) utilizamos parte da função de inserir alunos do exercício anterior, no entanto, o que fizemos de diferente foi pedir ao utilizador que inserisse os nomes dos alunos pois nesta alínea não nos é pedido para o utilizador inserir o número do aluno. A função é a seguinte:

Figura 31- Função para inserir os nomes dos alunos

O output desta função é o seguinte:

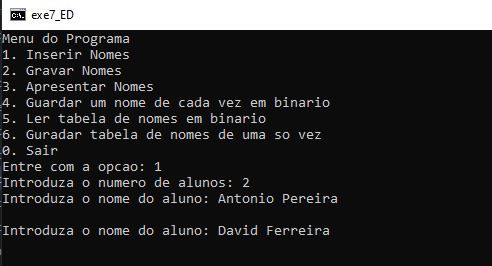


Figura 32- Interface da função inserir\_nomes

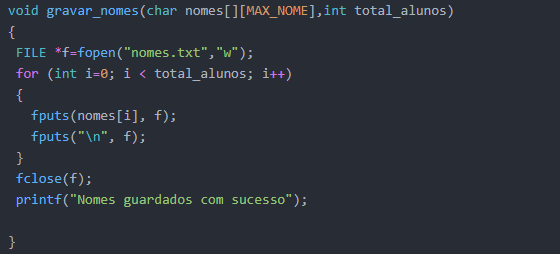
Para a alínea b), a função é a mesma do exercício anterior de guardar os alunos, só que aqui utilizamos um único fputs que é para gravar os nomes dos alunos. A função de ler os nomes dos alunos, alínea c), também é a mesma do exercício anterior, mas tal como fizemos na função de gravar os nomes só utilizamos um fgets que é também para ler os nomes. As funções são as seguintes:

Figura 33- Função para gravar os nomes dos alunos

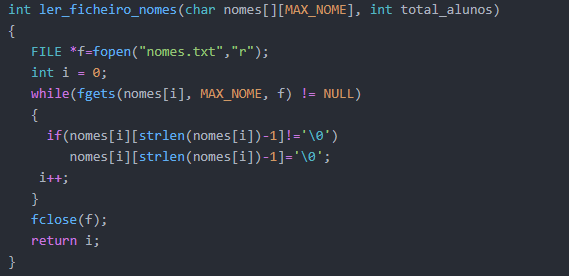
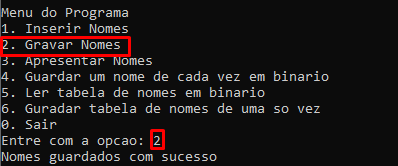


Figura 34- Função para ler os nomes dos alunos

O output da função de gravar nomes é o seguinte:

Figura 35- Interface da função gravar\_nomes



Para alínea d) foi-nos pedido para guardar os nomes dos alunos em dois ficheiros binários diferentes e de duas formas diferentes. Num tínhamos que guardar os nomes de uma só vez e no outro guardar um nome de cada vez. Para a primeira forma, guardar de uma só vez, usamos uma função e dentro desta função abríamos o ficheiro no modo de abertura de escrita(wb) e fizemos um único fwrite e dentro deste no número de elementos a escrever multiplicamos pela variável total\_alunos, que é a variável que diz os números de alunos que foram inseridos. Já para a segunda forma, guardar um nome de cada vez, também fizemos uma função e o modo de abertura também é o mesmo, só que dentro desta função fizemos um for a percorrer a lista dos alunos e dentro deste for fizemos um fwrite e no número de elementos a escrever pusemos 1 e assim vai guardar um nome de cada vez. As funções são as seguintes:

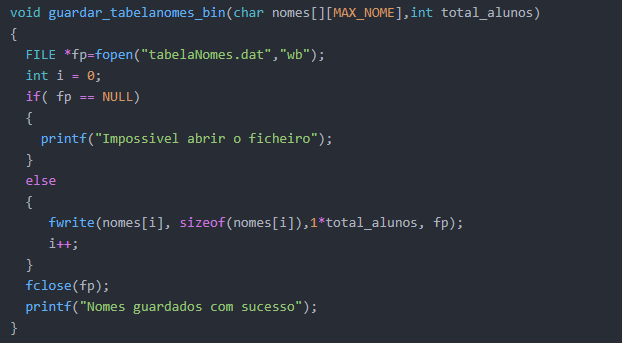


Figura 36- Função para gravar os nomes de uma só vez

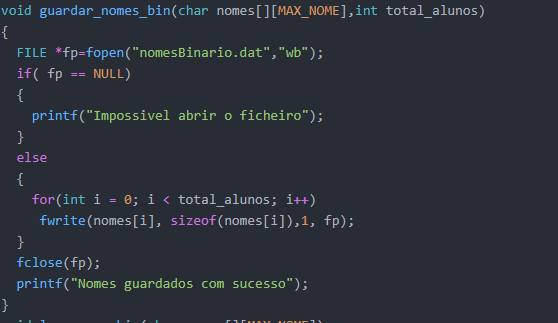
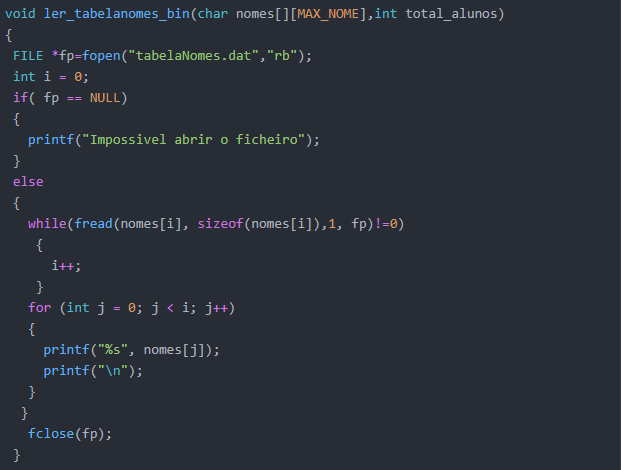


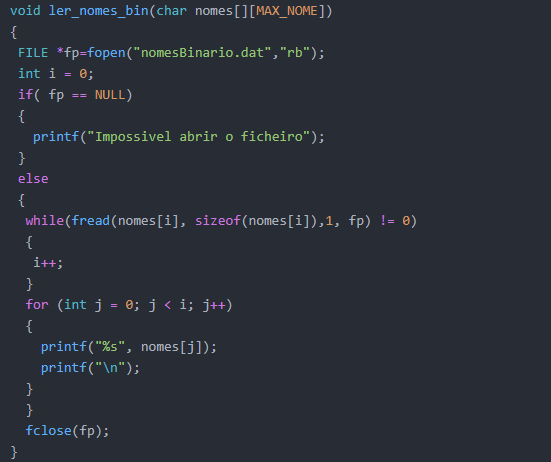
Figura 37- Função para gravar um nome de cada vez

Na última alínea é pedido para lermos os nomes dos alunos dos ficheiros da alínea anterior. Para isso, fizemos duas funções e dentro destas abríamos os ficheiros no modo de abertura de ler em binário(rb)e fizemos um while. No while é realizado um fread, que serve para ler os nomes dos ficheiros binários, para as duas funções. Para apresentar os nomes que estão guardados nos ficheiros binários, fizemos um for, a percorre a variável i, que é a que diz quantos nomes é que foram lidos, e dentro deste for é realizado um printf dos nomes. As funções desta última alínea são as seguintes:

Figura 38- Função para ler a tabela dos nomes

Figura 39- Função para ler os nomes de cada vez





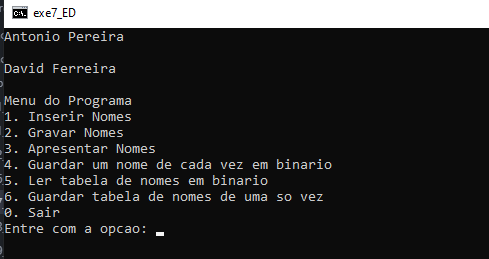
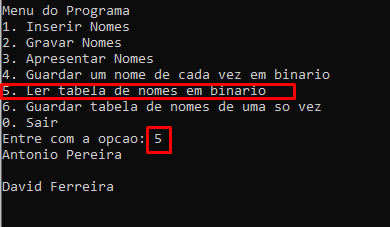
Os outputs destas funções são os seguintes:

Figura 40- Interface da função ler\_nomes\_bin

Figura 41- Interface da função ler\_tabelanomes\_bin

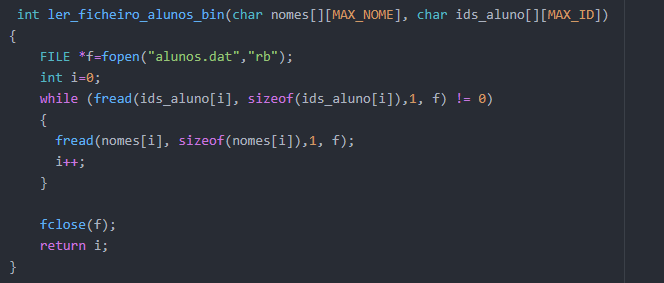


## Refaça o exercício 6: usando ficheiros binários, desenvolva um programa que faça a gestão dos alunos inscritos numa turma. Para cada aluno deve conhecer-se o nome e o número

**Nome do Ficheiro:** exe8\_ED.c

Neste exercício é pedido para utilizarmos as funções do exercício 6, só que em vez de guardar os dados dos alunos num ficheiro texto temos que guardar num ficheiro binário e ler do ficheiro binário esses dados. Para isso, para gravar os dados dos alunos fizemos uma função diferente em que dentro desta abríamos o ficheiro binário no modo de escrita em binário(wb) e fizemos um for a percorrer a lista dos alunos. Dentro deste for fizemos dois fwrite um para os nomes e outro para os números. Já para ler os dados dos alunos fizemos um while em que esta a ler o número dos alunos e dentro deste é realizado um fread para ler os nomes dos alunos. O resto é igual ao exercício 6. As funções de gravar e ler são as seguintes:

Figura 42- Função para gravar os dados dos alunos



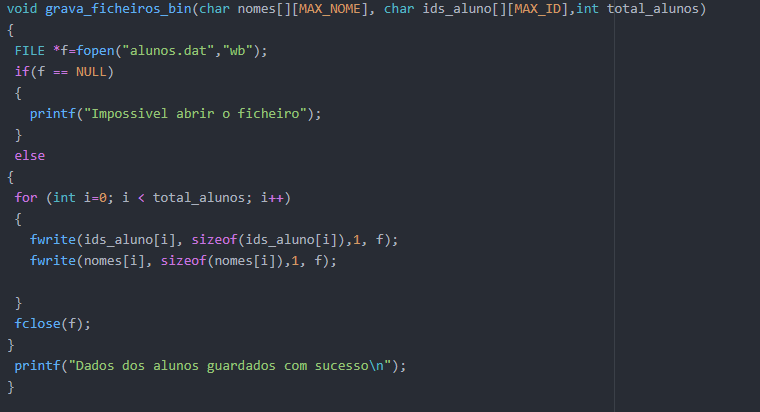
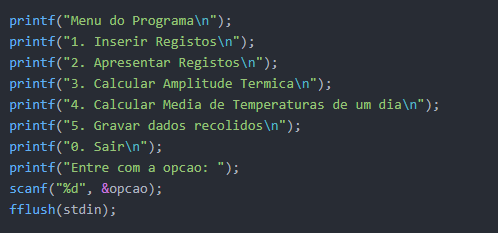


Figura 43- Função para ler os dados dos alunos

## Pretende-se guardar num ficheiro binário chamado *temperaturas.dat* os dados recolhidos por um termómetro digital automático que regista a temperatura de 15 em 15 minutos

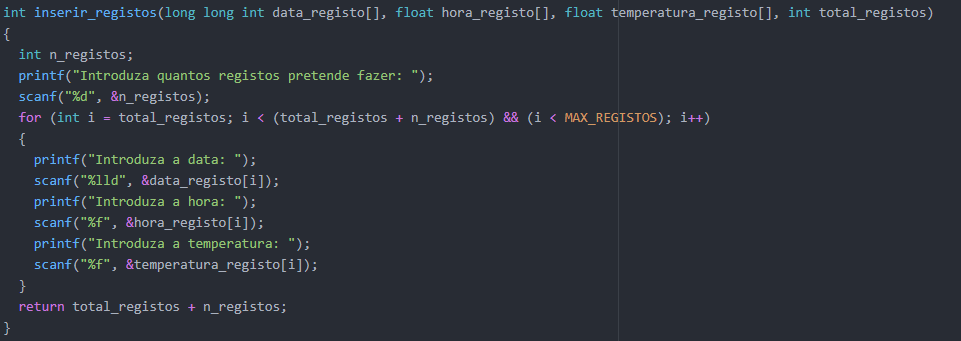
**Nome do Ficheiro:** exe9\_ED.c

Neste último exercício foi-nos pretendido pela professora fazer um programa que consiga inserir os dados introduzidos pelo utilizador como a data, a hora e a temperatura. O programa também tem que ser capaz de fazer a amplitude térmica e a media das temperaturas de um determinado dia. Por último o programa tem que gravar estes dados num ficheiro binário. Deste modo, fizemos um menu de opções a semelhança de alguns exercícios anteriores. O menu é o seguinte:



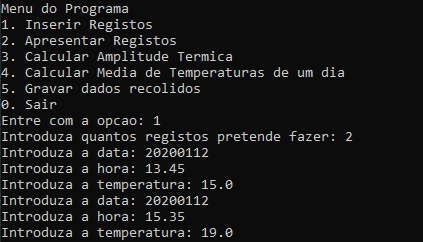
A primeira opção do nosso menu é a de inserir os dados do termómetro digital. Nesta função é realizado um for a percorrer a lista de registos mais o número de registos que o utilizador que inserir e dentro deste for pedimos ao utilizador que insira a data, a hora e a temperatura. Depois esta função retorna o total de registos mais o número de registos que o utilizador inseriu. A função é a seguinte:

Figura 44- Função para inserir registo do termómetro digital



Exemplo de output da função de inserir registos:

Figura 45- Interface da função inserir\_registos



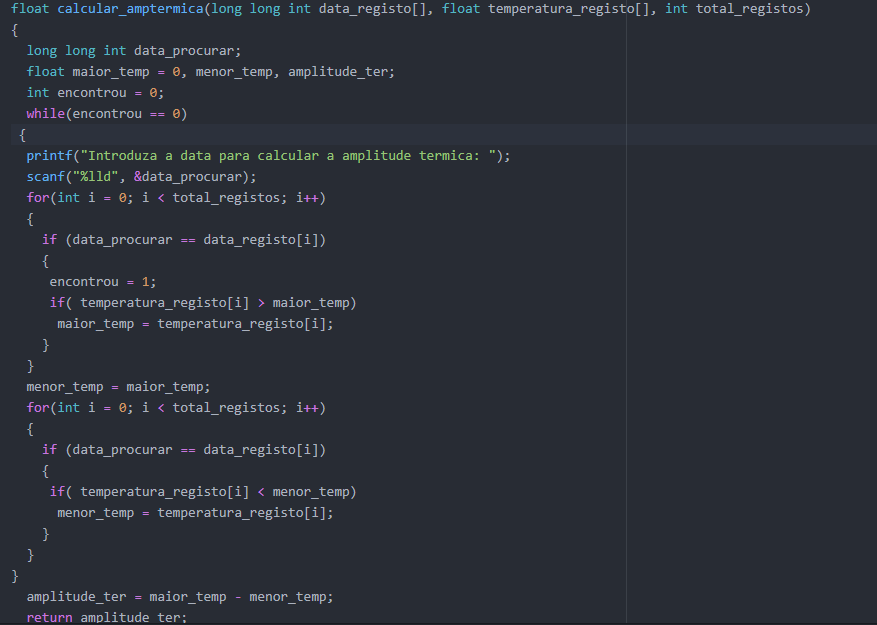
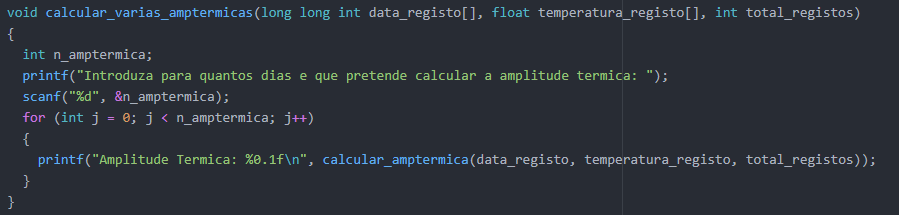
A segunda opção do nosso programa é para calcular a amplitude térmica durante um dia. Nesta função pedimos ao utilizador que insira o dia para calcular a amplitude depois o programa vai ver qual é a maior e menor temperatura desse dia, quando encontrar vai subtrair a maior temperatura com a menor temperatura e retorna a o valor da amplitude térmica. Esta função é chamada na função de calcular várias amplitudes térmicas. As funções são as seguintes:

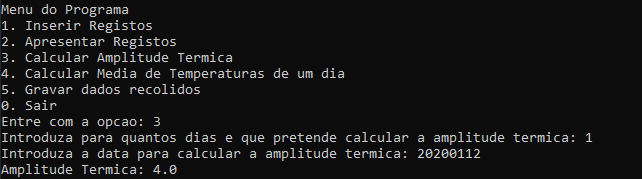
Figura 46- Função para calcular a amplitude térmica de um único dia

Figura 47- Função para calcular a amplitude térmica de vários dias



O output da função de calcular a amplitude térmica de vários dias é o seguinte:

Figura 48- Interface da função calcular\_varias\_amptermicas



A opção quatro do nosso menu é a de calcular a media de temperaturas de um determinado dia. Para isso, pedimos ao utilizador que insira a data para calcular a media, se existir essa data o programa vai somar todas as temperaturas dessa data e esta soma vai ser divida pelo número de temperaturas que tem nessa data para calcular assim a média. Esta função é chamada na função de calcular a media das temperaturas de vários dias. As duas funções são as seguintes:

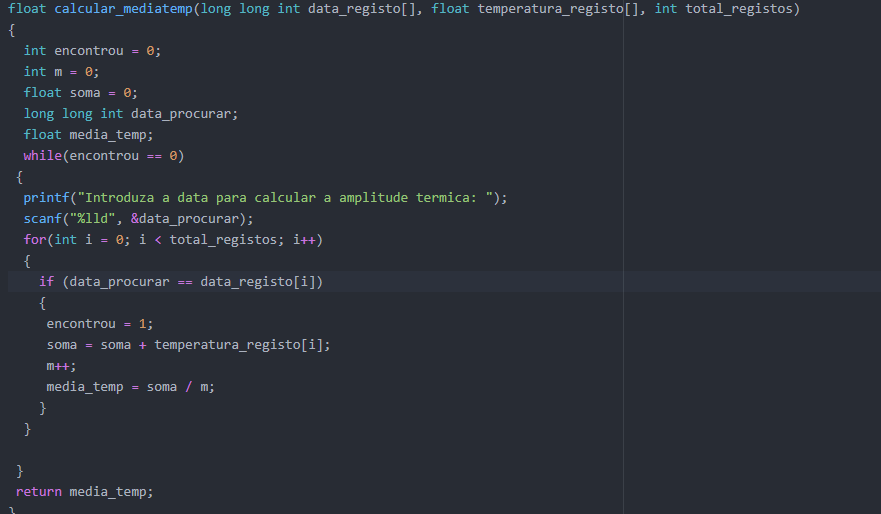


Figura 49- Função para calcular a media de temperaturas de um único dia

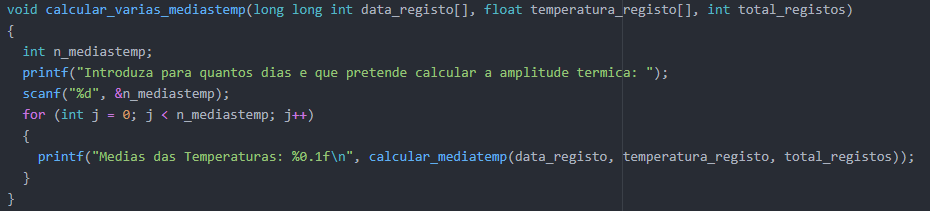


Figura 50- Função para calcular a media de temperaturas de vários dias

O output desta última função é o seguinte:



Figura 51- Interface da função calcular\_varias\_mediastemp

A última opção do nosso menu é a de gravar os dados recolhidos. Esta função é semelhante as funções de gravar em ficheiros binários dos exercícios anteriores. No entanto, nesta função só grava a data, a hora e a temperatura. A função de ler também é igual as outras funções de ler em binários dos exercícios anteriores só que aqui lê a data, a hora e a temperatura. As funções são as seguintes:

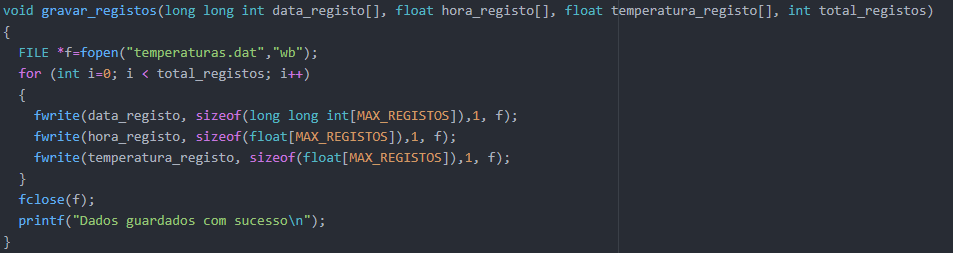


Figura 52- Função para guardar os registos recolhidos

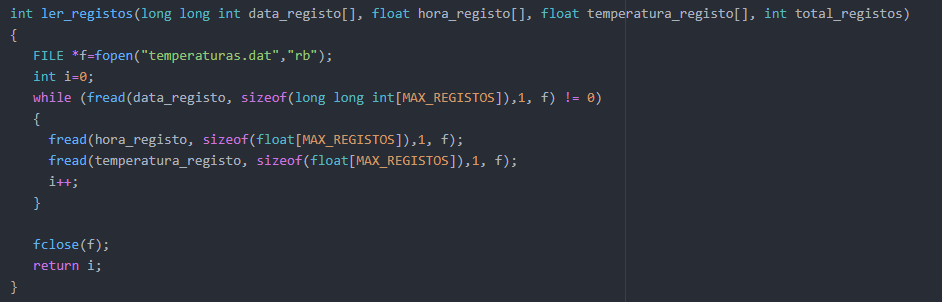


Figura 53- Função para ler os dados do ficheiro binário

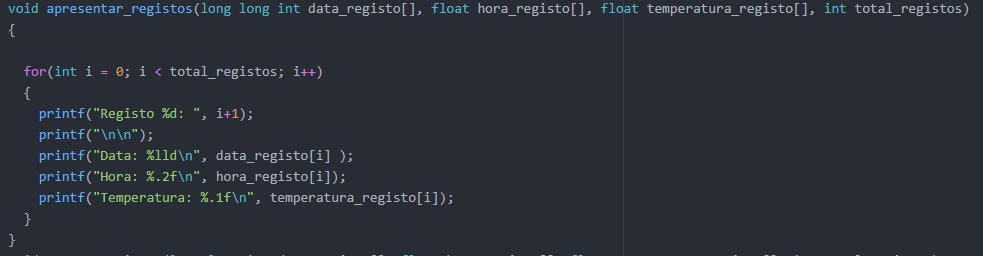
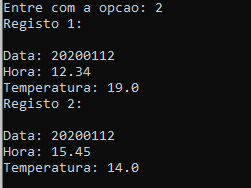
Neste exercício decidimos fazer uma função para apresentar os registos pois assim torna-se mais fácil para o utilizador saber se o ficheiro já tem alguns dados e quais são os dados que estão no ficheiro. A função é a seguinte:

Figura 54- Função para apresentar os registos do termómetro

O output desta função é o seguinte:

Figura 55- Interface da função apresentar\_registos



# Conclusão

Com este trabalho aprendemos melhor a trabalhar com ficheiros texto e binários como por exemplo guardar dados nestes ficheiros e ler os dados destes ficheiros. Assim, este trabalho foi uma mais valia para nos não só para nos relembrarmo-nos de matéria antiga como aprender coisa novas neste caso trabalhar com ficheiros binários.

# Referências

