

Licenciatura em Engenharia Informática – 19/20

Programação

4A: Manipulação Básica de Ficheiros

Francisco Pereira (xico@isec.pt)

Ficheiros

- Vantagens:

- Armazenamento permanente
- Leitura / Escrita de qualquer tipo de dados
- Tamanho variável
- Acesso sequencial / Acesso direto

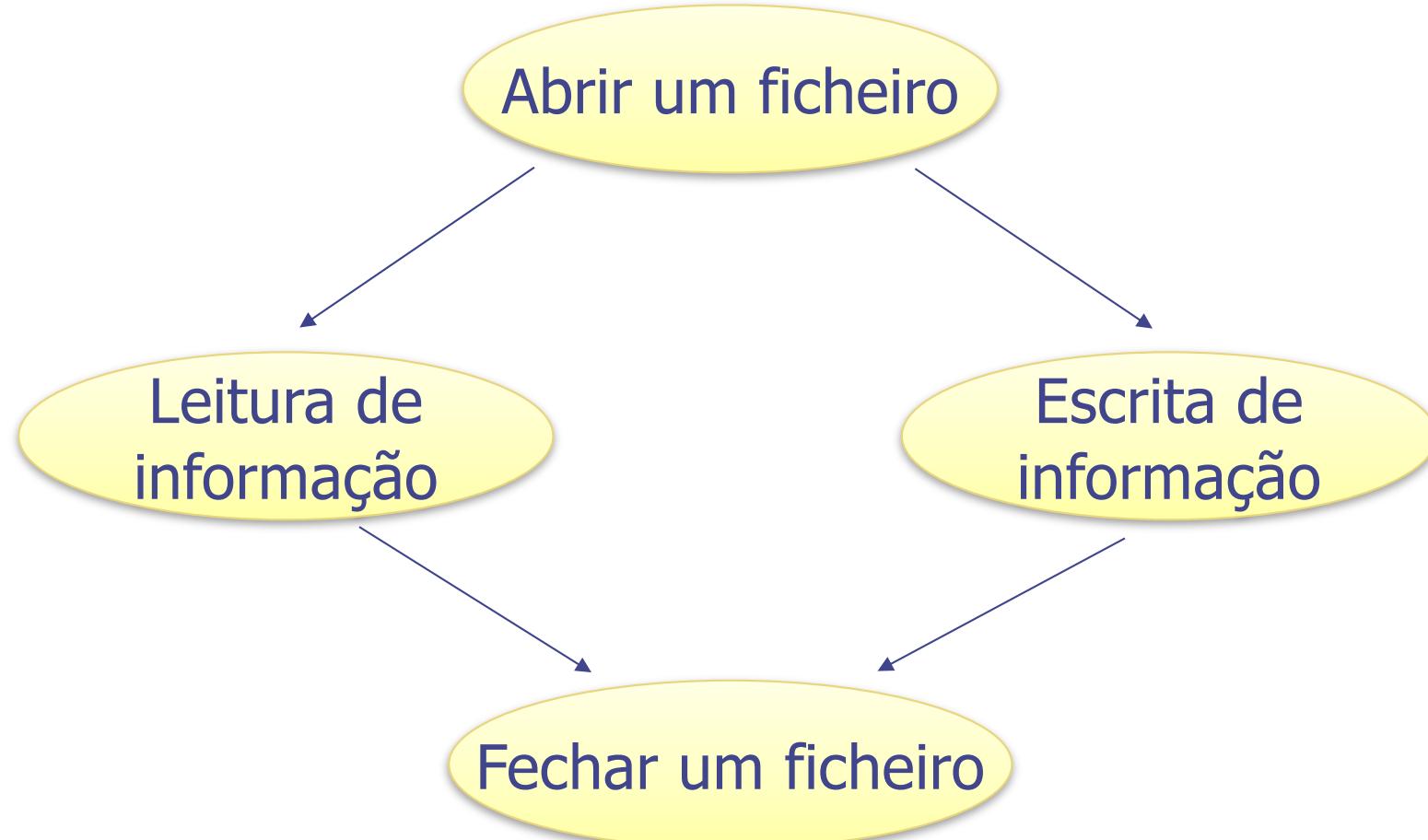
- Limitações:

- Acesso lento

Tipos de Ficheiros

- Que tipo de ficheiro utilizar?
 - Ficheiro binário:
 - Conjuntos de bytes que representam qualquer tipo de dados: caracteres, inteiros, estruturas, reais, ...
 - Não existe a noção de linha.
 - Ficheiro de texto:
 - Conjunto de caracteres divididos por linhas.
 - Editáveis.

Operações básicas



`<stdio.h>` possui funções para cada uma destas tarefas.

Exemplo

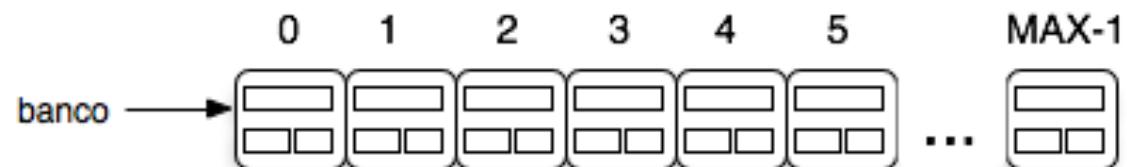
- Armazenar num ficheiro os clientes guardados no array de estruturas banco (ver capítulo anterior).

Array com
tamanho fixo

```
#define MAX 100
typedef struct dados cliente;

struct dados {
    char nome[100];
    char nconta[15];
    int montante;
};

cliente banco[MAX];
int total=0;
```



Exemplo

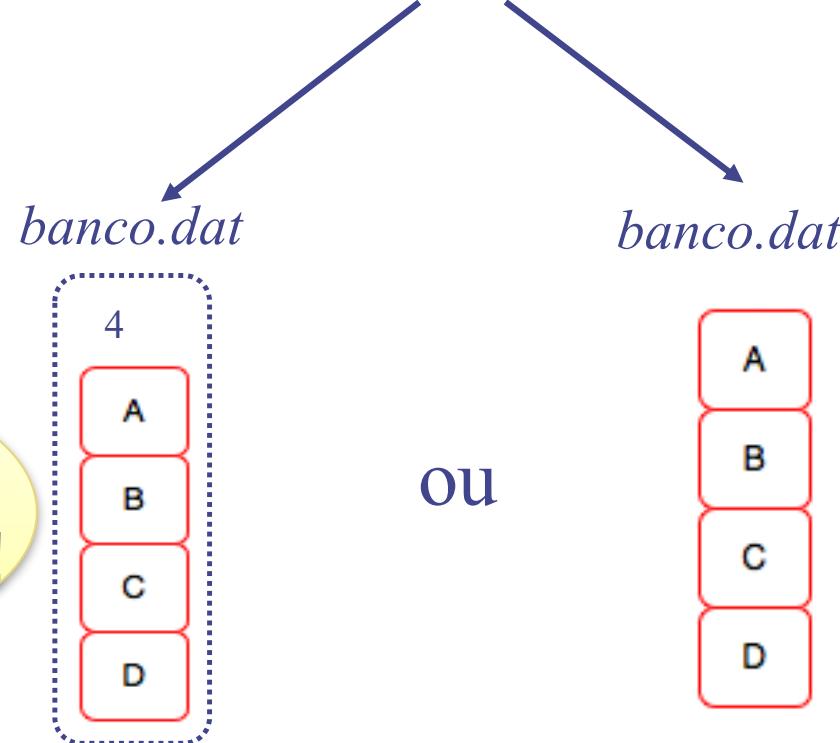
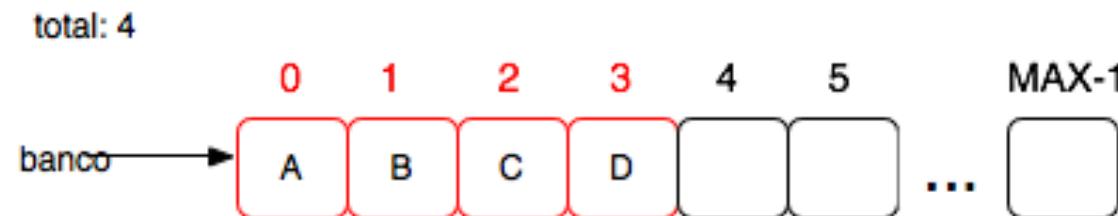
- Usar um ficheiro binário chamado `banco.dat`
 - Argumentos: ponteiro para o início do vetor e número de clientes.

```
void guardaDadosV1(cliente *b, int total);
```

- Algoritmo:
 - Abrir ficheiro `banco.dat` para escrita.
 - Guardar as estruturas no ficheiro.
 - Fechar o ficheiro.

Exemplo

- Duas alternativas para armazenar a informação:



Esta é a
melhor opção!

Abrir um ficheiro

Uma variável do tipo FILE* recebe o resultado devolvido, ficando associada ao ficheiro aberto.

Nome físico

```
FILE* fopen(char* nome, char* modo);
```

Devolve NULL em caso de erro

Modo de acesso

Acesso	Tipo de ficheiro
r: leitura	b: binário
w: escrita	t: texto (por defeito)
a: acrescentar	

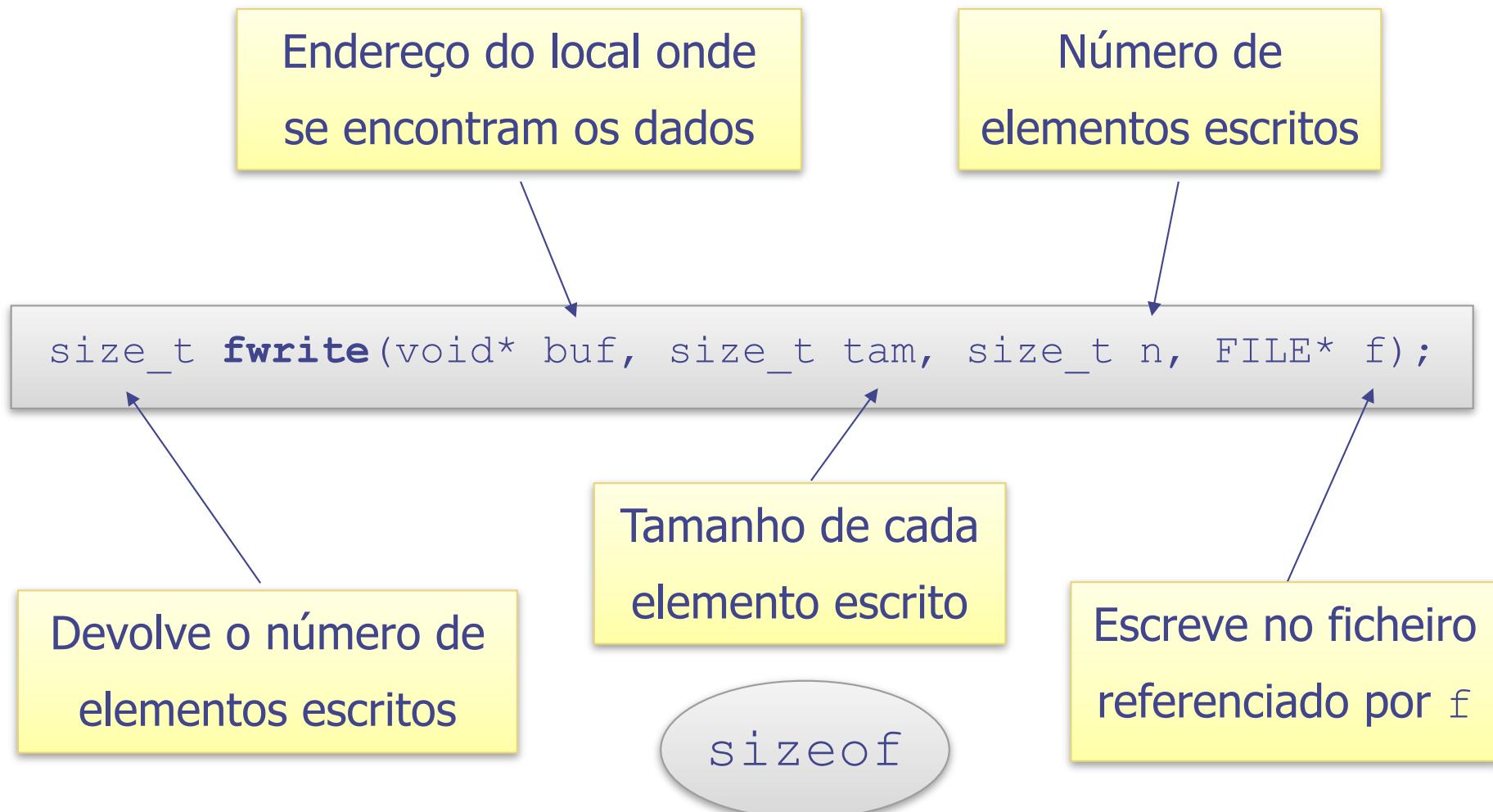
Fechar um ficheiro

Variável do tipo FILE* associada ao ficheiro

```
int fclose(FILE* f);
```

Devolve 0 se a operação correr bem

Escrever informação



Função guardaDadosV1 (): versão 1

```
void guardaDadosV1(cliente *b, int total)
{
    FILE *f;
    int i;

    f = fopen("banco.dat", "wb");
    if(f==NULL)
    {
        printf("Erro no acesso ao ficheiro\n");
        return;
    }
    fwrite(&total, sizeof(int), 1, f);
    for(i=0; i<total; i++)
        fwrite((b+i), sizeof(cliente), 1, f);
    fclose(f);
}
```

Guarda o número de estruturas no início do ficheiro

Função guardaDadosV2 (): versão 2

```
void guardaDadosV2(cliente *b, int total)
{
    FILE *f;
    int i;

    f = fopen("banco.dat", "wb");
    if(f==NULL)
    {
        printf("Erro no acesso ao ficheiro\n");
        return;
    }
    fwrite(&total, sizeof(int), 1, f);
    fwrite(b, sizeof(cliente), total, f);

    fclose(f);
}
```

Escreve todas as estruturas
com uma única instrução

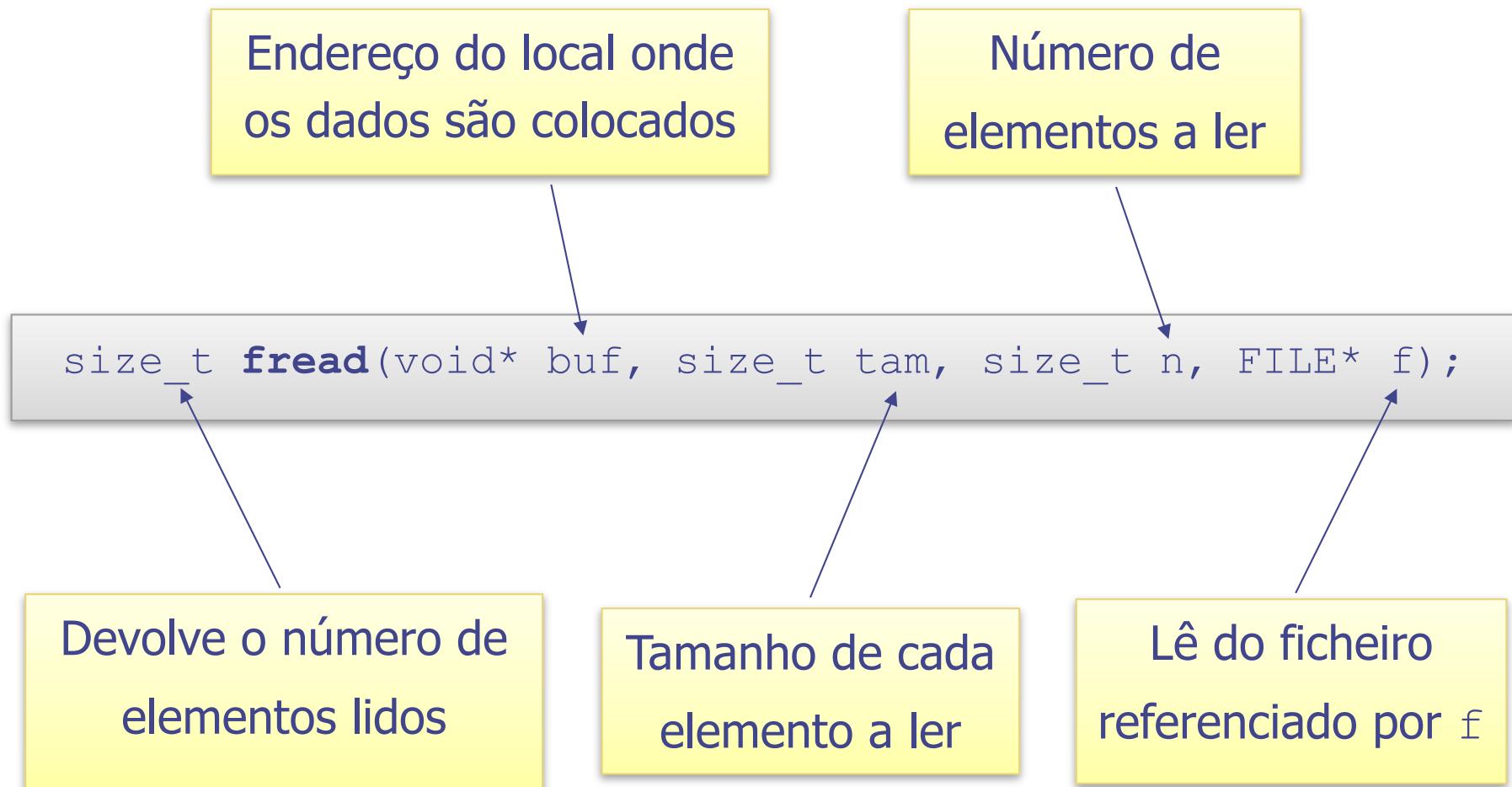
Exemplo

- Função que leia as estruturas armazenadas no ficheiro `banco.dat` e as coloque no vetor:
 - Argumentos: ponteiro para o início do vetor e endereço da variável total

```
void leDadosV1(cliente *b, int *total);
```

- Algoritmo:
 1. Abrir ficheiro `banco.dat` para leitura.
 2. Ler o número de estruturas armazenadas
 3. Atualizar a variável contador de clientes
 4. Ler as estruturas do ficheiro para o vetor
 5. Fechar o ficheiro.

Ler informação



Função leDadosV1(): versão 1

```
void leDadosV1(cliente *b, int *total){  
  
FILE *f;  
  
*total = 0;  
f = fopen("banco.dat", "rb");  
if(f==NULL)  
{  
    printf("Erro no acesso ao ficheiro\n");  
    return ;  
}  
  
fread(total, sizeof(int), 1, f);  
fread(b, sizeof(cliente), *total, f);  
  
fclose(f);  
}
```

Valor inteiro no início
do ficheiro indica quantas
estruturas existem

Exemplo

- E se o ficheiro não tiver o valor inteiro no início?

Não sabemos quantas estruturas existem no ficheiro.

Função guardaDadosV3(): versão 3

```
void guardaDadosV3(cliente *b, int total)
{
    FILE *f;
    int i;

    f = fopen("banco.dat", "wb");
    if(f==NULL)
    {
        printf("Erro no acesso ao ficheiro\n");
        return;
    }
    fwrite(b, sizeof(cliente), total, f);
    fclose(f);
}
```

Não é guardado
o valor inteiro

Função leDadosV2(): versão 2

```
void leDadosV2(cliente *b, int *total) {
    FILE *f;
    cliente aux;

    *total = 0;
    f = fopen("banco.dat", "rb");
    if(f==NULL) {
        printf("Erro no acesso ao ficheiro\n");
        return;
    }

    while(fread(&aux, sizeof(cliente), 1, f) == 1)
        b[(*total)++] = aux;

    fclose(f);
}
```

É lida uma estrutura
de cada vez

Função leDadosV3(): versão 3

- Função que verifica se foi efetuada uma leitura para além do final do ficheiro

```
int feof(FILE* f);
```

Devolve valor $\neq 0$ se o indicador
de final de ficheiro foi atingido

Função leDadosV3(): versão 3

```
void leDadosV3(cliente *b, int *total) {
    FILE *f;
    cliente aux;

    *total = 0;
    if( (f = fopen("banco.dat", "rb")) == NULL) {
        printf("Erro no acesso ao ficheiro\n");
        return;
    }
    fread(&aux, sizeof(cliente), 1, f);
    while (feof(f) == 0)
    {
        b[(*total)++] = aux;
        fread(&aux, sizeof(cliente), 1, f);
    }
    fclose(f);
}
```

Exemplo

- Armazenar num ficheiro os clientes guardados no array de estruturas banco (ver capítulo anterior).

Array Dinâmico

```
typedef struct dados cliente;

struct dados {
    char nome[100];
    char nconta[15];
    int montante;
};

cliente *banco = NULL;
int total=0;
```

Que alterações são necessárias?

Função guardaDadosDin()

```
void guardaDadosDin(cliente *b, int total){  
    FILE *f;  
    int i;  
  
    f = fopen("bancoD.dat", "wb");  
    if(f==NULL)  
    {  
        printf("Erro no acesso ao ficheiro\n");  
        return;  
    }  
    fwrite(&total, sizeof(int), 1, f);  
    fwrite(b, sizeof(cliente), total, f);  
    fclose(f);  
}
```

Igual à função anterior

Versão com inteiro no início do ficheiro

Função leDadosDin()

```
cliente* leDadosDIN(int *total) {
    FILE *f;
    cliente *b = NULL;

    *total = 0;
    f = fopen("bancoD.dat", "rb");
    if(f==NULL) {
        printf("Erro no acesso ao ficheiro\n");
        return b;
    }
    fread(total, sizeof(int), 1, f);
    b = malloc(sizeof(cliente) * (*total));
    if(b == NULL) {
        fclose(f);
        return b;
    }
    fread(b, sizeof(cliente), *total, f);
    fclose(f);
    return b;
}
```

Tem que fazer a alocação
inicial do array

Versão com inteiro no
início do ficheiro

Ficheiros de texto

- Ficheiros de texto mais comuns num programa em C:
 - stdin
 - stdout
 - stderr
- O que pode ser feito?
 - Leitura/escrita de caracteres
 - Leitura/escrita de conjuntos de caracteres (palavras/linhas)
 - Leitura/escrita formatada

Ficheiros de texto

- Escrita do carácter c no ficheiro f:

```
int fputc(int c, FILE* f);
```



Devolve carácter escrito (EOF se erro)

- Leitura do próximo carácter do ficheiro f:

```
int fgetc(FILE* f);
```



Devolve carácter lido (EOF se erro ou final do ficheiro)

Ficheiros de texto

- Escrita de uma linha num ficheiro:

```
int fputs(char* s, FILE* f);
```

- Leitura de um conjunto de caracteres de um ficheiro:

```
char* fgets(char* s, int n, FILE* f);
```

Termina quando surgir '\n' ou atingir n-1 caracteres

Funções para escrita/leitura formatada

- Escrita de dados num ficheiro:

```
int fprintf(FILE* f, ...);
```

Restantes argumentos são idênticos
aos da função `printf()`

- Leitura de dados de um ficheiro:

```
int fscanf(FILE* f, ...);
```

Restantes argumentos são idênticos
aos da função `scanf()`

Exemplo 1

```
#include <stdio.h>

int main()
{
    FILE *f;
    int i = 10;
    float x = -234.5;
    char st[20] = "Ola Mundo";

    f = fopen("ex1.txt", "w");
    if(f != NULL)
    {
        fprintf(f, "%s\n%d\t%3.3f\n", st, i, x);
        fputs("Ola outra vez", f);
        fputc('\n', f);
        fputc('A', f);
        fclose(f);
    }
    return 0;
}
```

Exemplo 2

```
#include <stdio.h>

int main()
{
    FILE *f;
    char st1[20], st2[20];
    int i;
    float x;

    f = fopen("e1.txt", "r");
    if(f != NULL)
    {
        fscanf(f, " %d ", &i);
        fgets(st1, 20, f);
        fscanf(f, " %f %c", &x, st2);
        st2[1] = fgetc(f);
        fgets(st2+2, 5, f);
        fclose(f);
    }
    return 0;
}
```

e1.txt

34
Ola Mundo
-12.467abcdefghijklm

Exemplo 3

- Processar um ficheiro de texto contendo as apostas efetuadas num concurso do euromilhões
- Informação de um apostador está dividida em 2 linhas:
 - Nome
 - 7 valores inteiros

dados.txt

Ana Mendes

3 5 7 12 20 1 3

Carlos Antunes Lima

2 5 12 21 24 4 5

Luis Silva

5 9 10 17 25 2 4

Exemplo 3

- Leitura da informação de um apostador:

```
int i, a[7];
char st[100];
FILE *f;

...
fscanf(f, " %[^\n]", st);

for(i=0; i<7; i++)
    fscanf(f, "%d", &a[i]);
...

```

Exemplo 4

- Inicializar algumas posições de uma matriz quadrada (tipo *float*):
 - As posições e os valores a preencher estão armazenados num ficheiro

```
#define N 4

float m[N][N] = {0.0};
int i,j;

// Chamar função de leitura

for(i=0; i<N; i++) {
    for(j=0; j<N; j++)
        printf("%.3f\t", m[i][j]);
    putchar('\n');
}
```

mat.txt

1	2	1283.434
1	0	-122.122
4	5	33.413
0	3	-319.605

mat.bin

1	2	1283.434
1	0	-122.122
4	5	33.413
0	3	-319.605

Exemplo 4:

Leitura do ficheiro de texto

```
void leTXT(char *nome, int n, float mat[n][n]) {
    FILE *f;
    int x[2];
    float v;

    f = fopen(nome, "r");
    if (f==NULL) {
        printf("Erro no acesso ao ficheiro\n");
        return;
    }
    while(fscanf(f, "%d %d %f", x, x+1, &v) == 3) {
        if(x[0]>=0 && x[0]<n && x[1]>=0 && x[1]<n)
            mat[x[0]][x[1]] = v;
    }
    fclose(f);
}
```

Exemplo 4: Leitura do ficheiro binário

```
void leBIN(char *nome, int n, float mat[n][n]) {
    FILE *f;
    int x[2];
    float v;

    f = fopen(nome, "rb");
    if (f==NULL) {
        printf("Erro no acesso ao ficheiro\n");
        return;
    }
    while(fread(x, sizeof(int), 2, f) == 2) {
        fread(&v, sizeof(float), 1, f);
        if (x[0]>=0 && x[0]<n && x[1]>=0 && x[1]<n)
            mat[x[0]][x[1]] = v;
    }
    fclose(f);
}
```

Exemplo 5

- Armazenar informação sobre os clientes do banco num ficheiro de texto
- Implementar a função:

```
void guarda_dados_txt(cliente *b, int total);
```

Exemplo 5

- Podemos adaptar as funções anteriores:

```
void escreve.todos(cliente tab[], int n)
{
    int i;
    for(i=0; i<n; i++)
        escreve_info(tab[i]);
}
```

```
void escreve_info(cliente a)
{
    printf("Nome: %s\nNº conta: %s\tSaldo: %d\n",
           a.nome, a.nconta, a.montante);
}
```

Exemplo 5

```
void guarda_dados_txt(cliente tab[], int n)
{
    int i;
    FILE *f;

    f = fopen("banco.txt", "w");
    if(f==NULL)
        return;
    fprintf(f, "%d\n", n);
    for(i=0; i<n; i++)
        escreve_info_txt(tab[i], f);
    fclose(f);
}
```

```
void escreve_info_txt(cliente a, FILE *f)
{
    fprintf(f, "Nome: %s\tNº conta: %s\tSaldo: %d\n",
            a.nome, a.nconta, a.montante);
}
```

Exercício

- Recuperar informação sobre os clientes do banco a partir de um ficheiro de texto
- Implementar funções que possam ser usadas com as 2 variantes de arrays de clientes:
 - Tamanho fixo
 - Array dinâmico