Ficheiros binários

Taxonomia de tipos de dados

Simples

- Numéricos
  - int (inteiros)
  - float (reais)
  - char (carateres)
- Apontadores
  - *

Compostos

- Padrão
  - arrays (1 dimensão e 2 dimensões)
  - **FILE** (ficheiros)
- Definidos pelo utilizador
  - struct (registos/estruturas)

Funções para manipulação de ficheiros

Abertura de um ficheiro

fopen

Fecho de um ficheiro

fclose

Escrita num ficheiro

fwrite

Leitura de um ficheiro

fread

Determinar a posição corrente do marcador

ftell

Deslocar o marcador para o início do ficheiro

rewind

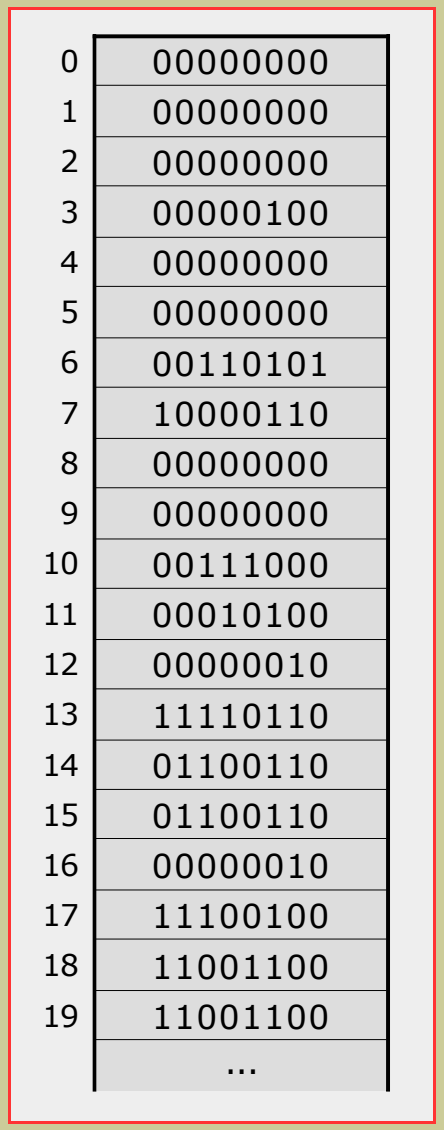
Deslocar o marcador um número de bytes desde uma certa posição

fseek

Biblioteca

stdio.h

Exemplo de um ficheiro binário



Abertura de um ficheiro

Sintaxe

FILE ***fopen** (const char ***filename**, const char ***mode**);

- Parâmetros da função
  - **filename** é a string com o nome externo (físico) do ficheiro que se pretende abrir
  - **mode** é a string com o modo de abertura do ficheiro, e que pode ser de 4 tipos:
    - **"rb"** (abertura do ficheiro para *leitura*)
    - **"wb"** (abertura do ficheiro para *escrita*)
    - **"ab"** (abertura do ficheiro para *escrita no fim* do ficheiro)
    - **"rb+"** (abertura do ficheiro para *leitura e escrita*)
- Devolução/retorno da função
  - o **endereço** de uma variável do tipo FILE que contém o endereço do ficheiro lógico, se aberto com sucesso, ou
  - **NULL**, se houver algum erro de abertura

Sintaxe

- Modos de abertura de um ficheiro
  - **"rb"** (abertura do ficheiro para leitura)
    - se o ficheiro existe, então coloca o **marcador** no início do ficheiro
  - **"wb"** (abertura do ficheiro para escrita)
    - se ficheiro existe, então esvazia ficheiro e coloca o **marcador** no início do ficheiro
    - se ficheiro não existe, então cria-o e coloca o **marcador** no início do ficheiro
  - **"ab"** (abertura do ficheiro para escrita no fim do ficheiro)
    - se ficheiro existe, então coloca o **marcador** no fim do ficheiro
    - se ficheiro não existe, então cria-o e coloca o **marcador** no início do ficheiro
  - **"rb+"** (abertura do ficheiro para *leitura e escrita*)

Fecho de um ficheiro

Sintaxe

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

- Parâmetros da função
  - **file** é a variável que contém o endereço do ficheiro lógico que se pretende fechar
- Devolução da função
  - 0 (zero), se o ficheiro foi fechado com sucesso,
  - EOF (-1), se houver algum erro no fecho do ficheiro

Escrita de dados num ficheiro

Sintaxe

```
int  fwrite (void *ptr, int size, int n, FILE *file);
```

- Parâmetros da função
  - **ptr** é o endereço inicial da zona de memória que contém os dados a escrever
  - **size** é o número de bytes que cada um dos dados ocupa
  - **n** é o número de dados a escrever
  - **file** é o endereço do ficheiro lógico onde escrever
- Devolução da função
  - o número de elementos que escreveu no ficheiro (e não o número de bytes)
- Consequências:
  - avança o marcador para o fim do ficheiro

Exemplos

```
int  fwrite (void *ptr, int size, int n, FILE *file);
```

fwrite (&X, sizeof(float), 1, f);

- escreve no ficheiro **f**, **1** número real (com **sizeof(float)** bytes) que se encontra no endereço de **X**; ou seja, escreve no ficheiro **f** o número real da variável **X**

fwrite (V, sizeof(int), 10, f);

- escreve no ficheiro **f** os **10** primeiros números inteiros (cada um com **sizeof(int)** bytes) do array **V**; ou seja, escreve no ficheiro **f** os inteiros **V[0], …., V[9]**

fwrite (&V[5], sizeof(int), 5, f);

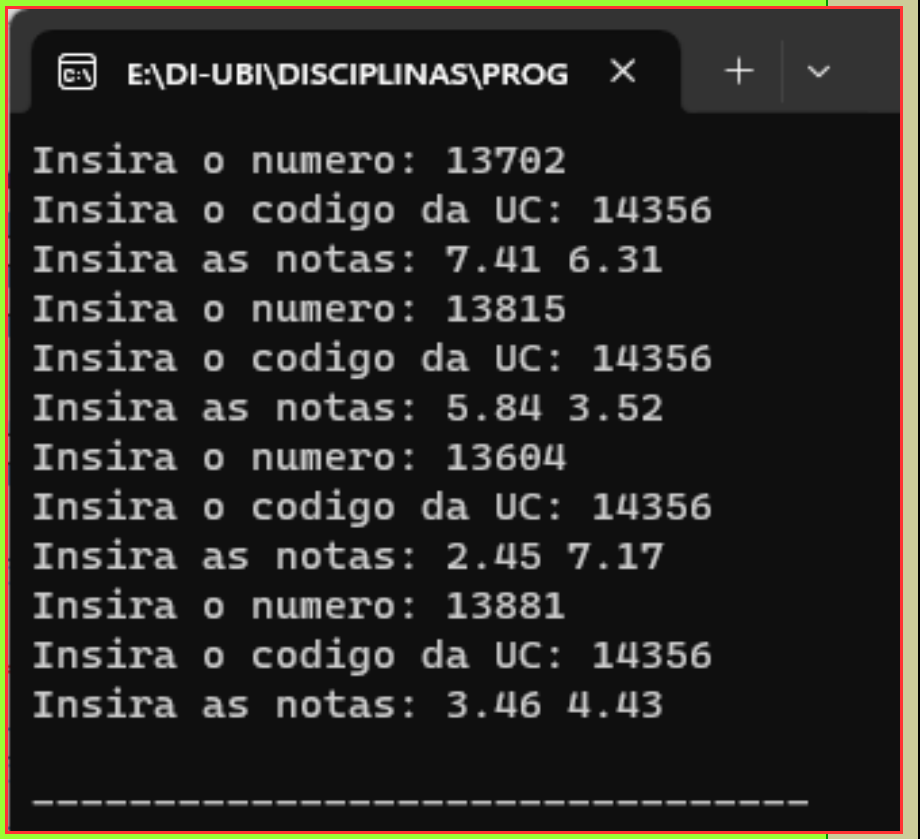
- escreve no ficheiro **f**, **5** valores inteiros (com **sizeof(int)** bytes) do array **V** a partir do 6º elemento; ou seja, escreve no ficheiro **f** os inteiros **V[5], …., V[9]**

Exemplo

```
#include <stdio.h>
#include <stdlib.h>
typedef struct {
   int  Numero;
   int  CodigoUC;
   float  Notas[2];
} ESTUDANTE;
...
```

Exemplo

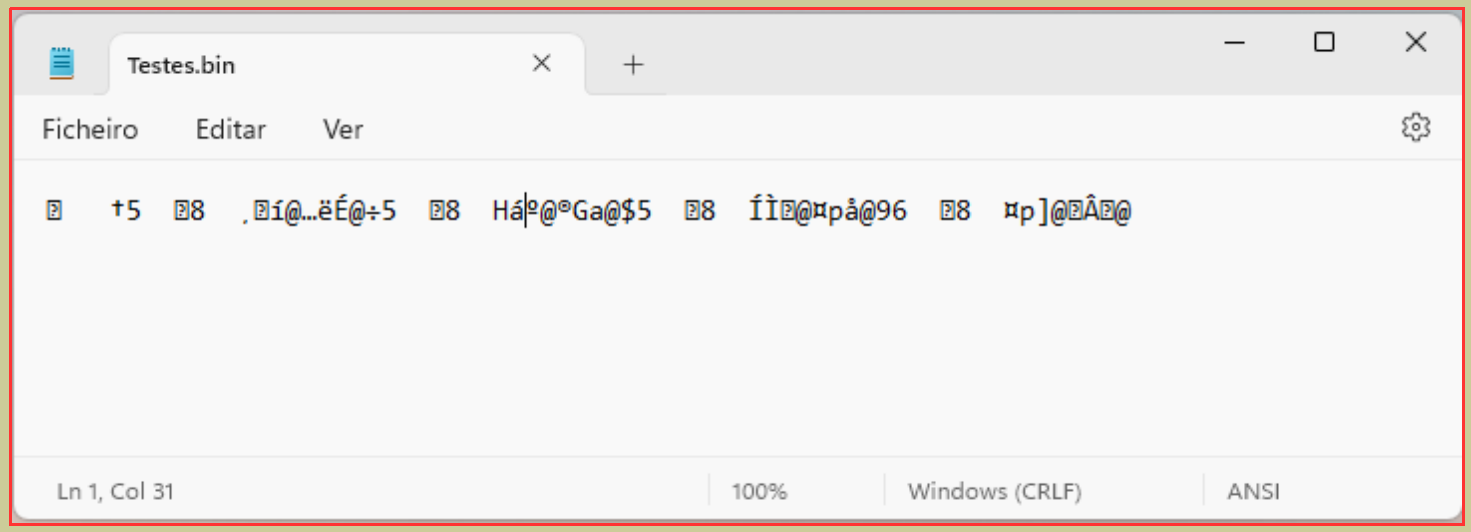
```
...
void main()
{
  int  k, TAM = 4;
  ESTUDANTE  *PROG;
  FILE *fout;
  PROG = malloc(TAM * sizeof(ESTUDANTE));
  for (k = 0; k < TAM; k++)
  {
     printf("Insira o numero: ");
     scanf("%d", &PROG[k].Numero);
     printf("Insira o codigo da UC: ");
     scanf("%d", &PROG[k].CodigoUC);
     printf("Insira as notas: ");
     scanf("%f%f", &PROG[k].Notas[0], &PROG[k].Notas[1]);
  }
  ...
```



Exemplo

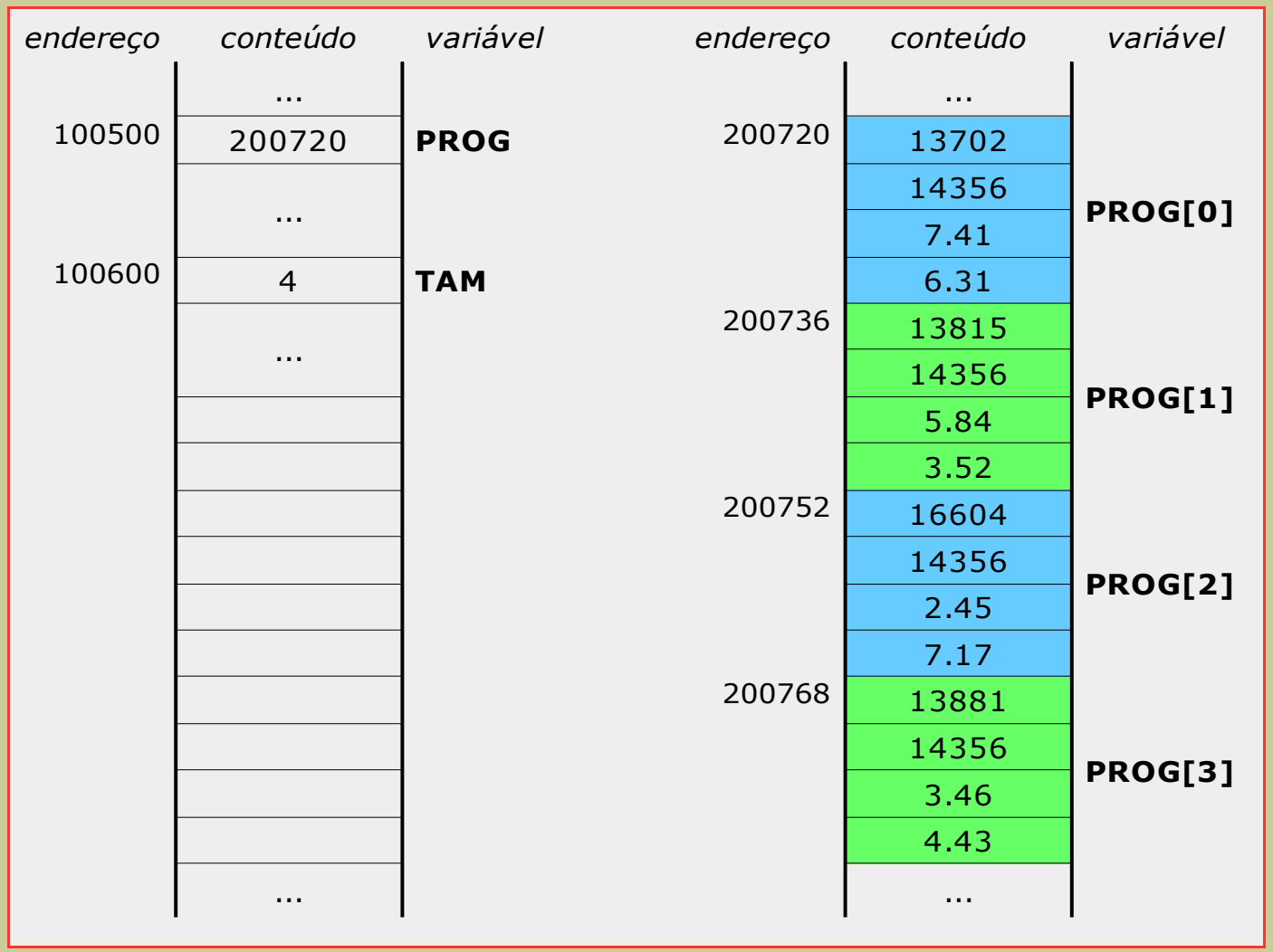
```
    ...
    fout = fopen("Testes.bin", "wb");
    fwrite(&TAM, sizeof(int), 1, fout);
    fwrite(PROG, sizeof(ESTUDANTE), TAM, fout);
    fclose(fout);
}
```

- Foi criado o ficheiro binário "Testes.bin"



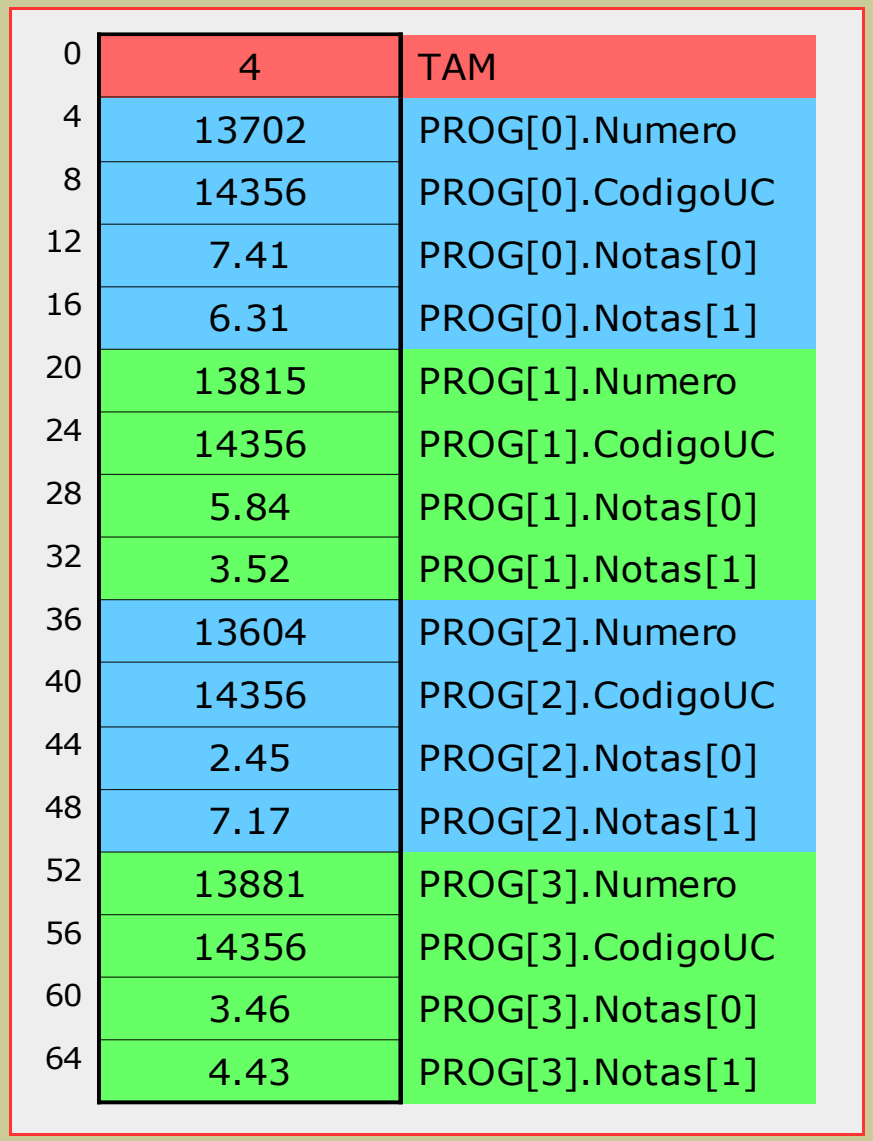
Exemplo

- Esquema dos dados guardados na memória



Exemplo

- Esquema dos dados guardados no ficheiro binário (formato decimal)



0	4	TAM
4	13702	PROG[0].Numero
8	14356	PROG[0].CodigoUC
12	7.41	PROG[0].Notas[0]
16	6.31	PROG[0].Notas[1]
20	13815	PROG[1].Numero
24	14356	PROG[1].CodigoUC
28	5.84	PROG[1].Notas[0]
32	3.52	PROG[1].Notas[1]
36	13604	PROG[2].Numero
40	14356	PROG[2].CodigoUC
44	2.45	PROG[2].Notas[0]
48	7.17	PROG[2].Notas[1]
52	13881	PROG[3].Numero
56	14356	PROG[3].CodigoUC
60	3.46	PROG[3].Notas[0]
64	4.43	PROG[3].Notas[1]

Exemplo

- Esquema dos dados guardados no ficheiro binário (formatos decimal e binário)

0	**4**	**TAM**
4	**13702**	**PROG[0].Numero**
8	**14356**	**PROG[0].CodigoUC**
12	**7.41**	**PROG[0].Notas[0]**
16	**6.31**	**PROG[0].Notas[1]**
20	13815	PROG[1].Numero
24	14356	PROG[1].CodigoUC
28	5.84	PROG[1].Notas[0]
32	3.52	PROG[1].Notas[1]
36	13604	PROG[2].Numero
40	14356	PROG[2].CodigoUC
44	2.45	PROG[2].Notas[0]
48	7.17	PROG[2].Notas[1]
52	13881	PROG[3].Numero
56	14356	PROG[3].CodigoUC
60	3.46	PROG[3].Notas[0]
64	4.43	PROG[3].Notas[1]

0	00000000	TAM
1	00000000	
2	00000000	
3	00000100	
4	00000000	PROG[0].Numero
5	00000000	
6	00110101	
7	10000110	
8	00000000	PROG[0].CodigoUC
9	00000000	
10	00111000	
11	00010100	
12	00000010	PROG[0].Notas[0]
13	11110110	
14	01100110	
15	01100110	
16	00000010	PROG[0].Notas[1]
17	11100100	
18	11001100	
19	11001100	
	...	...

Leitura de dados de um ficheiro

Sintaxe

```
int  fread (void *ptr, int size, int n, FILE *file);
```

- Parâmetros da função
  - **ptr** é o endereço inicial da zona de memória que recebe os dados a ler
  - **size** é o número de bytes que cada um dos dados ocupa
  - **n** é o número de dados a ler
  - **file** é o endereço do ficheiro lógico a ler
- Devolução da função
  - o número de elementos que leu do ficheiro (e não o número de bytes)
- Consequência:
  - avança o **marcador** no ficheiro o número de bytes lidos

Exemplos

```
int  fread (void *ptr, int size, int n, FILE *file);
```

```
fread (&X, sizeof(float), 1, f);
```

- lê do ficheiro **f**, **1** bloco de **sizeof(float)** bytes e converte-o num real que atribuí à variável do tipo real **X**

```
fread (V, sizeof(int), 10, f);
```

- lê do ficheiro **f**, **10** blocos de **sizeof(int)** bytes e converte-os em 10 números inteiros que atribuí a V[0], V[1], …, V[9], respetivamente

```
fread (&V[2], sizeof(int), 5, f);
```

- lê do ficheiro **f**, **5** blocos de **sizeof(int)** bytes e converte-os em 5 números inteiros que atribuí aos elementos do array **V**, V[2], V[3], …, V[6], respetivamente

Exemplo 1 – leitura dos registos de uma só vez

- Mostrar todos os dados contidos no ficheiro binário "Testes.bin" (ao lado), lendo todos os registos com uma só instrução de leitura

0	4	TAM
4	13702	PROG[0].Numero
8	14356	PROG[0].CodigoUC
12	7.41	PROG[0].Notas[0]
16	6.31	PROG[0].Notas[1]
20	13815	PROG[1].Numero
24	14356	PROG[1].CodigoUC
28	5.84	PROG[1].Notas[0]
32	3.52	PROG[1].Notas[1]
36	13604	PROG[2].Numero
40	14356	PROG[2].CodigoUC
44	2.45	PROG[2].Notas[0]
48	7.17	PROG[2].Notas[1]
52	13881	PROG[3].Numero
56	14356	PROG[3].CodigoUC
60	3.46	PROG[3].Notas[0]
64	4.43	PROG[3].Notas[1]

Exemplo 1 – leitura dos registos de uma só vez

```
void main()
{
  int  k, TAM;
  ESTUDANTE  *PROG;
  FILE  *fin;
  fin = fopen("Testes.bin", "rb");
  fread(&TAM, sizeof(int), 1, fin);
  printf("TAMANHO: %d\n", TAM);
  PROG = malloc(TAM * sizeof(ESTUDANTE));
  fread(PROG, sizeof(ESTUDANTE), TAM, fin);
  fclose(fin);
  for (k = 0; k < TAM; k++) {
     printf("Numero: %d\n", PROG[k].Numero);
     printf("Codigo da UC: %d\n", PROG[k].CodigoUC);
     printf("Notas: %5.2f e %5.2f\n", PROG[k].Notas[0], PROG[k].Notas[1]);
  }
}
```

```
E:\DI-UBI\DISCIPLINAS\PROG
TAMANHO: 4
Numero: 13702
Codigo da UC: 14356
Notas:  7.41 e  6.31
Numero: 13815
Codigo da UC: 14356
Notas:  5.84 e  3.52
Numero: 13604
Codigo da UC: 14356
Notas:  2.45 e  7.17
Numero: 13881
Codigo da UC: 14356
Notas:  3.46 e  4.43

--------------------------
```

Exemplo 2 – leitura dos registos um a um

- Mostrar dos dados contidos no ficheiro binário "Testes.bin" (ao lado), lendo todos os registos nele contido, cada um deles separadamente e com apenas uma instrução de leitura

0	4	TAM
4	13702	PROG[0].Numero
8	14356	PROG[0].CodigoUC
12	7.41	PROG[0].Notas[0]
16	6.31	PROG[0].Notas[1]
20	13815	PROG[1].Numero
24	14356	PROG[1].CodigoUC
28	5.84	PROG[1].Notas[0]
32	3.52	PROG[1].Notas[1]
36	13604	PROG[2].Numero
40	14356	PROG[2].CodigoUC
44	2.45	PROG[2].Notas[0]
48	7.17	PROG[2].Notas[1]
52	13881	PROG[3].Numero
56	14356	PROG[3].CodigoUC
60	3.46	PROG[3].Notas[0]
64	4.43	PROG[3].Notas[1]

Exemplo 2 – leitura dos registos um a um

```
void main() {
  int  k, TAM;
  ESTUDANTE *PROG;
  FILE *fin;
  fin = fopen("Testes.bin", "rb");
  fread(&TAM, sizeof(int), 1, fin);
  printf("TAMANHO: %d\n", TAM);
  PROG = malloc(TAM * sizeof(ESTUDANTE));
  for (k = 0; k < TAM; k++)
     fread(&PROG[k], sizeof(ESTUDANTE), 1, fin);
  fclose(fin);
  for (k = 0; k < TAM; k++) {
     printf("Numero: %d\n", PROG[k].Numero);
     printf("Codigo da UC: %d\n", PROG[k].CodigoUC);
     printf("Notas: %5.2f ## %5.2f\n", PROG[k].Notas[0], PROG[k].Notas[1]);
  }
}
```

```
E:\DI-UBI\DISCIPLINAS\PROG
TAMANHO: 4
Numero: 13702
Codigo da UC: 14356
Notas:  7.41 ##  6.31
Numero: 13815
Codigo da UC: 14356
Notas:  5.84 ##  3.52
Numero: 13604
Codigo da UC: 14356
Notas:  2.45 ##  7.17
Numero: 13881
Codigo da UC: 14356
Notas:  3.46 ##  4.43

--------------------------
```

Determinar a posição corrente/atual do marcador

Sintaxe

```
long  ftell (FILE *file);
```

- Parâmetros da função
  - **file** é o endereço do ficheiro lógico a tratar
- Devolução da função
  - a posição corrente/atual do marcador (em bytes)

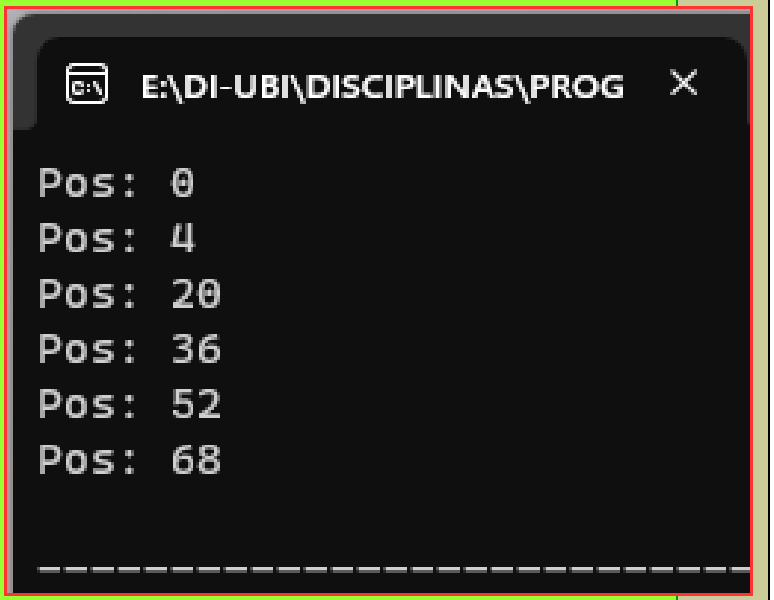
Exemplo

- Determinar e mostrar as posições do início de cada um dos seguintes dados:
  - TAM (INÍCIO DO FICHEIRO)
  - PROG[0]
  - PROG[1]
  - PROG[2]
  - PROG[3]
  - FIM DE FICHEIRO

0	4	TAM
4	13702	PROG[0].Numero
8	14356	PROG[0].CodigoUC
12	7.41	PROG[0].Notas[0]
16	6.31	PROG[0].Notas[1]
20	13815	PROG[1].Numero
24	14356	PROG[1].CodigoUC
28	5.84	PROG[1].Notas[0]
32	3.52	PROG[1].Notas[1]
36	13604	PROG[2].Numero
40	14356	PROG[2].CodigoUC
44	2.45	PROG[2].Notas[0]
48	7.17	PROG[2].Notas[1]
52	13881	PROG[3].Numero
56	14356	PROG[3].CodigoUC
60	3.46	PROG[3].Notas[0]
64	4.43	PROG[3].Notas[1]

Exemplo

```
void main()
{
  int  k, TAM;
  long  pos;
  ESTUDANTE *PROG;
  FILE *fin;
  fin = fopen("Testes.bin", "rb");
  pos = ftell(fin); printf("Pos: %ld\n", pos);
  fread(&TAM, sizeof(int), 1, fin);
  pos = ftell(fin); printf("Pos: %ld\n", pos);
  PROG = malloc(TAM * sizeof(ESTUDANTE));
  for (k = 0; k < TAM; k++) {
     fread(&PROG[k], sizeof(ESTUDANTE), 1, fin);
     pos = ftell(fin); printf("Pos: %ld\n", pos);
  }
  fclose(fin);
}
```



Deslocar o marcador para o início do ficheiro

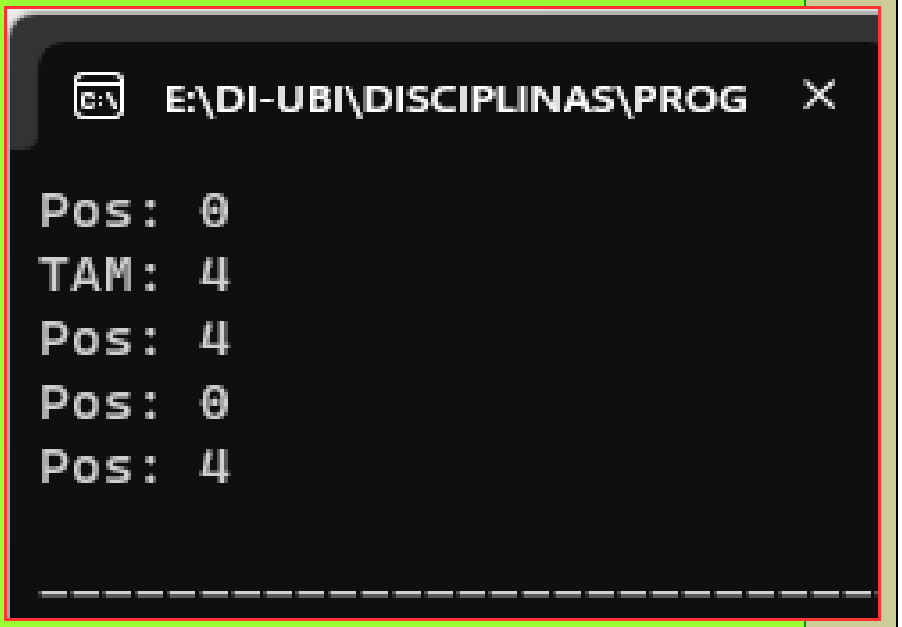
Sintaxe

```
void  rewind (FILE *file);
```

- Parâmetros da função
  - **file** é o endereço do ficheiro lógico a tratar

Exemplo

```
void main()
{
  int  TAM;
  long  pos;
  FILE  *fin;
  fin = fopen("Testes.bin", "rb");
  pos = ftell(fin); printf("Pos: %ld\n", pos);
  fread(&TAM, sizeof(int), 1, fin);
  printf("TAM: %d\n", TAM);
  pos = ftell(fin); printf("Pos: %ld\n", pos);
  rewind(fin);
  pos = ftell(fin); printf("Pos: %ld\n", pos);
  fread(&TAM, sizeof(int), 1, fin);
  pos = ftell(fin); printf("Pos: %ld\n", pos);
  fclose(fin);
}
```



Deslocar o marcador um número de bytes

Sintaxe

```
int  fseek (FILE *file, long salto, int origem);
```

- Parâmetros da função
  - **file** é o endereço do ficheiro lógico a tratar
  - **salto** designa o valor em bytes que pretende deslocar relativamente a *origem*
    - valor positivo ⇒ salto para a frente
    - valor negativo ⇒ salto para trás
  - **origem** tem três valores possíveis, representados pelas seguintes constantes:
    - **SEEK_SET**: início do ficheiro
    - **SEEK_CUR**: posição corrente/atual do ficheiro
    - **SEEK_END**: fim do ficheiro
- Devolução da função
  - 0 (zero), se a operação teve sucesso
  - um valor não nulo, se a operação não teve sucesso

Exemplo 1

- Mostrar os valores e as posições antes e depois da leitura dos seguintes dados:
  - TAM
  - PROG[1]

0	4	TAM
4	13702	PROG[0].Numero
8	14356	PROG[0].CodigoUC
12	7.41	PROG[0].Notas[0]
16	6.31	PROG[0].Notas[1]
20	13815	PROG[1].Numero
24	14356	PROG[1].CodigoUC
28	5.84	PROG[1].Notas[0]
32	3.52	PROG[1].Notas[1]
36	13604	PROG[2].Numero
40	14356	PROG[2].CodigoUC
44	2.45	PROG[2].Notas[0]
48	7.17	PROG[2].Notas[1]
52	13881	PROG[3].Numero
56	14356	PROG[3].CodigoUC
60	3.46	PROG[3].Notas[0]
64	4.43	PROG[3].Notas[1]

Exemplo 1

```
void main() {
  int  TAM;  long  pos;
  ESTUDANTE  Elem;  FILE *fin;
  fin = fopen("Testes.bin", "rb");
  pos = ftell(fin); printf("Pos TAM 1: %ld\n", pos);
  fread(&TAM, sizeof(int), 1, fin);
  printf("TAM: %d\n", TAM);
  pos = ftell(fin); printf("Pos TAM 2: %ld\n", pos);
  fseek(fin, sizeof(ESTUDANTE), SEEK_CUR);
  pos = ftell(fin); printf("Pos PROG[1] 1: %ld\n", pos);
  fread(&Elem, sizeof(ESTUDANTE), 1, fin);
  printf("Numero: %d\n", Elem.Numero);
  printf("Codigo da UC: %d\n", Elem.CodigoUC);
  printf("Notas: %5.2f ## %5.2f\n", Elem.Notas[0], Elem.Notas[1]);
  pos = ftell(fin); printf("Pos PROG[1] 2: %ld\n", pos);
  fclose(fin);
}
```

```
E:\DI-UBI\DISCIPLINAS\PROG
Pos TAM 1: 0
TAM: 4
Pos TAM 2: 4
Pos PROG[1] 1: 20
Numero: 13815
Codigo da UC: 14356
Notas:  5.84 ##  3.52
Pos PROG[1] 2: 36

--------------------------
```

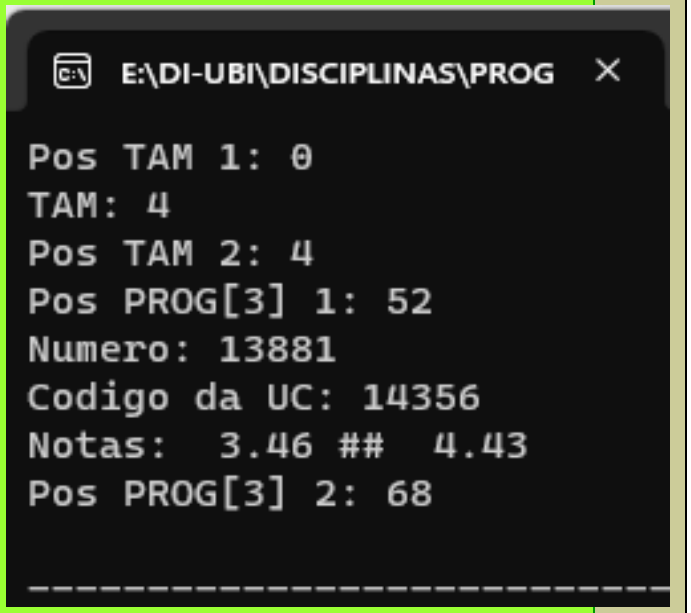
Exemplo 2

- Mostrar os valores e as posições antes e depois da leitura dos seguintes dados:
  - TAM
  - PROG[3]

0	4	TAM
4	13702	PROG[0].Numero
8	14356	PROG[0].CodigoUC
12	7.41	PROG[0].Notas[0]
16	6.31	PROG[0].Notas[1]
20	13815	PROG[1].Numero
24	14356	PROG[1].CodigoUC
28	5.84	PROG[1].Notas[0]
32	3.52	PROG[1].Notas[1]
36	13604	PROG[2].Numero
40	14356	PROG[2].CodigoUC
44	2.45	PROG[2].Notas[0]
48	7.17	PROG[2].Notas[1]
52	13881	PROG[3].Numero
56	14356	PROG[3].CodigoUC
60	3.46	PROG[3].Notas[0]
64	4.43	PROG[3].Notas[1]

Exemplo 2

```
void main() {
  int  TAM; long  pos;
  ESTUDANTE  Elem; FILE *fin;
  fin = fopen("Testes.bin", "rb");
  pos = ftell(fin); printf("Pos TAM 1: %ld\n", pos);
  fread(&TAM, sizeof(int), 1, fin);
  printf("TAM: %d\n", TAM);
  pos = ftell(fin); printf("Pos TAM 2: %ld\n", pos);
  fseek(fin, -16, SEEK_END);
  pos = ftell(fin); printf("Pos PROG[3] 1: %ld\n", pos);
  fread(&Elem, sizeof(ESTUDANTE), 1, fin);
  printf("Numero: %d\n", Elem.Numero);
  printf("Codigo da UC: %d\n", Elem.CodigoUC);
  printf("Notas: %5.2f ## %5.2f\n", Elem.Notas[0], Elem.Notas[1]);
  pos = ftell(fin); printf("Pos PROG[3] 2: %ld\n", pos);
  fclose(fin);
}
```



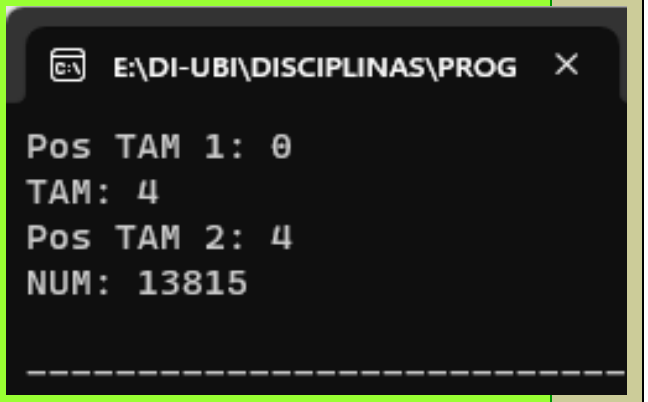
Exemplo 3

- Mostrar os seguintes dados:
  - TAM (posição inicial, valor e posição final)
  - Avançar 1 registo completo
  - NUM (PROG[1].Numero) - posição atual
  - Recuar para o início de PROG[0]
  - Posição inicial de PROG[0]
  - Posição inicial de PROG[1]
  - Posição inicial de PROG[2]
  - Posição inicial de PROG[3]
  - FIM DE FICHEIRO
  - Escrever PROG[0]
  - Escrever PROG[1]
  - Escrever PROG[2]
  - Escrever PROG[3]

0	4	TAM
4	13702	PROG[0].Numero
8	14356	PROG[0].CodigoUC
12	7.41	PROG[0].Notas[0]
16	6.31	PROG[0].Notas[1]
20	13815	PROG[1].Numero
24	14356	PROG[1].CodigoUC
28	5.84	PROG[1].Notas[0]
32	3.52	PROG[1].Notas[1]
36	13604	PROG[2].Numero
40	14356	PROG[2].CodigoUC
44	2.45	PROG[2].Notas[0]
48	7.17	PROG[2].Notas[1]
52	13881	PROG[3].Numero
56	14356	PROG[3].CodigoUC
60	3.46	PROG[3].Notas[0]
64	4.43	PROG[3].Notas[1]

Exemplo 3

```
void main()
{
  int  NUM, TAM, k;
  long  pos;
  ESTUDANTE  *PROG;
  FILE  *fin;
  fin = fopen("Testes.bin", "rb");
  pos = ftell(fin);
  printf("Pos TAM 1: %ld\n", pos);
  fread(&TAM, sizeof(int), 1, fin);
  printf("TAM: %d\n", TAM);
  pos = ftell(fin);
  printf("Pos TAM 2: %ld\n", pos);
  fseek(fin, sizeof(ESTUDANTE), SEEK_CUR);
  fread(&NUM, sizeof(int), 1, fin);
  printf("NUM = %d\n", NUM);
  ...
```



Exemplo 3

```
...
PROG = malloc(TAM * sizeof(ESTUDANTE));
fseek(fin, 4, SEEK_SET);
for (k = 0; k < TAM; k++) {
    pos = ftell(fin);
    printf("Pos PROG[%1d]: %ld\n", k, pos);
    fread(&PROG[k], sizeof(ESTUDANTE), 1, fin);
}
pos = ftell(fin);
printf("Pos FIM: %ld\n", pos);
for (k = 0; k < TAM; k++) {
    printf("Numero: %d\n", PROG[k].Numero);
    printf("Codigo da UC: %d\n", PROG[k].CodigoUC);
    printf("Notas: %5.2f ## %5.2f\n", PROG[k].Notas[0], PROG[k].Notas[1]);
}
fclose(fin);
}
```