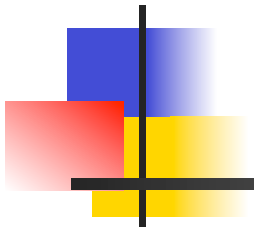
Manipulação de Arquivos Binários em C

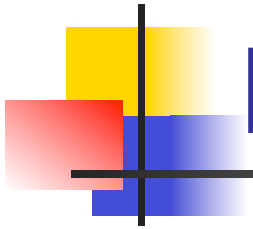
Cristina D. A. Ciferri

Anderson Chaves Carniel

João Pedro de Carvalho Castro

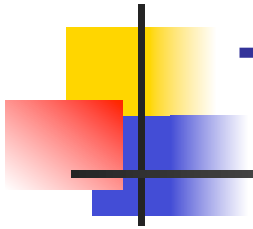


Introdução



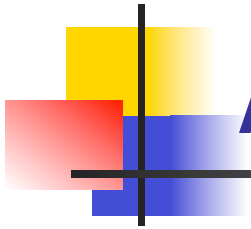
Fonte do Material

- Prof. Dr. Jander Moreira
  - (Universidade Federal de São Carlos)
- Link para acesso
  - http://wiki.icmc.usp.br/images/b/b4/SCC0603022016arquivos.pdf



Tipos de Arquivos

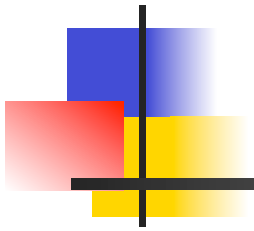
- Arquivo texto
  - Contém caracteres legíveis (texto, números)
  - Exemplo: Código em C
- Arquivo binário
  - Dados armazenados por meio de sua representação binária (não legível)
  - Exemplo: Arquivo DAT



Arquivos Binários

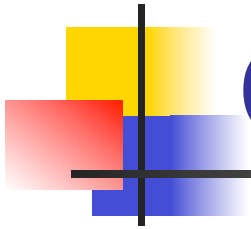
- Exemplo
  - Em um arquivo binário, o valor 127 pode usar a representação equivalente do tipo **int** da linguagem C
  - Nesse caso, o valor usaria 4 bytes

```
01111111 00000000 00000000 00000000
```



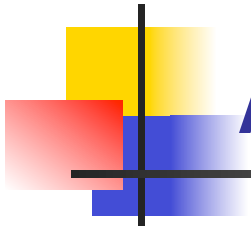
Operações em Arquivos

* ênfase em arquivos binários *



Operações em Arquivos

Manipulação	Descrição
Abertura de arquivo	Solicitação do direito de acesso ao arquivo físico, seja para usar um arquivo existente quanto para criar um arquivo novo.
Fechamento de arquivo	Solicitação do encerramento do direito de acesso ao arquivo.
Leitura de dados	Solicitação de cópia de um conjunto de dados (bytes) do dispositivo para variáveis na memória principal.
Escrita de dados	Solicitação da cópia de um conjunto de bytes armazenados em variáveis na memória principal para o dispositivo.



Abertura de Arquivo

Manipulação	Descrição
Abertura de arquivo	Solicitação do direito de acesso ao arquivo físico, seja para usar um arquivo existente quanto para criar um arquivo novo.
Fechamento de arquivo	Solicitação do encerramento do direito de acesso ao arquivo.
Leitura de dados	Solicitação de cópia de um conjunto de dados (bytes) do dispositivo para variáveis na memória principal.
Escrita de dados	Solicitação da cópia de um conjunto de bytes armazenados em variáveis na memória principal para o dispositivo.

```
FILE *fopen(const char *path, const char *mode);
```

Modo	Descrição
w	Um arquivo vazio é criado[a], permitindo-se apenas comandos de escrita. Leitura de dados não são permitidas.
w+	Um arquivo vazio é criado, permitindo-se comandos de escrita e leitura de dados.
r	Um arquivo é aberto, se ele existir e houver permissões corretas, permitindo apenas operações de leitura. O ajuste é feito para que o acesso seja feito no início dos dados.
r+	Um arquivo é aberto, se ele existir e houver permissões corretas, permitindo operações de leitura e escrita. O ajuste é feito para que o acesso seja feito no início dos dados.
a	Um arquivo é aberto, se ele existir e houver permissões corretas, permitindo apenas operações de leitura. O ajuste é feito para que o acesso seja feito no fim dos dados.
a+	Um arquivo é aberto, se ele existir e houver permissões corretas, permitindo operações de leitura e escrita. O ajuste é feito para que o acesso seja feito no fim dos dados.

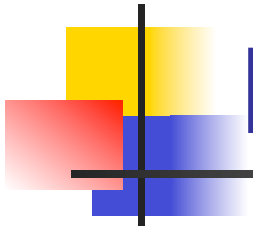
Para arquivos binários, é necessário adicionar a letra b ao final da *string* que indica o modo.

```
FILE *fopen(const char *path, const char *mode);
```

```
FILE *descritor;
char nomeArquivo[100];

printf("Nome do arquivo: ");
gets(nomeArquivo);

/* abertura do arquivo */
descritor = fopen(nomeArquivo, "r+");
```

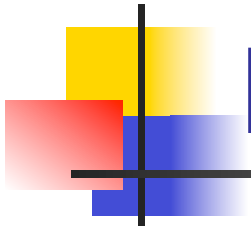


Fechamento de Arquivo

Manipulação	Descrição
Abertura de arquivo	Solicitação do direito de acesso ao arquivo físico, seja para usar um arquivo existente quanto para criar um arquivo novo.
Fechamento de arquivo	Solicitação do encerramento do direito de acesso ao arquivo.
Leitura de dados	Solicitação de cópia de um conjunto de dados (bytes) do dispositivo para variáveis na memória principal.
Escrita de dados	Solicitação da cópia de um conjunto de bytes armazenados em variáveis na memória principal para o dispositivo.

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

```
/* fechamento do arquivo */
fclose(descritor);
```



Escrita de Dados

Manipulação	Descrição
Abertura de arquivo	Solicitação do direito de acesso ao arquivo físico, seja para usar um arquivo existente quanto para criar um arquivo novo.
Fechamento de arquivo	Solicitação do encerramento do direito de acesso ao arquivo.
Leitura de dados	Solicitação de cópia de um conjunto de dados (bytes) do dispositivo para variáveis na memória principal.
Escrita de dados	Solicitação da cópia de um conjunto de bytes armazenados em variáveis na memória principal para o dispositivo.

```
typedef struct{
            int ra;
            float nota;
} tEntrada;
```

```
tEntrada aluno;
```

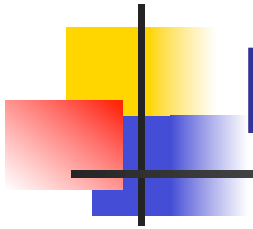
```
size_t fwrite(const void *ptr, size_t size, size_t nmemb,
                      FILE *stream);
```

```
printf("RA (0 para terminar): ");
scanf("%d", &aluno.ra);

while(aluno.ra != 0){
      printf("Nota: ");
      scanf("%f", &aluno.nota);

      /* copia dos dados para o arquivo */
      fwrite(&aluno, sizeof(aluno),1, arquivo);

      /* proximo aluno */
      printf("RA (0 para terminar): ");
      scanf("%d", &aluno.ra);
}
```



Leitura de Dados

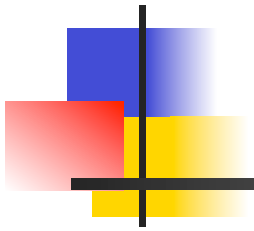
Manipulação	Descrição
Abertura de arquivo	Solicitação do direito de acesso ao arquivo físico, seja para usar um arquivo existente quanto para criar um arquivo novo.
Fechamento de arquivo	Solicitação do encerramento do direito de acesso ao arquivo.
Leitura de dados	Solicitação de cópia de um conjunto de dados (bytes) do dispositivo para variáveis na memória principal.
Escrita de dados	Solicitação da cópia de um conjunto de bytes armazenados em variáveis na memória principal para o dispositivo.

```
typedef struct{
            int ra;
            float nota;
} tEntrada;
```

```
tEntrada aluno;
```

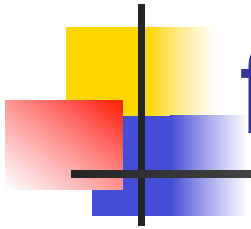
```
size_t fread(void *ptr, size_t size, size_t nmemb,
                        FILE *stream);
```

```
while(fread(&aluno, sizeof(aluno),
                            1, arquivo) != 0){
      printf("Registro #%d\n", cont++);
      printf("  > RA: %d\n", aluno.ra);
      printf("  > Nota: %.1f\n\n", aluno.nota);
}
```



Operações Adicionais

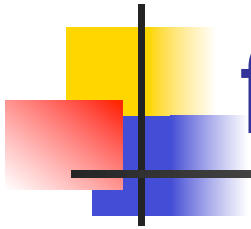
* ênfase em arquivos binários *



fseek: Posicionamento

```
int fseek(FILE *stream, long offset, int whence);
```

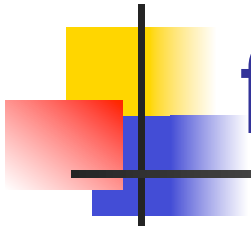
- OFFSET:
  - Deslocamento que deve ser feito (em bytes)
- WHENCE:
  - Ponto de referência usado para o deslocamento



fseek: Posicionamento

```
int fseek(FILE *stream, long offset, int whence);
```

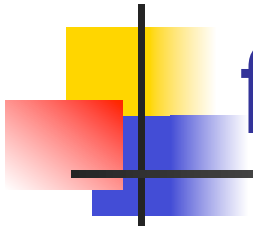
Referência (whence)	Descrição
SEEK_CUR	O deslocamento em bytes é calculado em relação à posição corrente, isto é, em relação à posição atual. Valores positivos para **offset** indicam avanço na posição corrente, enquanto valores negativos indica retrocesso.
SEEK_SET	O deslocamento é feito em relação ao início do arquivo, ou seja, em relação à posição zero. Neste caso, **offset** deve ser maior ou igual a zero.
SEEK_END	O deslocamento é feito em relação ao fim do arquivo. Os valores de **offset** devem ser negativos, pois valores positivos indica posições além do fim do arquivo.[a]



fseek: Posicionamento

```
int fseek(FILE *stream, long offset, int whence);
```

Comando	Resultado
fseek(arq, 0, SEEK_SET)	Posiciona no início do arquivo, ou seja, no byte zero.
fseek(arq, 0, SEEK_END)	Posiciona no fim do arquivo. Se um comando **fwrite()** for executado, um novo registro será acrescentado no fim do arquivo.
fseek(arq, 0, SEEK_CUR)	Não faz nada.
fseek(arq, -sizeof(reg), SEEK_END)	Posiciona no início do último registro do arquivo.
fseek(arq, n*sizeof(reg), SEEK_SET)	Posiciona no início do registro **n**, sendo que o primeiro registro é o registro 0.



ftell: Posição Corrente

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