Manipulação de Arquivos em C

Prof. Dr. Márcio Castro
marcio.castro@ufsc.br

UNIVERSIDADE FEDERAL
DE SANTA CATARINA

1 Introdução

Introdução

- O sistema de E/S da **biblioteca padrão do C** utiliza o conceito de **streams**
- Uma **stream** é uma abstração que representa um **canal de comunicação**
- **Streams** podem ser associadas a:
  - **Arquivos** armazenados em disco
  - **Dispositivos**, tais como o próprio **terminal** e dispositivos **periféricos**
  - **Processos**, através do uso de **pipes**
- O **tipo de dados** usado para representar stream é `FILE *`

Introdução

- Um programa em C possui **três streams** criadas **automaticamente**
  - Elas representam os **canais de comunicação padrão de um processo**

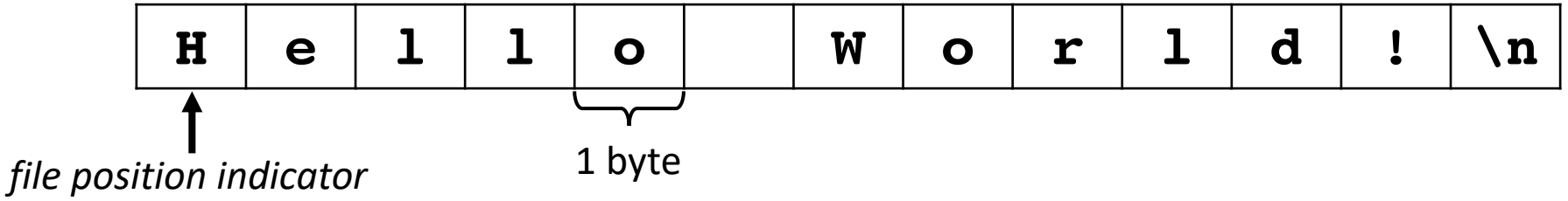
Stream	Descrição	Normalmente conectada ao
`FILE *stdin`	*Standard input stream*	Teclado
`FILE *stout`	*Standard output stream*	Terminal
`FILE *stderr`	*Standard error stream*	Terminal

- As **streams padrão** podem ser **associadas** (ou **redirecionadas**) a **arquivos**, **dispositivos** ou **processos** específicos

Introdução

- **Streams** precisam ser **associadas** a um **arquivo**, **dispositivo** ou **processo** para que possam ser usadas
  - A associação é feita através de uma operação de **abertura** (**open**)
  - A desassociação é feita através de uma operação de **fechamento** (**close**)
- Quando uma stream é **aberta**, um **indicador de posição de arquivo** apontará para o **início do arquivo**
  - Operações subsequentes serão realizadas a partir da posição indicada

Exemplo de um arquivo de texto contendo “Hello World!\n”



2 Operações básicas em arquivos

Operações básicas em arquivos: **fopen**

```
FILE *fopen(const char *pathname, const char *mode)
```

Abre uma stream, a associa a um arquivo e retorna um ponteiro para a stream

- **pathname**: caminho absoluto ou relativo para um arquivo
- **mode**: controla como o arquivo será aberto

Exemplo

```
FILE *arquivo;
if((arquivo = fopen("teste", "r")) == NULL) {
    printf("Erro ao abrir arquivo.\n");
}
```

Controle de abertura de arquivo

Modo	Descrição
`r`	Abre um arquivo em **modo somente leitura** (o arquivo precisa existir).
`w`	Abre um arquivo em **modo somente escrita**. Se o arquivo já existir ele será sobrescrito. Caso contrário, um novo arquivo será criado.
`a`	Abre um arquivo em **modo append** (operações de escrita ocorrem somente no fim do arquivo). Se o arquivo não existir, ele será criado.
`r+, w+, a+`	Abre um arquivo em **modo leitura e escrita**. Aplicam-se, nesse caso, as regras dos modos `r`, `w` ou `a`.

- Um caractere `b` pode ser utilizado em conjunto com o modo para indicar a criação de uma **stream binária** (**ao invés de uma stream de texto**)
  - Porém, sistemas POSIX **não fazem distinção entre esses dois tipos de stream**

Operações básicas em arquivos: `fclose`

```
int fclose(FILE *stream)
```

Fecha uma stream que estava associada a um arquivo, retornando 0 em caso de sucesso

- **`stream`**: ponteiro para a stream que foi aberta via **`fopen`**

Exemplo

```
fclose(arquivo);
```

Operações básicas em arquivos: **fseek**

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

Modifica o indicador de posição de arquivo associado a uma stream

- **stream**: ponteiro para a stream que foi aberta via **fopen**
- **whence**: indica a partir de onde o offset será aplicado
  - **SEEK_SET**: a partir do início do arquivo
  - **SEEK_CUR**: a partir da posição atual do indicador de posição de arquivo
  - **SEEK_END**: a partir do fim do arquivo
- **offset**: deslocamento a ser feito a partir de **whence**

Exemplo

```
fseek(arquivo, 10*sizeof(char), SEEK_SET);
fseek(arquivo, -20*sizeof(char), SEEK_END);
```

Operações básicas em arquivos: `feof`

```
int feof(FILE *stream)
```

Verifica se o indicador de posição de arquivo está apontando para o final do arquivo

- `stream`: ponteiro para a stream que foi aberta via `fopen`

Exemplo

```
if (feof(arquivo))
  printf("Final do arquivo!\n")
```

Operações básicas em arquivos: **rename**

```
int rename(const char *oldpath, const char *newpath)
```

Renomeia um arquivo

- **oldpath**: caminho absoluto ou relativo do arquivo atual
- **newpath**: caminho absoluto ou relativo do novo arquivo

Exemplo

```
rename("file1.txt", "file2.txt");
rename("file2.txt", "/home/usr1/file-new.txt");
```

Operações básicas em arquivos: **remove**

```
int remove(const char *pathname)
```

Remove um arquivo

- **pathname**: caminho absoluto ou relativo do arquivo a ser removido

Exemplo

```
remove("file3.txt");
remove("/home/usr1/file-new.txt");
```

Operações básicas em arquivos: `fflush`

```
int fflush(FILE *stream)
```

Todas as operações em arquivos são realizadas **temporariamente em memória** (em **buffers**) para melhorar o desempenho geral do sistema

A função `fflush` esvazia o conteúdo do buffer

- **`stream`**: ponteiro para a stream que foi aberta via **`fopen`**

Exemplo

```
fflush(stdout);
fflush(arquivo);
```

3 Escrita e leitura de caracteres e strings

Escrita e leitura de caracteres

```
int fgetc(FILE *stream)
int fputc(int c, FILE *stream)
```

Lê (`fgetc`) ou escreve (`fputc`) um caractere na posição indicada pelo indicador de posição do arquivo na stream e o avança

- `c`: caractere a ser escrito
- `stream`: ponteiro para a stream que foi aberta via `fopen`

Exemplo

```
do {
  caractere = fgetc(arquivo);
  if (caractere != EOF)
    fputc(caractere, stdout);
}
while(!feof(arquivo));
```

Escrita e leitura de strings

```
char *fgets(char *s, int size, FILE *stream)
int fputs(const char *s, FILE *stream)
```

Lê (`fgets`) ou escreve (`fputs`) uma string na posição indicada pelo indicador de posição do arquivo na stream o avança.

- **`s`**: string a ser escrita ou que receberá a leitura
- **`size`**: número de caracteres a serem lidos, incluindo-se o `'\0'`
- **`stream`**: ponteiro para a stream que foi aberta via **`fopen`**

A string é **truncada** quando o caractere `'\n'` é encontrado!

Exemplo

```
do {
   fgets(string, 256, stdin);
   fputs(string, arquivo);
} while(string[0] != '\n');
```

4 Escrita e leitura de blocos de dados

Escrita e leitura de blocos de dados

- É possível manipular dados em arquivos utilizando-se operações de escrita e leitura em **blocos de tamanho arbitrário**
  - O tamanho de um bloco é definido em **bytes**

- Operações em blocos permitem uma **maior flexibilidade** na manipulação de dados em arquivos
  - São **necessárias** para manipular dados de **arquivos binários**
  - Mas também **podem ser utilizadas em arquivos de texto**

Escrita e leitura de blocos de dados

- Dados podem ser **estruturados de diferentes formas** em arquivos **binários**
- A **estrutura interna** de armazenamento dos arquivos binários deve ser **conhecida pelo programa que os manipula**

Exemplos de arquivos binários com diferentes estruturas de armazenamento

Arquivo 1

byte	
0	5512
4	998
8	8
12	72

int (4 bytes)

Arquivo 2

byte	
0	5512.1300
8	998.6761
16	8.0000
24	72.9999

double (8 bytes)

Arquivo 3

byte			
0	5512	5512.1300	“aa\0”
15	998	998.6761	“bb\0”
30	8	8.0000	“cc\0”
45	72	72.9999	“dd\0”

struct item (15 bytes)

```
struct item {
   int x;
   double y;
   char z[3];
}
```

Escrita e leitura de blocos de dados

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

Lê (`fread`) ou escreve (`fwrite`) um bloco de bytes na posição indicada pelo indicador de posição do arquivo na stream e o avança

- **`ptr`**: endereço de memória a partir do qual serão lidos (**`fwrite`**) ou escritos (**`fread`**) os dados
- **`size`**: tamanho de um item a ser lido/escrito (em bytes)
- **`nmemb`**: quantidade de itens a serem lidos/escritos
- **`stream`**: ponteiro para a stream que foi aberta via **`fopen`**

Exemplo

```
int dado;
fread(&dado, sizeof(int), 1, arquivo);
dado *= 2;
fseek(arquivo, -1*sizeof(int), SEEK_CUR);
fwrite(&dado, sizeof(int), 1, arquivo);
```

Obrigado pela atenção!

Dúvidas? Entre em contato:

- marcio.castro@ufsc.br
- www.marciocastro.com

UNIVERSIDADE FEDERAL
DE SANTA CATARINA

LaPeSD

Distributed Systems Research Lab

www.lapesd.inf.ufsc.br