

Arquivos

Introdução à Ciência da Computação I

Prof. Denis F. Wolf

Entrada e Saída de Dados

- Em C não existem comandos de Entrada e Saída, sendo estas tarefas executadas por funções especialmente criadas para esta finalidade e armazenadas em bibliotecas específicas.
- Dados podem ser manipulados em dois diferentes tipos de fluxos:
 - fluxo de texto
 - fluxo binário.

Fluxos (streams)

- **Fluxo de texto:** composto por uma seqüência de caracteres, que pode ou não ser dividida em linhas terminadas por um caractere de final de linha.
 - Um detalhe que deve ser considerado é que na última linha não é obrigatório o caractere de fim de linha.
- **Fluxo binário:** composto por uma seqüência de bytes lidos, sem tradução, diretamente do dispositivo externo.
 - Não ocorre nenhuma tradução e existe uma correspondência um para um entre os dados do dispositivo e os que estão no fluxo.

Arquivos

- Os arquivos são entendidos como *streams* cujos dados estão guardados em um dispositivo de armazenamento secundário
- Operações comuns em arquivos são:
 - abertura e fechamento de arquivos;
 - apagar um arquivo;
 - leitura e escrita de um caractere;
 - indicação de que o fim do arquivo foi atingido;
 - posicionar o arquivo em um ponto determinado
- Importante:
 - Ao final das operações necessárias o programa deve fechar o arquivo.
 - Quando um programa é encerrado todos os arquivos associados são fechados automaticamente e os conteúdos dos buffers são descarregados para o dispositivo externo.

Funções da biblioteca *stdio.h*

Função	Descrição
<code>fopen()</code>	Abre um arquivo
<code>fclose()</code>	Fecha um arquivo
<code>fseek()</code>	Posiciona o ponteiro do arquivo
<code>feof()</code>	Retorna VERDADE se chegou ao fim do arquivo
<code>fflush()</code>	Descarrega o buffer associado com o arquivo
<code>fprintf()</code>	Equivalente a <code>printf()</code> , usando stream
<code>fscanf()</code>	Equivalente a <code>scanf()</code> , usando stream
<code>fgets()</code>	Equivalente a <code>gets()</code> , usando stream

Abrindo e fechando um arquivo

`FILE *arq;`

Declara um ponteiro para arquivo (FILE)

`arq = fopen("nome_do_arq", modo);`

Abre/cria arquivo. Retorna NULL se ocorrer algum erro.

`fclose(arq);`

Fecha um arquivo


Abrindo e fechando um arquivo

```
void main ()
{
    FILE *arq;
    arq=fopen("dados.txt", "w+")
    if (arq==NULL)
    {
        printf("Erro na abertura do      arquivo");
        exit(0);
    }
    fclose(arq);
}
```

Modos de abertura

- r: abre arquivo para leitura
- w: cria arquivo para escrita
- a: abre arquivo para escrita (anexando)
- r+: abre arquivo para leitura e escrita
- w+: cria/sobrescreve arquivo para leitura e escrita, apagando os dados existentes anteriormente
- a+: cria/abre arquivo para leitura e escrita (anexando)

Funções para manipulação de arquivo

- fprintf(arq, "string", variáveis) – escreve no arquivo (semelhante ao printf)
 - fscanf(arq, "string", variáveis) – lê do arquivo (semelhante ao scanf)
 - fgets(*dados, tamanho dos dados, arq) – lê do arquivo (semelhante ao gets)
- 

Funções para manipulação de arquivo

- fseek(arq, posição a ser buscada, posição de referencia) – busca posição em arquivo aberto
posição a ser buscada: em bytes
posição de referência: SEEK_SET para início do arquivo
SEEK_CUR para posição atual
SEEK_END para fim do arquivo
- feof(arq) – retorna zero se ponteiro não aponta para o fim do arquivo
- fflush(arq) – descarrega o buffer no arquivo

Exemplo: escrita de texto no arquivo

```
#include <stdio.h>

void main ()
{
    FILE *arq;
    int i, n;

    if((arq=fopen("test.txt", "w+"))==NULL)
        printf("\nErro abrindo arquivo.\n");

    for(i=0; i<5; i++) {
        scanf("%d", &n);
        fprintf(arq, "%d\n", n);
    }

    fclose(arq);
}
```

Exemplo: leitura de texto no arquivo

```
#include <stdio.h>

void main ()
{
    FILE *arq;
    int i, n;

    if((arq=fopen("test.txt", "r+"))==NULL)
        printf("\nErro abrindo arquivo.\n");

    for(i=0; i<5; i++) {
        fscanf(arq, "%d", &n);
        printf("%d\n", n);
    }

    fclose(arq);
    system("PAUSE");
}
```

Exercícios

- 1) Criar um programa que lê uma frase e escreve em um arquivo de texto. Após a escrita no arquivo, o programa deve ler o conteúdo do arquivo e exibi-lo na tela.
- 2) Criar um programa que lê uma palavra digitada, abre um arquivo de texto já existente e verifica se a palavra está contida no arquivo.

Exercícios

2) Solução:

```
#include <stdio.h>

void main ()
{
    FILE *arq;
    char s[20], p[20];
    int i, ok=0;

    if((arq=fopen("test.txt", "r+"))==NULL)
        printf("\nErro abrindo arquivo.\n");

    printf("Digite a palavra: ");
    scanf("%s", p);

    while (feof(arq)==0) {
        fscanf(arq, "%s", s);
        if (strcmp(p, s)==0) ok=1;
        printf("%s ", s);
    }

    if (ok==0) printf("\n\nPalavra
    nao encontrada.\n");
    else printf("\n\nPalavra
    encontrada!\n");

    fclose(arq);
    system("PAUSE");
}
```

Exercícios

- 3) Faça um programa que crie uma cópia de um arquivo. Para isso solicite ao usuário dois nomes de arquivos, sendo o primeiro o arquivo de origem e o segundo o de destino.

Exercícios

3) Solução:

```
#include <stdio.h>
void main ()
{
    FILE *or, *dest;
    char origem[20], destino[20],
        buffer[100], st_arq[20];

    printf("Arquivo de origem: ");
    scanf("%s", origem);
    printf("Arquivo de destino: ");
    scanf("%s", destino);

    if((or=fopen(origem, "r+"))==NULL)
        printf("\nErro abrindo arquivo.\n");
    if((dest=fopen(destino, "w+")==NULL)
        printf("\nErro abrindo arquivo.\n");

    while (feof(or)==0) {
        fgets(buffer, 200, or);
        fprintf(dest, "%s ", buffer);
    }
    fclose(or);
    fclose(dest);
    sprintf(st_arq, "notepad %s",
    destino);
    system("PAUSE");
}
```

Exercícios

- 4) Faça um programa que abra 2 arquivos de texto já existentes e mostre as palavras que estão presentes em ambos os arquivos.