

Programação de Computadores I – Arquivos

Profa. Mercedes Gonzales
Márquez

Arquivos

- Existem dois tipos de arquivos: arquivo texto e arquivo binário.
- Um arquivo texto armazena os dados em ASCII, na forma de sequência de caracteres.
- Um arquivo binário armazena os dados em forma binária (sequência de bits), e normalmente ocupa bem menos espaço em memória do que um arquivo texto para armazenar a mesma informação.

Arquivos

- Os arquivos são manipulados com variáveis do tipo apontador para arquivo, as quais são declaradas da seguinte forma:
- `FILE *arq;`
- Antes de ler ou escrever dados em um arquivo, precisamos endereçá-lo na variável `arq`. Este endereçamento é realizado com o comando `fopen`, que pode abrir o arquivo para leitura e/ou escrita.
- Após leitura e/ou escrita, o comando `fclose` é usado para fechar o arquivo, liberando o apontador `arq`.

Arquivos

■ Arquivo texto – Abrir e Fechar

```
FILE *arquivo;  
arquivo = fopen(nome, modo);
```

- Unidade de disco e diretório
- Caminho Relativo
- Nome do arquivo

```
fclose(arquivo);
```

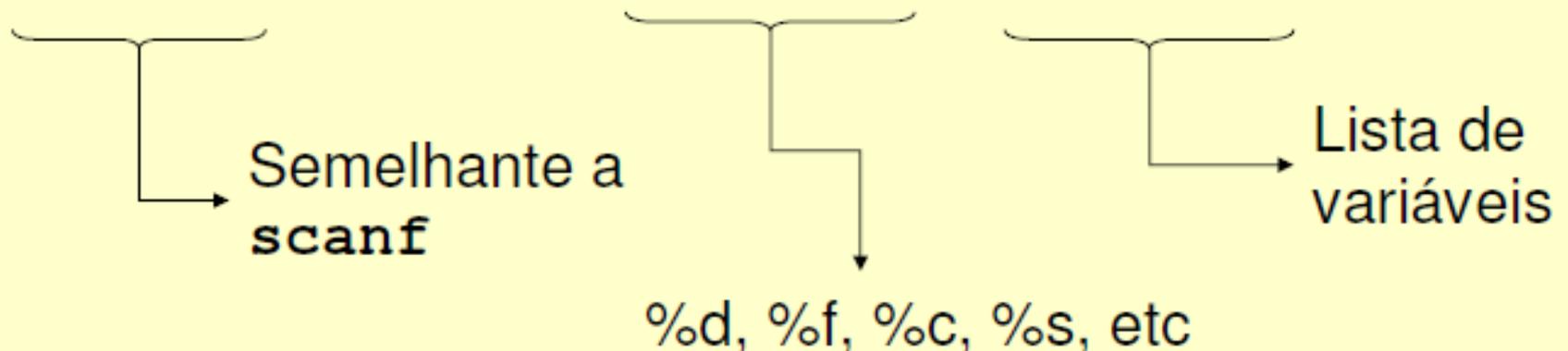
“r”	“r+”
“w”	“w+”
“a”	“a+”
“b”: binário	
“t”: texto	

Arquivos

■ Arquivo texto - Leitura.

```
FILE *arquivo;  
arquivo = fopen(nome, modo);
```

```
fscanf(arquivo, "fomato", &variavel);
```



```
FILE *arquivo;  
char c, linha[102];  
arquivo = fopen(nome, modo);
```

Ler um caractere: `c = fgetc(arquivo);`

Ler uma linha: `fgets(linha, 100, arquivo);`

★★ `FILE *arquivo;`
`arquivo = fopen(nome, modo);`

`fprintf(arquivo, "texto", &variavel);`

Semelhante a
`printf`

Lista de
variáveis

`%d, %f, %c, %s, etc`

```
FILE *arquivo;  
arquivo = fopen(nome, modo);  
char c, texto[100];
```

Escrever um caractere: `fputc(c, arquivo);`

Escrever um texto: `fputs(linha, arquivo);`

Garantir escrita no disco: `fflush(arquivo);`

Arquivos

1. Escreva um PROGRAMA em C que leia um arquivo texto (o usuário deverá digitar o nome do arquivo) e imprima a quantidade de caracteres do arquivo.

Arquivos

```
# include <stdio.h>
```

```
#include <stdlib.h>
```

```
int main() {
```

```
    FILE *ptr;
```

```
    char nomeArquivo[20];
```

```
    int total=0;
```

```
    scanf("%s", nomeArquivo);
```

```
    ptr = fopen(nomeArquivo, "rt");
```

```
    if (ptr==NULL) {
```

```
        printf("Arquivo não existe.");
```

```
        exit(1);
```

```
    }
```

```
    while (getc(ptr)!=EOF) total++;
```

```
    printf("%d", total);
```

```
}
```

Arquivos

2. Escreva um programa em C que leia um arquivo texto (o usuário deverá digitar o nome do arquivo) e imprima o conteúdo na tela. O programa deve imprimir uma linha de cada vez.

Arquivos

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
int main() {
```

```
    FILE *ptr; char nomeArquivo[20], character;
```

```
    scanf("%s", nomeArquivo);
```

```
    ptr = fopen(nomeArquivo, "rt");
```

```
    if (ptr==NULL) {
```

```
        printf("Arquivo não existe.");
```

```
        exit(1); }
```

```
    character=getc(ptr);
```

```
    while (character!=EOF){
```

```
        printf("%c", character);
```

```
        character=getc(ptr); }
```

```
}
```

Arquivos

3. Escreva um programa em C que crie um arquivo texto com números aleatórios. A quantidade de números e o nome do arquivo será fornecido pelo usuário. Os números aleatórios gerados deverão ser menores que 100. Cada número deverá ficar em uma linha diferente.

Arquivos

```
#include <stdio.h>
```

```
#include <math.h>
```

```
#include <stdlib.h>
```

```
Int main() {
```

```
    FILE *arquivo;
```

```
    char nome[10];
```

```
    int quantidade, i;
```

```
    printf("Digite o nome do arquivo:");
```

```
    scanf("%s", nome);
```

```
    printf("Digite a quantidade de numeros:");
```

```
    scanf("%d", &quantidade);
```

Arquivos

```
arquivo = fopen(nome, "w");
if (arquivo==NULL) {
    printf("Falha ao criar o arquivo.");
    exit(1); }
else {
    for (i=1;i<=quantidade;i++)
        fprintf(arquivo, "%d\n", rand()%100);
}
fclose(arquivo);
}
```

Arquivos

4. Escreva um programa em C que armazene os dados de alunos de uma faculdade com a seguinte representação:

```
struct aluno {  
    char nome[10];  
    int matricula;  
    float nota; }  
}
```

A quantidade de alunos será questionada ao usuário antes do preenchimento. O nome do arquivo deverá ser alunos.txt.

```
#include<stdio.h>
#include<stdlib.h>
int main(){
    FILE *arq;
    char *nome = "alunos.txt";
    int i, num;
    typedef struct{
        char nome[10];
        int matricula;
        float nota;
    }cadaluno;

    cadaluno aluno;

    arq= fopen(nome, "w");
    if (arquivo==NULL) {
        printf("Falha ao criar o arquivo.");
        exit(1); }
    else {
        printf ("Informe o numero de alunos: ");
        scanf ("%d", &num);
        printf ("Entrada dos dados dos %d
alunos da faculdade\n", num);
```

```
for(i = 0; i < num; i++){
    printf ("Aluno No %d\n", i+1);
    printf ("Nome do aluno: ");
    scanf ("%s", aluno.nome);
    printf ("Numero de matricula: ");
    scanf ("%d", &aluno.matricula);
    printf ("Nota: ");
    scanf ("%f", &aluno.nota);
    fprintf (arq, "%s %d %2.1f\n",
aluno.nome, aluno.matricula,
aluno.nota);
    }
}

fclose(arq);
}
```

Arquivos

5. O arquivo dados.txt contém uma sequência de números inteiros separados por brancos. O programa seguinte calcula a média destes números.

Arquivos

```
#include <stdio.h>
#include <stdlib.h>
#define TRUE 1
int main (void) {
    int x, n, k; double soma;
    FILE *entrada;
    entrada = fopen ("dados.txt", "r");
    if (entrada == NULL){
        printf ("\nNão encontrei arquivo\n");
        exit (EXIT_FAILURE);
    }
    soma = n = 0;
    while (TRUE) {
        k = fscanf (entrada, "%d", &x);
        if (k != 1) break;
        soma += x; n ++;
    }
    fclose (entrada);
    printf ("A média dos números é %f\n", soma/n);
}
```

Arquivos

A função `fscanf` devolve o número de objetos efetivamente lidos. O programa usa isso para detectar o fim do arquivo (assume-se que o arquivo contém pelo menos um número).

Abrindo o arquivo para escrita e leitura (ou seja `w+`), faça um único programa que crie o arquivo `dados.txt` e imprima a média desses números. Use o comando `rewind` que permite ir no início do arquivo depois que todos os números foram escritos.

Arquivos

```
#include <stdio.h>
#include <stdlib.h>
#define TRUE 1
int main (void) {
    FILE *arquivo;
    char *nome="dados.txt";
    int quantidade, i;
    int x,k; double soma = 0;

    printf("Digite a quantidade de numeros:");
    scanf("%d", &quantidade);
    arquivo = fopen(nome, "w+");
```

Arquivos

```
if (arquivo==NULL) {
    printf("Falha ao criar o arquivo.");
    exit(1); }
else {
    for (i=1;i<=quantidade;i++)
        fprintf(arquivo, "%d ", rand()%100);

    rewind (arquivo);
    soma = 0;
    while (TRUE) {
        k = fscanf (arquivo, "%d", &x);
        if (k != 1) break;
        soma += x;
    }
    fclose (arquivo);
    printf ("A media dos numeros eh %f\n",
soma/quantidade);
}
}
```

Arquivos

6. O programa abaixo, mostra um exemplo que le e escreve vários tipos de dados em um arquivo.

```
#include<stdio.h> #include<stdlib.h>
#define MAX 20
int main () {
    char palavra[MAX];
    int i; float f; FILE *pa;
    char *nome = "format.txt";
    /* Abre o arquivo para leitura e escrita */
    if (( pa = fopen(nome, "w+")) == NULL) {
        printf("\n\nNao foi possivel abrir o arquivo.\n"); exit(1);
    }
    puts ("Entre com uma palavra."); scanf ("%s", palavra);
    puts ("Entre com um numero inteiro."); scanf("%d", &i);
    puts ("Entre com um numero flutuante."); scanf("%f", &f);
```

Arquivos

```
/* Escreve os dados no arquivo */  
fprintf(pa, "%s %d %f", palavra, i, f);  
rewind(pa); /* volta ao inicio do arquivo */  
printf("\nTerminei de escrever, agora vou ler.\n");  
fscanf(pa, "%s %d %f", palavra, &i, &f);  
printf("Palavra lida: %s\n", palavra);  
printf("Inteiro lido: %d\n", i);  
printf("Float lido: %f\n", f);  
fclose(pa); getchar(); /* Espera o usuario digitar alguma coisa */  
}
```

Arquivos

7. Faça um programa que leia 10 caracteres e armazene em um arquivo 10 cópias de cada um. Exiba o conteúdo do arquivo.

```
#include<stdio.h>
```

```
#include<stdlib.h>
```

```
int main(){
```

```
    FILE *parq;
```

```
    char *nome = "dados.txt";
```

```
    char c; int i, j;
```

```
    //Abre o arquivo
```

```
    if((parq = fopen(nome, "w+")) == NULL){
```

```
        printf("\n\nO arquivo nao pode ser aberto."); exit (1);
```

```
    }
```

Arquivos

7. Faça um programa que leia 10 caracteres e armazene em um arquivo 10 cópias de cada um. Exiba o conteúdo do arquivo.

```
//Grava os caracteres
for(i = 0; i < 10; i++){
    c = getche();
    for(j = 0; j < 10; j++)
        fputc(c, parq);
}
rewind(parq);
//Volta ao inicio do arquivo
//Exibe o conteudo do arquivo
while(!feof(parq)) printf("%c", fgetc(parq));
fclose(parq);
getche();
}
```

Arquivos

8. Segue um programa que emite uma listagem de pontuação dos candidatos de um concurso. Assume-se que um arquivo cand.txt com o gabarito do concurso já está disponível.

```
int main() {
    FILE *arq;
    int i, insc, ptos;
    char gab[6], resp[6];
    if( (arq=fopen("cand.txt","r"))==NULL ) {
        printf("Arquivo não pôde ser aberto\n");
        exit(1);
    }
    printf("\nGabarito? ");
    gets(gab);
    while( 1 ) {
        fscanf(arq, "%d %s", &insc, resp);
        if( feof(arq) ) break;
        for(ptos=i=0; i<5; i++)
            if( resp[i]==gab[i] )
                ptos++;
        printf("\n%d %d",insc,ptos);
    }
    fclose(arq);
}
```

Arquivos

9. Faça o programa que realize a criação do arquivo `cand.txt` e depois o leia para emitir a listagem do exercício anterior.

10. Escreva um programa para gerenciar uma agenda de contatos. Para cada contato armazene o nome, o telefone e o aniversário (dia e mês). O programa deve permitir

- (a) inserir contato
- (b) remover contato
- (c) pesquisar um contato pelo nome
- (d) listar todos os contatos
- (e) listar os contatos cujo nome inicia com uma dada letra
- (f) imprimir os aniversariantes do mês.

Sempre que o programa for encerrado, os contatos devem ser armazenados em um arquivo de texto. Quando o programa iniciar, os contatos devem ser inicializados com os dados contidos neste arquivo.