Arquivos

Programação de Computadores I

Natália Batista
nataliabatista@decom.cefetmg.br

1. Arquivos

- Estruturas de dados armazenadas fora da memória principal do computador, por exemplo em discos.
- Usados quando:
  - informações são muito numerosas para caber na memória principal.
  - necessidade de armazenamento permanente de informações.

1. Arquivos

- Qualquer dado não volátil tem que ser guardado em arquivos:
  - textos,
  - programas: fonte e binário,
  - valores de entrada,
  - valores calculados,
  - bancos de dados,
  - imagens,
  - filmes, etc.

2. Tipos de arquivos

- Arquivos binários:
  - Usam o sistema binário de numeração para armazenar informações numéricas ou literais.
- Arquivos texto:
  - Armazenam informações numéricas e literais através de códigos de seus caracteres.
  - São agrupados em linhas, que por sua vez são agrupadas em páginas.

3. Manipulação de arquivos

1. Declarar o ponteiro.
2. Abrir.
3. Ler / Escrever.
4. Fechar.

4. Arquivos: abertura

- Todo arquivo tem dois nomes:
  - interno: declarado dentro do programa.
  - externo: usado pelo sistema operacional.
- Associação dos nomes é feita por meio do ponteiro de arquivo.
- Exemplo:

```
...
FILE *fp; // Declaração da estrutura
 fp = fopen("exemplo.bin","wb");
/* o arquivo se chama exemplo.bin e está localizado no diretório corrente */
if (!fp)
   printf ("Erro na abertura do arquivo.");   //no caso de um erro fopen() retorna um ponteiro (NULL)
...
```

4. Arquivos: abertura

- Protótipo:

  FILE *fopen (char *nome_do_arquivo,char *modo);

Modo	Significado
"r"	Abre um arquivo texto para leitura. O arquivo deve existir antes de ser aberto.
"w"	Abrir um arquivo texto para gravação. Se o arquivo não existir, ele será criado. Se já existir, o conteúdo anterior será destruído.
"a"	Abrir um arquivo texto para gravação. Os dados serão adicionados no fim do arquivo ("append"), se ele já existir, ou um novo arquivo será criado, no caso de arquivo não existente anteriormente.
"rb"	Abre um arquivo binário para leitura. Igual ao modo "r" anterior, só que o arquivo é binário.
"wb"	Cria um arquivo binário para escrita, como no modo "w" anterior, só que o arquivo é binário.
"ab"	Acrescenta dados binários no fim do arquivo, como no modo "a" anterior, só que o arquivo é binário.
"r+"	Abre um arquivo texto para leitura e gravação. O arquivo deve existir e pode ser modificado.
"w+"	Cria um arquivo texto para leitura e gravação. Se o arquivo existir, o conteúdo anterior será destruído. Se não existir, será criado.
"a+"	Abre um arquivo texto para gravação e leitura. Os dados serão adicionados no fim do arquivo se ele já existir, ou um novo arquivo será criado, no caso de arquivo não existente anteriormente.
"r+b"	Abre um arquivo binário para leitura e escrita. O mesmo que "r+" acima, só que o arquivo é binário.
"w+b"	Cria um arquivo binário para leitura e escrita. O mesmo que "w+" acima, só que o arquivo é binário.
"a+b"	Acrescenta dados ou cria uma arquivo binário para leitura e escrita. O mesmo que "a+" acima, só que o arquivo é binário

5. Arquivos: fechamento

- int fclose(FILE *fp);
  - No final da utilização, grava os dados no arquivo.
  - Faz com que qualquer caracter que tenha permanecido no "buffer" associado ao fluxo de saída seja gravado.

6. Arquivo texto

- O programa a seguir **lê uma string do teclado e escreve-a, caractere por caractere em um arquivo** em disco (o arquivo arquivo.txt, que será aberto no diretório corrente).

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

int main() {
  FILE *fp;
  char string[100];
  int i;
  fp = fopen("arquivo.txt","w"); // Arquivo ASCII, para escrita
  if(!fp) {
    printf("Erro na abertura do arquivo.\n");
    return 0;
  };

  printf("Digite a string a ser gravada no arquivo:");
  gets(string);

  // Grava a string, caractere a caractere
  for(i=0; i<strlen(string); i++)
    fputc(string[i], fp);

  fclose(fp);
  return 0;
}
```

6. Arquivo texto

- O programa a seguir **abre um arquivo já existente e o lê, caracter por caracter**, até que o final do arquivo seja atingido. Os caracteres lidos são apresentados na tela.

```
#include <stdio.h>

int main() {
  FILE *fp;
  char c;

  fp = fopen("arquivo.txt","r"); // Abre arquivo texto para leitura
  if(!fp) {
    printf("Erro na abertura do arquivo.\n");
    return 0;
  }

  c = fgetc(fp);

  while(!feof(fp)){     // Enquanto não chegar ao final do arquivo
    printf("%c", c);
    c = fgetc(fp);      // Imprime o caracter lido
  }

  fclose(fp);
  return 0;
}
```

7. Arquivo texto: funções

- int **fputc** (int ch,FILE *fp);
  - Escreve um caractere no arquivo.
- int **fgetc** (FILE *fp);
  - Retorna um caractere lido do arquivo.
- char ***fgets** (char *str, int tamanho,FILE *fp);
  - Lê uma string num arquivo. A função lê a string até que um caracter de nova linha seja lido ou tamanho-1 caracteres tenham sido lidos.
  - A string resultante sempre terminará com '\0' (por isto tamanho-1 caracteres serão lidos).
- char ***fputs** (char *str,FILE *fp);
  - Escreve uma string num arquivo.
- int **feof** (FILE *fp);
  - Retorna não-zero se o arquivo chegou ao EOF, caso contrário retorna zero.

7. Arquivo texto: funções

- int **fprintf**(FILE *fp,char *str,...);
  - Funciona como a função printf(), mas a saída é um arquivo.
- int **fscanf**(FILE *fp,char *str,...);
  - Funciona como a função scanf(), porém lê de um arquivo.

Programa que escreve duas strings em um arquivo para formar uma frase.

```c
#include <stdio.h>

int main() {
  FILE *p;
  char str[30], resposta[80];
  char frase[] = "Este e um arquivo chamado: ";
  int i;

  //Le um nome para o arquivo a ser aberto:
  printf("Entre com um nome para o arquivo: ");
  gets(str);

  if (!(p = fopen(str,"w"))){ //Caso ocorra algum erro na abertura do arquivo
    printf("Erro! Impossivel abrir o arquivo!\n");
    return 0;
  }

  //Se nao houve erro, imprime no arquivo
  fputs(frase, p);
  fputs(str,p);

  fclose(p);
  return 0;
}
```

Programa que escreve uma strings em um arquivo usando a função *fprintf*.

```
#include <stdio.h>

int main() {
  FILE *p;
  char str[80],c;

  // Le um nome para o arquivo a ser aberto:
  printf("Entre com um nome para o arquivo:\n");
  gets(str);

  if (!(p = fopen(str,"w"))) {
    printf("Erro! Impossivel abrir o arquivo!\n");
    return 0;
  }

  // Se nao houve erro, imprime no arquivo e fecha
  fprintf(p,"Este e um arquivo chamado: %s\n", str);
  fclose(p);

  return 0;
}
```

Programa que lê uma string de um arquivo e imprime na tela, caracter por caracter com a função *fscanf*.

```
#include <stdio.h>

int main() {
  FILE *p;
  char str[80], c;

  // Le um nome para o arquivo a ser aberto:
  printf("Entre com um nome para o arquivo: ");
  gets(str);

  // Abre para a leitura
  if (!(p = fopen(str,"r"))) {
    printf("Erro! Impossivel abrir o arquivo!\n");
    return(0);
  }

  // Imprime conteudo do arquivo
  while (!feof(p)) {
     fscanf(p,"%c",&c);
     printf("%c",c);
  }

 fclose(p);
 return 0;
}
```

8. Arquivo binário

- unsigned **fread**(*void *buffer, int numero_de_bytes, int count, FILE *fp*);
  - O *buffer* é a região de memória na qual serão armazenados os dados lidos. O número de bytes é o tamanho da unidade a ser lida. *count* indica quantas unidades devem ser lidas.
  - A função retorna o número de unidades efetivamente lidas.
- unsigned **fwrite**(*void *buffer,int numero_de_bytes,int count,FILE *fp*);
  - Escrita no arquivo. A função retorna o número de itens escritos. Este valor será igual a *count* a menos que ocorra algum erro.

Programa que escreve um registro (cujos dados foram lidos do teclado) em um arquivo binário.

```
#include <stdio.h>

typedef struct CONTA {
  char nome[81];
  float saldo;
} Conta;

int main() {
  FILE *arq;
  Conta cliente;

  //Le os dados da conta do cliente do teclado
  printf("Digite o nome do cliente: ");
  gets(cliente.nome);
  printf("Digite o saldo: ");
  scanf("%f", &cliente.saldo);

  arq = fopen("banco.dat", "wb");
  //Caso ocorra algum erro na abertura do arquivo
  if (arq == NULL){
    printf("Erro! Impossivel abrir o arquivo!\n");
    return 0;
  }

  //Se nao houve erro, grava registro no arquivo
  fwrite(&cliente, sizeof(Conta), 1, arq);

  fclose(arq);
  return 0;
}
```

```c
#include <stdio.h>

typedef struct CONTA {
  char nome[81];
  float saldo;
} Conta;

int main() {
  FILE *arq;
  Conta cliente;

  arq = fopen("banco.dat", "rb");

  //Caso ocorra algum erro na abertura do arquivo
  if ( arq == NULL){
    printf("Erro! Impossivel abrir o arquivo!\n");
    return 0;
  }

  //Le os dados do arquivo
  fread (&cliente, sizeof(Conta), 1, arq);

  printf("Nome do cliente: %s\n", cliente.nome);
  printf("Saldo: %f\n", cliente.saldo);

  fclose(arq);
  return 0;
}
```

Programa que lê um registro de um arquivo binário e escreve os dados na tela.

9. Arquivo binário: exemplo com vetor

```
struct Conta {
  char nome[81];
  float saldo;
}
...
Conta clientes[100];
fwrite (clientes, sizeof (Conta), 100, ponteiro_arq);
```

- O primeiro parâmetro é o ponteiro do vetor cujos elementos queremos gravar no arquivo (no caso, o vetor clientes).
- O segundo, é o tamanho de cada registro.
- O terceiro, é a quantidade de registros que queremos escrever.
- O quarto é o ponteiro para o arquivo já aberto com fopen.

9. Arquivo binário: exemplo com vetor

```
struct Conta {
  char nome[81];
  float saldo;
}
...
Conta clientes[100];
fread (clientes, sizeof (Conta), 100, ponteiro_arq);
```

- O primeiro parâmetro é o ponteiro para onde queremos mandar o que for lido (no caso, o vetor clientes).
- O segundo, é o tamanho de cada registro.
- O terceiro, é a quantidade de registros que queremos ler.
- O quarto é o ponteiro para o arquivo já aberto com fopen.

```c
#include <stdio.h>
#define TAM 10

typedef struct CONTA {
  char nome[81];
  float saldo;
} Conta;


int main() {
  FILE *arq;
  Conta clientes[TAM];

  //Le os dados das contas dos clientes do teclado
  for (int i=0; i<TAM; i++){
    printf("Digite o nome do cliente: ");
    gets(clientes[i].nome);
    printf("Digite o saldo: ");
    scanf("%f%*c", &clientes[i].saldo);
  }

  arq = fopen("banco.dat", "wb");
  //Caso ocorra algum erro na abertura do arquivo
  if ( arq == NULL){
    printf("Erro! Impossivel abrir o arquivo!\n");
    return 0;
  }

  //Se nao houve erro, grava registro no arquivo
  fwrite(clientes, sizeof(Conta), TAM, arq);

  fclose(arq);
  return 0;
}
```

Programa que escreve todos os elementos de um vetor de registros em um arquivo binário.

```c
#include <stdio.h>
#define TAM 10

typedef struct CONTA {
  char nome[81];
  float saldo;
} Conta;

int main() {
  FILE *arq;
  Conta clientes[TAM];

  arq = fopen("banco.dat", "rb");
  //Caso ocorra algum erro na abertura do arquivo
  if ( arq == NULL){
    printf("Erro! Impossivel abrir o arquivo!\n");
    return 0;
  }

  //Se nao houve erro, le registros do arquivo
  fread(clientes, sizeof(Conta), TAM, arq);


  //Imprime na tela os dados das contas dos clientes
  for (int i=0; i<TAM; i++){
    printf("Nome do cliente: %s   ", clientes[i].nome);
    printf("Saldo: %f\n", clientes[i].saldo);
  }

  fclose(arq);
  return 0;
}
```

Programa que lê os registros de um arquivo e os armazena em um vetor de registros no programa.

10. Arquivo binário: acesso

- Acesso sequencial: dados são lidos ou escritos seguidamente. Organização do arquivo texto é sempre sequencial.
- Acesso direto: int **fseek** (*FILE *fp,long numbytes,int origem*);
  - Move a posição corrente de leitura ou escrita no arquivo de um valor especificado, a partir de um ponto especificado.
  - Origem:

Nome	Valor	Significado
SEEK_SET	0	Início do arquivo
SEEK_CUR	1	Ponto corrente no arquivo
SEEK_END	2	Fim do arquivo