Arquivos

Informática – IFSULDEMINAS

Primeiro Semestre de 2013

Roteiro

Estrutra de Decisão

- Já vimos como podemos receber e enviar dados para usuário por meio do teclado e da tela.
- Agora veremos também como ler e gravar dados em arquivos.
- As funções para entrada/saída em arquivo estão declaradas na biblioteca *stdio.h*.
- Uma leitura ou gravação em arquivo envolve abrir e fechar arquivos.

Estrutra de Decisão

- Já vimos como podemos receber e enviar dados para usuário por meio do teclado e da tela.
- Agora veremos também como ler e gravar dados em arquivos.
- As funções para entrada/saída em arquivo estão declaradas na biblioteca *stdio.h*.
- Uma leitura ou gravação em arquivo envolve abrir e fechar arquivos.

Estrutra de Decisão

- Já vimos como podemos receber e enviar dados para usuário por meio do teclado e da tela.
- Agora veremos também como ler e gravar dados em arquivos.
- As funções para entrada/saída em arquivo estão declaradas na biblioteca *stdio.h*.
- Uma leitura ou gravação em arquivo envolve abrir e fechar arquivos.

Estrutra de Decisão

- Já vimos como podemos receber e enviar dados para usuário por meio do teclado e da tela.
- Agora veremos também como ler e gravar dados em arquivos.
- As funções para entrada/saída em arquivo estão declaradas na biblioteca *stdio.h*.
- Uma leitura ou gravação em arquivo envolve abrir e fechar arquivos.

Função **fopen**

fopen

variável_arquivo = fopen(nome_arquivo, modo_abertura);

- A função recebe como parâmetros o nome do arquivo e o modo de leitura.
- Alguns modos de leitura:
  - **r**: leitura (*read*)
  - **w**: gravação sobrescrevendo (*write*)
  - **a**: gravação a partir do final (*append*)
- A função retorna um ponteiro para um arquivo.

Função **fopen**

fopen

variável_arquivo = fopen(nome_arquivo, modo_abertura);

- A função recebe como parâmetros o nome do arquivo e o modo de leitura.
- Alguns modos de leitura:
  - **r**: leitura (*read*)
  - **w**: gravação sobrescrevendo (*write*)
  - **a**: gravação a partir do final (*append*)
- A função retorna um ponteiro para um arquivo.

Função **fopen**

fopen

variável_arquivo = fopen(nome_arquivo, modo_abertura);

- A função recebe como parâmetros o nome do arquivo e o modo de leitura.
- Alguns modos de leitura:
  - **r**: leitura (*read*)
  - **w**: gravação sobrescrevendo (*write*)
  - **a**: gravação a partir do final (*append*)
- A função retorna um ponteiro para um arquivo.

Função **fopen**

fopen

variável_arquivo = fopen(nome_arquivo, modo_abertura);

- A função recebe como parâmetros o nome do arquivo e o modo de leitura.
- Alguns modos de leitura:
  - **r**: leitura (*read*)
  - **w**: gravação sobrescrevendo (*write*)
  - **a**: gravação a partir do final (*append*)
- A função retorna um ponteiro para um arquivo.

Declarando variáveis FILE

- Um arquivo é representado por uma variável do tipo FILE.
- Para abrir um arquivo precisamos de um ponteiro (endereço de memória) para variáveis do tipo FILE.
- Para declarar um ponteiro utilizamos um * antes do nome da variável.

Declarando ponteiro para arquivo

```
FILE * nome_da_variável;
```

Abrindo um arquivo

```
FILE * arquivo;
arquivo = fopen("texto.txt", "w");
```

Declarando variáveis FILE

- Um arquivo é representado por uma variável do tipo FILE.
- Para abrir um arquivo precisamos de um ponteiro (endereço de memória) para variáveis do tipo FILE.
- Para declarar um ponteiro utilizamos um * antes do nome da variável.

Declarando ponteiro para arquivo

```
FILE * nome_da_variável;
```

Abrindo um arquivo

```
FILE * arquivo;
arquivo = fopen("texto.txt", "w");
```

Declarando variáveis FILE

- Um arquivo é representado por uma variável do tipo FILE.
- Para abrir um arquivo precisamos de um ponteiro (endereço de memória) para variáveis do tipo FILE.
- Para declarar um ponteiro utilizamos um * antes do nome da variável.

Declarando ponteiro para arquivo

```
FILE * nome_da_variável;
```

Abrindo um arquivo

```
FILE * arquivo;
arquivo = fopen("texto.txt", "w");
```

Declarando variáveis FILE

- Um arquivo é representado por uma variável do tipo FILE.
- Para abrir um arquivo precisamos de um ponteiro (endereço de memória) para variáveis do tipo FILE.
- Para declarar um ponteiro utilizamos um * antes do nome da variável.

Declarando ponteiro para arquivo

FILE * nome_da_variável;

Abrindo um arquivo

```
FILE * arquivo;
arquivo = fopen("texto.txt", "w");
```

Declarando variáveis FILE

- Um arquivo é representado por uma variável do tipo FILE.
- Para abrir um arquivo precisamos de um ponteiro (endereço de memória) para variáveis do tipo FILE.
- Para declarar um ponteiro utilizamos um * antes do nome da variável.

Declarando ponteiro para arquivo

FILE * nome_da_variável;

Abrindo um arquivo

```
FILE * arquivo;
arquivo = fopen("texto.txt", "w");
```

Função **fprintf**

fprintf

fprintf(nome_arquivo, mensagem);

- A função **fprintf** funciona como a função **printf**, mas a saída é no arquivo.
- A função recebe como parâmetros o nome do arquivo e a mensagem que será escrita no arquivo.

Escrevendo no arquivo

```
FILE * arquivo;
arquivo = fopen("texto.txt", "w");
fprintf(arquivo, "IFSULDEMINAS");
```

Função **fprintf**

fprintf

fprintf(nome_arquivo, mensagem);

- A função **fprintf** funciona como a função **printf**, mas a saída é no arquivo.
- A função recebe como parâmetros o nome do arquivo e a mensagem que será escrita no arquivo.

Escrevendo no arquivo

```
FILE * arquivo;
arquivo = fopen("texto.txt", "w");
fprintf(arquivo, "IFSULDEMINAS");
```

Função **fprintf**

fprintf

fprintf(nome_arquivo, mensagem);

- A função **fprintf** funciona como a função **printf**, mas a saída é no arquivo.
- A função recebe como parâmetros o nome do arquivo e a mensagem que será escrita no arquivo.

Escrevendo no arquivo

```
FILE * arquivo;
arquivo = fopen("texto.txt", "w");
fprintf(arquivo, "IFSULDEMINAS");
```

Função **fprintf**

fprintf

fprintf(nome_arquivo, mensagem);

- A função **fprintf** funciona como a função **printf**, mas a saída é no arquivo.
- A função recebe como parâmetros o nome do arquivo e a mensagem que será escrita no arquivo.

Escrevendo no arquivo

```
FILE * arquivo;
arquivo = fopen("texto.txt", "w");
fprintf(arquivo, "IFSULDEMINAS");
```

Função **fclose**

fclose

fclose(nome_arquivo)

- A função **fclose** é responsável por fechar um arquivo.
- A função recebe como parâmetro o nome do arquivo a ser fechado.
- Todas as vezes que um arquivo é aberto, ele deve ser fechado.

Fechando o arquivo

```
FILE * arquivo;
arquivo = fopen("texto.txt", "w");
...
fclose(arquivo);
```

Função **fclose**

fclose

fclose(nome_arquivo)

- A função **fclose** é responsável por fechar um arquivo.
- A função recebe como parâmetro o nome do arquivo a ser fechado.
- Todas as vezes que um arquivo é aberto, ele deve ser fechado.

Fechando o arquivo

```
FILE * arquivo;
arquivo = fopen("texto.txt", "w");
...
fclose(arquivo);
```

Função **fclose**

fclose

fclose(nome_arquivo)

- A função **fclose** é responsável por fechar um arquivo.
- A função recebe como parâmetro o nome do arquivo a ser fechado.
- Todas as vezes que um arquivo é aberto, ele deve ser fechado.

Fechando o arquivo

```
FILE * arquivo;
arquivo = fopen("texto.txt", "w");
...
fclose(arquivo);
```

Função **fclose**

fclose

fclose(nome_arquivo)

- A função **fclose** é responsável por fechar um arquivo.
- A função recebe como parâmetro o nome do arquivo a ser fechado.
- Todas as vezes que um arquivo é aberto, ele deve ser fechado.

Fechando o arquivo

```
FILE * arquivo;
arquivo = fopen("texto.txt", "w");
...
fclose(arquivo);
```

Função **fclose**

fclose

fclose(nome_arquivo)

- A função **fclose** é responsável por fechar um arquivo.
- A função recebe como parâmetro o nome do arquivo a ser fechado.
- Todas as vezes que um arquivo é aberto, ele deve ser fechado.

Fechando o arquivo

```
FILE * arquivo;
arquivo = fopen("texto.txt", "w");
...
fclose(arquivo);
```

Praticando ...

Problema

Leia dez números inteiros do usuário e escreva estes números no arquivo.

Escrevendo dez números no arquivo

```
int count, n;
FILE * arquivo;
arquivo = fopen("meuArq.txt", "w");
for (count = 1; count <= 10; count++) {
    printf("Digite um número:\n");
    scanf("%d", &n);
    fprintf(arquivo,"%d\n", n);
}
fclose(arquivo);
```

- O programa abre o arquivo **meuArq.txt** no modo **w** para escrita.
- Lê dez número e escreve estes números no arquivo.
- Como na função *printf*, a função *fprintf* usa símbolos especiais para escrever valores de variáveis no arquivo.

Praticando ...

Problema

Leia dez números inteiros do usuário e escreva estes números no arquivo.

Escrevendo dez números no arquivo

```
int count, n;
FILE * arquivo;
arquivo = fopen("meuArq.txt", "w");
for (count = 1; count <= 10; count++) {
    printf("Digite um número:\n");
    scanf("%d", &n);
    fprintf(arquivo,"%d\n", n);
}
fclose(arquivo);
```

- O programa abre o arquivo **meuArq.txt** no modo **w** para escrita.
- Lê dez número e escreve estes números no arquivo.
- Como na função *printf*, a função *fprintf* usa símbolos especiais para escrever valores de variáveis no arquivo.

Praticando ...

Problema

Leia dez números inteiros do usuário e escreva estes números no arquivo.

Escrevendo dez números no arquivo

```
int count, n;
FILE * arquivo;
arquivo = fopen("meuArq.txt", "w");
for (count = 1; count <= 10; count++) {
    printf("Digite um número:\n");
    scanf("%d", &n);
    fprintf(arquivo,"%d\n", n);
}
fclose(arquivo);
```

- O programa abre o arquivo **meuArq.txt** no modo **w** para escrita.
- Lê dez número e escreve estes números no arquivo.
- Como na função *printf*, a função *fprintf* usa símbolos especiais para escrever valores de variáveis no arquivo.

Praticando ...

Problema

Leia dez números inteiros do usuário e escreva estes números no arquivo.

Escrevendo dez números no arquivo

```
int count, n;
FILE * arquivo;
arquivo = fopen("meuArq.txt", "w");
for (count = 1; count <= 10; count++) {
    printf("Digite um número:\n");
    scanf("%d", &n);
    fprintf(arquivo,"%d\n", n);
}
fclose(arquivo);
```

- O programa abre o arquivo **meuArq.txt** no modo **w** para escrita.
- Lê dez número e escreve estes números no arquivo.
- Como na função *printf*, a função *fprintf* usa símbolos especiais para escrever valores de variáveis no arquivo.

Praticando ...

Problema

Leia dez números inteiros do usuário e escreva estes números no arquivo.

Escrevendo dez números no arquivo

```
int count, n;
FILE * arquivo;
arquivo = fopen("meuArq.txt", "w");
for (count = 1; count <= 10; count++) {
    printf("Digite um número:\n");
    scanf("%d", &n);
    fprintf(arquivo,"%d\n", n);
}
fclose(arquivo);
```

- O programa abre o arquivo **meuArq.txt** no modo **w** para escrita.
- Lê dez número e escreve estes números no arquivo.
- Como na função *printf*, a função *fprintf* usa símbolos especiais para escrever valores de variáveis no arquivo.

Função **fscanf**

fscanf

fscanf(nome_arquivo, símbolo_especial, variável)

- A função **fscanf** funciona como a função **scanf**, mas a leitura é feita no arquivo.
- A função recebe como parâmetros o nome do arquivo, o símbolo especial que indica o tipo de dado que será lido (%d, %f, %c, %s), e a variável.

Lendo do arquivo

```
char str[20];
FILE * arquivo;
arquivo = fopen("texto.txt", "r");
fscanf(arquivo, "%s", str);
fclose(arquivo);
```

Função **fscanf**

fscanf

fscanf(nome_arquivo, símbolo_especial, variável)

- A função **fscanf** funciona como a função **scanf**, mas a leitura é feita no arquivo.
- A função recebe como parâmetros o nome do arquivo, o símbolo especial que indica o tipo de dado que será lido (%d, %f, %c, %s), e a variável.

Lendo do arquivo

```
char str[20];
FILE * arquivo;
arquivo = fopen("texto.txt", "r");
fscanf(arquivo, "%s", str);
fclose(arquivo);
```

Função **fscanf**

fscanf

fscanf(nome_arquivo, símbolo_especial, variável)

- A função **fscanf** funciona como a função **scanf**, mas a leitura é feita no arquivo.
- A função recebe como parâmetros o nome do arquivo, o símbolo especial que indica o tipo de dado que será lido (%d, %f, %c, %s), e a variável.

Lendo do arquivo

```
char str[20];
FILE * arquivo;
arquivo = fopen("texto.txt", "r");
fscanf(arquivo, "%s", str);
fclose(arquivo);
```

Função **fscanf**

fscanf

fscanf(nome_arquivo, símbolo_especial, variável)

- A função **fscanf** funciona como a função **scanf**, mas a leitura é feita no arquivo.
- A função recebe como parâmetros o nome do arquivo, o símbolo especial que indica o tipo de dado que será lido (%d, %f, %c, %s), e a variável.

Lendo do arquivo

```
char str[20];
FILE * arquivo;
arquivo = fopen("texto.txt", "r");
fscanf(arquivo, "%s", str);
fclose(arquivo);
```

Praticando ...

Problema

Leia dez números inteiros do arquivo.

Lendo dez números do arquivo

```
int cont, n;
FILE * arquivo;
arquivo = fopen("meuArq.txt", "r");
for (cont = 1; cont <= 10; cont++) {
    fscanf(arquivo, "%d", n);
    printf("Numero %d: %d\n", i, n);
}
fclose(arquivo);
```

- O programa abre o arquivo **meuArq.txt** no modo **r** para leitura.
- Lê dez número do arquivo e escreve estes números na tela.
- Como na função *scanf*, a função *fscanf* usa símbolos especiais para ler valores do arquivo para variáveis.

Praticando ...

Problema

Leia dez números inteiros do arquivo.

Lendo dez números do arquivo

```
int cont, n;
FILE * arquivo;
arquivo = fopen("meuArq.txt", "r");
for (cont = 1; cont <= 10; cont++) {
    fscanf(arquivo, "%d", n);
    printf("Numero %d: %d\n", i, n);
}
fclose(arquivo);
```

- O programa abre o arquivo **meuArq.txt** no modo **r** para leitura.
- Lê dez número do arquivo e escreve estes números na tela.
- Como na função *scanf*, a função *fscanf* usa símbolos especiais para ler valores do arquivo para variáveis.

Praticando ...

Problema

Leia dez números inteiros do arquivo.

Lendo dez números do arquivo

```
int cont, n;
FILE * arquivo;
arquivo = fopen("meuArq.txt", "r");
for (cont = 1; cont <= 10; cont++) {
    fscanf(arquivo, "%d", n);
    printf("Numero %d: %d\n", i, n);
}
fclose(arquivo);
```

- O programa abre o arquivo **meuArq.txt** no modo **r** para leitura.
- Lê dez número do arquivo e escreve estes números na tela.
- Como na função *scanf*, a função *fscanf* usa símbolos especiais para ler valores do arquivo para variáveis.

Praticando ...

Problema

Leia dez números inteiros do arquivo.

Lendo dez números do arquivo

```
int cont, n;
FILE * arquivo;
arquivo = fopen("meuArq.txt", "r");
for (cont = 1; cont <= 10; cont++) {
    fscanf(arquivo, "%d", n);
    printf("Numero %d: %d\n", i, n);
}
fclose(arquivo);
```

- O programa abre o arquivo **meuArq.txt** no modo **r** para leitura.
- Lê dez número do arquivo e escreve estes números na tela.
- Como na função *scanf*, a função *fscanf* usa símbolos especiais para ler valores do arquivo para variáveis.

Função **fgets**

fgets

fgets(variável_string, número_caracteres, nome_arquivo)

- A função **fgets** funciona como a função **gets**, mas a leitura é feita no arquivo.
- A função recebe como parâmetros uma variável string que receberá o texto, o número de caracteres que será lido e o nome do arquivo.

Lendo do arquivo

```
char str[110];
FILE * arquivo;
arquivo = fopen("texto.txt", "r");
fgets(str, 100, arquivo);
fclose(arquivo);
```

Função **fgets**

fgets

fgets(variável_string, número_caracteres, nome_arquivo)

- A função **fgets** funciona como a função **gets**, mas a leitura é feita no arquivo.
- A função recebe como parâmetros uma variável string que receberá o texto, o número de caracteres que será lido e o nome do arquivo.

Lendo do arquivo

```
char str[110];
FILE * arquivo;
arquivo = fopen("texto.txt", "r");
fgets(str, 100, arquivo);
fclose(arquivo);
```

Função **fgets**

fgets

fgets(variável_string, número_caracteres, nome_arquivo)

- A função **fgets** funciona como a função **gets**, mas a leitura é feita no arquivo.
- A função recebe como parâmetros uma variável string que receberá o texto, o número de caracteres que será lido e o nome do arquivo.

Lendo do arquivo

```
char str[110];
FILE * arquivo;
arquivo = fopen("texto.txt", "r");
fgets(str, 100, arquivo);
fclose(arquivo);
```

Função **fgets**

fgets

fgets(variável_string, número_caracteres, nome_arquivo)

- A função **fgets** funciona como a função **gets**, mas a leitura é feita no arquivo.
- A função recebe como parâmetros uma variável string que receberá o texto, o número de caracteres que será lido e o nome do arquivo.

Lendo do arquivo

```
char str[110];
FILE * arquivo;
arquivo = fopen("texto.txt", "r");
fgets(str, 100, arquivo);
fclose(arquivo);
```

Praticando ...

Problema

Leia um texto de 200 caracteres do arquivo.

Lendo texto do arquivo

```
char str[210];
FILE * arquivo;
arquivo = fopen("meuArq.txt", "r");
fgets(str, 200, arquivo);
printf("Texto: %s\n", str);
fclose(arquivo);
```

- O programa abre o arquivo **meuArq.txt** no modo **r** para leitura.
- Lê 200 caracteres do arquivo usando *fgets* e imprime a string na tela.

Praticando ...

Problema

Leia um texto de 200 caracteres do arquivo.

Lendo texto do arquivo

```
char str[210];
FILE * arquivo;
arquivo = fopen("meuArq.txt", "r");
fgets(str, 200, arquivo);
printf("Texto: %s\n", str);
fclose(arquivo);
```

- O programa abre o arquivo **meuArq.txt** no modo **r** para leitura.
- Lê 200 caracteres do arquivo usando *fgets* e imprime a string na tela.

Praticando ...

Problema

Leia um texto de 200 caracteres do arquivo.

Lendo texto do arquivo

```
char str[210];
FILE * arquivo;
arquivo = fopen("meuArq.txt", "r");
fgets(str, 200, arquivo);
printf("Texto: %s\n", str);
fclose(arquivo);
```

- O programa abre o arquivo **meuArq.txt** no modo **r** para leitura.
- Lê 200 caracteres do arquivo usando *fgets* e imprime a string na tela.

Praticando ...

Problema

Leia um texto de 200 caracteres do arquivo.

Lendo texto do arquivo

```
char str[210];
FILE * arquivo;
arquivo = fopen("meuArq.txt", "r");
fgets(str, 200, arquivo);
printf("Texto: %s\n", str);
fclose(arquivo);
```

- O programa abre o arquivo **meuArq.txt** no modo **r** para leitura.
- Lê 200 caracteres do arquivo usando *fgets* e imprime a string na tela.

Função **feof**

feof

feof(nome_arquivo);

- A função **feof** verifica se o arquivo chegou ao final.
- A função recebe como parâmetro o nome do arquivo.

Lendo caracteres até o final do arquivo

```
char c;
FILE * arquivo;
arquivo = fopen("texto.txt", "r");
while (!feof(arquivo)) {
   fscanf(arquivo,"%c", &c);
   printf("%c",c);
}
fclose(arquivo);
```

- O programa lê caracteres enquanto não chegou ao final do arquivo.

Função **feof**

feof

feof(nome_arquivo);

- A função **feof** verifica se o arquivo chegou ao final.
- A função recebe como parâmetro o nome do arquivo.

Lendo caracteres até o final do arquivo

```
char c;
FILE * arquivo;
arquivo = fopen("texto.txt", "r");
while (!feof(arquivo)) {
   fscanf(arquivo,"%c", &c);
   printf("%c",c);
}
fclose(arquivo);
```

- O programa lê caracteres enquanto não chegou ao final do arquivo.

Função **feof**

feof

feof(nome_arquivo);

- A função **feof** verifica se o arquivo chegou ao final.
- A função recebe como parâmetro o nome do arquivo.

Lendo caracteres até o final do arquivo

```
char c;
FILE * arquivo;
arquivo = fopen("texto.txt", "r");
while (!feof(arquivo)) {
   fscanf(arquivo,"%c", &c);
   printf("%c",c);
}
fclose(arquivo);
```

- O programa lê caracteres enquanto não chegou ao final do arquivo.

Função **feof**

feof

feof(nome_arquivo);

- A função **feof** verifica se o arquivo chegou ao final.
- A função recebe como parâmetro o nome do arquivo.

Lendo caracteres até o final do arquivo

```
char c;
FILE * arquivo;
arquivo = fopen("texto.txt", "r");
while (!feof(arquivo)) {
   fscanf(arquivo,"%c", &c);
   printf("%c",c);
}
fclose(arquivo);
```

- O programa lê caracteres enquanto não chegou ao final do arquivo.

Função **feof**

feof

feof(nome_arquivo);

- A função **feof** verifica se o arquivo chegou ao final.
- A função recebe como parâmetro o nome do arquivo.

Lendo caracteres até o final do arquivo

```
char c;
FILE * arquivo;
arquivo = fopen("texto.txt", "r");
while (!feof(arquivo)) {
   fscanf(arquivo,"%c", &c);
   printf("%c",c);
}
fclose(arquivo);
```

- O programa lê caracteres enquanto não chegou ao final do arquivo.