Trabalhando com arquivos em C

Eduardo S. Dobay

Outubro de 2007

O processo de trabalhar com arquivos em C consiste em três etapas:

1. Abrir o arquivo;
2. Ler e/ou gravar as informações desejadas no arquivo;
3. Fechar o arquivo.

1 Abrindo arquivos

Em C, para poder trabalhar em um arquivo, precisamos abri-lo, associando-o a uma variável interna do programa. Para isso, usamos variáveis do tipo `FILE *` (cujo funcionamento interno não nos importa), que podem ser declaradas assim:

```
FILE *entrada;
FILE *saida;
```

O nome das variáveis é, repito, *interno* ao programa, de modo que poderíamos ter escolhido `bacalhau`, `Dom_Pedro_II` ou `trifosfato_de_adenosina`, independentemente do nome do arquivo com que fôssemos trabalhar.

Mas, calma, a associação entre variável e arquivo ainda não foi feita. Quem faz isso é a função `fopen`, que funciona da seguinte maneira:

```
entrada = fopen("arquivo_entrada.txt", "r");
saida = fopen("arquivo_saida.txt", "w");
```

Essa função precisa de dois parâmetros, dos quais o primeiro é o mais óbvio: o nome do arquivo. O segundo parâmetro diz ao computador o que pretendemos fazer com o arquivo: gravar ("`w`", de *write*) ou ler dados ("`r`", de *read*). No final, se tudo tiver dado certo, essas variáveis conterão um tipo de referência aos arquivos que abrimos. São essas variáveis que iremos usar quando formos ler e gravar nossos dados.

Veja que, ao abrir um arquivo para gravação, pode acontecer de já existir um arquivo com o mesmo nome que você pediu. Se isso ocorrer, o arquivo existente será apagado, e o que você gravar ficará no lugar do arquivo antigo. Caso contrário, o programa simplesmente criará um arquivo novo, com o nome que você pediu. Se o que você quer é apenas adicionar dados ao final do arquivo, sem apagar nada, você pode usar, no lugar da letra `w`, a letra "`a`" (de *append*).

1.1 Problemas

Existem algumas situações que podem impedi-lo de abrir um arquivo:

- O arquivo não existe e você tentou abri-lo para leitura;
- Você não tem permissão para ler ou gravar o arquivo pedido;
- O arquivo já está aberto e/ou bloqueado por outro programa.

Nesses casos, quando você chamar a função `fopen`, a variável referente ao arquivo receberá o valor `NULL` (um valor mais ou menos especial em C, que geralmente indica algum objeto inválido). Para verificar se isso ocorreu, você pode fazer o seguinte teste:

```
if (entrada == NULL)
{
   printf("Socorro! O arquivo não pôde ser aberto!\n");
}
```

2 Gravando em arquivos

Gravar dados em arquivos é realmente fácil, se você estiver acostumado com a função `printf`. A única diferença é que você usa a função `fprintf`, e precisa dizer o arquivo no qual as coisas devem ser escritas. Mas atenção, você não vai dizer o nome do arquivo, mas sim a variável à qual você associou o arquivo agora há pouco!

Veja como se faz isso com alguns exemplos simples:

```
fprintf(saida, "Bom dia!\n");
fprintf(saida, "Resultado: %d\n", resultado);
fprintf(saida, "soma = %10.6f      quociente = %10.6f\n",
        soma, quociente);
```

3 Lendo arquivos

Para ler arquivos também não há nada de extraordinário: a função `scanf` é trocada pela função `fscanf`, e especificamos (da mesma maneira que com a `fprintf`) o arquivo que queremos usar. Por exemplo:

```
int n1, n2;
...
fscanf(entrada, "%d %d", &n1, &n2);
```

3.1 Cuidados

Quer esteja lendo de um arquivo, quer esteja lendo do teclado, você deve ter cuidado (especialmente no começo de uma linha) ao usar o formato de leitura "`%c`", usado para ler caracteres. O que ocorre é que o C não faz nenhuma distinção entre os tipos de caracteres quando esse formato é usado.

```
char c;
...
fscanf(entrada, " %c", &c);
```

Similarmente, se você quiser ler dois caracteres separados por espaços, você deve usar o formato `" %c %c"`.

4 Fechando arquivos

Finalmente, depois que você terminou de se divertir, é necessário fechar os arquivos que você abriu. O fechamento do arquivo serve, basicamente, para liberar a memória que foi utilizada para trabalhar com o arquivo, além de desbloquear o arquivo para uso em outros programas.

Essa tarefa é bem simples (e deve ser executada para cada arquivo que foi aberto):

```
fclose(entrada);
fclose(saida);
```

Note que você não deve tentar fechar um arquivo que sequer foi aberto — ou seja, se você constatou que ocorreu um problema (a tal da comparação com `NULL`).