DCC–IME–USP

Routo Terada
Sala 288A Tel.: 3091 6144
e-mail rt@ime.usp.br

Introdução à Computação I
Curso de C. Moleculares – Segundo Semestre de 2003

Exercício-Programa 11, Peso 1 Data de entrega: até a aula de 99 de novembro de 2003.

1 Contagem de palavras em texto

Faça um programa em C que lê e imprime um texto e posteriormente imprime o número de palavras nesse texto.

O texto será dado em um arquivo. Isso signi…ca que você deverá ler os caracteres de um arquivo texto (e não do teclado). A leitura poderá ser feita caractere a caractere.

A seguir explicaremos os conceitos básicos que você necessita para fazer a leitura caractere a caractere de um arquivo texto.

Arquivos

Um arquivo texto é simplesmente uma seqüência de caracteres. Para utilizar um arquivo no seu programa em C, você deve declarar uma variável do tipo FILE como mostramos a seguir: FILE *texto;

A variável texto é uma variável interna ao seu programa. Para fazer a associação desta variável a um arquivo externo de nome, por exemplo, "entrada.txt", o seu programa deve conter, entre os comandos de inicialização, a seguinte linha: texto = fopen("entrada.txt", "r");

Essa linha deve ser executada apenas uma vez pelo seu programa, e deve vir antes de qualquer leitura do arquivo. Por exemplo, você pode colocar esta linha como sendo a primeira (depois das declarações de variáveis) do seu programa.

Existe uma função que pode ser usada para testar se não há mais caracteres para serem lidos no arquivo, ou seja, se o programa leu o texto armazenado no arquivo até o …m. Essa função se chama feof() e tem como parâmetro a variável que indica o arquivo. Mostraremos em mais detalhes o uso da função feof() abaixo.

Finalmente, um arquivo deve ser fechado depois de usado. Para isso, no …nal do seu programa ponha a linha: fclose(texto);

Caracteres

Para manipular caracteres em um programa, você pode usar variáveis do tipo char. Em C, para declarar uma variável c do tipo char basta a seguinte linha: char c;

Tal variável c pode armazenar um caractere, por exemplo, 'a' ou 'B' ou '5'. Além dos caracteres usuais que você encontra num texto, um arquivo texto pode conter alguns caracteres especiais, como o caractere indicador de …m de linha. O caractere indicador de …m de linha é representado em C por 'n0', e de linha nova é 'nn'.

Para fazer a leitura do próximo caractere de um arquivo indicado pela variável texto, sugerimos o uso da função getc(). Mais especi…camente, suponha que em um programa em C queremos ler um caractere de um arquivo indicado pela variável texto, e guardar o caractere lido em uma variavel c. Isso pode ser feito através da seguinte instrução: c = getc(texto);
onde texto e c foram declarados como acima. Essa instrução lê o próximo caractere do arquivo indicado por texto e armazena na variável c.

Para imprimir um caractere, use o formato de impressão ‘ ‘%c’ ’. Por exemplo, para imprimir o caractere armazenado na variável c, use o comando: printf(“%c”, c);

Exemplo

Em seguida apresentamos um exemplo de um programa que lê e imprime o texto armazenado em um arquivo chamado “entrada.txt” e conta o número de caracteres no texto dado.

```
#include <stdio.h>
main() {
 FILE *texto; /* Apontador para o arquivo de entrada. */
 char c; /* Armazena o último caractere lido. */
 int cont; /* Contador do número de caracteres. */
 if ((texto = fopen(“entrada.txt”, “r”)) == NULL)
      printf(“Erro: não pode abrir o arquivo de entrada.nn”);
 cont = 0;
 c = getc(texto); /* Lê o primeiro caractere do texto. */
 while (!feof(texto)) {
   printf(“%c”, c);
   if (c != ‘nn’&& c != ‘n0’)
   /* Se o caractere não é um indicador de linha nova e
     de …m de linha */
   cont = cont + 1;
   c = getc(texto); /* Lê o próximo caractere do texto. */
   } /* …m while */
 printf(“O número de caracteres no texto dado é: %d.nn”, cont);
 fclose(texto);
} // …m main
```

O seu programa deve ter uma estrutura parecida com essa. Mais especi…camente, a declaração do arquivo, inicialização e …nalização podem ser feitas exatamente da mesma forma, bem como a leitura dos caracteres e o controle de quando todos os caracteres foram lidos.

Algumas coisas que você pode usar no programa

Para que o seu programa leia o texto de um arquivo qualquer, ele deve ler o nome do arquivo de entrada. Para isso, siga a seguinte receita.

```
#include <stdio.h>
main() {
```

```
FILE *texto;
char nome[40];
printf("Digite com o nome do arquivo: ");
scanf("%s", nome);
if ((texto = fopen(nome, "r")) == NULL)
   printf("Erro: não pode abrir o arquivo de entrada.nn");
...
fclose(texto);
} // ...m main
```

Para converter um caractere maiúsculo para minúsculo, você pode usar o seguinte "truque":

```
if ((ch > = 'A') && (ch < = 'Z')) /* Se maiúscula */
 ch = ch + ( 'a'- 'A');
```

É possível usar caracteres para indexar um vetor da seguinte forma: se ch é um caractere minúsculo, X[ch - 'a'] é a posição 0 do vetor X se ch=='a', posição 1 se ch=='b', e assim por diante.

Observações

- Este exercício é para ser feito individualmente.
- Entregue um envelope com o seu nome e com os seguintes itens:
  - uma listagem em papel com o programa em linguagem C
  - uma descrição simples (cerca de 5 linhas) explicando como usar o programa
  - um disquete com os seguintes arquivos
    - o programa em linguagem C,
    - o programa compilado,
    - arquivos com os dados de entrada , pelo menos 4 arquivos, chamados ENT1, ENT2, etc., e
    - arquivos com os dados de saída, pelo menos 4 arquivos, correspondentes, chamados SAI1, SAI2, etc.
    - para redirecionar os arquivos para disco, veja o ...m da página 9 da apostila.
- Coloque comentários em seu programa explicando o que cada etapa do programa signi...ca! Isso será levado em conta na sua nota.
- Coloque como comentário o seu nome, número USP, qual o compilador (gcc, TURBO-C, ou outro), qual o sistema operacional (LINUX, MS-DOS, UNIX, ou outro) e qual o modelo de computador (Intel x86, SUN, ou outro) que V usou.
- Faça uma saída clara! Isso será levado em conta na sua nota.

- Não deixe para a última hora. Planeje investir 70 porcento do tempo total de dedicação em escrever o seu programa todo e simular o programa SEM computador (eliminando erros de lógica) ANTES de digitar e compilar no computador. Isso economiza muito tempo e energia.