Introdução à Computação

Arquivos

Arquivo: tipo texto

- *Stream* de texto
- Um arquivo texto é uma sequência de linhas, onde cada linha contém zero ou mais caracteres e termina com um ou mais caracteres que assinalam o fim da linha
- O comprimento máximo de uma linha é 255 caracteres
- Na linguagem C, o final de linha é indicado pelo caractere ‘\n’ e ao salvar num arquivo, o ‘\n’ é convertido para CR-LF (Carrieage-return/linefeed). Na leitura ocorre o inverso.

Arquivo: tipo binário

- É um cujo conteúdo deve ser interpretado por um programa que entende com antecedência como ele é formatado.
- Um executável é um exemplo de um arquivo binário.
- Um arquivo binário pode ser transmitido na rede
  - Mais seguro, pois poucos programas conseguem interpreta-lo.
- Um arquivo de documento Microsoft Word é arquivo do tipo binário ou do tipo texto?

Nome de arquivos

- Os nomes são armazenados em *strings*
- Deve-se seguir as regras de nomenclatura do sistema operacional
  - Note que no Windows o caminho é indicado pela barra invertida
- O nome do arquivo pode ser informado junto com sua localização no disco
  - Ex: c:\data\conta.txt
- Como em C a barra invertida também tem um significado especial, para representar o caminho numa string é necessário preceder cada barra com outra barra invertida
  - char *filename = “c:\\data\\conta.txt”;

Arquivo: abrir

- Modos de abrir um arquivo:
  - r (abre p/ leitura; arquivo deve existir)
  - w (cria um novo p/ escrita; exclui tudo se existir)
  - a (cria um novo se n existir; *append operation*)

- Veja o significado de cada parâmetro em:

http://www.cplusplus.com/reference/clibrary/cstdio/fopen/

Exemplo

```
FILE *fp;
char *filename = "conta.txt";
char *mode = "w";
if ((fp = fopen(filename,mode)) != NULL)
   printf("Arquivo aberto com sucesso \n");
else
   printf("Erro na abertura do arquivo \n");
fclose(fp);
```

fopen retorna um ponteiro para o tipo FILE, que é uma estrutura declarada em STDIO.H

Tipos de Arquivo

```
00000000 68 65 6C 6C 6F 0D 0A                                     hello..
```

```
00000000 7B 5C 72 74 66 31 5C 61 6E 73 69 5C 61 6E 73 69 {\rtf1\ansi\ansi
00000010 63 70 67 31 32 35 32 5C 64 65 66 66 30 5C 64 65 cpg1252\deff0\de
00000020 66 6C 61 6E 67 33 30 38 31 7B 5C 66 6F 6E 74 74 flang3081{\fontt
00000030 62 6C 7B 5C 66 30 5C 66 73 77 69 73 73 5C 66 63 bl{\f0\fswiss\fc
00000040 68 61 72 73 65 74 30 20 41 72 69 61 6C 3B 7D 7D harset0 Arial;}}
00000050 0D 0A 7B 5C 2A 5C 67 65 6E 65 72 61 74 6F 72 20 ..{\*\generator
00000060 4D 73 66 74 65 64 69 74 20 35 2E 34 31 2E 31 35 Msftedit 5.41.15
00000070 2E 31 35 30 33 3B 7D 5C 76 69 65 77 6B 69 6E 64 .1503;}\viewkind
00000080 34 5C 75 63 31 5C 70 61 72 64 5C 66 30 5C 66 73 4\uc1\pard\f0\fs
00000090 32 30 20 68 65 6C 6C 6F 5C 70 61 72 0D 0A 5C 70 20 hello\par..\p
000000A0 61 72 0D 0A 7D 0D 0A 00                           ar..}...
```

Arquivo: abrir do tipo binário

- Por padrão, o arquivo manipulado é do tipo texto

- Para trabalhar com arquivo binário, deve-se acrescentar o caractere ‘b’ no final do modo de abertura
  - Ex: char *mode = "wb";

Entrada e saída de dados formatados

- Arquivos do tipo texto

- Função de saída:

```
int fprintf ( FILE * stream, const char * format, ... );
```

- Função de entrada:

```
int fscanf ( FILE * stream, const char * format, ... );
```

- O *fprintf* e o *fscanf* funcionam de modo semelhante ao *printf* e o *scanf*, porém a partir de um *stream* especificado, e não do *stdin (padrão)*

Exemplo com fprintf

```
FILE *fp;
char *filename = "conta";
char *mode = "w";
float salario = 4000.25;
char *nome = "Joao";
float aux_nome[5], aux_sal;

if ((fp = fopen(filename,mode)) != NULL){
   printf("Arquivo texto aberto com sucesso \n");
        fprintf(fp,"%s %f\n", nome, salario);

}
else{
   printf("Erro na abertura do arquivo texto \n");
}
fclose(fp);
```

Exemplo com fscanf

```
FILE *fp;
char *filename = "conta";
char *mode = “r";
float salario = 4000.25;
char *nome = "Joao";
float aux_nome[5], aux_sal;

if ((fp = fopen(filename,mode)) != NULL){
   fscanf(fp,"%s %f", aux_nome, &aux_sal);
   printf("Salario = %s %f\n", aux_nome, aux_sal);
}
else{
   printf("Erro na abertura do arquivo texto \n");
}
fclose(fp);
```

Entrada e saída direta de arquivos

- Arquivos do tipo binário
- Normalmente utilizado para salvar dados que serão lidos pelo mesmo programa
- Blocos de dados da memória podem ser lidos/gravados diretamente de/para arquivo

Saída direta

int fwrite(void *buf, int size, int count, FILE *fp);

- O argumento *buf* é um ponteiro para a região da memória que deseja salvar (por ser void, pode apontar para qualquer tipo)
- *Size* especifica o tamanho em bytes dos itens individuais
- *Count* indica a quantidade de itens
- **fp* indica o arquivo
- A função retorna a qtde de itens salvos com sucesso

Implemente e compare o tamanho dos arquivos do tipo texto e binário

```
FILE *fp, *fpb;
char *filename = "conta_tex", *filenameb = "conta_bin";
char *mode = "w", *modeb = "wb";
float salariob = 4000.123;
float salario = 4000.123;
if ((fp = fopen(filename,mode)) != NULL){
        fprintf(fp,"%f",salario);

}
else{
   printf("Erro na abertura do arquivo texto \n");
}
if ((fpb = fopen(filenameb,modeb)) != NULL){
        fwrite(&salariob, sizeof(float),1,fpb);

}
else{
   printf("Erro na abertura do arquivo texto \n");
}
fclose(fp);
fclose(fpb);
```

Leitura direta

```
int fread (void *buf, int size, int count,
    FILE *fp);
```

- De modo oposto ao fwrite, o fread lê um bloco de dados de um arquivo em modo binário para a memória
- Nesse caso o dado é lido do arquivo apontado pelo *fp para a variável *buf

Posição de acesso do arquivo

- Especificada em termos de bytes a partir do início do arquivo
- Ao abrir um arquivo o indicador de posição aponta para o início do arquivo (posição 0), a não ser que o arquivo foi aberto no modo *append* ou e o seu tamanho for > 0;
- Por padrão, a posição muda sequencialmente de acordo com a quantidade de bytes lidos/escritos do arquivo
- Há funções que permitem realizar o reposicionamento

Funções de reposicionamento

- Para obter a posição atual de leitura/escrita
  - long int ftell (FILE * stream);

- Para retornar à posição zero
  - void rewind (FILE * stream);

- Mover para uma posição específica
  - int fseek (FILE * stream, long int offset, int origin);

http://www.cplusplus.com/reference/clibrary/cstdio/fseek/

fseek()

```
FILE * pFile;
pFile = fopen ( "example.txt" , "w" );
fputs ( "This is an apple." , pFile );
fseek ( pFile , 9 , SEEK_SET );
fputs ( " sam" , pFile );
fclose ( pFile );
```

Exercício

- Escrever um programa para ler um texto do usuário e no final de digitá-lo (sinalizado por um ponto de exclamação), grave o texto em um arquivo. Imprimir no final o texto do arquivo.
- Escreva um programa que grave no arquivo cada caractere digitado pelo usuário (finalizado por uma exclamação tb) e imprima no final o texto completo

Referência

- http://www.cplusplus.com/reference/clibrary/