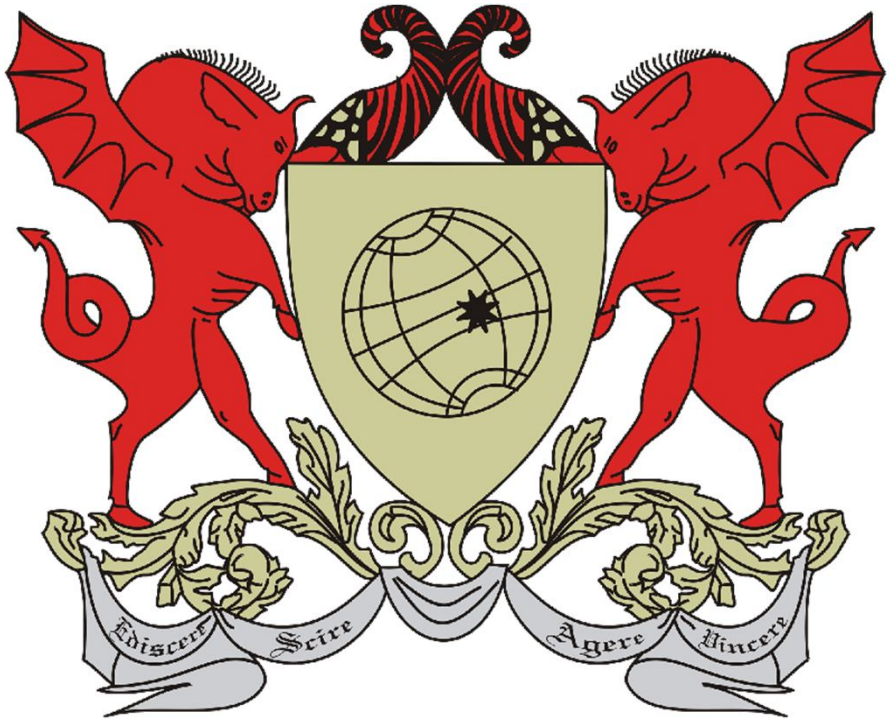

CCF 110 – Programação

Aula 10 – Manipulação de arquivos/Linguagem C

Prof. José Augusto Nacif – jnacif@ufv.br

Conceitos Básicos

- Um arquivo em disco consiste em um conjunto de informações que são mantidas na memória secundária.
- Comparando-se memória principal (RAM) e memória secundária (discos), os aspectos mais relevantes são eficiência e persistência.
- Para contornar os problemas decorrentes da eficiência o sistema operacional “bufferiza” as informações lidas/gravadas em disco.

Conceitos Básicos

- Em C os arquivos podem ser vistos/tratados de 2 maneiras: em modo texto (uma sequência de caracteres) ou binário (uma sequência de bytes).
- Um arquivo tipo texto pode ser lido/tratado por qualquer editor de textos (bloco de notas, Word, Dev C++)
- Um arquivo binário pode manipular grandes quantidades de informação de modo mais eficiente

Conceitos Básicos

- Para qualquer um dos tipos, o sistema operacional disponibiliza um conjunto de serviços necessários à manipulação de arquivos:
  - Abertura do arquivo (localização, alocação de buffer)
  - Leitura (disponibilização das informações do buffer para o programa)
  - Gravação (alteração de dados preexistentes ou acréscimo de novas informações)
  - Fechamento do arquivo (atualização das informações mantidas no buffer e liberação da área de memória utilizada)

Função para abertura de arquivos

- A função básica para abertura de um arquivo é:
  - `FILE *fopen (char* nome_arquivo, char* modo)`
- `nome_arquivo` – nome completo do arquivo incluindo diretório/subdiretório/nome.extensão
- modo

```
r     read      Modo leitura (O arquivo precisa existir)
w     write     Modo gravação (Gera um novo arquivo)
a     append    Modo gravação no final do arquivo
                (Adiciona num arquivo existente ou gera um novo)
t     text      Modo texto
b     binary    Modo binário
+               modificador dos modos r , w e  a  (r+, w+ e a+)
```

Função para abertura de arquivos

- EXEMPLOS

```
FILE *fp;
fp = fopen(“entrada.txt”,”rt”);
if (fp==NULL) {
   printf(“Erro na abertura do arquivo.\n”);
   exit(1); //aborta o programa
} //abre o arquivo de texto entrada.txt para leitura

FILE *arq;
arq = fopen(“saida.txt”,”wt”);
if (arq==NULL) {
  printf(“Erro na abertura do arquivo.\n”);
  exit(1); //aborta o programa
}//abre o arquivo de texto saida.txt para gravação
```

Função para fechamento de arquivos

- `int  fclose (FILE *fp);`
- O valor de retorno é `NULL` se o arquivo for fechado com sucesso ou a constante `EOF` (definida pela biblioteca) indicando a ocorrência de erro.

- EXEMPLOS:
  - `fclose(fp);`
  - `fclose(arq);`

Funções para leitura de arquivos em modo texto

- `int fscanf(FILE *fp, char* formato,...);`
  - Semelhante ao scanf, especifica, além do ponteiro para o arquivo, o formato dos dados a serem lidos. Retorna o número de informações lidas
- `int fgetc(FILE *fp);`
  - Esta função captura os dados caractere a caractere, até o final do arquivo (EOF).
- `char fgets(char* s, int n, FILE *fp);`
  - Lê a partir do arquivo uma sequência de caracteres, até que um '\n' seja encontrado.
  - int n deve ser especificado de tal modo que acomode o finalizador de string '\0'. Retorna NULL se não leu nada.

Funções para leitura de arquivos - exemplos

- `int fgetc(FILE *fp);`
- EXEMPLO

```
/*conta o numero de linhas de um arquivo*/
#include <stdio.h>
int main () {
    int c,nlinhas=0;
    FILE *arq;
    arq=fopen("entrada.txt", "rt"); //abre o arquivo
     if (arq==NULL) {
      printf("Não foi possível abrir o arquivo.\n");
      exit(1);
      }
     while((c=fgetc(arq))!=EOF) {
            if(c=='\n') nlinhas++;
     }
    fclose(arq); //fecha o arquivo
    printf("Numero de linhas = %d\n", nlinhas); /* exibe o resultado na tela */
return 0;
}
```

Funções para leitura de arquivos -exemplos

- `int fscanf(FILE *fp, char* formato, ...);`
- EXEMPLO:

```
/*conta o numero de linhas de um arquivo*/
#include <stdio.h>
int main () {
    int nl=0; char c;
    FILE *arq;
    arq=fopen(“entrada.txt”, “rt”); //abre o arquivo
    if (arq==NULL) {
      printf(“Não foi possível abrir o arquivo.\n”);
      exit(1);
    }
    while(fscanf(arq,”%c”,&c)==1) {
        if(c==‘\n’) nl++;
    }
   fclose(arq); //fecha o arquivo
   printf(“Numero de linhas = %d\n”, nl); /* exibe o resultado na tela */
   return 0;
}
```

Funções para leitura de arquivos -exemplos

- `char fgets(char* s, int n, FILE *fp);`
- EXEMPLO:

```
/*mostra linhas de arquivo de texto*/
#include <stdio.h>
int main () {
    char c[41];
    FILE *arq;
    arq=fopen("entrada.txt", "rt"); //abre o arquivo
   if (arq==NULL) {
      printf("Não foi possível abrir o arquivo.\n");
      system("pause");
      exit(1);
   }
    while(fgets(c, 40, arq)!= NULL) {
          printf("%s",c); // ou puts(c);
  }
  fclose(arq); //fecha o arquivo
}
```

Funções para gravação de arquivos de modo texto

- `int fprintf( FILE* fp, char* formato, ...);`
  - Análoga à função printf grava as informações especificadas no arquivo
- `int fputc(int c, FILE* fp);`
  - Análoga à função putc grava no arquivo especificado a informação caractere a caractere.
- `char* fputs(char* s, FILE* fp);`
  - Análoga à função puts grava no arquivo especificado a informação string a string.

Funções para gravação de arquivos -exemplos

- `int fprintf( FILE* fp, char* formato, ...);`
- Exemplo:

```
#include <stdio.h>
//grava a frase no arquivo tipo texto
int main(){
    FILE *fp;
    char frase[]=“Exemplo de fprintf";
    fp=fopen("saida.txt","wt");
    for(int i=0; frase[i]!='\0';i++)
       fprintf(fp,"%c",frase[i]);
    fclose(fp);
    printf("arquivo gravado\n");
    system("pause");
    return 0;
}
```

Funções para gravação de arquivos -exemplos

- `int fputc( int  c, FILE* fp);`
- Exemplo:

```
#include <stdio.h>
int main(){
    FILE *arq;
    char frase[]=“Exemplo de fputc";
    arq=fopen("saida2.txt","wt");
    for(int i=0; frase[i]!='\0';i++)
       fputc(frase[i],arq);
    fclose(arq);
    return 0;
}
```

Funções para gravação de arquivos -exemplos

- `char* fputs( char* s, FILE* fp);`
- Exemplo:

```
#include <stdio.h>
int main(){
    FILE *fp;
    char frase[]="Programação Avançada";
    char endline='\n';
    fp=fopen("saida3.txt","wt");
    fputs(frase,fp);
    fputc(endline,fp);
    fputs(frase,fp);
    fclose(fp);
    return 0;
}
```

Arquivos em modo Binário

- Servem para salvar(recuperar) as informações tais como se encontram na memória principal;
- A vantagem neste tipo de utilização é o manuseio de grandes quantidades de dados de forma mais eficiente.
- Um arquivo gravado em binário permite ainda a recuperação randômica de parte da informação (fseek)

Arquivos em modo Binário

- Função para leitura:
  - `int fread (void* p, int tam, int num, FILE *fp);`
    - `void* p` é o endereço de memória que contém a informação a ser gravada
    - `int tam` é o tamanho em bytes de cada elemento
    - `int num` é o número de elementos
    - `FILE *fp` é o ponteiro para o arquivo
- Função para gravação
  - `int fwrite (void* p, int tam, int num, FILE *fp);`
- Função para posicionamento
  - `int fseek (FILE *fp, long offset, int origem);`

Arquivos em modo Binário – Exemplo 1

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
typedef struct {
     int num;
     char nome[10];
     float nota;
     }aluno;
int main(){
    FILE *fp;
    aluno a,b;
    a.num=100;
    strcpy(a.nome, "Aluno");
    a.nota=9.5;
    fp=fopen("saidaBin.bin","wb");//gravação binario
    fwrite(&a, sizeof(aluno),1,fp);
    fclose(fp);
    fp=fopen("saidaBin.bin","rb");//leitura binário
    fread(&b, sizeof(aluno),1,fp);
    printf("\nDados gravados:\nNum: %d, Nome: %s, Nota= %.1f\n\n",b.num,b.nome,b.nota);
    system("pause");
    return 0;
}
```

Arquivos em modo Binário – Exemplo 2

```
#include <stdio.h>
#include <stdlib.h>
//uso do fseek

int main(){
    FILE *fp;
    int num[20];
    int vet[5];
    for(int i=0;i<20;i++)
       num[i]=i;
    fp=fopen("vetBin.bin","wb");//gravação binario (gera novo arq)
    fwrite(num, sizeof(int),20,fp);
    fclose(fp);
    fp=fopen("vetBin.bin","rb");
    fseek(fp,-5*sizeof(int),SEEK_END); //SEEK_CUR ou SEEK_SET
    fread(vet, sizeof(int),5,fp);
    printf("\nVetor resultante\n");
    for(int i=0; i<5;i++)
        printf("%d\t",vet[i]);
    printf("\n\n");
    system("pause");
    return 0;
}
```

Exercício1

- Faça um programa que decodifique um determinado texto gravado em um arquivo, a partir da seguinte tabela de substituição de caracteres, gerando um novo arquivo:

CARACTER EXISTENTE	SUBSTITUIR POR
A	Z
E	Y
I	X
O	W
U	*

Exercício 1 - Resolução

```
//copia o conteudo do arquivo tmp.txt em tmpCopy.txt, com alteracoes
#include<stdio.h>
#include<stdlib.h>
main(){
     char c;
     FILE *ent, *sai;
     ent = fopen("tmp.txt", "r");
     if(ent){
        sai = fopen( "tmpCopy.txt", "w" );
        c = fgetc( ent );
        while( c != EOF ) {
           switch (c) {
                 case 'A': c='Z';break;
                 case 'E': c='Y';break;
                 case 'I': c='X';break;
                 case 'O': c='W';break;
                 case 'U': c='*';break;
           }
           fputc( c, sai );
           c = fgetc( ent );
        }
        fclose( ent );
        fclose( sai );
      }
      else printf ("deu erro na abertura do arquivo tmp.txt\n");
      system("pause");
}
```

Exercício 2

- Considere um arquivo tipo texto já gravado. Gere um novo arquivo com uma nova linha, com dados lidos do teclado, entre a terceira e a quarta linha deste texto. Mostre o conteúdo do arquivo original e do novo.

Exercício 2 - Resolução

```
//copia o conteudo do arquivo original.txt em copia.txt
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#define MAX 150
main(){
     int i;
     char linha[MAX];
     FILE *entrada, *saida;
     entrada = fopen( "original.txt", "r" );
     if(entrada){
        saida = fopen( "copia.txt", "w" );
        for (i=1;i<4;i++){
            fgets(linha, MAX, entrada);//le as primeiras linhas
            fputs(linha, saida );     // copia para o arquivo de saida
        }
        printf("Informe o conteudo da linha a ser inserida: ");
        gets(linha);
        strcat(linha,"\n");
        fputs(linha, saida ); // escreve a linha adicionada
```

Exercício 2 - Resolução

```
// copia o resto do arquivo, sem alteracao
        fgets(linha, MAX, entrada);
        while(!feof(entrada) ) {
            fputs( linha, saida );
            fgets(linha, MAX, entrada);
        }
        fputs( linha, saida ); //para escrever a ultima linha (do EOF)
        fclose( entrada );
        fclose( saida );
      }
      else printf ("deu erro na abertura do arquivo original.txt\n");
      system("pause");
}
```