Fundamentos de Programação 1

Linguagem C
“Arquivos Binários”.

Slides 19

Prof. SIMÃO

Jean Marcelo SIMÃO

Arquivo de Escrita e Leitura

Binário
(Dados ‘Homogêneos’)

Arquivo (Binário) de Escrita e Leitura

```c
/* copia um arquivo */
#include <stdio.h>
#include <stdlib.h>

int main()
{
   FILE *entrada, *saida;

   char ch;
   char nomearq1[300], nomearq2[300];

   printf ( "Programa para copiar arquivos. \n" );

   // Arquivo fonte
   printf ("Digite o nome do arquivo fonte: \n");
   gets (nomearq1);

   entrada = fopen (nomearq1, "w+b");

   if ( entrada == NULL )
   {
      printf ( "O arquivo-fonte nao pode ser aberto (1).\n" );
      system ("Pause");
      exit (1);
   }

   printf( "Digite o conteudo do arquivo-fonte. \n" );
   do
   {
      ch = getchar();
      fputc ( ch, entrada );
   }while (ch != '\n' );
```

```c
   // Reposiciona o indicador de posiccao no inicio no arquivo.
   rewind ( entrada );

   // Arquivo destino
   printf ("Digite o nome do arquivo destino: \n");
   gets ( nomearq2 );

   saida = fopen ( nomearq2, "w+b");
   if ( saida == NULL )
   {
      printf ("O arquivo-destino nao pode ser aberto.");
      system("Pause");
      exit(1);
   }

   // Esse codigo copia de fato o arquivo.

   while ( !feof ( entrada ) )
   {
        ch = fgetc (entrada) ;
        if ( !feof ( entrada ) )
        {
           fputc (ch, saida);
        }
   }

   fclose (entrada);
   fclose (saida);

   system ( "Pause" );
   return 0;
}
```

Arquivo de Escrita e Leitura

Binário – fwrite & fread
(Dados Heterogêneos)

```c
/* Escreve alguns dados nao caracteres em um arquivo em disco e le de volta. */
#include <stdio.h>
#include <stdlib.h>

int main()
{
  FILE*   fp;
  double  d = 12.23;
  int     i = 101;
  long    l = 123023L;

  fp = fopen ( "TesteBin", "wb+");

  if ( fp == NULL )
  {
     printf ( "O arquivo nao pode ser aberto. \n" );
     system ( "Pause" );
     exit (1);
  }

  fwrite ( &d, sizeof ( double ), 1, fp) ;
  fwrite ( &i,  sizeof ( int ),         1, fp) ;
  fwrite ( &l,  sizeof ( long ),     1, fp) ;

  rewind ( fp );

  fread ( &d, sizeof ( double ), 1, fp);
  fread ( &i,  sizeof ( int ),        1, fp);
  fread ( &l,  sizeof ( long ),     1, fp);

  printf ("\n");
  printf ("%f %d %ld \n", d, i, l);
  fclose ( fp );

  system ("Pause");
  return 0;
}
```

Exemplo

Lista Postal

```c
/* Um programa de lista postal muito simples */

#include <stdio.h>
#include <ctype.h>
#include <stdlib.h>
#include <string.h>

#define TAM 2

struct Elemento
{

      char nome [100];
      char rua [100];
      char cidade [100];
      char estado [2];
      char cep [10];

};

struct Elemento  Lista [TAM];

char menu ();

void inicia_lista ();

void cadastra ();

void mostra ();

void salva ();

void carrega ();
```

```c
int main()
{
   char escolha;
   inicia_lista();

   for ( ;; )
   {
      escolha = menu();
      switch (escolha)
      {
         case 'c':
         case 'C':   { cadastra();  }  break;

         case 'm':
         case 'M':   { mostra();    } break;

         case 's':
         case 'S':   { salva();       } break;

         case 'a':
         case 'A':  { carrega();    } break;

         case 't':
         case 'T':  { exit (0 );       } break;

         default : { printf ( "Opcao invalida. \n" );  }

      printf ( "\n \n \n" );
   }
   system ( "Pause" );
   return 0;
}
```

```c
/* Um programa de lista postal muito simples */

#include <stdio.h>
#include <ctype.h>
#include <stdlib.h>
#include <string.h>

#define TAM 2

struct Elemento
{

      char nome [100];
      char rua [100];
      char cidade [100];
      char estado [2];
      char cep [10];

} Lista [TAM];

char menu ();

void inicia_lista ();

void cadastra ();

void mostra ();

void salva ();

void carrega ();
```

```c
int main()
{
   char escolha;
   inicia_lista();

   for ( ;; )
   {
      escolha = menu();
      switch (escolha)
      {
         case 'c':
         case 'C':   { cadastra();  }  break;

         case 'm':
         case 'M':   { mostra();    } break;

         case 's':
         case 'S':   { salva();       } break;

         case 'a':
         case 'A':  { carrega();    } break;

         case 't':
         case 'T':  { exit (0 );       } break;

         default : { printf ( "Opcao invalida. \n" );  }

      printf ( "\n \n \n" );
   }
   system ( "Pause" );
   return 0;
}
```

```c
char menu()
{
   printf ("\n \n \n");
   char opcao;

   printf ( " (C)adastrar.    \n"  );
   printf ( " (M)ostrar.      \n"  );
   printf ( " C(A)arregar.    \n"  );
   printf ( " (S)alvar.       \n"   );
   printf ( " (T)erminar.     \n"  );

   fflush ( stdin );
   scanf ( "%c", &opcao );

   return opcao;
}
```

```c
void mostra()
{
    printf ("\n \n \n");

    register int t;

    for( t = 0; t < TAM; t++ )
    {
       if (*Lista[t].nome )
       {
          printf ( "%s \n", Lista[t].nome);
          printf ( "%s \n", Lista[t].rua);
          printf ( "%s \n", Lista[t].cidade);
          printf ( "%s \n", Lista[t].estado);
          printf ( "%s \n", Lista[t].cep);
       }
       printf ("\n");
    }
}
```

```c
void inicia_lista()
{
    register int t;
    for (t = 0; t < TAM; t++)
    {
       *Lista[t].nome = '\0';      // Lista[t].nome[0] = '\0';
    }
}
```

```c
void cadastra ()
{
    printf ("\n \n \n");

    for ( int i = 0; i < TAM; i++ )
    {
       printf ( "Nome: \n" );
       fflush ( stdin );
       gets ( Lista[i].nome );

       printf ( " Rua: \n" );
       fflush ( stdin );
       gets ( Lista[i].rua );

       printf ( "Cidade: \n" );
       fflush (stdin);
       gets ( Lista[i].cidade );

       printf ( "Estado: \n" );
       fflush ( stdin );
       gets ( Lista[i].estado );

       printf ( "CEP: \n" );
       fflush ( stdin );
       gets ( Lista[i].cep );
    }
}
```

```
void salva ()
{

    printf ("\n \n \n");

    FILE *fp;
    int result;

    fp = fopen ("cadastro", "wb");

    if ( fp == NULL )
    {

      printf ( "O arquivo nao pode ser aberto. \n" );
      return;

    }

    for ( int i = 0; i < TAM; i++ )
    {

       if ( *Lista[i].nome )
       {

          result = fwrite ( &Lista[i], sizeof ( struct Elemento ), 1, fp );

          if ( result != 1 )
          {
             printf ( "Erro de escrita no arquivo. \n" );
          }

       }

    }

    fclose (fp);

}
```

```
void carrega ()
{

    printf ("\n \n \n");

    FILE  *fp;
    int resp;

    fp = fopen ( "cadastro", "rb“ );

    if ( fp == NULL )
    {

      printf ( "O arquivo nao pode ser aberto. \n" );
      return;

    }

    inicia_lista ();
    for ( int i = 0; i < TAM; i++ )
    {

       resp = fread ( &Lista[i], sizeof (struct Elemento), 1, fp );

       if ( resp != 1 )
       {

          if ( feof ( fp ) )
          {
            break;
          }
          printf ( " Erro de leitura no arquivo. \n" );
       }

    }
    fclose ( fp );

}
```

Exercício

- Naquele exercício de estruturas e ponteiros... fazer com que os ‘registros’ (informações) do funcionários possam ser gravados em um arquivo, bem como recuperados do arquivo.