fclose

Cierra un archivo.

Prototipo

int fclose (FILE *archivo);

Descripción

Retorna el valor cero (0) si resulto satisfactoriamente, de lo contrario retorna EOF si cualquier error es detectado.

Ejemplo

```
#include <string.h>
#include <stdio.h>

int main()
{
   FILE *arch;
   char buf[11] = "0123456789";

   /* crea un archivo conteniendo 10 bytes */
   arch = fopen("EJEMPLO.DAT", "w");
   fwrite(&buf, strlen(buf), 1, arch);

   /* cierra el archivo */
   fclose(arch);
}
```

feof

Macro que retorna un valor distinto de cero si el fin de archivo es encontrado.

Prototipo

```
int feof (FILE *archivo);
```

Ejemplo

```
#include <stdio.h>

int main()
{
  FILE *archivo;

  /* abre un archivo para lectura */
  archivo = fopen("EJEMPLO.DAT", "r");

  /* lee un carácter del archivo */
  fgetc(archivo);

  /* chequear para EOF */
  if (feof(archivo)) printf("Fin de archivo\n");

  /* cierra el archivo */
  fclose(archivo);
}
```

ferror

Macro que retorna un valor distinto de cero si un error ha ocurrido en la última operación de archivo.

Prototipo

```
int ferror (FILE *archivo);
```

Ejemplo

```
#include <stdio.h>

 int main()
 {
   FILE *archivo;

   /* abre un archivo para escritura */
   archivo = fopen("EJEMPLO.DAT", "w");

   /* Forzar para una ocurrencia de un error */
   (void) getc(archivo);

   /* verificar un error en la operación de archivo */
   if ferror(archivo)
   {
     /* imprime un mensaje de error */
     printf("Error leyendo de EJEMPLO.DAT\n");

     /* Resetea el error */
     clearerr(archivo);
   }

   fclose(archivo);
 }
```

fgetpos

Recupera por medio del parámetro (pos) la posición actual en la que se encuentra el puntero de archivo.

Prototipo

```
int fgetpos (FILE *archivo, long int *pos);
```

Descripción

Retorna el valor cero (0) si todo resulta satisfactoriamente, de lo contrario un valor distinto de cero si ocurre algún error. En el parámetro (pos) es retornado la posición actual del puntero de archivo.

Ejemplo

```
#include <string.h>
#include <stdio.h>

 int main()
 {
   FILE *archivo;
   char string[] = "Esto es una prueba";
   fpos_t filepos;

   /* abre un archivo para actualizar */
   archivo = fopen("EJEMPLO.DAT", "w+");

   /* escribe una cadena en el archivo */
   fwrite(string, strlen(string), 1, archivo);

   /* recupera la posición del puntero de archivo*/
   fgetpos(archivo, &filepos);
   printf("El puntero de archivo esta en el byte\%ld\n", filepos);

   fclose(archivo);
 }
```

fgets

Realiza la lectura de la cadena (s) desde un archivo, hasta un máximo de (n) caracteres.

Prototipo

```
char *fgets (char *s, int n, FILE *archivo);
```

Descripción

Si todo resulta satisfactoriamente, retorna la cadena almacenada en el parámetro (s), NULL ó fin de archivo si ocurre algún error en la operación. El parámetro (n) indica el número máximo de caracteres a leer del archivo.

Ejemplo

```
#include <string.h>
#include <stdio.h>

int main()
{
   FILE *archivo;
   char string[] = "Esto es una prueba";
   char msg[20];

   archivo = fopen("EJEMPLO.DAT", "w+");

   /* escribe una cadena en el archivo */
   fwrite(string, strlen(string), 1, archivo);

   /* posiciona el puntero al principo del archivo*/
   fseek(archivo, 0, SEEK_SET);

   /* lee una cadena del archivo */
   fgets(msg, strlen(string)+1, archivo);

   printf("%s", msg);

   fclose(archivo);
}
```

fopen

Abre un archivo, según el modo determinado.

Prototipo

FILE *fopen (const char *nombre, const char *modo);

Descripción

Retorna un puntero si el archivo se pudo abrir satisfactoriamente, en caso contrario retorna NULL. El parámetro (modo) debe ser alguno de los que a continuación se presentan:

Tipo	Descripción
r	Abre sólo para lectura
w	Crea para escritura, sobreescribe archivo existente
a	Agrega. Abre para escritura al final del archivo o crea archivo para escribir.
+	Agrega simbolo para actualizar lectura/escritura
b	Abre en modo binario
t	Abre en modo texto

Ejemplo

```
 #include <stdio.h>
int  main()  {
   FILE *in, *out;

   if ((in = fopen("\\AUTOEXEC.BAT", "rt")) == NULL)  {
      fprintf(stderr, "No puedo abrir archivo de entrada\n");
      return 1;  }

   if ((out = fopen("\\AUTOEXEC.BAK", "wt")) == NULL)  {
      fprintf(stderr, "No puedo abrir archivo de salida\n");
      return 1;
   }
   while (!feof(in)) fputc(fgetc(in), out);
   fclose(in);
   fclose(out);
return 0;

 }
```

fprintf

Realiza la escritura de una salida formateada hasta un archivo.

Prototipo

int fprintf (FILE *archivo, const char *formato[, argument, ...]);

Descripción

Utiliza el mismo formato que la función printf, pero fprintf envía la salida a un archivo. fprint retorna el número de bytes de salida. Si ocurre algún error, esta retorna EOF.

Ejemplo

```
#include <stdio.h>

int main()
{
  FILE *archivo;
  int i = 100;
  char c = 'C';
  float f = 1.234;

  /* abre un archivo para actualizar */
  archivo = fopen("EJEMPLO.DAT", "w+");

  /* escribe alguna información en el archivo */
  fprintf(archivo, "%d %c %f", i, c, f);

  /* cierra el archivo */
  fclose(archivo);
}
```

fputs

Realiza la escritura de una cadena (s) hasta un archivo.

Prototipo

```
int fputs (const char *s, FILE *archivo);
```

Descripción

Si la escritura es completada de forma satisfactoria, retorna el último cáracter escrito; en caso contrario retorna el valor EOF.

fread

Realiza la lectura de información por medio del parámetro (ptr) de (n) bloques de (size) bytes cada uno desde un archivo.

Prototipo

unsigned int fread (void *ptr, unsigned int size, unsigned int n, FILE *archivo);

Descripción

Realiza la lectura de (n) bloques de (size) bytes cada uno. Retorna el número de bloques (no bytes) actualmente leídos.

Ejemplo

```
#include <string.h>
#include <stdio.h>

main()  {
  FILE *archivo;
  char msg[] = "Esto es una prueba";
  char buf[20];

  if ((archivo = fopen("EJEMPLO.DAT", "w+")) == NULL)  {
    fprintf(stderr, "No puedo abrir archivo de salida\n");
    return 1;
  }

  /* escribir alguna información en el archivo */
  fwrite(msg, strlen(msg)+1, 1, archivo);

  /* mover el puntero hasta el principio del archivo */
  fseek(archivo, SEEK_SET, 0);

  /* realiza la lectura de la información  e imprime la información */
  fread(buf, strlen(msg)+1, 1, archivo);
  printf("%s\n", buf);

  fclose(archivo);  }
```

fseek

Mueve el puntero de archivo hasta una posición determinada.

Prototipo

int fseek (FILE *archivo, long desplazamiento, int posición);

Descripción

El parámetro (desplazamiento) es la nueva posición relativa a partir del parámetro (posición). El parámetro (posición) puede tomar uno de los siguientes valores:

SEEK_SET	Desplazamiento a partir del principio de archivo
SEEK_CUR	Desplazamiento a partir de la posición actual en la que se encuentra el puntero
SEEK_END	Desplazamiento a partir del fin de archivo

Retorna un valor cero (0) si tuvo éxito, o un valor distinto de cero si se produjo algún error.

Ejemplo

```
#include <stdio.h>

int main() {
  FILE *archivo;

  archivo = fopen("PRUEBA.TXT", "w+");
  fprintf(archivo, "Esto es una prueba");
  printf("Tamaño de PRUEBA.TXT is %ld bytes\n", filesize(archivo));
}

long filesize (FILE *archivo)
{
  long curpos, length;
  curpos = ftell(archivo);
  fseek(archivo, 0L, SEEK_END);
  length = ftell(archivo);
  fseek(archivo, curpos, SEEK_SET);
  return (length);
}
```

fsetpos

Posiciona el puntero en la posición (pos) del archivo.

Prototipo

int fsetpos (FILE *archivo, const long int *pos);

Descripción

La nueva posición del puntero de archivo almacenada en el parámetro (pos), es obtenida previamente mediante una llamada a la función fgetpos. Retorna un valor cero (0) si tuvo éxito, o un valor distinto de cero si se produjo algún error.

Ejemplo

```
#include <stdlib.h>
#include <stdio.h>

int main()
{
  FILE *archivo;
  fpos_t filepos;

  /* abrie un archivo para actualización */
  archivo = fopen("EJEMPLO.DAT", "w+");

  /* recupera la posición actual del puntero de archivo */
  fgetpos(archivo, &filepos);

  /* escribe alguna información en el archivo */
  fprintf(archivo, "Esto es una prueba");

  /* imprime actual posición del puntero de archivo */
  showpos(archivo);

  /* selecciona una posición del puntero de archivo, e imprime esta */
  if (fsetpos(archivo, &filepos) == 0) showpos(archivo);
  else
  {
    fprintf(stderr, "Error actualizando puntero de archivo\n");
    exit(1);
```

```
  }
  fclose(archivo);
}

void showpos (FILE *archivo)
{
  fpos_t pos;

  /* imprime la actual posición del puntero de archivo */
  fgetpos(archivo, &pos);
  printf("Posición de archivo: %ld\n", pos);
}
```

ftell

Retorna la actual posición del puntero de archivo.

Prototipo

long int ftell (FILE *archivo);

Descripción

Si la operación resulta satisfactoria retorna la posición actual del puntero de archivo, en caso contrario retorna –1l.

Ejemplo

```
#include <stdio.h>

int main()
{
  FILE *archivo;

  archivo = fopen("PRUEBA.TXT", "w+");
  fprintf(archivo, "Esto es una prueba");
  printf("El puntero de archivo se encuentra en el byte %ld\n", ftell(archivo));
}
```

fwrite

Realiza la escritura hasta un archivo.

Prototipo

unsigned int fwrite (const void *ptr, unsigned int size, unsigned int n, FILE *archivo);

Descripción

Escribe (n) bloques de (size) bytes cada uno. Retorna el número de bloques (no bytes) actualmente escritos.

Ejemplo

```
#include <stdio.h>

struct mystruct {
  int i;
  char ch;
};

int main()
{
   FILE *archivo;
   struct mystruct s;

   /* abrir archivo PRUEBA.TXT */
   if ((archivo = fopen("PRUEBA.TXT", "wb")) == NULL)
   {
      fprintf(stderr, "No puedo abrir archivo de salida\n");
      return 1;
   }
   s.i = 0;
   s.ch = 'A';

   /* escribe la estructura (s) en el archivo */
   fwrite(&s, sizeof(s), 1, archivo);
   fclose(archivo);
}
```

remove

Macro que remueve un archivo.

Prototipo

int remove (const char *nombre);

Descripción

Esta macro realiza una llamada a la función unlink. Retorna un valor cero (0) si tuvo éxito, o un valor –1 si se produjo algún error.

Ejemplo

```
#include <stdio.h>

int main()
{
  char archivo[80];

  /* pregunta al usuario el nombre del usuario */
  printf("Archivo a eliminar: ");
  gets(archivo);

  /* elimina el archivo */
  if (remove(archivo) == 0) printf("Eliminado %s.\n",archivo);
  else perror("remove");
}
```

rename

Renombra un archivo.

Prototipo

int rename (const char *oldname, const char *newname);

Descripción

Retorna un valor cero (0) si tuvo éxito, o un valor –1 si se produjo algún error.

Ejemplo

```
#include <stdio.h>

int main()
{
  char oldname[80], newname[80];

  printf("Archivo a renombrar: ");
  gets(oldname);
  printf("Nuevo nombre: ");
  gets(newname);

  /* Renombrar el archivo */
  if (rename(oldname, newname) == 0)
    printf("Renombrado %s to %s.\n", oldname, newname);
  else
    perror("rename");
}
```

rewind

Reposiciona el puntero de archivo al principio del mismo.

Prototipo

```
void rewind (FILE *archivo);
```

Ejemplo

```
#include <stdio.h>
#include <dir.h>

int main()
{
  FILE *arch;
  char *fname = "PRUEBA.TXT", *newname, first;

  newname = mktemp(fname);
  arch = fopen(newname,"w+");
  fprintf(arch,"abcdefghijklmnopqrstuvwxyz");
  rewind(arch);
  fscanf(arch,"%c",&first);
  printf("El primer carácter es : %c\n",first);
  fclose(arch);
  remove(newname);
}
```

unlink

Elimina un archivo.

Prototipo

int unlink (const char *nombre);

Descripción

If the filename archivo has the read-only attribute, unlink will fail. Call chmod first to change the archivo's attribute.

Retorna un valor cero (0) si tuvo éxito, o un valor de –1 si se produjo algún error.

Ejemplo

```
#include <stdio.h>
#include <io.h>

int main()
{
  FILE *arch = fopen("PRUEBA.TXT","w");
  int status;

  fprintf(arch,"Hola");

  status = access("PRUEBA.TXT",0);
  if (status == 0) printf("Archivo existe\n");
  else printf("Archivo no existe\n");

  fclose(arch);
  unlink("PRUEBA.TXT");
  status = access("PRUEBA.TXT",0);
  if (status == 0) printf("Archivo existe\n");
  else printf("Archivo no existe\n");
}
```