***Libreria standard di I/O***, le sue funzioni individuano il file su cui fare operazioni di I/O attraverso uno stream (flusso di dati) e non più attraverso il file descriptor. Quando si crea o si apre un file con le funzioni di standard I/O, si dice che si associa uno stream al file, la struttura contiene tutte le info per trattare lo stream.

Ogni processo ha 3 *stream* predefiniti che sono individuati attraverso i puntatori:

* ***stdin*** che punta allo *standard input*;
* ***stdout*** che punta allo *standard output*;
* ***stderr*** che punta allo *standard error*.

Lo scopo del ***buffering*** è quello di usare il minimo numero di chiamate a read e write, le funzioni della libreria standard di I/O utilizzano 3 tipi di buffering :

* ***Fully buffered***, le operazioni di I/O avvengono quando il buffer è pieno, il termine ***flush*** (far scorrere) descrive la scrittura di un buffer standard di I/O (es *stream input* e *output*).
* ***Line buffered***, le operazioni di I/O avvengono quando è incontrato il carattere di *newline* sull'input o sull’output.
* ***Unbuffered***, la libreria standard di I/O non bufferizza i caratteri e le operazioni di I/O avvengono immediatamente (esempio lo *stream error*).

**MODIFICA DEL BUFFERING**:

* void ***setbuf*** (FILE \**fp*, char \**buf* );
* int ***setvbuf*** (FILE \**fp*, char \**buf*, int *mode*, size\_t *size*);

*Descrizione*: Devono essere chiamate dopo che lo stream è stato aperto e prima di ogni altra operazione su di esso.

*Restituiscono*: 0 se OK, diverso da 0 in caso di errore.

**FUNZIONI FLUSH:**

* int ***fflush***(FILE \**fp*);

Descrizione: scrive il contenuto del buffer sul file puntato da *fp*, forzando il flush di uno stream.

Restituisce: 0 se OK, *EOF* in caso di errore.

* int ***fflush***(*NULL*) = Effettua il flush di tutti gli stream aperti!

**APRIRE UNO STREAM:**

* FILE \****fopen***(const char \**pathname*, const char \**type*);

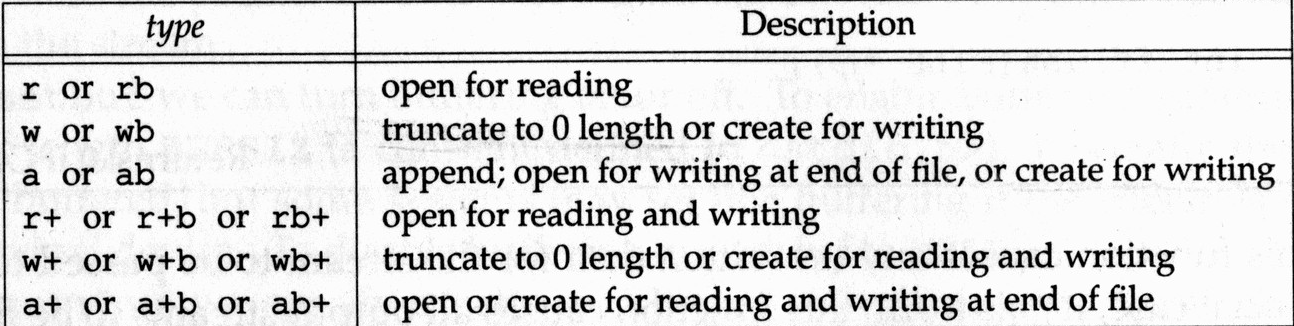
*Descrizione*: apre il file pathname.

* FILE \****freopen***(const char \**pathname*, const char \**type*, FILE \**fp*);

*Descrizione*: apre il file pathname sullo stream fp, chiudendo questo se era già aperto.

* FILE \****fdopen***(int *fd*, const char \**type*);

*Descrizione*: prende un file descriptor (che è stato ottenuto per esempio con una open) e gli associa uno standard I/O stream.



**CHIUDERE UNO STREAM:**

* int ***fclose***(FILE \**fp*);

*Descrizione*: chiude uno *stream* aperto.

*Restituisce*: 0 se OK, *EOF* in caso di errore.

**POSIZIONAMENTO DI UNO STREAM:**

* long ***ftell***(FILE \**fp*);

*Restituisce*: l'indicatore della posizione corrente (misurato in byte) se OK, -1 su errore.

* long ***fseek***(FILE \**fp*, long *offset*, int *whence*); /\* simile a lseek \*/

Restituisce: 0 se OK, diverso da 0 su errore.

* void ***rewind***(FILE \**fp*); /\* lo stream è settato all’inizio del file \*/
* int ***fgetpos***(FILE \**fp*, fpos\_t \**pos*);
* int ***fsetpos***(FILE \**fp*, const fpos\_t \**pos*);

*Descrizione*: fgetpos (fsetpos) pone nell’oggetto (prende dall’oggetto) puntato da pos l’indicatore della posizione del file fp.

*Restituiscono*: 0 se OK, diverso da 0 su errore.

**PRELEVARE UN FILE DESCRIPTOR:**

* int ***fileno***(FILE \**fp*);

*Restituisce*: il file descriptor associato allo stream.