File I/O

- File is a stream of characters or sequence of bytes
- Files are used for persistent or permanent storage
- In C, accessing a file is not that straightforward, there are functions available in the stdio.h header file.
- With the help of standard I/O function we can read and write the content from a file.

To open a file

- To open a file fopen function will be used and prototype of that function is:

  FILE *fopen(const char *path, const char *mode);

  where, → *path - name of a file/ file along path enclosed within double quotes
  e.g., "text.txt" , "/usr/desktop/text.txt", "c:/folder/text.txt"

  → *mode - mode of file, which can be any one from below list

  - **"r" mode**
    - read only mode
    - can only read information from a file / only read operation
    - If file exists, then it will open the file Else returns a NULL
    - If success -> returned pointer will points to first character of the file points to beginning of the file

  - **"w" mode**
    - Write only mode
    - can only write information to a file / only write operation
    - If file exists, then it will open the file and can write the info Else it will create a new file in the given path
    - If success –
      - File pointer will points to first character of the file
      - If already file contains the information then write mode will replace the previous contents with new contents

  - **"a" mode**
    - append mode
    - Append mode is used to add the content to a file.
    - If file exists, then it will open the file and add the info Else it will create a new file
    - If success –
    - File pointer will points to last character of the file
    - If file already contains the info then append mode will concat or add the previous contents with new contents

- **“r+” mode**
  - both read and write operation
  - If file exists it open the file else it will return NULL

- **“w+” mode**
  - both write and read operation
  - If file exists open the file else create a new file

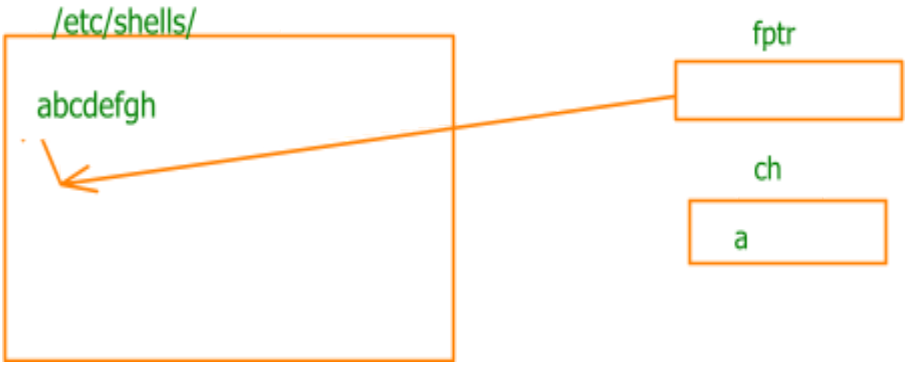
- **“a+” mode**
  - both read and write operation
  - If file exists open the file else create a new file
  - new content will be concatenated with previous

To close a file

- It's recommended to close the file pointer after all the file operation is completed.
- To close a file fclose or fcloseall will be used
- Function prototype:
  - fclose(filepointername)
  - fcloseall()
- fclose will close the particular file only, but fclose all will close all the file pointers opened in the program.

How to read content from a file?

- To read content from file, fgetc is used
- Fgetc will fetch a character from file
- To check the end of file, feof function will be helpful
  - feof(filepointer)
- fputc(ch,stdout) -> it will print a character on stdout/on screen



Error check in files

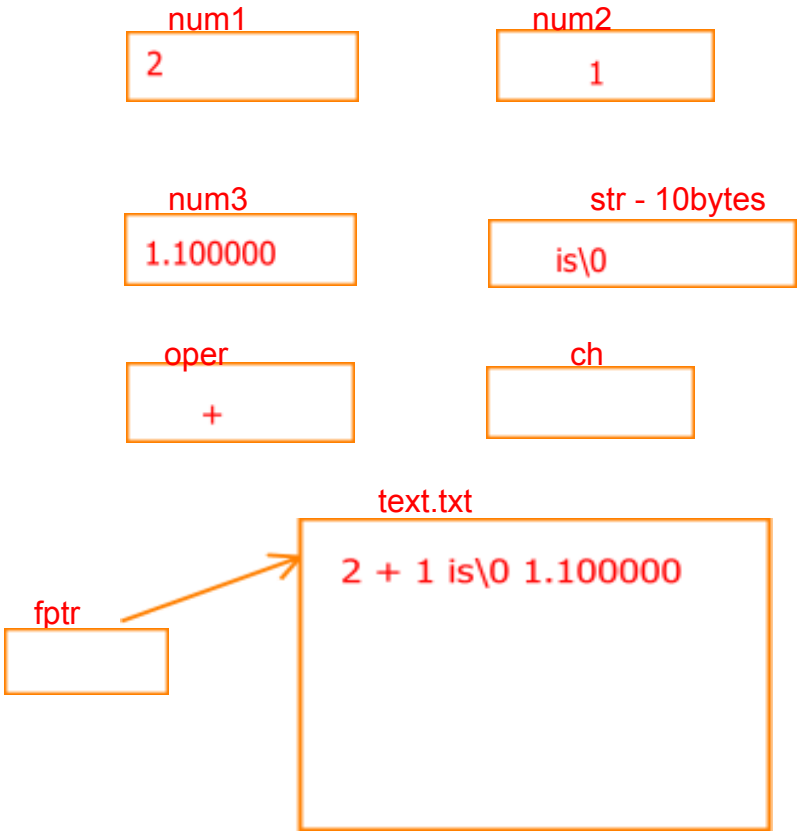
- Initially whenever a file pointer opens there is an error flag associated with it.
- That error flag will be 0 in the initial stage.
- After a certain FILE I?O operation, consider an example
    - fopen("file.txt","w");
    - File.txt is opened in write mode, if you try to read content from file it will be an error
    - In such scenario the error flag will be set to 1, to check the errors we use function

      ferror(FILE *)

To know the file pointer position in the file

- ftell(FILE *)
- By using ftell we can fetch the position of file pointer
- Provide some info/convey the message
- it will give information about where the file pointer is pointing
- fptr position will always be starting 0

006.c -- rewind(FILE *) --> make the file pointer to point to starting position



To move file pointer anywhere in the file:

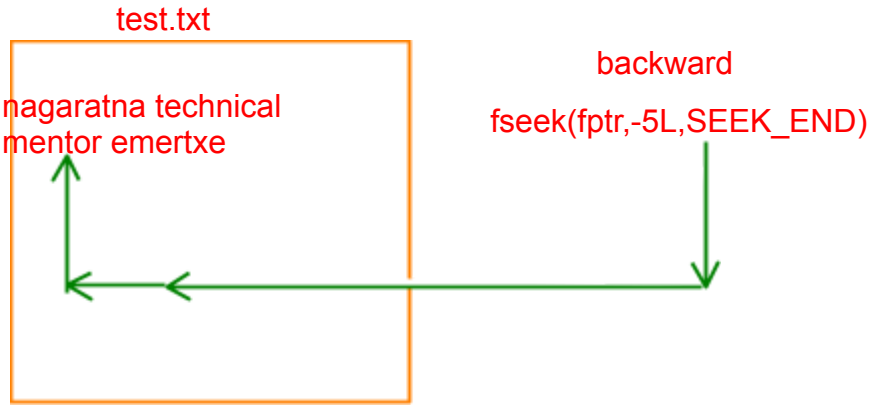
- Fseek function is used to move file pointer
- fseek(fptr, how_much, from_where)

how much u need to move --> 0L (long value), 10L, -10L

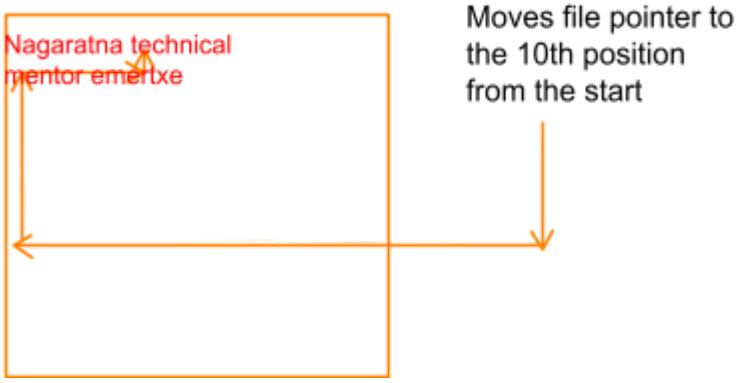
from where --> SEEK_SET --> file pointer will start from beginning of file

SEEK_END --> file pointer will be set to end of the file

SEEK_CUR --> file pointer will move from current position



fseek(fptr,10,SEEK_SET)



fseek(fptr,10,SEEK_CUR);

→ moves file pointer to the 10th position forward from current position

fseek(fptr,0L,SEET_SET); // equivalent to rewind() method

→ brings back the file pointer to the beginning of the position

Fwrite and fread

- These 2 functions are another way of reading or writing the content from a file.
- Both the function reads and writes the content as raw data which is non-human readable.
- Syntax:
  - size_t fwrite(const *ptr, size_t size, size_t nitems, FILE *stream)
  - size_t fread(const *ptr, size_t size, size_t nitems, FILE *stream)