C LIBRARY - <STDIO.H>

The **stdio.h** header defines three variable types, several macros, and various functions for performing input and output.

Library Variables

Following are the variable types defined in the header stdio.h −

S.N.	Variable & Description
1	**size_t** This is the unsigned integral type and is the result of the **sizeof** keyword.
2	**FILE** This is an object type suitable for storing information for a file stream.
3	**fpos_t** This is an object type suitable for storing any position in a file.

Library Macros

Following are the macros defined in the header stdio.h −

S.N.	Macro & Description
1	**NULL** This macro is the value of a null pointer constant.
2	**_IOFBF, _IOLBF** and **_IONBF** These are the macros which expand to integral constant expressions with distinct values and suitable for the use as third argument to the **setvbuf** function.
3	**BUFSIZ** This macro is an integer, which represents the size of the buffer used by the **setbuf** function.
4	**EOF** This macro is a negative integer, which indicates that the end-of-file has been reached.
5	**FOPEN_MAX** This macro is an integer, which represents the maximum number of files that the system can guarantee to be opened simultaneously.

6	**FILENAME_MAX** This macro is an integer, which represents the longest length of a char array suitable for holding the longest possible filename. If the implementation imposes no limit, then this value should be the recommended maximum value.
7	**L_tmpnam** This macro is an integer, which represents the longest length of a char array suitable for holding the longest possible temporary filename created by the **tmpnam** function.
8	**SEEK_CUR, SEEK_END,** and **SEEK_SET** These macros are used in the **fseek** function to locate different positions in a file.
9	**TMP_MAX** This macro is the maximum number of unique filenames that the function **tmpnam** can generate.
10	**stderr, stdin,** and **stdout** These macros are pointers to FILE types which correspond to the standard error, standard input, and standard output streams.

Library Functions

Following are the functions defined in the header stdio.h −

> *Follow the same sequence of functions for better understanding and to make use of* ***Try it****Onlinecompiler option, because file created in the first function will be used in subsequent functions.*

S.N.	Function & Description
1	int fclose*FILE * stream* Closes the stream. All buffers are flushed.
2	void clearerr*FILE * stream* Clears the end-of-file and error indicators for the given stream.
3	int feof*FILE * stream* Tests the end-of-file indicator for the given stream.
4	int ferror*FILE * stream* Tests the error indicator for the given stream.
5	

int fflush*FILE * stream*

Flushes the output buffer of a stream.

6

int fgetpos*FILE * stream, fpos_t * pos*

Gets the current file position of the stream and writes it to pos.

7

FILE *fopen*constchar * filename, constchar * mode*

Opens the filename pointed to by filename using the given mode.

8

size_t fread*void * ptr, size_tsize, size_tnmemb, FILE * stream*

Reads data from the given stream into the array pointed to by ptr.

9

FILE *freopen*constchar * filename, constchar * mode, FILE * stream*

Associates a new filename with the given open stream and same time closing the old file in stream.

10

int fseek*FILE * stream, longintoffset, intwhence*

Sets the file position of the stream to the given offset. The argument *offset* signifies the number of bytes to seek from the given *whence* position.

11

int fsetpos*FILE * stream, constfpos_t * pos*

Sets the file position of the given stream to the given position. The argument *pos* is a position given by the function fgetpos.

12

long int ftell*FILE * stream*

Returns the current file position of the given stream.

13

size_t fwrite*constvoid * ptr, size_tsize, size_tnmemb, FILE * stream*

Writes data from the array pointed to by ptr to the given stream.

14

int remove*constchar * filename*

Deletes the given filename so that it is no longer accessible.

15

int rename*constchar * old_filename, constchar * new_filename*

Causes the filename referred to, by old_filename to be changed to new_filename.

16

void rewind*FILE * stream*

Sets the file position to the beginning of the file of the given stream.

17

void setbuf*FILE * stream, char * buffer*

Defines how a stream should be buffered.

18

int setvbuf*FILE * stream, char * buffer, intmode, size_tsize*

Another function to define how a stream should be buffered.

19

FILE *tmpfile*void*

Creates a temporary file in binary update mode *wb+*.

20

char *tmpnam*char * str*

Generates and returns a valid temporary filename which does not exist.

21

int fprintf*FILE * stream, constchar * format, . . .*

Sends formatted output to a stream.

22

int printf*constchar * format, . . .*

Sends formatted output to stdout.

23

int sprintf*char * str, constchar * format, . . .*

Sends formatted output to a string.

24

int vfprintf*FILE * stream, constchar * format, va_listarg*

Sends formatted output to a stream using an argument list.

25

int vprintf*constchar * format, va_listarg*

Sends formatted output to stdout using an argument list.

26

int vsprintf*char * str, constchar * format, va_listarg*

Sends formatted output to a string using an argument list.

27

int fscanf*FILE * stream, constchar * format, . . .*

Reads formatted input from a stream.

28

int scanf*constchar ∗ format, . . .*

Reads formatted input from stdin.

29

int sscanf*constchar ∗ str, constchar ∗ format, . . .*

Reads formatted input from a string.

30

int fgetc*FILE ∗ stream*

Gets the next character *anunsignedchar* from the specified stream and advances the position indicator for the stream.

31

char *fgets*char ∗ str, intn, FILE ∗ stream*

Reads a line from the specified stream and stores it into the string pointed to by str. It stops when either $n - 1$ characters are read, the newline character is read, or the end-of-file is reached, whichever comes first.

32

int fputc*intchar, FILE ∗ stream*

Writes a character *anunsignedchar* specified by the argument char to the specified stream and advances the position indicator for the stream.

33

int fputs*constchar ∗ str, FILE ∗ stream*

Writes a string to the specified stream up to but not including the null character.

34

int getc*FILE ∗ stream*

Gets the next character *anunsignedchar* from the specified stream and advances the position indicator for the stream.

35

int getchar*void*

Gets a character *anunsignedchar* from stdin.

36

char *gets*char ∗ str*

Reads a line from stdin and stores it into the string pointed to by, str. It stops when either the newline character is read or when the end-of-file is reached, whichever comes first.

37

int putc*intchar, FILE ∗ stream*

Writes a character *anunsignedchar* specified by the argument char to the specified stream and advances the position indicator for the stream.

38

int putchar*intchar*

Writes a character *anunsignedchar* specified by the argument char to stdout.

39

int puts*constchar ∗ str*

Writes a string to stdout up to but not including the null character. A newline character is appended to the output.

40

int ungetc*intchar, FILE ∗ stream*

Pushes the character char *anunsignedchar* onto the specified stream so that the next character is read.

41

void perror*constchar ∗ str*

Prints a descriptive error message to stderr. First the string str is printed followed by a colon and then a space.