Learning C/C++ Step-By-Step - Page 12

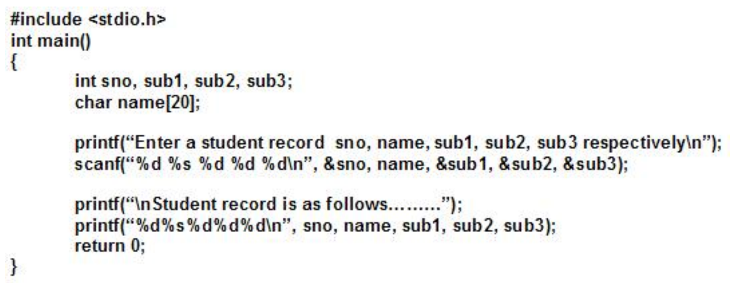
Submitted by ganesh35 (Contact Author) (Forums) on Wed, 2009-01-07 18:38. ::

12. Step-by-Step CorC++ --- C Programming - Files

File Handling

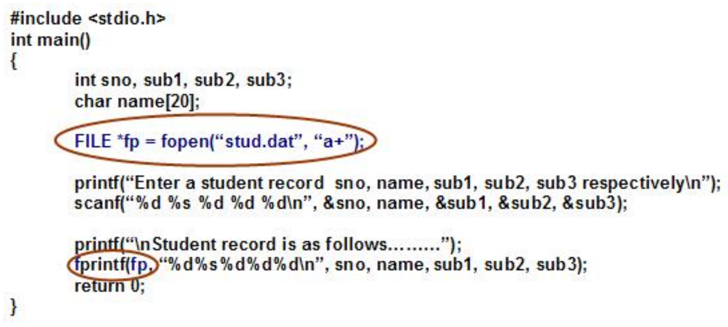
Introduction

Let's find the output of the following program.



```
#include <stdio.h>
int main()
{
        int sno, sub1, sub2, sub3;
        char name[20];

        printf("Enter a student record  sno, name, sub1, sub2, sub3 respectively\n");
        scanf("%d %s %d %d %d\n", &sno, name, &sub1, &sub2, &sub3);

        printf("\nStudent record is as follows.........");
        printf("%d%s%d%d%d\n", sno, name, sub1, sub2, sub3);
        return 0;
}
```

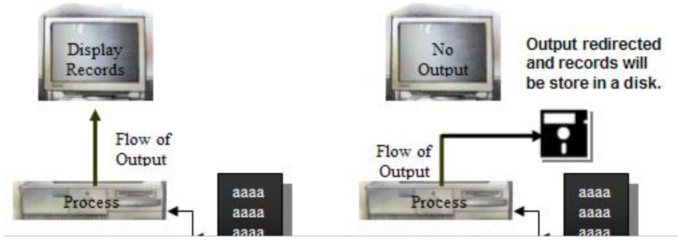
Yes, it accepts a record of student information and displays it.
Here is the same program, but included statements with a few modifications.



```
#include <stdio.h>
int main()
{
        int sno, sub1, sub2, sub3;
        char name[20];

        FILE *fp = fopen("stud.dat", "a+");

        printf("Enter a student record  sno, name, sub1, sub2, sub3 respectively\n");
        scanf("%d %s %d %d %d\n", &sno, name, &sub1, &sub2, &sub3);

        printf("\nStudent record is as follows.........");
        fprintf(fp, "%d%s%d%d%d\n", sno, name, sub1, sub2, sub3);
        return 0;
}
```

Above two programs are same, but the second program contains a highlighted statement (**FILE *fp = fopen("stud.dat", "a+");**) and a few modifications like **'f**printf', **'fp'.** Only few modifications included. These modifications affect data to transfers from console to diskette in the file **stud.dat**. This process is known as **file control/file management/file organization**.

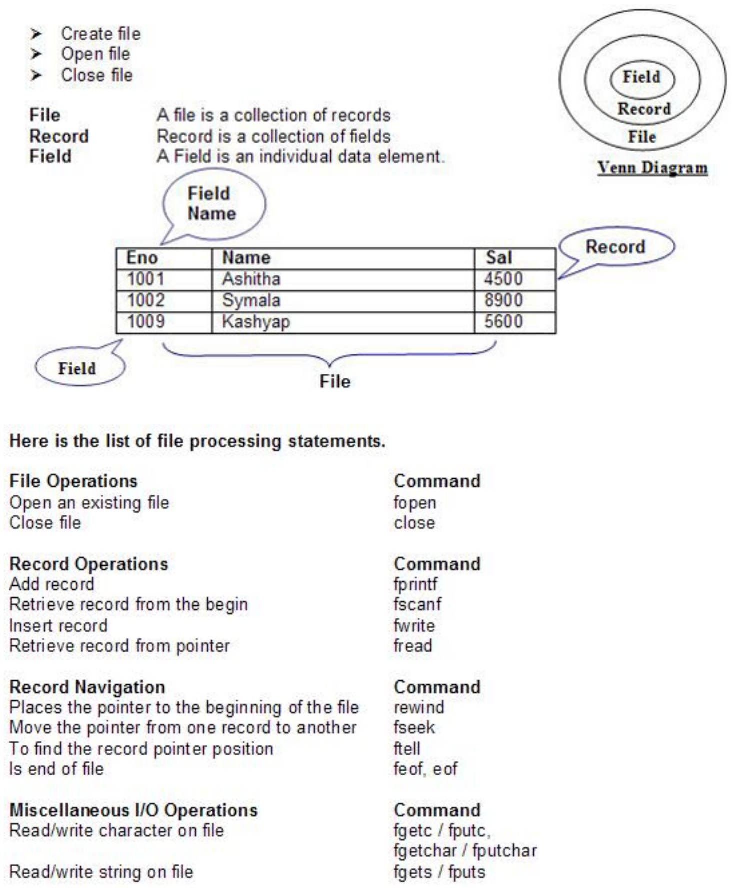
This is an easiest way to transfer the output from monitor to file using file control statement.



Actually file processing involved with a lot of operations as well as methods to implement. Here is the actual process to handle files.

File Handling

Generally every program has to present the resulted values on the screen (1st program illustrates this). But those values are removed from the memory whenever the program is terminated. If we want to keep records permanently, save them in a file. Every file has a few operations, here is a few;


Field
Record
File
Venn Diagram
Field Name
Record
Field
File

- Create file
- Open file
- Close file

File	A file is a collection of records
Record	Record is a collection of fields
Field	A Field is an individual data element.

Eno	Name	Sal
1001	Ashitha	4500
1002	Symala	8900
1009	Kashyap	5600

Here is the list of file processing statements.

File Operations	**Command**
Open an existing file	fopen
Close file	close

Record Operations	**Command**
Add record	fprintf
Retrieve record from the begin	fscanf
Insert record	fwrite
Retrieve record from pointer	fread

Record Navigation	**Command**
Places the pointer to the beginning of the file	rewind
Move the pointer from one record to another	fseek
To find the record pointer position	ftell
Is end of file	feof, eof

Miscellaneous I/O Operations	**Command**
Read/write character on file	fgetc / fputc, fgetchar / fputchar
Read/write string on file	fgets / fputs

File Operations

fopen	Opens the stream filename in the mode mode & if succeeded, Returns a pointer to the newly open stream; or Null other wise.
Syntax	**FILE *fopen(const char *filename, const char *mode);**

E.g.	FILE *fp = fopen("stud.dat", "r"); /* Read from file */ FILE *fp = fopen("emp.dat", "w"); /* Write to file */ FILE *fp = fopen("emp.dat", "a+"); /* Read and Write on file */ **Mode:** The mode string used in calls to fopen, is one of the following values: **Mode** **Description** **r** Open for reading only **w** Create for writing (If a file by that name already exists, it will be overwritten). **a** Append; open for writing at end of file, or create for writing if the file does not exist. **r+** Open an existing file for update (reading and writing) **w+** Create a new file for update (reading and writing). If a file by that name already exists, it will be overwritten. **a+** Open for append; open for update at the end of the file, or create if the file does not exist. To specify that a given file is being opened or created in text mode, append "**t**" to the string (**rt, w+t,** etc.). To specify binary mode, append "**b**" to the string (**wb, a+b,** etc.).
fclose	Closes the file pointed to by fp & returns 0 on success, EOF is returned in case of error
Syntax	**Int fclose(FILE *fp);**
e.g.	Fclose(fp); fclose(stud); fcloseall();
fprintf	Sends formatted output to a stream. Uses the same format specifiers as printf, but sends output to the specified stream. Returns the number of bytes output or EOF in case of error.
Syntax	**Fprintf(fptr, “Control String”, list);**

E.g	Fprintf(fp, “%d %s %d %d %d”, sno, name, sub1, sub2, sub3);
	fprintf(emp, “%d %s %d”, eno, name, sal);

fscanf	This function is used to read a formatted data from a specified file.
Syntax:	**Fscanf(fptr, “Control String”, list);**

E.g	Fscanf(fp, “%d %s %d %d %d”, &sno, name, &sub1, &sub2, &sub3);
	fscanff(emp, “%d %s %d”, &eno, name, &sal);

fwrite	Fwrite appends a specified number of equal-sized data items to an output file. **Size_t fwrite(const void *ptr, size_t size, size_t n, FILE*stream);** **Argument What It Is/Does** **Ptr** Pointer to any object; the data written begins at ptr **Size** Length of each item of data **N** Number of data items to be appended **stream** Specifies output file The total number of bytes written is (**n * size**)
Syntax:	
fread	Fread retrieves a specified number of equal-sized data items from an input file.
Syntax	**Size_t fread(void *ptr, size_t size, size_t n, FILE*stream);** **Argument What It Is/Does** **Ptr** Pointer to any object; the data written begins at ptr **size** Length of each item of data **n** Number of data items to be appended **stream** Specifies output file The total number of bytes written is (**n * size**)
rewind	Repositions file pointer to stream's beginning

Syntax	**Void rewind(FILE *stream);**
	E.g. fewind(fp);
	Rewind(stream) is equivalent to fseek(stream, 0L, SEEK_SET)
	except that rewind clears the end-of-file and error indicators, while fseek only clears the end-of-file indicator. After rewind, the next operation on an update file can be either input or output.

fseek	The file pointer for the stream is positioned at offset number of bytes calculated from the position specified by whence. Offset may be zero, negative, or positive. The defined symbols SEEK_CUR, SEEK_SET & SEEK_END are used as whence specifiers to indicate current position. BOF & EOF respectively. Returns 0 if successful or nonzero on failure.
Syntax	**Int fssek(FILE *stream, long offset, int whence);**
ftell	Returns the current file pointer position on success or Negative value on error.
Syntax	**Long ftell(FILE *stream);**
feof	It is a macro to return nonzero if end-of-file has been reached on the stream.

Syntax	**Int feof(FILE *stream);**
eof	Checks whether the position marker in the file given by its handle is at the end-of-file. If yes, returns 0, 1 is returned if position marker is NOT at eof & an error is indicated by setting of errno & return value of –1.
Syntax	**Int eof(int handle);**
fgets / fputs	The function fgets/fputs gets/puts a string(of size n bytes) on the file pointed to by stream and returns end-of-file on error.
Syntax	**Char *fgets(char *s, int n, FILE *stream);**
fgetc/fputc	Reads/writes a character from a stream.
Syntax	**Int fgetc/fputc(FILE *stream);**
fgetchar/ fputchar	These are equivalent to the above fgetc/fputc.

Write a program to read a student data and store it in a data file.

```
/* Program to create a student data file */
/* 85_write.c */
#include <stdio.h>
#include <ctype.h>
#include <conio.h>
int main()
{
      int sno, sub1, sub2, sub3;
      char name[10],ch;
      FILE *fp = fopen("stud.dat", "w");
      do{
            clrscr();
            printf("Enter Student number         "); scanf("%d", &sno);
            printf("Enter Student name           "); scanf("%s", name);
            printf("Enter 3 Subjects Marks       ");
            scanf("%d%d%d", &sub1, &sub2, &sub3);
            fprintf(fp, "%d %s %d %d %d\n", sno, name, sub1, sub2, sub3);
            printf("\n\nDo you want to cont... (y/n)"); ch = getche();
      }while(toupper(ch) != 'N');
      fclose(fp);
      return 0;
}
```

Write a program to retrieve data from a student data file.

```
/* Program to retrieve data from a student data file */
/* 86_read.c */
#include <stdio.h>
#include <conio.h>
int main()
{
      int sno, sub1, sub2, sub3;
      char name[10];
      FILE *fp = fopen("stud.dat", "a+");
      clrscr();
      printf("Student Records are as follows....\n");
      do{
            fscanf(fp, "%d%s%d%d%d\n", &sno, name, &sub1, &sub2, &sub3);
            printf("%5d%15s%3d%3d%3d\n", sno, name, sub1, sub2, sub3);
      }while(!feof(fp));
      fclose(fp);
      return 0;
}
```

previous | up | next
Learning C/C++ Step-By-Step - Page 11 | Learning C/C++ Step-By-Step - Page 13

0 Tweet

add comment | view as pdf | print: this | all page(s)

Related Tutorials

- Beginner's Guide To c++
- An Explanation of Pointers (C++)