Rajdhani

DATE / /

Roll No. 23355

Title : Assignment 10 : File Handling

Aim : Implementation of sequential file

Problem statement : Department maintains student's database. The file contains roll number, name, division and adress. Write a program to create a sequential file to store and maintain student data.
It should allow user to
a) create a database
b) Add information of student
c) Delete information of student
d) Search and display information of particular student.
   i) If record of student does not exist an appropiate message is displayed.
   ii) If student record is found, it should display the student's details.
e) Display records of all students in tabular form.

Theory :

- What is file data structure :
File structure is organization of data in secondary storage device in such a way that minimize the access time & the storage space. A file structure allows application to read, write & modify data.

- Need for file data structure :

1) Storing data in files will preserves data even if program terminates.

2) you can easily access the contents of file using few commands.

3) You can easily move your data from one computer to another without any changes.

- Types of file :

1) Data and code files:
- A data file is a computer file which stores data to be used by computer application or system, including input and output data.
- It usually does not contain instructions or code to be executed.

2) variable vs fixed length files:
Fixed length files:
- All records in the files are of same size.
- Leads to memory wastage.
- Access of record is easier & faster.

Variable length files:
- Different records in file have different size.
- Memory efficient
- Access to record is slow.

3) text vs binary files:
- A text file stores data in the form of alphabates, digits and other special symbol by storing their ASCII values & are in human-readable format.

- Binary files contain a collection of bytes which are not in human-readable format.

4) Based on how data is organized in the file.
Sequential file: A sequential file is one that contains & stores data in chronological order.

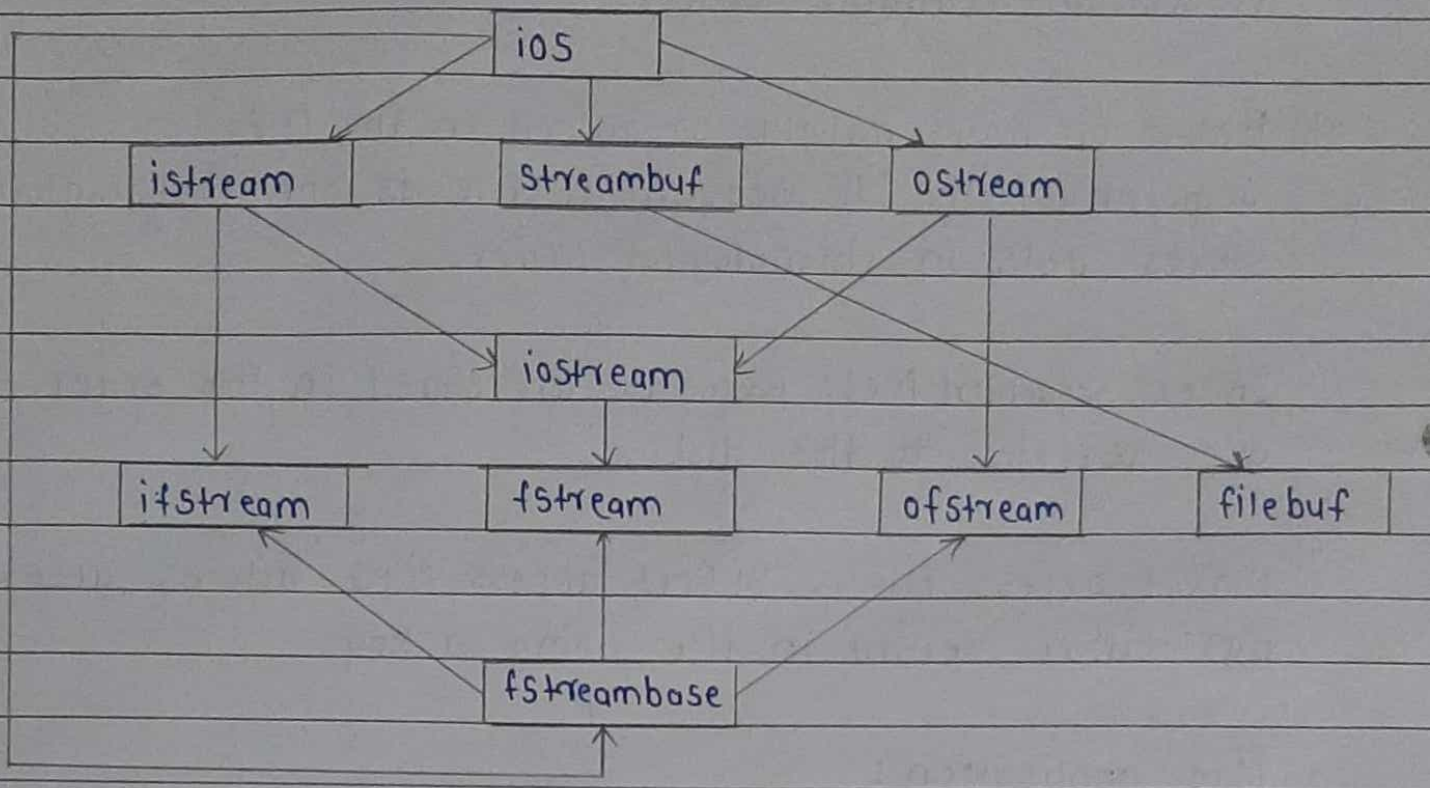
Index Sequential files: Records are stored in the order that they are written to the disk.

Direct access files: Direct access files allows access to a particular record in file using a key.

- File application:
- Read, store, update data on the file.

- List down various operations on the file:
  Create
  Write
  Read
  Delete
  Truncate
  Modify
  Append
  Rename

- C++ file basic : class hierarchy



Files are mainly dealt by using three classes fstream, ofstream, ifstream available in fstream header file.

ofstream : stream class to write on files
ifstream : stream class to read from files.
fstream : stream class to both read & write from/to files

- File open with modes, close, read, write syntaxes :

1) Open a file :
We can open a file by
i) Passing file name in constructor at the time of object creation.
ii) using open ~~mode~~ method

Rajdhani
DATE / /

i) Using constructor:
Syntax: fStream (filename, openmode)
Example: fStream file("myfile.txt", ios::in);

ii) Using open method:
Syntax: fstream objectName;
objectName.open (filename, openmode)
Example: fstream file;
file.open ("myfile.txt", ios::in);

2) closing a file:
fileobject.close();
example: file.close();

- Modes of file open:

1) in : Opens file for reading
2) out : opens file for writing
3) binary: operations are performed in binary mode rather than text.
4) ate : The o/p position starts at the end of file.
5) app : Appends the contents at the end of file.
6) trunc : Any contents that existed in the file, before it is opened are discarded.

Example of reading from file :

```
#include <iostream>
#include <fstream>
using namespace std;
int main ()
{
   char c;
   fstream ("myfile.txt", ios::in);
   fstream file1("myfile.txt", ios::in);
   if (!file1)
   {
      cout << "File not found";
   }
   else
   {
      while (file1)
      {
         file1 >> c;
         cout << c;
      }
   }
} file1.close();
}
```

Rajdhani
DATE / /

Example of writing to file:

```
#include<iostream>
#inclue <fstream>
using namespace std;
int main()
{
   char c;
   fstream file1 ("myfile.txt", ios::out);
   if (!file1)
     cout <<" unable to create file";
   else
    {
      char yn='y';
      while (yn=='y' || yn=='Y')
      {
          cin >> c;
          file1 << c;
          cout <<" Want to enter more:";
          cin >> yn;
      }
      file1.close();
   }
}
```

- isopen meaning:
Returns whether the stream is currently associated ~~with~~ to a file. Return true if file is open.

- eof meaning:
Returns nonzero value (meaning True) when there are no more data to read & zero otherwise.

- File pointer reposition functions:
tellg() : used with i/p stream & returns the current get position in the stream.

tellp() : used with o/p stream & returns the current put position in the stream

Seekg() : used to move pointer to desired location w.r.t. referen pointer.

Seekp() : used to put pointer to desired location w.r.t. reference pointer.

- Algorithm / pseudocode:

1) create student database:
i) Open file in write ~~mode~~ and binary mode.
ii) If there is any error while opening file, throw error message
iii) Else take student data from user & write to file using write() method
iv) close file.

Rajdhani
DATE / /

2) Add information of student :
i) Open file in append & binary mode.
ii) If there is any error, throw error message.
iii) Else take student's data from user & write it to file using write() method
iv) Close file.

3) Delete information of a student :
i) Open file in read mode.
ii) Create newfile with write mode.
iii) Write all records from file to newfile except that which is to be deleted.
iv) ~~De~~ close both files
v) Delete file.
vi) Rename newfile with original file name.

4) Search & display information of particular student :
i) Open file in read mode
ii) Repeate till end of file
a. If roll number of current record = search roll number
Display student record
Set flag
b. Else
Read next record

iii) If flag is not set, ~~No~~ record not found
iv) close file.

Rajdhani
DATE / /

3. Test cases / validations:

1. File open validation
2. Input data validation for roll number

- Conclusion :

Sequential files are suitable for applications that requires sequential processing of entire data. It is simple to program & easy to design. Operations like searching, deletion, updating consumes more time. It is not possible to add a record in middle easily.

It to access a record it takes more time than direct access files.

When the order in which you keep records ~~in~~ is not important then sequential ~~cho~~ files is best choice.