DATA FILE HANDLING IN C++

What if a FILE?

A file is a stream of bytes stored on some secondary storage devices.

NEED FOR DATA FILES

- ❑ Many real life problems requires handling of large amount of data.
- ❑ Earlier we used arrays to store bulk data.
- ❑ The problem with the arrays is that arrays are stored in RAM.
- ❑ The data stored in arrays is retained as long as the program is running. Once the program is over the data stored in the arrays is also lost.
- ❑ To store the data permanently we need files.

Files are required to save our data (on a secondary storage device) for future use, as RAM is not able to hold our data permanently.

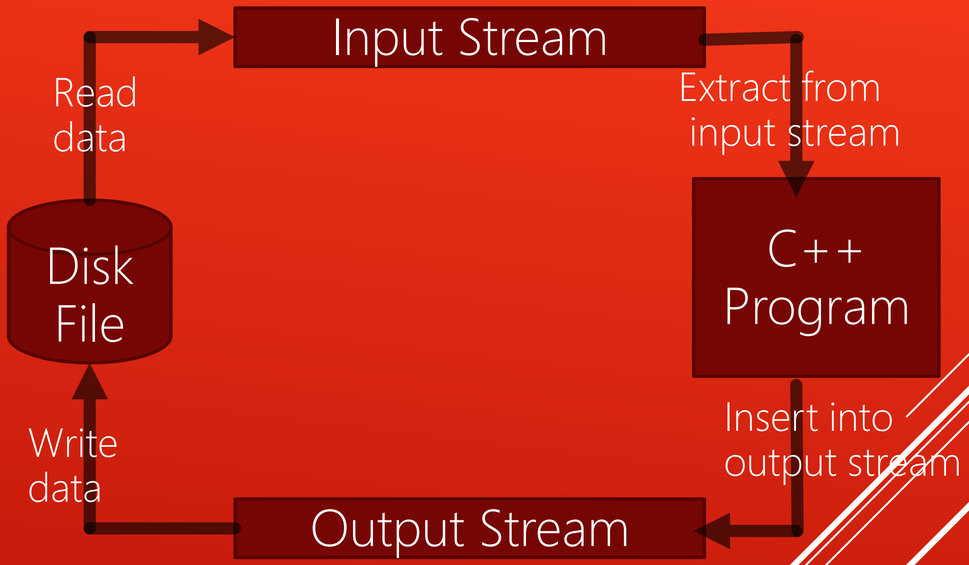
DIFFERENCE BETWEEN ARRAYS AND FILES

Difference between Files and Arrays(graduating to files):

ARRAYS	FILES
Arrays are stored in RAM	Files are stored on Hard Disk
Data is stored temporarily	Data is stored permanently
Arrays have fixed size	File can have variable size
Arrays can not be used to share data between programs.	Files can be used to share data between programs.

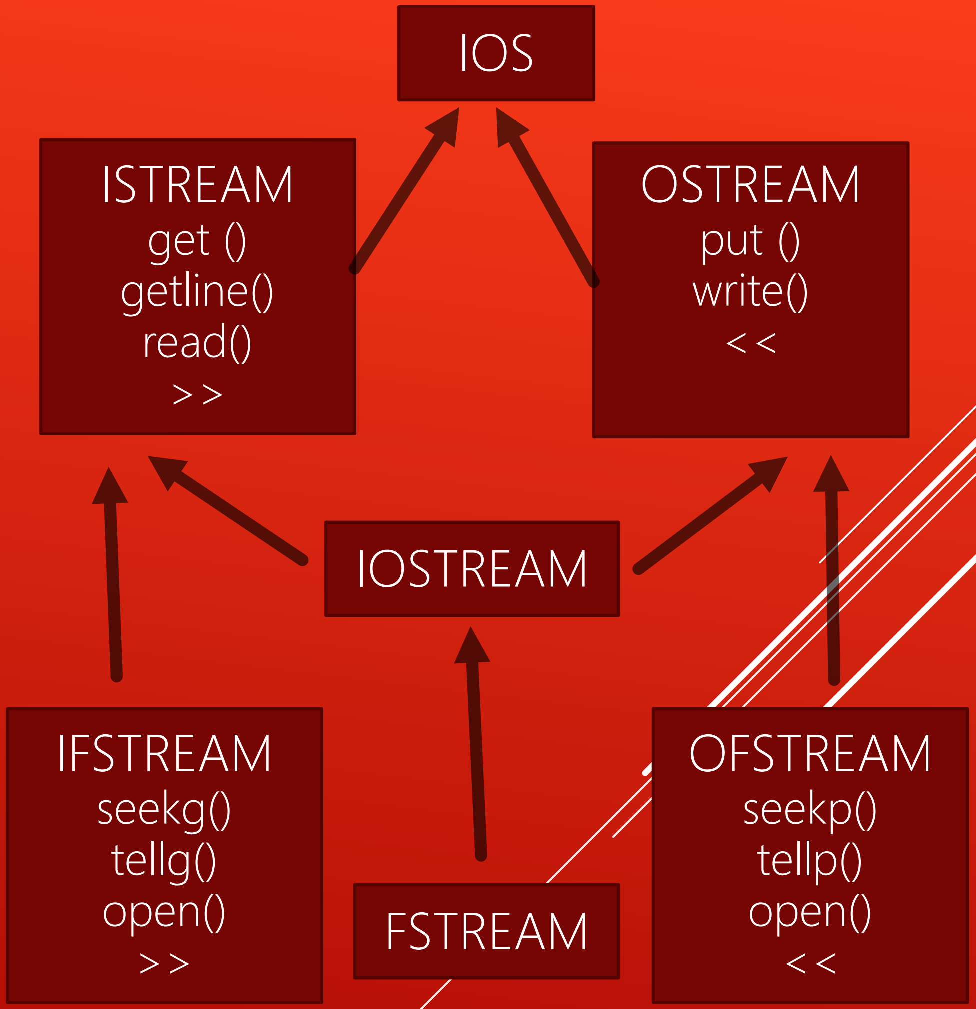
INPUT/OUTPUT IN C++ STREAMS

- ❑ The input/output system of C++ handles file I/O operations in the same way it handles console I/O operations.
- ❑ It uses file stream as an interface between programs and files.
- ❑ A stream is defined as the flow of data.
- ❑ Different kinds of stream are used to represent different kinds of data flow.
  - ❖ Output stream: The stream which controls the flow of data from the program to file is called output stream.
  - ❖ Input stream: The stream which controls the flow of data from the file to the program is called input stream.



INPUT/OUTPUT IN C++ CLASSES

- Each stream is associated with a particular class which contains definitions and methods for dealing with that particular kind of data
- These include fstream, ifstream and ofstream. These classes are defined in the header file fstream.h. Therefore it is necessary to include this header file while writing file programs.
- The classes contained in fstream.h are derived from iostream.h. Thus it is not necessary to include iostream.h in our program, if we are using the header file fstream.h in it.



INPUT/OUTPUT IN C++ CLASSES contd….

- The ifstream class contains open() function with default input mode and inherits the functions get(), getline(), read(), seekg() and tellg().
- The ofstream class contains open() function with default output mode and inherits functions put(), write(), seekp() and tellp() from ostream.
- The fstream class contains open() function with default input/output mode and inherits all I/O functions from iostream.h.

TYPES OF DATA FILES

There are two types of data files in C++: Text files and Binary files

- ❑ **Text files** store the information in ASCII characters. Each line of text in text files is terminated by a special character called EOL. In text files some internal translations take place while storing data.

- ❑ **Binary files** store information in binary format. There is no EOL character in binary files and no character translation takes place in binary files.

DIFFERENCE BETWEEN TEXT FILES AND BINARY FILES

These differ on six main parameters:

	TEXT FILES	BINARY FILES
Handling of new lines	In text files various character translations are performed such as "\r+\f"(carriage return-linefeed combination)is converted into "\n"(new line) while reading from a file and vice-versa while writing.	In binary files no such translations are performed.
Portability	Portable: one can easily transfer text file from one computer to the other	Non portable: Binary files are dependent. If the new computer uses a different internal representation for values they cannot be transferred.
Storage of numbers	In text files when we store numbers they are stored as characters eg if we store a decimal no 42.9876 in a text file it occupies 7 bytes	In a binary file 42.9876 is stored in 4 bytes

DIFFERENCE BETWEEN TEXT FILES AND BINARY FILES contd…

	Text Files	Binary Files
Readability	Are readable and thus can be easily edited using any word editor.	Not readable
Storage	Occupy more space due to character conversions	Occupy less space.
Accuracy	While reading/writing of numbers, some conversion errors may occur.	Highly accurate for numbers because it stores the exact internal representation of values.

OPENING FILES

Opening of files can be achieved in two ways:

- ❑ Using Constructor function: This method is useful when we open only one file in a stream. To open a file using a constructor fuction we create an object of desired stream and initialize that object ith the desired file name. For eg. The statement

```
            ofstream fout(“ABC.TXT”);
```

will create an onject fout of class ofstream, opens the file ABC.TXT and attaches it to the output stream for writing. Similarly the statement

```
            ifstream fin(“ABC.TXT);
```

will create an object fin of class ifstream, opens the file “ABC.TXT” and attaches it to the input stream for reading.

- ❑ Using open() function: This method is useful when we want to open multiple files using a single stream. For eg.

```
   ifstream fin;      //creates input stream
   fin.open(“ABC.TXT”); // associates ABC.TXT to this stream
   fin.close();  // closes the file
   fin.open(“XYZ.TXT”); // associates the input stream with file XYZ.TXT
```

CLOSING FILES

The connections with a file are automatically closed when the input and output stream objects expires ie when they go out of scope. However we can close the file explicitly by using the close() method:

```
    fin.close();
```

Closing a file flushes the buffer which means the data remaining in the buffer of input/output stream is moved to its appropriate place. For example, when an input files connection is closed, the data is moved from the input bufferto the program and when an output file connection is closed the data is moved from the output buffer to the disk file.

FILE MODES

File modes describes the way in which a file is to be used. The most common file modes are :

File Modes	Exolanation
ios::in	Opens file for reading. File pointer is at the beginning of the file
ios::out	Opens file for writing. If the file is already created and opened in this mode all the previous content gets erased from the file.
ios::app	Opens file for adding new records. File pointer is at the end of the file. New records can be added only at the end of the file.
ios::ate	Opens the file for both reading and writing. File pointer is at the end of the file when file is opened in his mode but can be moved to any location in the file using file pointer methods.
ios::binary	Opens file in binary mode. By default the file is opened n text mode.

Two or more modes can be combined using the bitwise operator |

TEXT FILE FUNCTIONS

- **Reading/writing a single character from/to file :**

  get() – read a single character from text file and store in a buffer.
  e.g **file.get(ch);**
  put() - writing a single character in text file.
  e.g. **file.put(ch);**

- **Reading/writing a line from/to file:**

  getline() - read a line of text from text file stored in a buffer.
  e.g **file.getline(s,80,"\n");**
  <<(insertion operator) – write a line to a file.
  fin<<s;

- **Reading/writing a word from/to file:**

  ```
  char ch[20];
  fin.getline(ch,20, ' ');
  ```

  We can use **file>>ch** for reading and **file<<ch** writing a word in text file. The >> operator does not accept white spaces so it will stop when it encounters a space after word and stores that word in ch.

A PROGRAM TO CREATE A TEXT FILE

```
#include<fstream.h>
void main()
{
ofstream fout("abc.txt");
fout<<" i am creating a new text file\n";
fout<<"this text file contains alphabets and numbers\n";
fout.close();
}
```

The above program will create a text file "abc.txt" and store two lines in it. You can store as many lines as you want.

A PROGRAM TO READ A TEXT FILE CHRACTER BY CHARACTER

```
#include<fstream.h>
void main()
{
ifstream fin("abc.txt");
char ch;
while(!fin.eof())
{
fin.get(ch);
cout<<ch;
}
fin.close();
}
```

The above program will read a text file "abc.txt" one character at a time and display it on the screen.

A PROGRAM TO READ A TEXT FILE WORD BY WORD

```
#include<fstream.h>
void main()
{
ifstream fin("abc.txt");
char ch[20];
while(!fin.eof())
{
fin>>ch;
cout<<ch;
}
fin.close();
}
```

The above program will read a text file "abc.txt" one word at a time and display it on the screen.

A PROGRAM TO READ A TEXT FILE LINE BY LINE

```
#include<fstream.h>
void main()
{
ifstream fin(“abc.txt”);
char ch[80];
while(!fin.eof())
{
fin.getline(ch,80,”\n”);
cout<<ch;
}
fin.close();
}
```

The above program will read a text file “abc.txt” one line at a time and display it on the screen.