File Handling

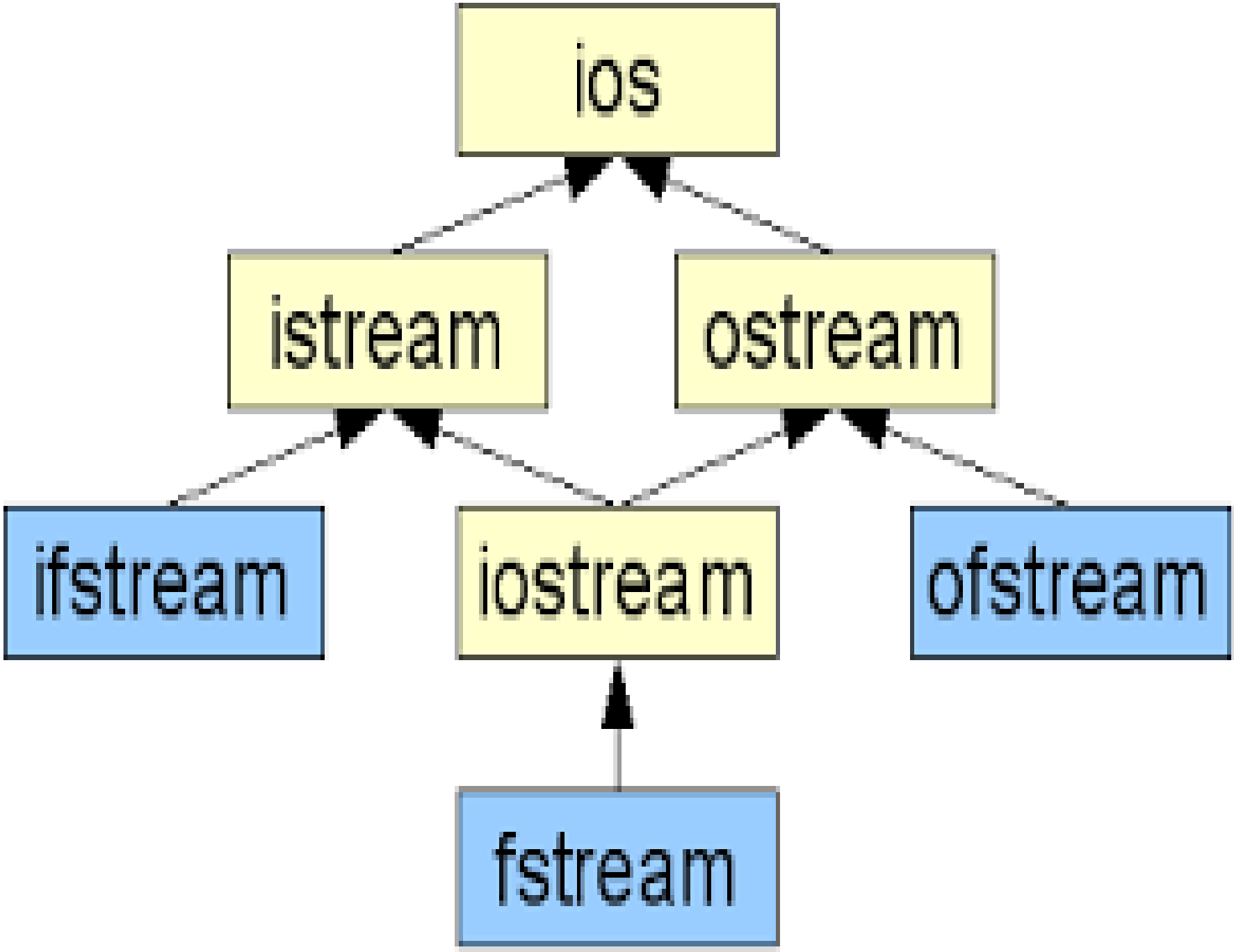
Type of Files

- Text File : hello how r u

- Binary File : ∞≈ôáó

File stream <fstream.h>

1	**ofstream** This data type represents the output file stream and is used to create files and to write information to files.
2	**ifstream** This data type represents the input file stream and is used to read information from files.
3	**fstream** This data type represents the file stream generally, and has the capabilities of both ofstream and ifstream which means it can create files, write information to files, and read information from files.


ios
istream
ostream
ifstream
iostream
ofstream
fstream

Files operations

- Open or Create a File
- Reading from File
- Writing to a file
- Append a File
- Close a File

File Creation and Opening

```
 ifstream fin;
ofsteram fout;

fin.open(“san.dat”);
fout.open(“san.dat”);
```

File Creation and Opening

```
fstream f1;

f1.open("san.dat", modes);
```

Text File opening Modes

1	**ios::app** Append mode. All output to that file to be appended to the end.
2	**ios::ate** Open a file for output and move the read/write control to the end of the file.
3	**ios::in** Open a file for reading.
4	**ios::out** Open a file for writing.
5	**ios::trunc** If the file already exists, its contents will be truncated before opening the file.
	ios::binary Operations are performed in binary mode rather than text.

Text File opening Modes

You can combine two or more modes by **Or**ing them together.

For example :

```
ofstream of;
of.open(“san.dat", ios::out | ios::trunc );
```

Writing to file

```
int roll;
ofstream fout;
fout.open(“san.dat”);
cin>>roll;
fout<<roll<<endl;
```

Reading from file

```
int roll;
ifstream fin;
fin.open(“san.dat”);
fin>>roll;
cout<<roll;
```

Closing a file

```
fsetram f1;

f1.close();
```

Some more options

- f1.seekg(0, ios::beg);

```
// point read pointer at beginning of file

 while (f1) {

      // Read a Line from File
      getline(f1, line);

      // Print line in Console
      cout << line << endl;
   }
```

Reading a class data

Class structure

```
class student
 {   int roll;   char name[30];
  float marks;
public:
 void getData(); // get student data from user
    void displayData(); // display data
};
```

Write as object

```
file.open(“san.txt", ios :: out); // open file for
   writing
cout << "\nWriting Student information to the
   file :- " << endl;
for (i = 0; i < 3; i++)
{      s[i].getData();
      file.write((char *)&s[i], sizeof(s[i])); //write object to file
}
```

Read an object

```
file.open(“san.txt", ios :: in); // open file for
   reading
cout << "\nReading Student information to the
   file :- " << endl;
for (i = 0; i < 3; i++) {      // read an object from a
   file
file.read((char *)&s[i], sizeof(s[i]));
   s[i].displayData();
  }
```