C++ in depth

File handling

Saurabh Shukla (MySirG)

Agenda

1. Data
2. Life of Data
3. Secondary memory
4. File
5. File Handling
6. Streams
7. File Handling Implementation
8. File types
9. File opening modes
10. Practical

Data

Data is any piece of information which is required in data processing in a computer program.

Life of data

- Program needs memory in RAM for execution.

Variables are reserved memory location in a program to store data

On the basis of scope variables are of different types like global variable, local variable, static local variable, instance variable and static member variable.

Life of data depends on the life of variables.

Secondary Memory

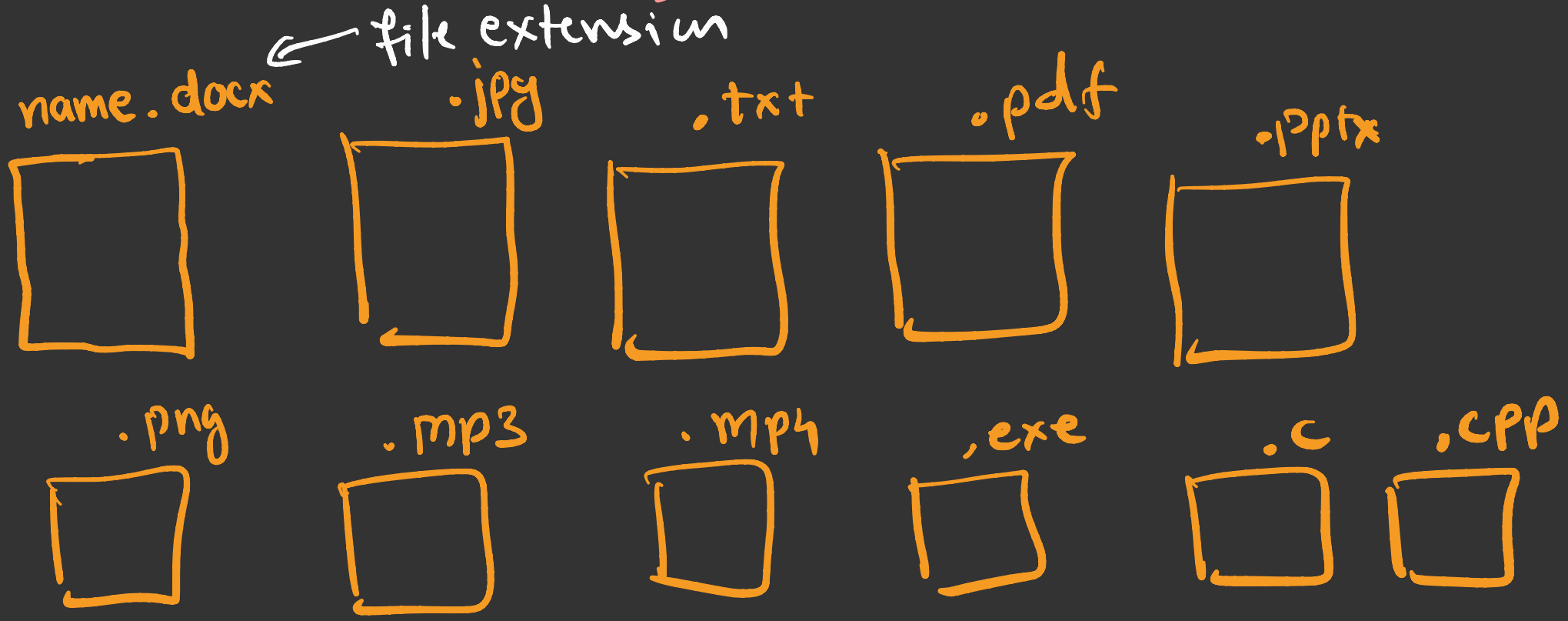
Sometimes you need variable data to sustain for longer period of time, even beyond the life of the program.

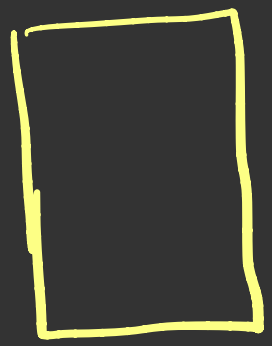
Life of variables cannot be more than the life of a program, therefore you need to store variable's data outside the program's memory

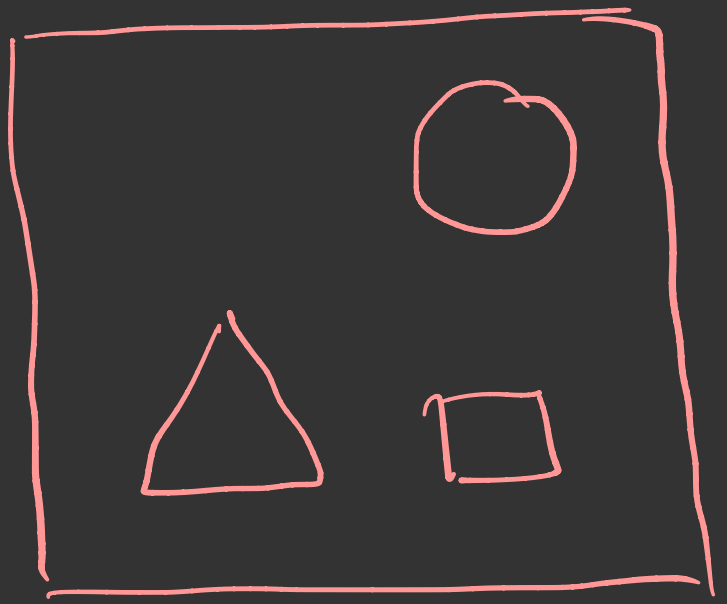
Secondary memory like hard disk or SSD or pen drive, etc.

File

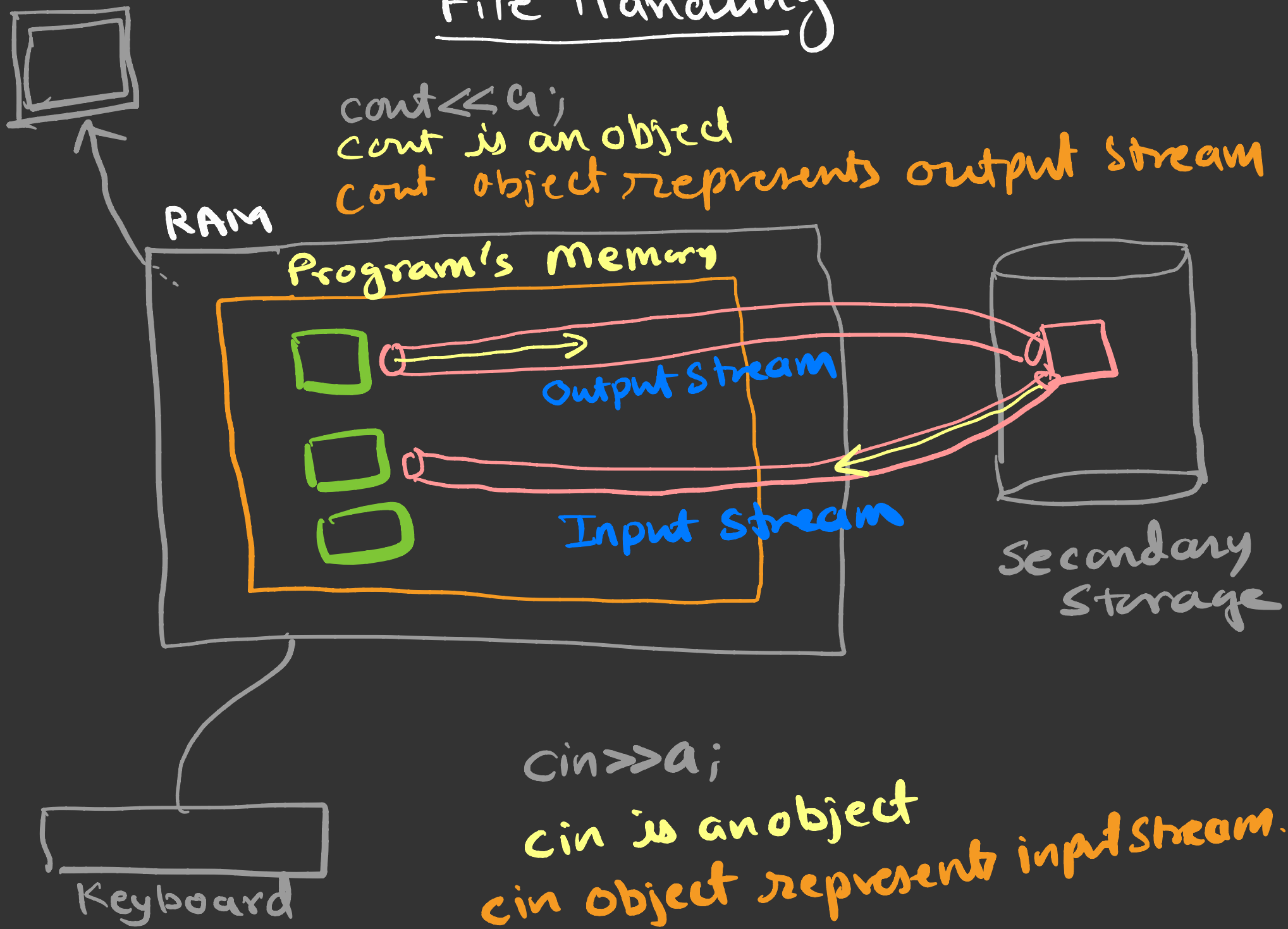
File is a logical entity in a computer's secondary memory to store data.

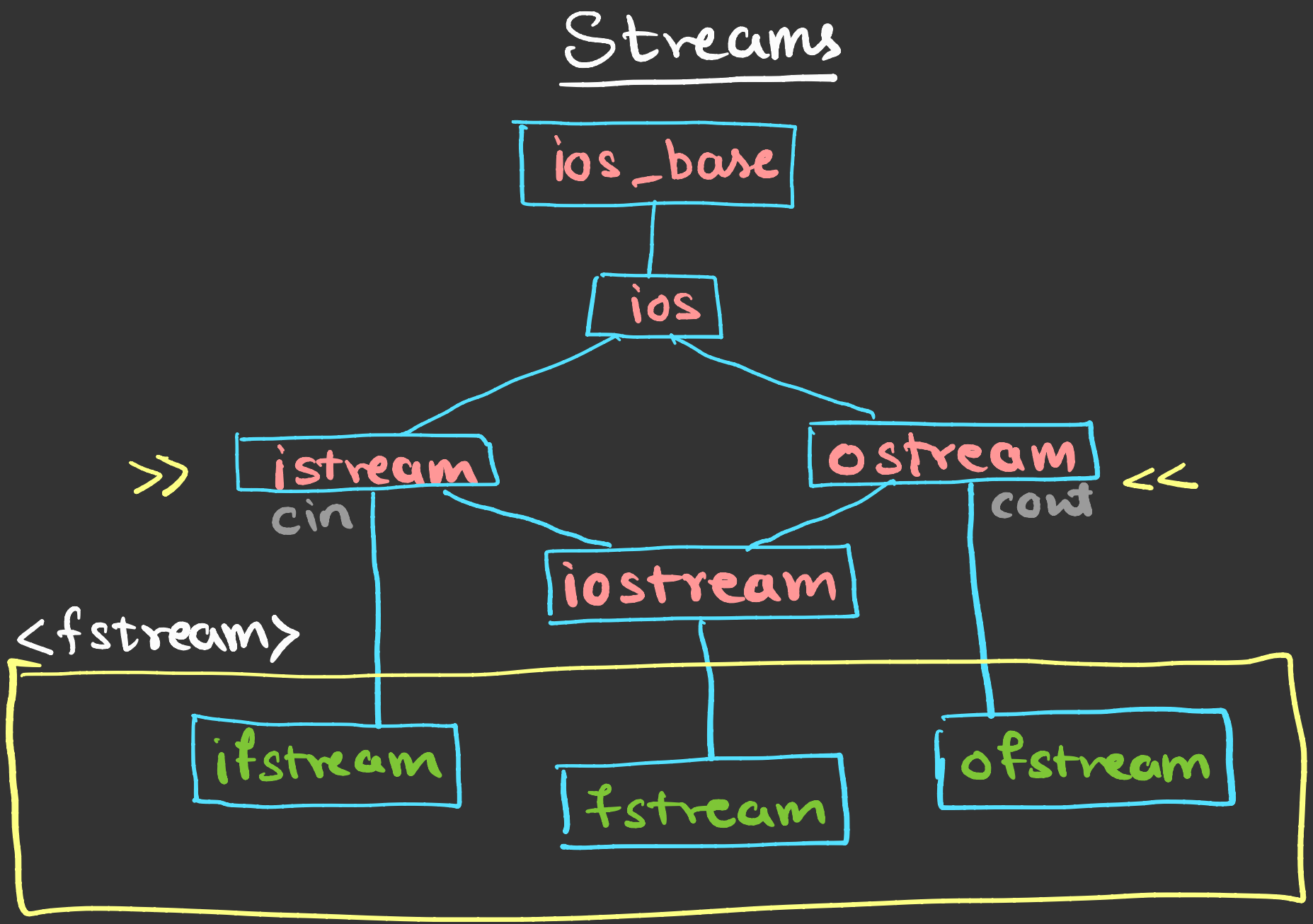






File Handling




Streams
ios_base
ios
istream
cin
>>
ostream
cout
<<
iostream
<fstream>
ifstream
fstream
ofstream

File Handling Implementation

1. Create an object of input or output stream
2. Open a file
3. Read or write
4. close file

File Types

There are two types of files

1. Text files — sequence of characters
2. Binary files — sequence of non-characters

File Opening modes

in	open for reading
out	open for writing
app	open to append
binary	open file as binary.

How to write data in a file?

```
fout << data;
```

How to read data from a file?

```
ch= fin.get();
while ( ! fin.eof())
{
        cout<<ch;
        ch= fin.get();
}
```

How to append data in a file?

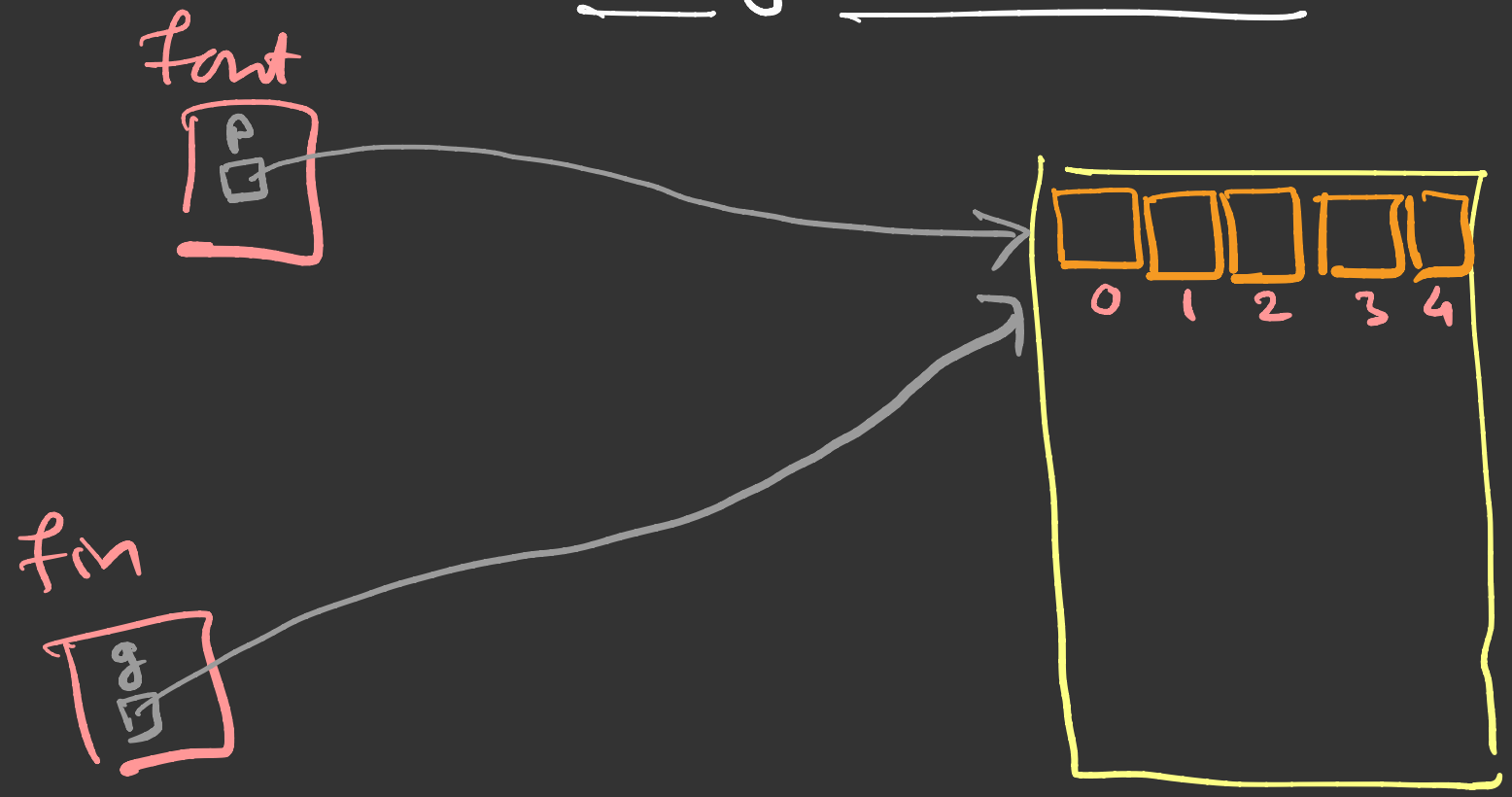
```
fout.open("file1.txt", ios::app);
fout << data;
```

```
Student S1;
S1.inputData();

ofstream fout;
fout.open("f1", ios::app|ios::binary);
fout.write((char*)&S1, sizeof(S1));
fout.close();
```

```
Student s;
ifstream fin;
fin.open("f1", ios::in | ios::binary);
if (!fin)
    cout<<"file not found";
else
{
    fin.read( (char *)&s, sizeof(s));
    while( !fin.eof())
    {
        s.showData();
        fin.read( (char *)&s, sizeof(s));
    }
    fin.close();
}
```

tellg() & tellp()




```
cout<< fout.tellp();
```

seekg() and seekp()

```
fout.seekp(0);
fin.seekg(2);
```