POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

☆Exceptions are unusual or unexpected problems that may arise during the execution of program

there are two types of exceptions:

Synchronous Exceptions

① Exceptions due to events within program

② Divide by zero, underflow overflow of data structure etc.

Asynchronous Exceptions

① Exceptions due to faults which are beyond the control of program

② diskfailure, input output interrupt etc

we can deal with synchronous exceptions

Exception handeling

fault handeling Mechanism

Steps

Main Ideas, Questions & Summary:

Library / Website Ref.:-

POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

(1) Detect No problem

(2) Report the problem

(3) handel error

~~We can using~~

we have 3 constructs try, throws and catch

Try block: has the code which can have errors or exceptions

Throw block: raises an exception and passes it to catch block

catch block: deals with the exception occ to arguments defined

Multiple catch blocks can be there after try block

```
try {
throws; throws;
}
catch(arg)
{
}
catch(arg) {}
```

Main Ideas, Questions & Summary:

Library / Website Ref.:-

POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

if exception doesnot have a match program terminates

```
try {
    if (a==0)
    { throw a;
    }
    else { cout<<a;
    }
}
catch(int i )
{
  cout <<"a is 0 "}
cout << "a is valid"
}
```

Streams : sequence of bits which enables smdly and recievig data to and from a program.
It provides an interface which takes data from input stream and delivers data to the outputstream.

Main Ideas, Questions & Summary:

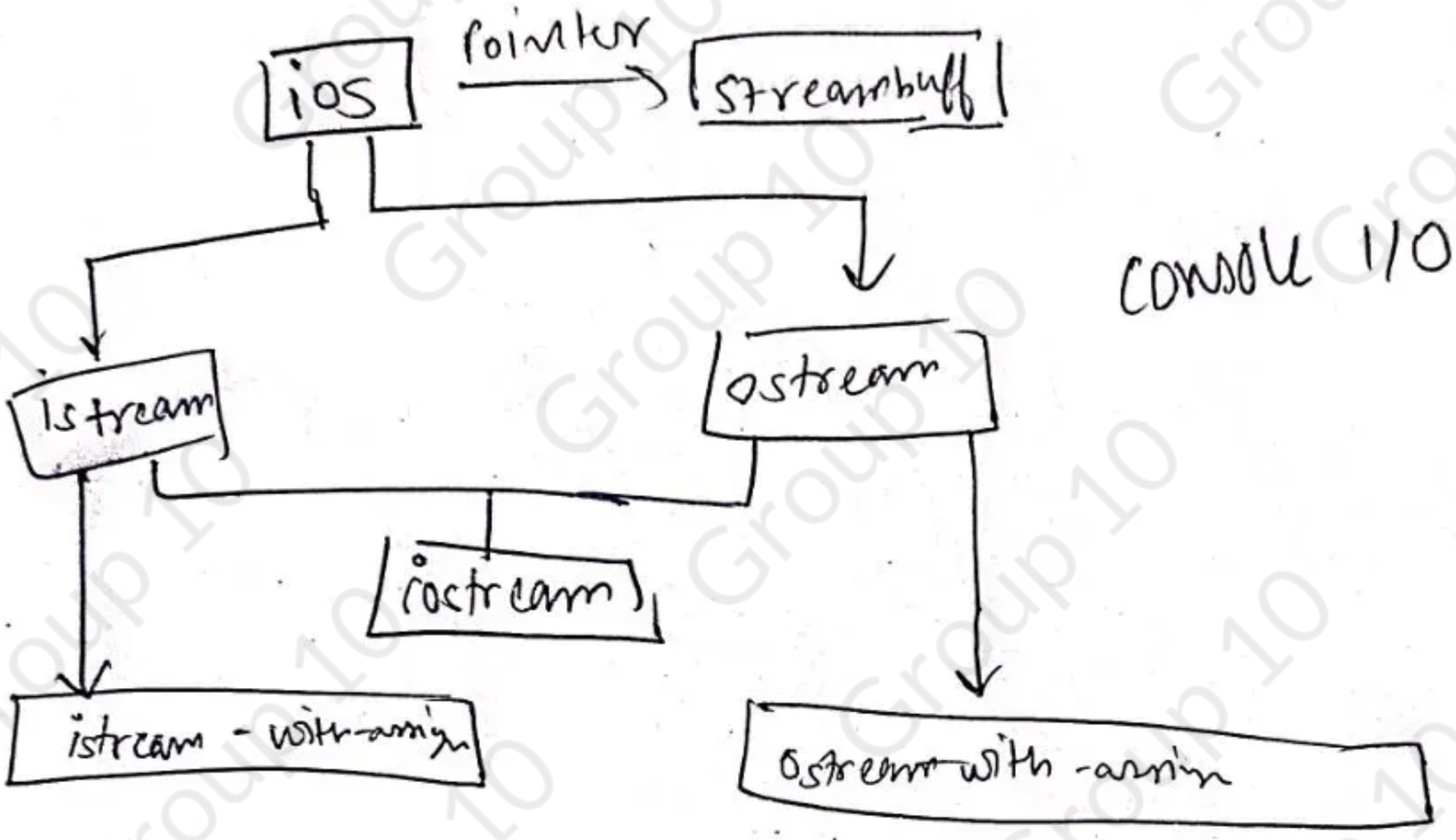
Library / Website Ref.:-

POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

Input Stream : recives data from source

Output Stream : It delivers data to the destination



CONSOLE I/O instream class

ios : This base class contains all basic facilities of io utilities by child classes

istream : input functions, defines >> operator

ostream : output functions, defines << operator

istream_with assign and ostream_with assign
⇒ Similar to istream & ostream but provides the facility of assignment operator.

Main Ideas, Questions & Summary:

Library / Website Ref.:-

File handling

① ios :-

- ios stands for input output stream.
- This class is the base class for other classes in this class hierarchy.
- This class contains the necessary facilities that are used by all the other derived classes for input & output operations.

② istream :-

- istream stands for input stream.
- This class is derived from the class 'ios'.
- This class handle input stream.

③ ostream :-

- ostream stands for output stream.
- This class is derived from the class 'ios'.
- This class handle output stream.
- This class declare output functions such as put() and write().

④ Streambuf :-

- This class contains a pointer which points to the buffer which is used to manage the input and output streams.

⑤ fstreambase :-

- This class provides operations common to the file streams.
- This class contains open() and close() function.

⑥ fstream :-

- This class provides input operations.
- It contains open() function with default input mode.

POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

⑦ ofstream :-

↳ This class provides output operations.

↳ This contains open() function with default output mode.

⑧ fstream :-

↳ This class provides support for simultaneous input and output operations.

↳ Inherits all the functions from istream and ostream through iostream.

⑨ filebuf :-

↳ Its purpose is to set the file buffers to read and write.

↳ We can also use file buffer member function to determine the length of the file.

for eg:

Open file by using constructor

ifstream (const char* filename, ios_base::openmode mode = ios_base...

ofstream(filename, openmode) by default openmode = ios::in

ifstream fin("filename");

pen file by using open method

alling of default constructor

ifstream fin;

fin.open (filename, openmode)
fin.open ("filename");