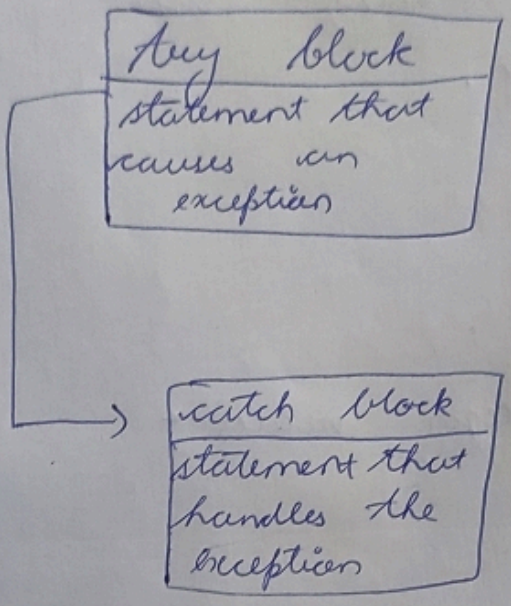
Group-9

POORNIMA

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

Exception handling



throwing will cause the normal program flow to be aborted in a raised exception

An exception is an unexpected problem that occurs during the execution of the program & terminates suddenly due to some issues.

Try: It represents a block of code that is executed when a particular exception is ~~thrown~~ placed from the try block.
It is followed by one or more catch blocks

Catch:
This statement executes a particular exception when is thrown by a try block
code to handle the exception is written inside catch.

Main Ideas, Questions & Summary:

Group 9

Throw:
An exception can be thrown using a throw keyword. When a program encounters a throw statement then it terminates the current function & starts finding matching catch block to handle the thrown exception.

Example:

```
int main()
{
int a, b;
cout << "Enter 2 integer values";
cin >> a >> b;
try
{
  if (b != 0)
  {
    cout << a/b;
  else
  {
    throw b;
  }
}
catch (int exp)
{
cout << "Divide by zero";
}
}
return 0;
}
```

Group - 9

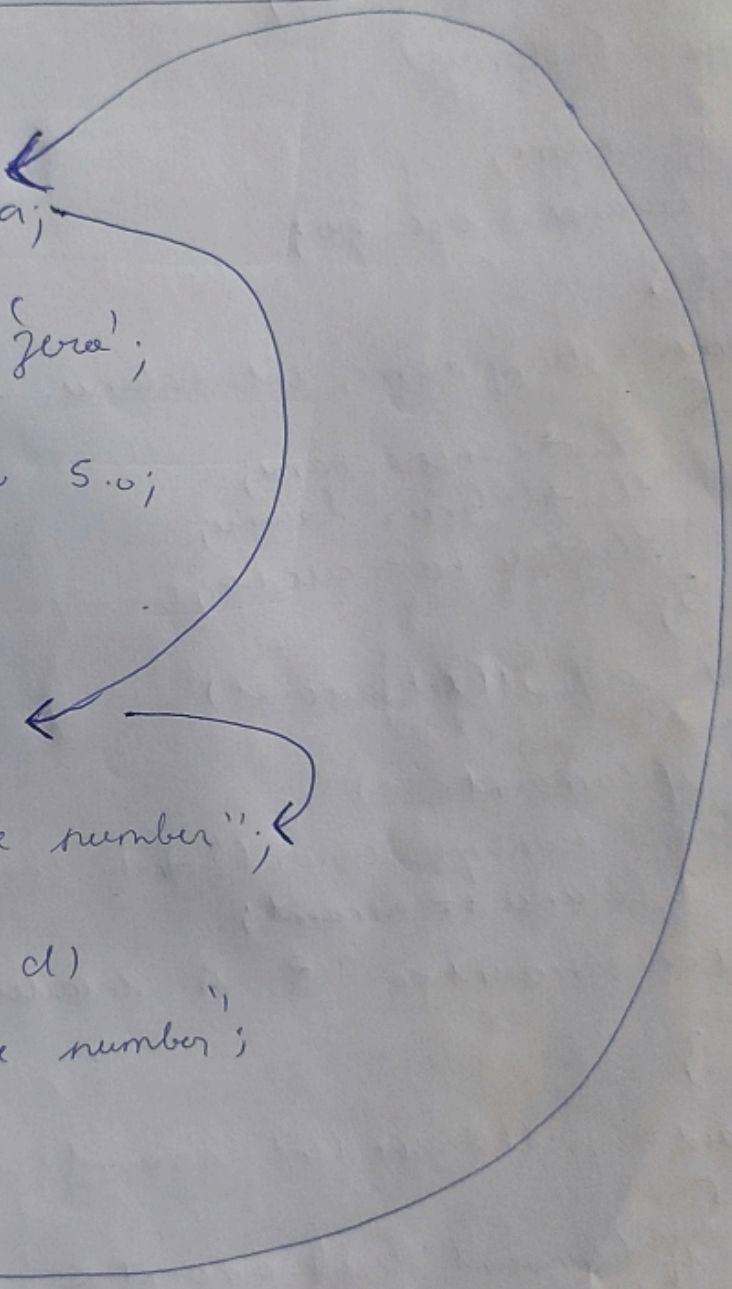
POORNIMA

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

Example of multiple catch statements:

```
void abc( int a)
{
  try {
  if (a>=1) throw a;
   else
  if (a==0) throw 'zero';
   else
   if (a <0) throw 5.0;
   catch (char c)
    {
    cout <<c
    }
   catch (int n)
    {
    cout<< "+ve number";
    }
   catch (double d)
    {
    cout<< "-ve number";
    }

  int main()
  {
    abc(1);
    abc(0);
    abc(-5);
    return 10;
  }
```



Main Ideas, Questions & Summary:

Library / Website Ref.:-

Full-fledged example of exception handling using bank account question

```
#include <iostream>
using namespace std;
class customer
{
string name;
int balance, acc_no;
public:
customer(string name, int balance, int acc_no)
    {
      this->name = name;
      this->balance = balance;
      this->acc_no = acc_no;
    };
void deposit(int amount)
  {
    if (amount<=0)
throw runtime_error("amt. should be greater than 0");
      balance += amount;
cout<< amount << "₹ is credited successfully\n";
    }
void withdraw(int amount)
  {
    if (amount>0 && amount<= balance)
        { balance -= amount;
cout<< amount << "₹ is debited successfully";
    }
    else if (amount<0)
    {
throw runtime_error("amt should be greater than 0");
    }
```

Group-9

POORNIMA

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

```
    else
    {throw runtime_error(" Your balance is low");
    }
}

};
int main()
{
Customer C1("Aditya", 5000, 10);
    try{
C1.deposit(100);
C1.withdraw(6000);
C1.deposit(100);
}
catch(const runtime_error& e)
    {
    cout << "Exception occured" << e.what() << endl;
    }

}
```

Main Ideas, Questions & Summary:

Library / Website Ref.:-

Group-9

Templates

* Allow us to write generic programs
* function & class template can be created
* Generic template can work with variety of datatypes.
* Redundancy ↓↓ (decreases)
* Flexibility ↑↑ (increases)
* Reusability ↑↑ (increases)

function templates

```
int abc (int a, int b)
abc (float a, float b)
abc (char a, char b)
```

} ( )

T abc, (T a, T b) } Generic datatypes

for example:

```
template <typename T> T abc (T a, T b)
        {
            return (a>b);
int main()
{
cout<< abc <int> (1,2);
cout<< abc <float> (1.0, 3.0);
cout<< abc <char> ('a', '6');
    return 0;
}
```

Group 11

POORNIMA

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

class templates

```
template < class T)
  class ABC
  {
   T x;
   public:
   abc (T n)
     {
       x=n;
      }
   T get()
    {
      return x;
    }
};
int main()
{
abc <int> obj1(5);
abc < float) obj2 (5.5);
 return 0;
}
```

class template with multiple parameters

```
template <class T, class U, class V=int)
 class ABC {
    T x1;
    U x2;
    V x3;
  public:
abc (T a1, U a2, V a2)
     { x1=a1;
       x2 = a2;
       x3=a3;
     }
 };
 int main()
  {
abc < float, char) obj1 (5.5, 'S', 5);
abc < float, char; bool )
 obj2 (5.5, 'S', true);
 return 0;
 }
```

Main Ideas, Questions & Summary:

file handling

GROUP: 9

file handling in cpp allows reading from and writing to files using the standered file streams:

-> ifstream (input file stream)
To read files.

-> ofstream (output file stream)
To write files.

-> fstream (file stream)
for both read & write in file.

Basic operations in file handling :—

-> Open a file
-> read/write data
-> close the file

Eg: Writing to a file

```
#include <iostream>
#include<fstream> //required for file handling.
using namespace std;
int main(){
    ofstream my("example.txt");
    if(my.is_open()){
        my<<"Hello, this is first line\n";
        my<<"This is second line of my file";
        cout<<"Data written to a file successfully.\n";
    }
    else{
        cout<<"Error opening file.\n";
    }
}
```

Group: 09

POORNIMA

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

Eg: Reading from a file

```
#include <iostream>
#include<fstream>
using namespace std;
int main(){
    ifstream my("example.txt");
    if(my.is_open()){
        string st;
        while(getline(my,st)){
            cout<<line<<endl;
        }
        my.close();
    }else{
        cout<<"Error opening file.\n";
    }
    return 0;
}
```

if you wants to read & write in file at same time then you have to use fstream function

→PTO

Main Ideas, Questions & Summary:

Library / Website Ref.:-

Some functions in fstream:— GROUP: 9

ios:: beg
ios:: curr
ios:: end
} pointers in fstream

here "ios" stands for input-output stream
"beg" → beginning
"curr" → current
"end" → end.

In c++ we have get & put pointers for getting data from a file and putting data into the file respectively.

for eg.

(i) seekg() :— "seek get"
It is used to move the get pointer to it desired location w.r.t. reference point.

eg: seekg(bytes, reference pointer);

(ii) tellg() :— "tell get"
It is used to know where is get pointer in the file
it returns integer value (location of pointer)

(iii) seekp() :— "seek put"

(iv) tellp() :— "tell put".

POORNIMA

GROUP :9

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

eg: fstream:—

```
#include<iostream>
#include<fstream>
using namespace std;
int main(){
    fstream my("data.txt", ios::in|ios::out|ios::app);
    if(my.is_open()){
    my<<"This is first line\n";
    my<<"This is second line";
    }
    // Reading from file
    my.seekg(0); // move pointer to beginning
    string st;
    while(getline(my, st)){
        cout<<st<<endl;
    }
    my.close();
    }else{
        cout<<"Error file opening.\n";
    }
    return 0;
}
```

Main Ideas, Questions & Summary:

Library / Website Ref.:-

Some file modes:—

GROUP: 9

ios:: in → open for reading
ios:: out → open for writing
ios:: app → open for append
ios:: trunc → Truncate file to 0 length (default for ofstream)
ios:: binary → Binary mode