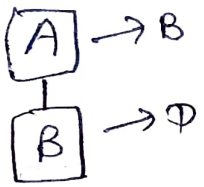
POORNIMA

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

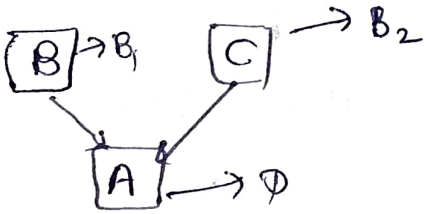
Inheritance: one of the most important concept in oop is that Inheritance. It allow us to define it class in terms of another class. which makes it easier to create and maintain and application. This also provide an opportunity to reuse the code functionality and fast Implementation type.

Types →

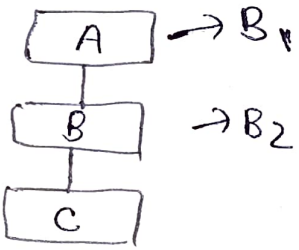
Single →



multiple →



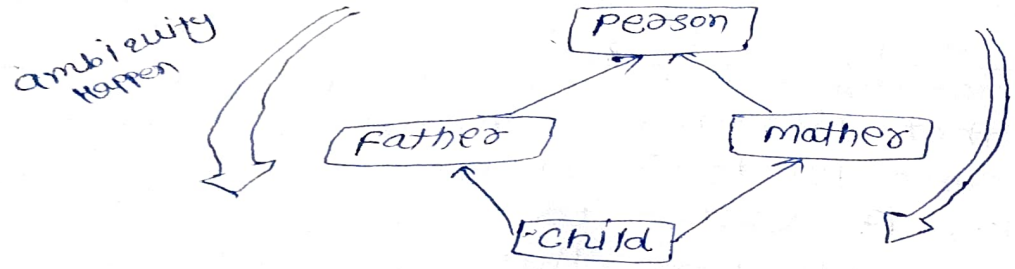
multilevel →



s, Questions & Summary:

ebsite Ref.:-

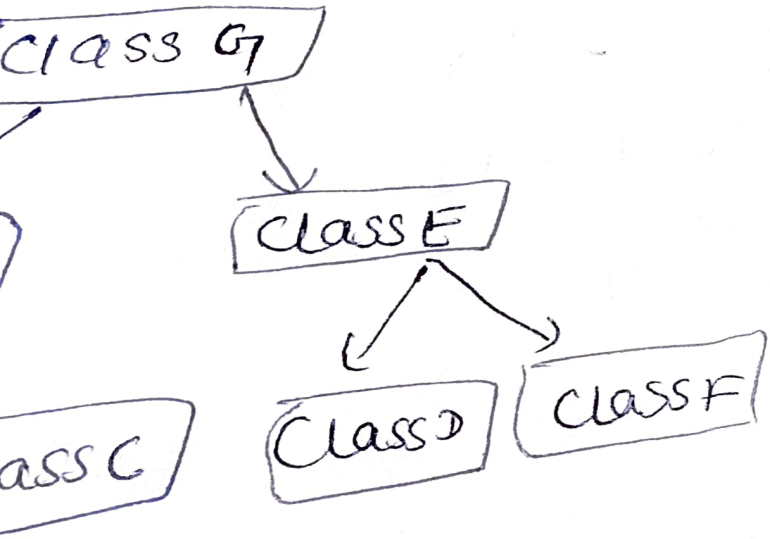
Diamond Problem :



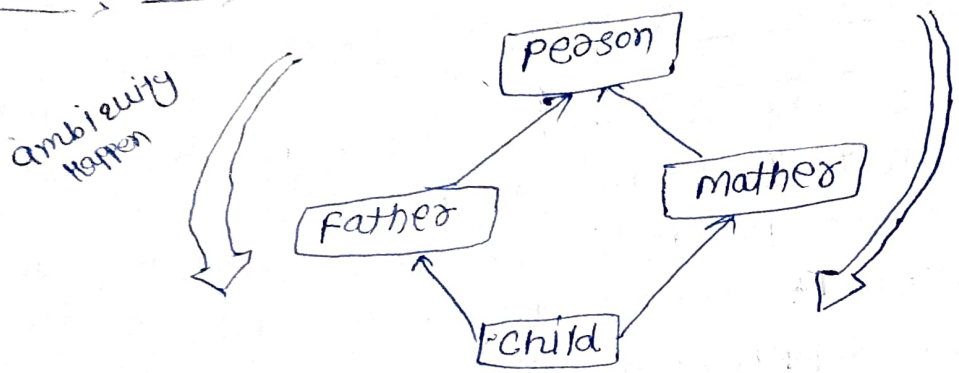
ion : use virtual keyword. Virtual class remove th ambiguity.

INHERITANCE :

In this type of inheritan one sub class is inherited from a sing . i.e. more than one derived class is cre ingle Base class.



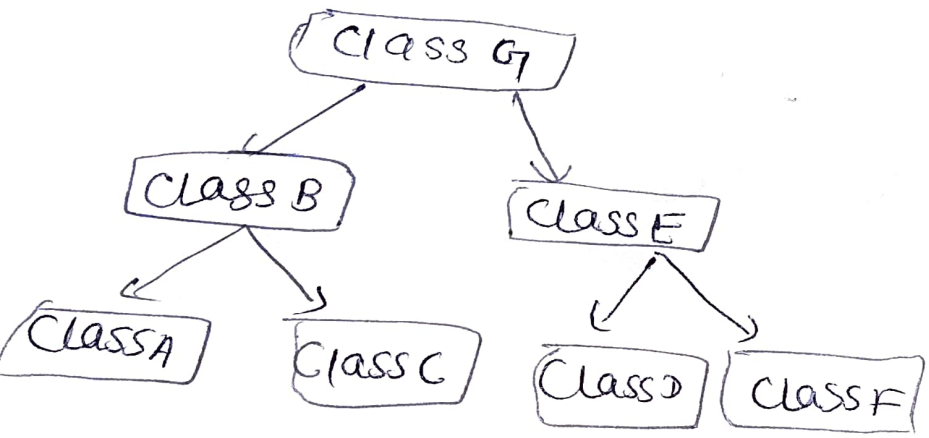
Diamond Problem :



Solution :- use virtual keyword. virtual class remove the ambiguity.

HIERARICAL INHERITANCE :

In this type of inheritance more than one sub class is inherited from a single Base class. i.e. more than one derived class is created from a single Base class.



POORNIMA

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

```
Class Shape
   { Public :
       Shape ( )
       {
        Cout << " Base class shape " << endl;
       }
   };
Class Rectangle : Public Shape
       { Public :
           Rectangle ( )
        {
         Cout << " Derived Class - Rectangle " << endl;
        }
   };
Class Triangle : Public Shape
 {     public :
        Triangle ( )
        { Cout << " Derived Class - Triangle " << endl
};      }
Class Circle : Public Shape
```

s, Questions & Summary:

Website Ref.:-

```
{   public :
        circle ( )
    {   cout << " Derived class - circle " << endl ;
    }
};
int main ( )
{   Rectangle r ;
 cout << " _____ " << endl ;
 Triangle t ;
 cout << " _____ " << endl ;
 circle c ;
 cout << " _____ " << endl ;
 return 0 ;
}
```

POORNIMA

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

```
Class A
{ Public :
    int x, y ;
    Void getdata ()
    {
      ~~cout~~ Cout << " Enter value " ;
               cin >> x >> y ;
    }
};
Class B : public A
   { public :
       Void Product ()
       { Cout << "product = " << x * y <<e
       }
};
Class C : Public A
    { Publi C :
        Void sub ( )
        {
        Cout << "\n sub = " << x-y
    }
};
```

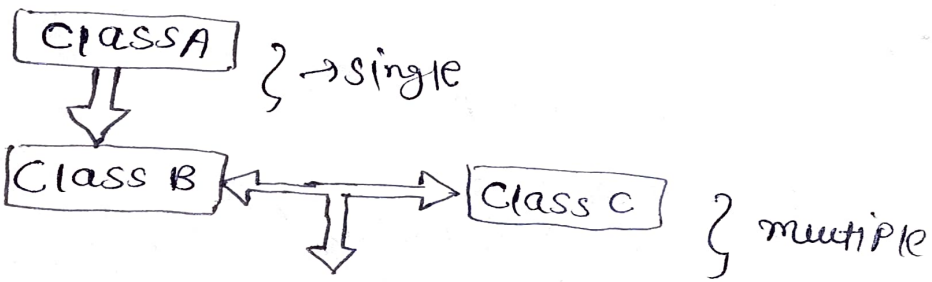
estions & Summary:

Ref.:-

```
int main ( )
{
    B b1 ;
    C c1 ;
  b1.getdata ( ) ;
  b1.product
  b1.getdata ( )
    c1.sub ( )
    return 0;
}
```

HYBRID :

The process of combining more than one of inheritance together while driving subclass program is called hybrid Inheritance. It is also called multiple inheritance.



POORNIMA

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

```
Class A
{ public:
    int x;
};
Class B : public A
{ public:
    B()   // Constructor
    { x=10;
    }
};
Class C
{ public:
    int y;
    C()   // Constructor
    { y=4;
    }
};
Class D: public B, public C
{ public:
    void sum()
    {
        cout << " sum =" << x+y;
    }
};
```

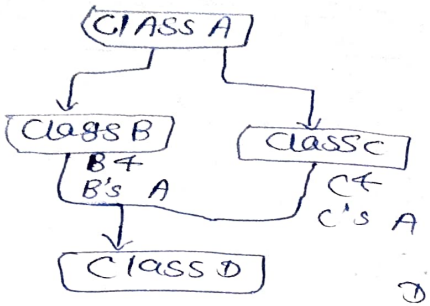
```
int main()
{
    D obj;
obj.sum();
    return 0;
}
```

s, Questions & Summary:

ebsite Ref.:-

~~int mai~~

Virtual Base class :



D has now 2A
- B's A & C's A
{ This lead to ambiguity }

```
Class A
{   Public:
      Void show( )
      { Cout << "Hello from A \n";
      }
};
Class B : Public A
{
};
Class C : Public A
{
};
Class D : Public B; Public C
{
};
int main ( )
{   D obj;
    obj.show ( );
```

O/P → Compile error

POORNIMA

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

How to resolve this issue of ambiguity when class A is inherited ~~Bat~~ in Bath class B and class C. It is declared as a virtual Base class By placing a keyword virtual as.

~~class~~

```
class A
{ public:
    int a;
    A() // constructor
  { a=10;
  }
};
class B: public virtual A
   {
   };
Class C: public virtual A
{
};
Class D : public B , public C
   { };
```

```
int main()
 { obj;
 Cout << "a=" << obj.a
 return 0;
 }
```

O/P = 10

{2 property → reusable, extendable}

Abstract Class :

An Abstract class in C++ is a class that have atleast one pure virtual function (i.e. a function has no definition). the classes inheriting the abstract class must provide a definition for the pure virtual function otherwise the sub class would become an abstract class itself.

```
class Shape
{
    protected :
        int width ;
        int height ;
    public :
    virtual int getArea()=0;
    // virtual int getArea(){};
        Vaid setwidth (int w)
        {   width = w;
        }
    Vaid setheight (int h)
     {   height = h ;
     }
};
Class Rectangle : public Shape
  { public:
    int getArea ( )
     {
     return (width * height);
     }
};
```

```
int main ( )
 { Rectangle r ;
   r. setwidth (5);
   r. setheight (7);
Cout << "Rectangle Area" << r.ge
                          << en
   return 0;
}
```

pure virtual function → derive class
implementation
POORNIMA
pure virtual function → use → base class
redefine → drive class

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

Virtual function : It is a member function of the Base class that you Redefine in a derive class. It is declare using a virtual keyword.

- It is use to tell the compiler to perform dynamic binding or late binding of the function.
- It is necessity to use the single pointer to refer to all the object of different classes. so we create a pointer to the Base class that refer to all the derive objects.

```
Ex: Class A
{ int x=5;
  public:
  void display ()
   { cout << " value of x" << endl;
   }
};
class B: public A
{ int y=10;
  public:
  void display ()
  {
   cout << " value of y" << endl;
  }
};
```

```
int main ()
{
   A *a;
   B b;
   a = &b;
   a -> display ();
   return 0;
}
```

o/p → value of x=5

In this ex.: pointer A is the Base class pointer the pointer can only access the Base class member But not the member of derive class

In C++ permit Base pointer to any object derive from the base class. It cannot directly access to pointer member of the derive class. therefore

Visibility mode → Private, protected, Public

Ideas, Questions & Summary:

/ Website Ref.:-

```
Class A
{
    int x=5;
    public:
  virtual void display()
   {
  cout<<" value of x"<<endl;
   }
};
Class B : public A
        int y=10;
        public:
  void display
{
 cout<<" value of y"<<endl;
}
};
int main()
 {
    A *a;
    B b;
    a=&b;
   a->display();
    return 0;
}
```

o/p → value of y=10

We need virtual ... to access the ... derive class ...

Pure virtual function :

When the function has no definition ... function known as 'do-nothing' function. It is also known as pure-virtual function.

POORNIMA

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

ex: Abstract class

```
class Base
{ public:
  virtual void show()=0;
};
class Derived: Public Base
 { void show()
   {
    Cout<<"Derived Class"<<endl;
   }
};
```

```
int main()
{
  Base *b;
  Derive d;
    b = &d;
  b->show();
  return 0;
}
```

Function overriding :

It is a concept by which you can define a function of same name and same function signature in both base class & derive class which a function definition.

It redefine a function of the base class inside the derive class which override the Base class function.

It override the function at the runtime of the program

in Ideas, Questions & Summary:

rary / Website Ref.:-

ex:

```
class A
{ public:
  void display() //signature
  {
    cout<<"Base class "<<endl;
  }
};
class B : public A
{ public:
  void display()
  {
  cout<<"derived class";
  }
};
```

```
int main()
{
  B obj;
obj.display();
  return 0;
}
```

o/p → derived class

POORNIMA

UNIT-4

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

Constant data member :

Const keyword is used to make any element of a program to constant.

Const variable : If we declare a variable as a Const we can not change its value.

Case I :

```
int main ()
{
    const int a=5;
    a=10;
    return 0;
}
```

⇒ give error {compilation error

Case II :

```
int main ()
{
    const int a=5
    a++;
    return 0;
}
```

⇒ give error (compilation error)

a Const variable must be assigned a value at the ti of declaration.

Ideas, Questions & Summary:

/ Website Ref.:-

Constant are have two types.

1. Constant data member
2. Constant member function

Defining class data member as const :

A Const data member can not be initialized time of declaration or within the member function.

Initialization →

Class Name() : Constant member Name (value)

{

}

Ex:

```
Class Number
{
  private:
   const int x;   // Declaration
   public:
    Number() : x(36) { } // Initialization
   Void display()
   {
    Cout << "x is " << x << endl;
};
};
   int main()
   {
      number num;
       num.display();
       return 0;
   }
```

O/P = 36

POORNIMA

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

Constant member function :

The const member function are declarare which are a constant in the program, the obj called by these function can not be modified.
It is recommended to use const keyword so that the accidently changes to object are avoided

Syntex : data type func-name const ( ) ;

```
class Number
{ private :
  int a ; int b;
  public:
  number ( ) {a=0; b=0;}
  void ~~setter~~ set_a (int num1 )
      { a= num1;
      }
  void set_b (int num2)
      { b = num2;
      }
int get_a (void) const
{ return a;
}
int get_b (void) const
{ return b;
}
};
int main ()
{ Number Num;
  Num.set_a (100) ;
  Num. set_b (200)
Cout << "value of a: " <<
        num.get_a()<<en
Cout << "value of b: " <<
        num.get_b()<<end
return 0;
getch ( );
}
```

Ideas, Questions & Summary:

ry / Website Ref.:-

Static Data member :

static data member are class member that are declared using static keyword
static data member as certain special characteristic (only one value is send to all objects)

Syntax: data type class name :: member_name=value;

```
class x
{
  public:
  static int i;
};
int x :: i = 0;
```

program :

```
#include <iostream>
using namespace std;
struct x
{
  static
  ~~static~~ const int a=76;
};
const int x :: a;
int main()
{
cout << x :: a << endl;
return 0;
}
```

O/P → 76

```
# include <iostream>
Using namespace std;
Class demo
{
  public:
    static int abc;
};
int demo :: abc=10;
int main()
{ cout << " value of abc"
    << demo :: abc << endl;
return 0;
}
```

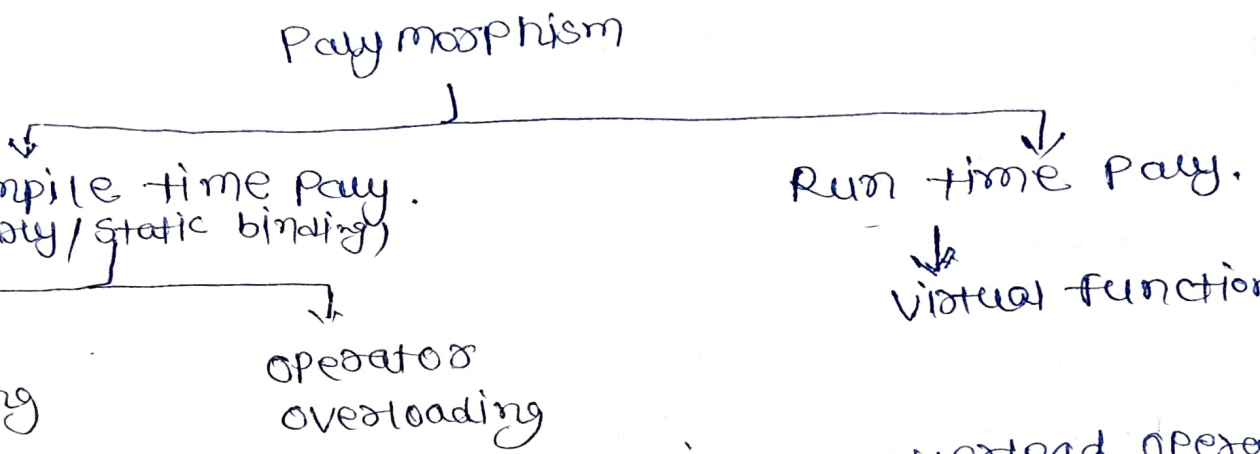
POORNIMA

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

POLYMORPHISM :

It means "many forms".

It occur when we have many classes that are related to each other by inheritance. Polymorphism uses those method that perform different class. This allow us to perform a single action in different way.



Operator overloading : C++ also provide to overload operator

Ex: use ~~digit~~ of + operator in digits

use + operator in the addition of string

(string Hello' + string 'word' = Hello w

unary operator + Binary operator overloading

n Ideas, Questions & Summary:

ary / Website Ref.:-

Unary operator : operator which work on a single operand are called unary operator

ex: Increment operator (++) ; logical not operator (!)
Decrement operator ("--")
Unary minus operator (-)

Binary operator : operator which work on two operands are called binary operator.
ex: addition, subtraction, division

Unary operator overloading :

```
Class Count
{   int value;
    Public :
      Count () : value (5) { }
    Void operator ++ ()
      {  ++ value;
      }
   Void display ( )
{
Cout << "Count" << value << endl;
  }
};
 Void main ( )
{  Count c1 ;
    ++ c1 ;
    c1.display ();
}  getch ();
```

POORNIMA

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

Runtime Polymorphism : this type of polymorphism is achiev by function ~~overloading~~ overriding.
function overriding on the other hand occurs when a derive class has a definition for one of the member function of the base class that base functio said to be overridden.

~~There~~

There are four operator than can not be overload in C++
1. Scope resaluation operator (::)
2. member ~~function~~ selection (.)
3. member selection through pointer to ~~fu~~unction (.*)
4. ternary operator (?:)

Ideas, Questions & Summary:

y / Website Ref.:-

UNIT - 5

POORNIMA

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

Q. What is EXCEPTION Handling in C++.

Ans EXCEPTION Handling in C++ is define as a method that take care of the surprising condition like user time cusses at whatever point a sudden situation happen, there is a moment of the progra control to a uniform function as handler.

Need of EXCEPTION Handling :

Basic keywords in EXCEPTION Handling :

1. throw
2. catch
3. Try

} → ~~What is try throw~~

3. What is try throw catch in C++.

It is define as-

1. throw : ~~when~~ a program throws an exception where problem show up. this is done using a throw keyword.

2. Catch : program that utilize in exception handler t catch an exception. It is ~~add~~ added to the pa of the program where we need this to deal with error.

n Ideas, Questions & Summary:

rary / Website Ref.:-

Try : the try block recognize the code block of which exception will be activated. It fallawed by one or more catch blocks.

Syntax in try/catch Block :

```
Try
  { // the protection code
  }
catch (Exception_name P1)
    { // Catch Block
    }
Catch (Exception_name P2)
    { // Catch Block
    }
```

{ one try block have the multiple catch block }

UNDERSTANDING NEED OF EXCEPTION HANDLING WITH EX..

```
Void main ()
{
  int a=10, b=0, c;
  c = a/b;
Cout << " output is " << c;
  getch ();
}
```

POORNIMA

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

#

```
double divison (int a, int b)
  { if (b==0) {
  throw " divison by zero Condition ! ";
    }
    { return (a/b);
    }
  Void main ()
  { int x = 50;
      Y = 0;
    double z=0;
  try{
      z = divison (x, y);
   Cout << z << endl;
  }
  Catch ( Const char *msg )
  { Cout << msg << endl;
  }
  getch ( );
  }
```

ain Ideas, Questions & Summary:

brary / Website Ref.:-