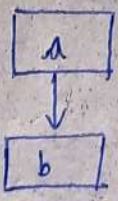
28-06-2024

INHERITANCE:- It is a property of which 1 class objects the properties of another class objects.

TYPES OF INHERITANCE:-

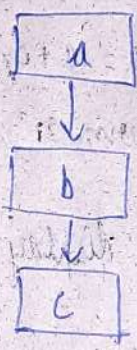
1. Single :-

```
class a
{
}
class b : public a
{
}
```



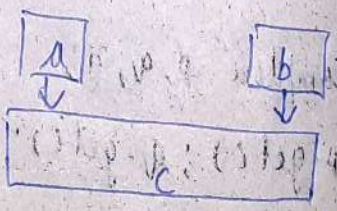
2. Multi level :-

```
class a
{
}
class b : public a
{
}
class c : public b
```



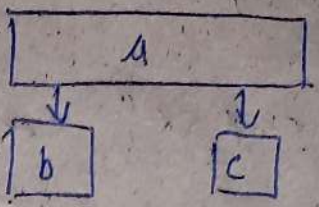
3. Multiple :-

```
class a
{
}
class b
{
}
class c : public a, public b
{
}
```



4. Hirarchical :-

```
class a { };
class b : public a
{
}
class c : public a
{
}
```



Ex :- .cpp

```
-> using namespace std;
#include <iostream>
class abc
{
  private : int a, b;
  public : void get()
           {
             cout << "enter a, b"; cin >> a >> b;
           }
         void display()
         {
           cout << a << "  " << b;
         } cout << endl;
};
class xyz : public abc
{
  private : int x, y;
  public : void read()
           {
             cout << "enter x, y"; cin >> x >> y;
           }
```

~~void display~~

```
void write ()
 {
   cout << x << " " << y;
 }
};
main ()
{
   x y z p;
   p.get (); p.read (); p.display (); p.write ();
}
```

O/P :-

Enter a, b : 1 2
Enter x, y : 3 4
1 2
3 4

MULTI LEVEL INHERITANCE :-

```
-> using namespace std;
# include <iostream>
 class abc                                          .cpp
 {
  private: int a,b;
  public: void get ()
            {
              cout << "enter a,b"; cin >> a >> b;
            }
          void display ()
          {
            cout << a << " " << b;
          }
};
```

```
class xyz : public abc
{
  private : int x, y;
  public : void read ( )
           {
             cout << " enter x, y "; cin >> x >> y;
           }
           void display ( )
           {
             cout << x << "   " << y;
           }
};
class mno : public xyz
{
  private : int m, n;
  public : void in ( )
           {
             cout << " enter m, n "; cin >> m >> n;
           }
           void out ( )
           {
             cout << m << "   " << n;
           }
};
main ( )
{
  mno p;
  p.get ( ); p.read ( ); p.cin (-) p.write ( ); p.in ( );
  p.out ( ); p.display ( );
}
```

-> using namespace std;
#include <iostream>
class abc
{
private: int a, b;
public: void get()
{
cout << "enter a, b";
cin >> a >> b;
}
void display()
{
cout << a << " " << b;
}
};
class xyz
{
private: int x, y;
public: void read()
{
cout << "enter x, y"; cin >> x >> y;
}
void display()
{
cout << x << " " << y;
}
};

.cpp

ab: 1 2
xy: 3 4
m n: 5 6

1 2 3 4 5 6

```
main ()
{
class mno : public x y z, public a b c
{
  private : int m, n;
  public : void in ()
           {
             cout << "enter: m, n";
             cin >> m >> n;
           }
           void out ()
           {
             cout << m << " " << n;
           }
};
main ()
{
 mno p;
p.get (); p.read (); p.in (); p.out (); p.display ();
 p.write ();
}
```

-> Using namespace std;
include <iostream>
class abc
{
private: int a,b;
public: void read()
{
cout << "enter a, b";
cin >> a >> b;
}
void write()
{
cout << a << " " << b;
}
};
class xyz
{
private: int x, y;
public: void read()
{
cout << "enter x, y";
cin >> x >> y;
}
void write
{
cout << x << " " << y;
}
};

```
class mno : public xyz, public abc
{
  private: int m, n;
  public : void in()
           {
             cout << "enter m, n";
             cin >> m >> n;
           }
           void out()
           {
             cout << m << "  " << n;
           }
};

main()
{
  mno p;
  p.abc :: read(); p.xyz :: read(); p.in();
  p.abc :: write(); p.xyz :: write(); p.out();
}
```

:: -> scope resolution operator