Inheritance

Syntax of derived class declaration

Class DerivedClass : [visibility mode]Bare class

1. Class B: public A // public derivation
   {
      // members of B
   }
2. Class B: private A // private derivation
   {
      // members of B
   }
3. Class B : A // private derivation by default
   {
      // members of B
   }

When a base class is privately inherited by a derived class, ‘public members’ of the base class become ‘private members’ of the derived class and therefore the public members of the base class can only be accessed by the member function of the derived class. They are inaccessible to the objects of the derived class.

```
# include<iostream.h>
 Class B
{
   int a ;
   public :
     int b ;
     void get_ab (void) ;
     int show_a (void) ;
} ;
 Class D : public B
{
    int c ;
    public c :
      void mul (void) ;
      void display (void);
} ;
 void B :: get_ab (void)
{
    a=5 ; b= 10 ;
}
 int B :: get_a ()
{
    return a ;
}
```

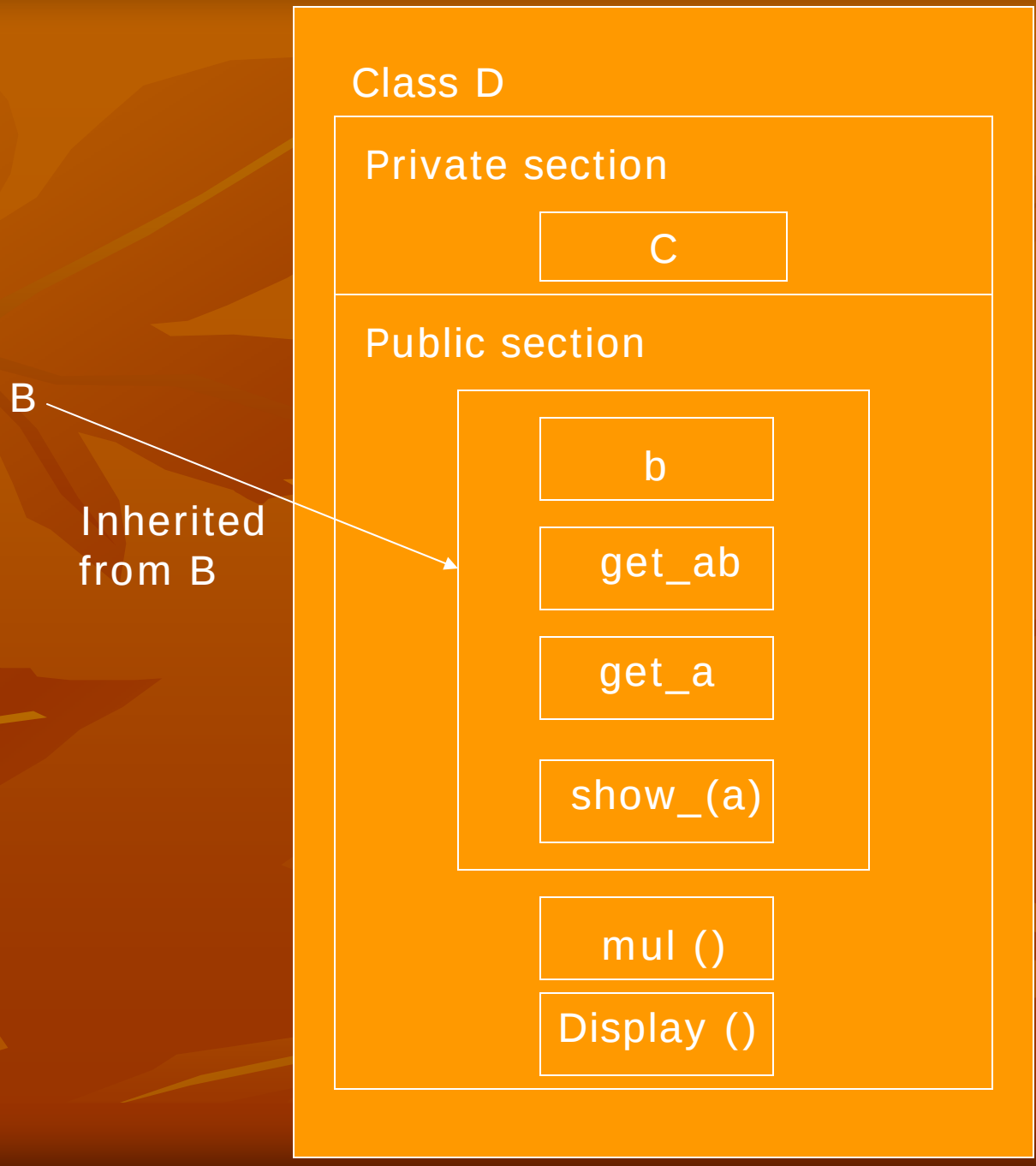
```
Void B :: show_a ()
{
  cout << “a=“ << a << “\n” ;
}
Void D :: mul ()
{
  c=b*get_a () ;
}
Void D :: display()
{
  cout << “a=“ << get_a () << “\n” ;
  cout << “b=“ << b << “\n” ;
  cout << “c=“ << c << “\n\n” ;
}
int main()
{
   D d;
   d. get_ab () ;
   d. mul () ;
   d. show_a () ;
   d. display () ;
   d.b =20
   d. mul () ;
   d. display () ;
  return 0 ;
}
```

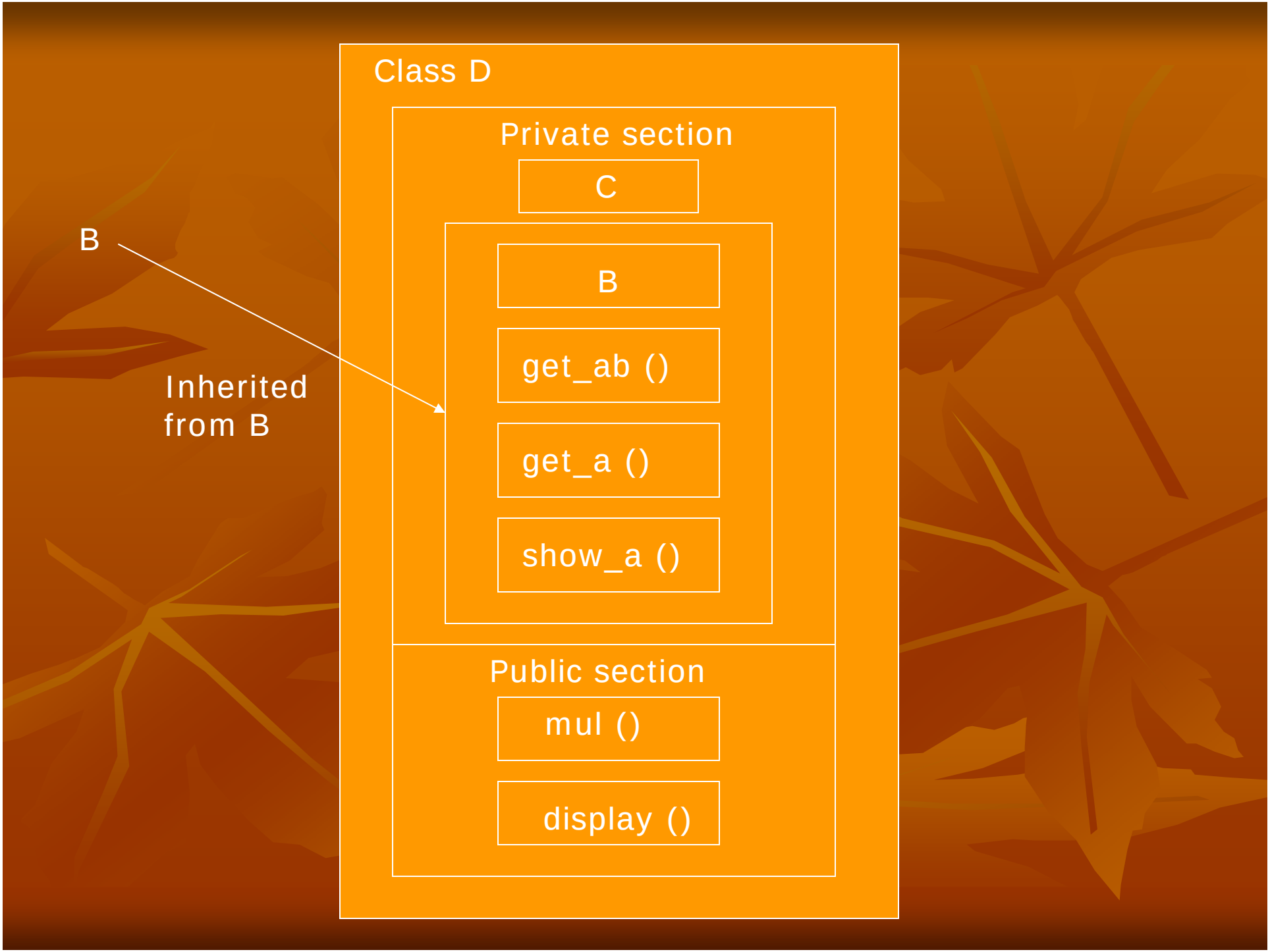
Output :

a=5

a=5
b=10
c=50

a=5
b=20
c=100




Class D
Private section
C
B
get_ab ()
get_a ()
show_a ()
Public section
mul ()
display ()
B
Inherited
from B

```
#include<iostream.h>
Class B
{
   int a ;
   public :
     int b;
     void get_ab () ;
     int get_a (void) ;
     void show_a (void) ;
} ;
Class D : private B
{
   int c ;
   public :
      void mul (void) ;
      void display (void) ;
} ;
Void B : get_ab (void)
{
   cout << “Enter values for a and b :” ;
   cin >> a >> b ;
}
```

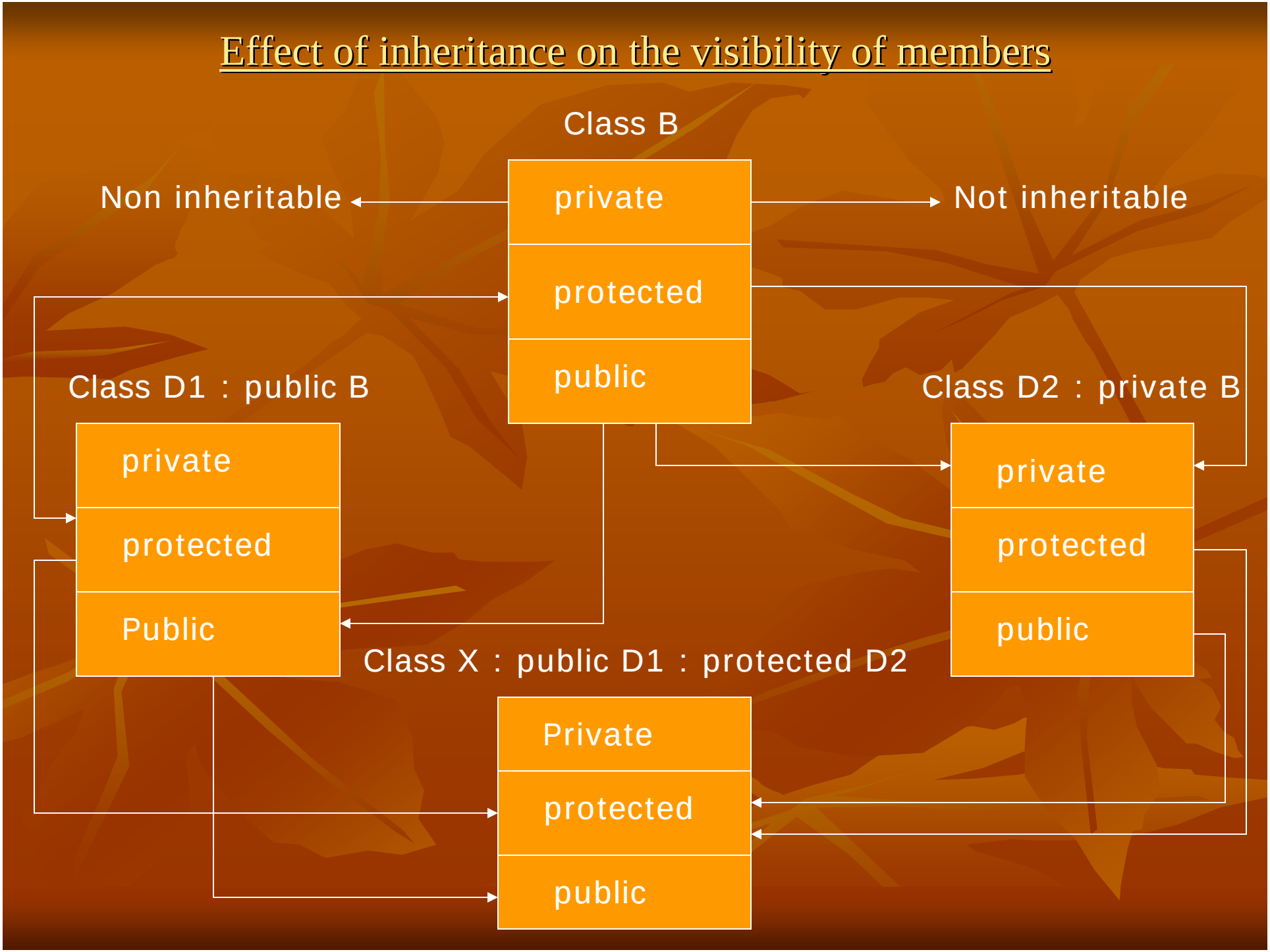
```
int B :: get_a()
{
   return a ;
}
Void B :: show_a()
{
   cout << “a=“ << a << “\n” ;
 }
Void D :: mul ()
{
   get_ab() ;
   c=b*get_a() ;
}
Void D :: display()
{
   show_a ;
   cout << “b=“ << b << “\n”
        << “c=“ << c << “\n\n” ;
}
```

```
int main()
{
  D d ;
 // d.get_ab() ;        WON'T WORK
    d.mul() ;
 // d.show_a() ;        WON'T WORK
    d.display() ;
 // d.b = 20 ;          WON'T WORK
    d.mul() ;
    d.display() ;
    return 0 ;
}
```

Output

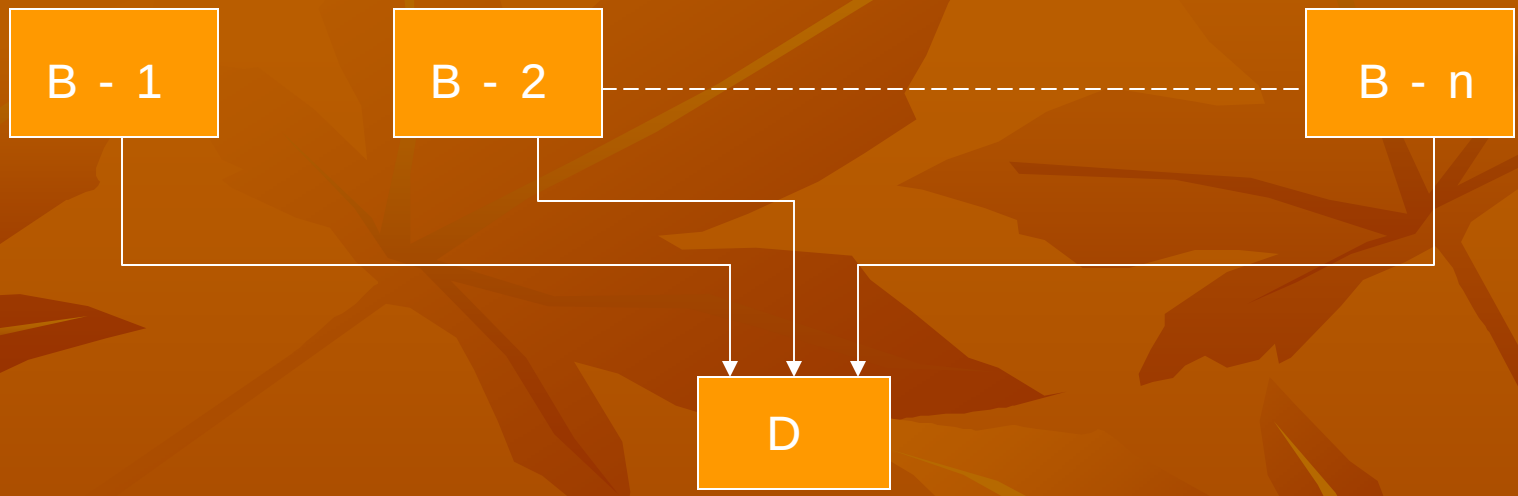
```
Enter values for a and b : 5 10
a = 5
b = 10
c = 50

Enter values for a and b : 12 20
a = 12
b = 20
c = 240
```

Effect of inheritance on the visibility of members



Base class visibility	Derived class visibility		
	Public derivation	Private derivation	Protected derivation
Private	Not inherited	Not inherited	Not inherited
Protected	Protected	Private	Protected
Public	Public	Private	Protected

Multiple inheritance



Class D : visibility B-1, visibility B-2 ----

{

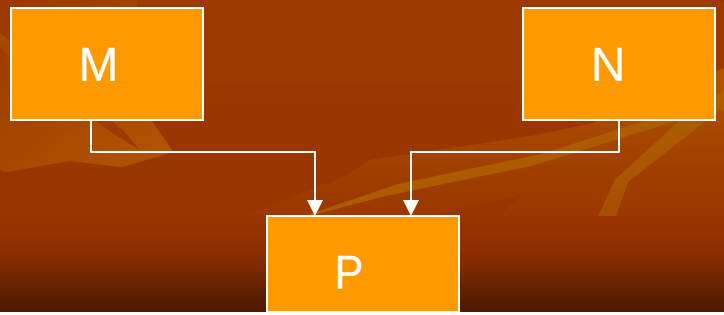
// Body of D

}

visibility → public

visibility → private

e.g. Class P : public M , public N



Ambiguity Resolution :

```
Class M
{
   public :
     void display (void)
     {
        cout << “class M \n” ;
     }
} ;
Class N
{
   public :
     void display (void)
      {
         cout << “class N \n” ;
      }
} ;
```

```
Class P : public M , public N
{
   public :
      void display (void)    // overrides display of M & N
   {
       M :: display () ;
   }
} ;
```

We can now use the derived class as follows :

```
int main ()
{
    P p ;
    p.display () ;
}
```