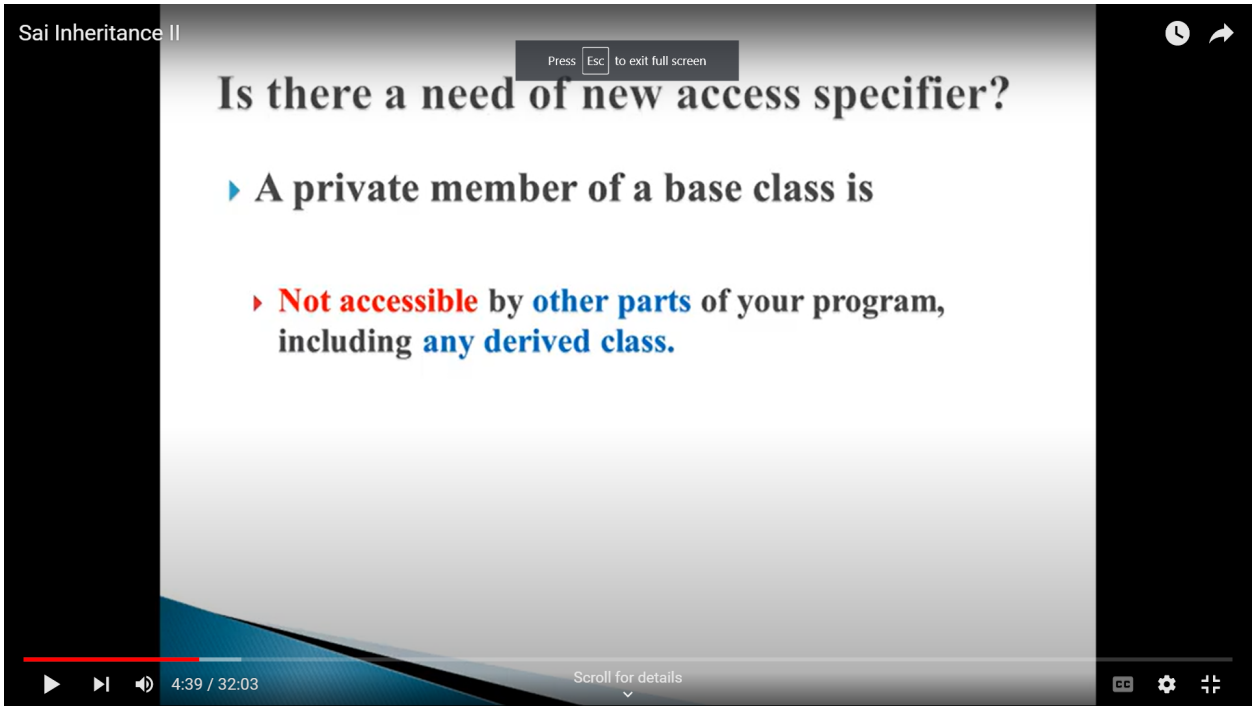
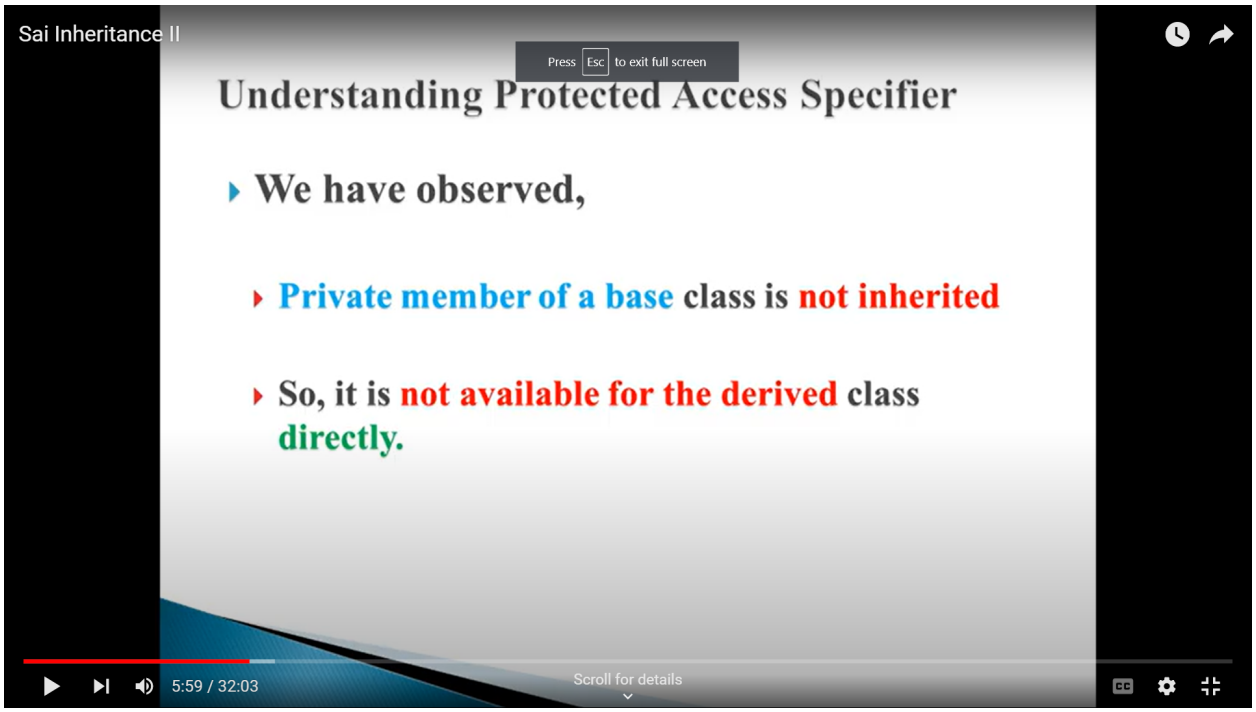
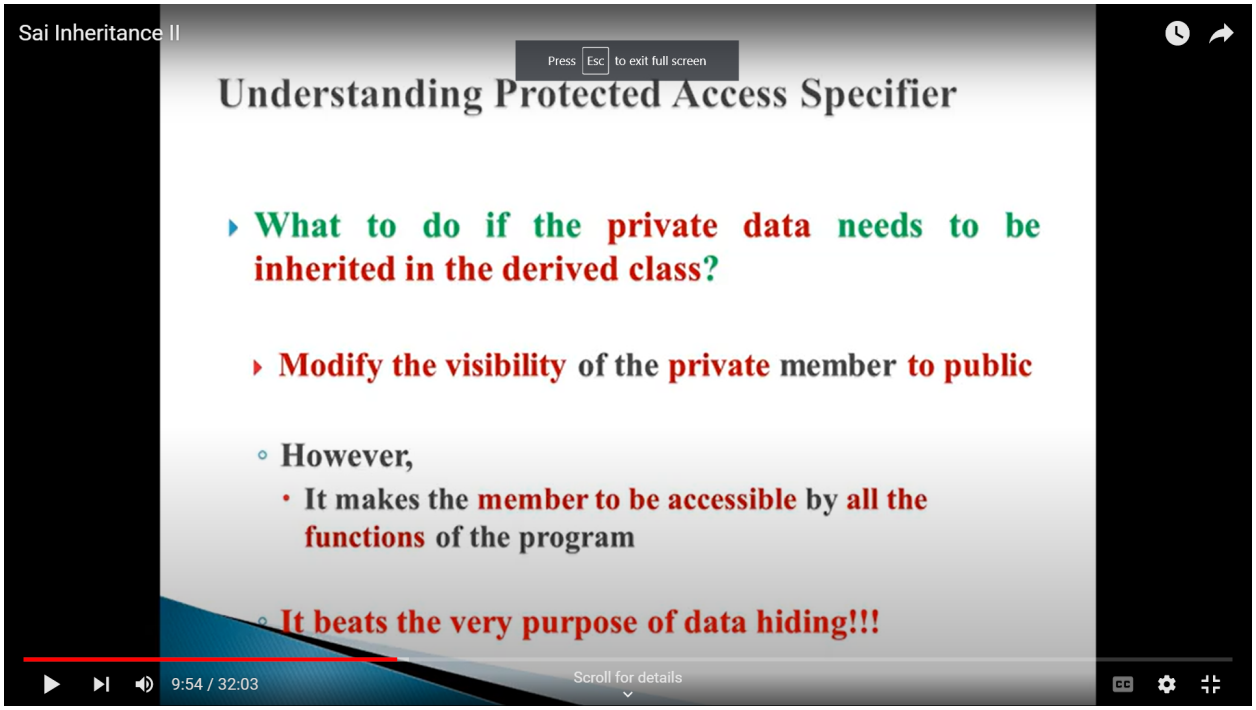
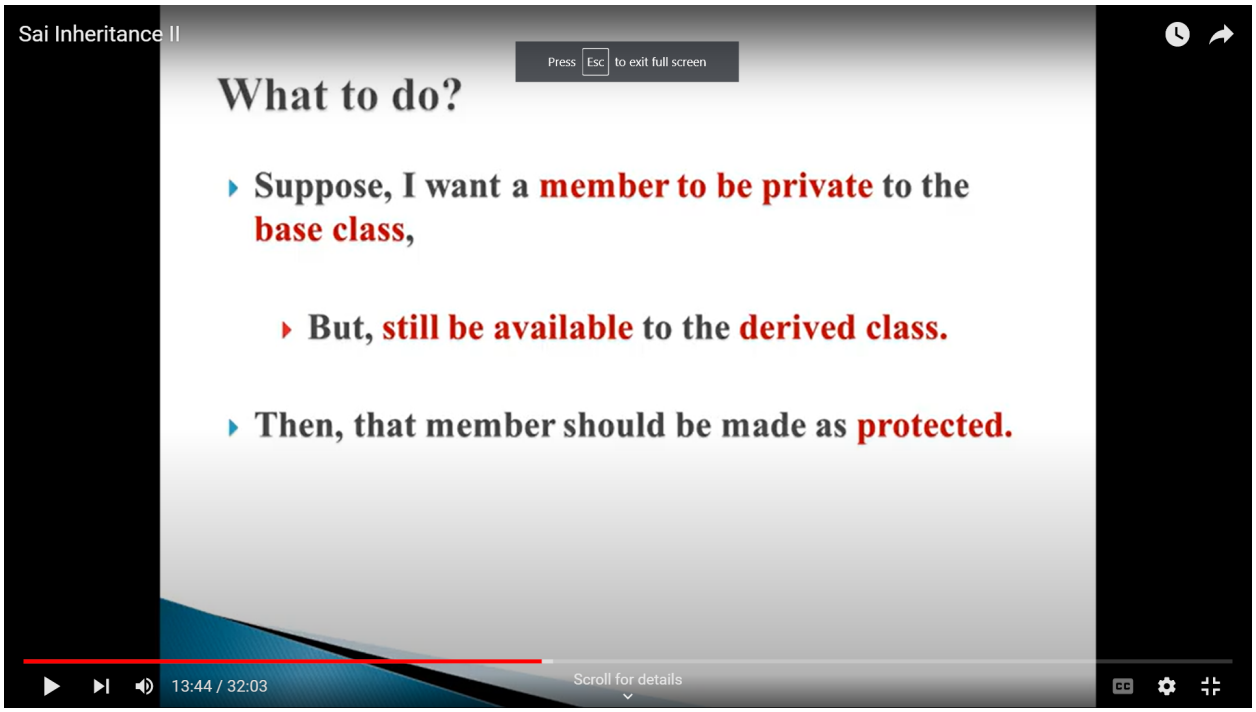
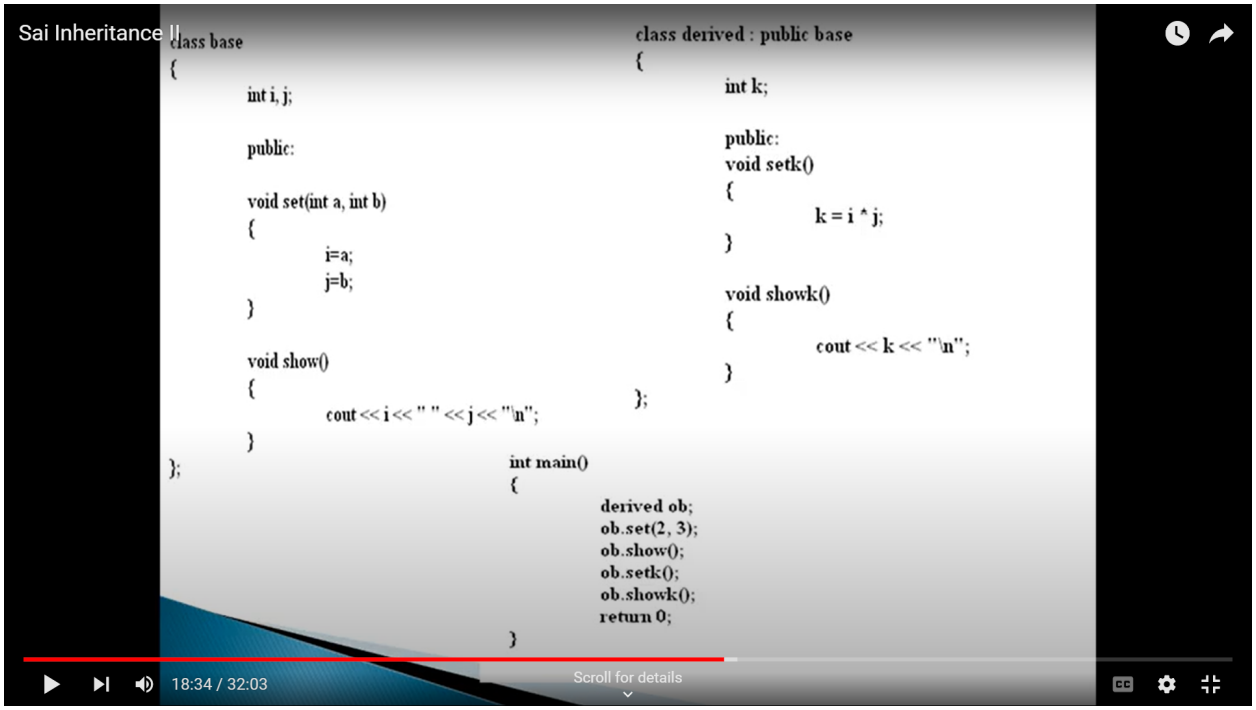
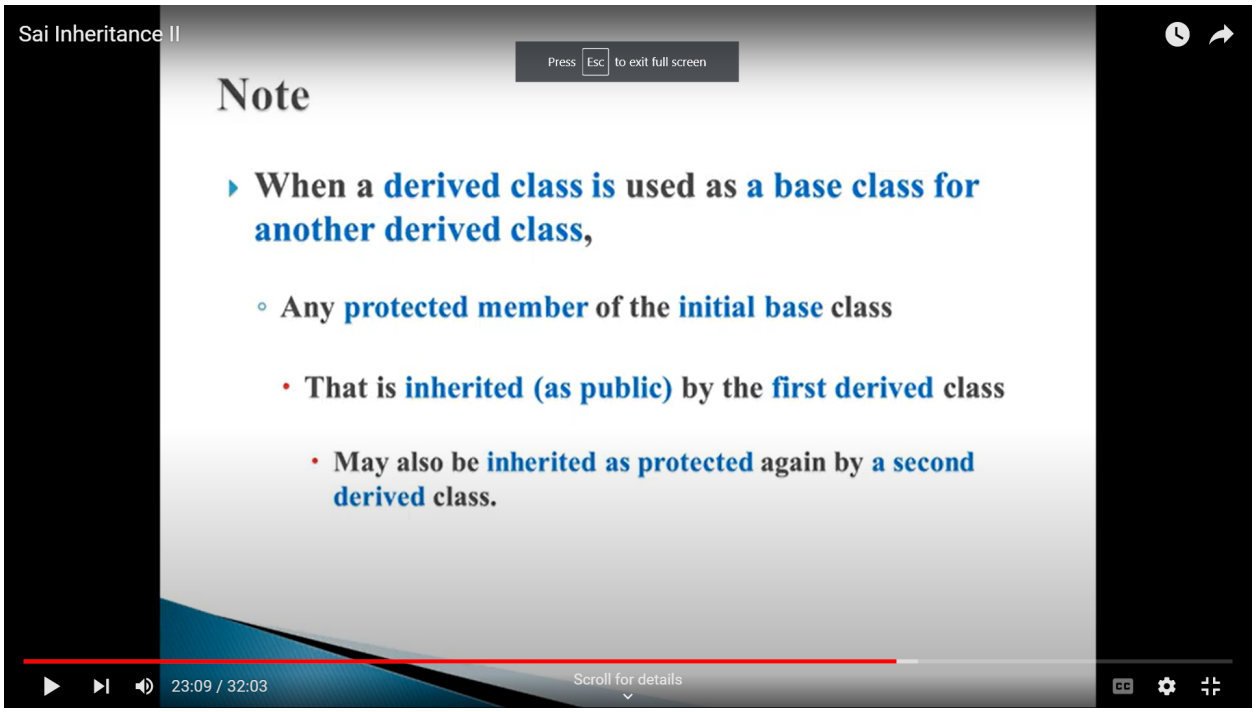

Sai Inheritance II
Press Esc to exit full screen
Is there a need of new access specifier?
A private member of a base class is
Not accessible by other parts of your program, including any derived class.
4:39 / 32:03
Scroll for details


Sai Inheritance II
Press Esc to exit full screen
Understanding Protected Access Specifier
We have observed,
Private member of a base class is not inherited
So, it is not available for the derived class directly.
5:59 / 32:03
Scroll for details

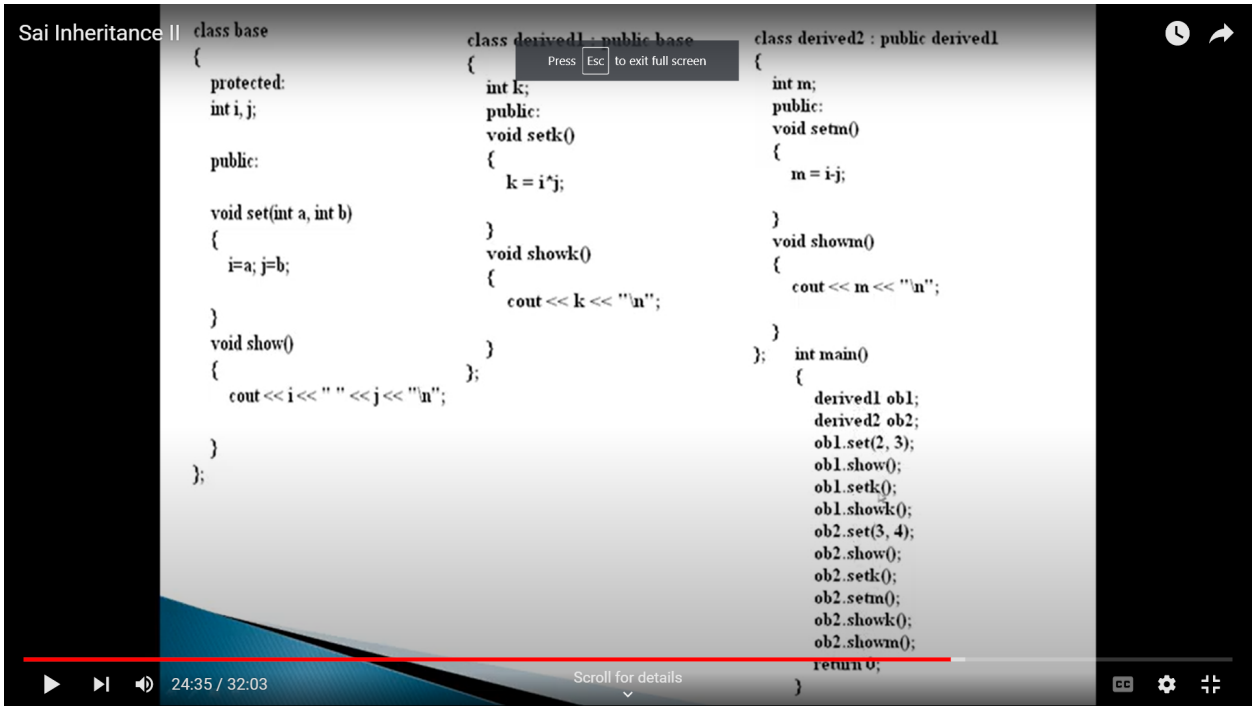

Sai Inheritance II
Press Esc to exit full screen
Understanding Protected Access Specifier
What to do if the private data needs to be inherited in the derived class?
Modify the visibility of the private member to public
However,
It makes the member to be accessible by all the functions of the program
It beats the very purpose of data hiding!!!
9:54 / 32:03
Scroll for details

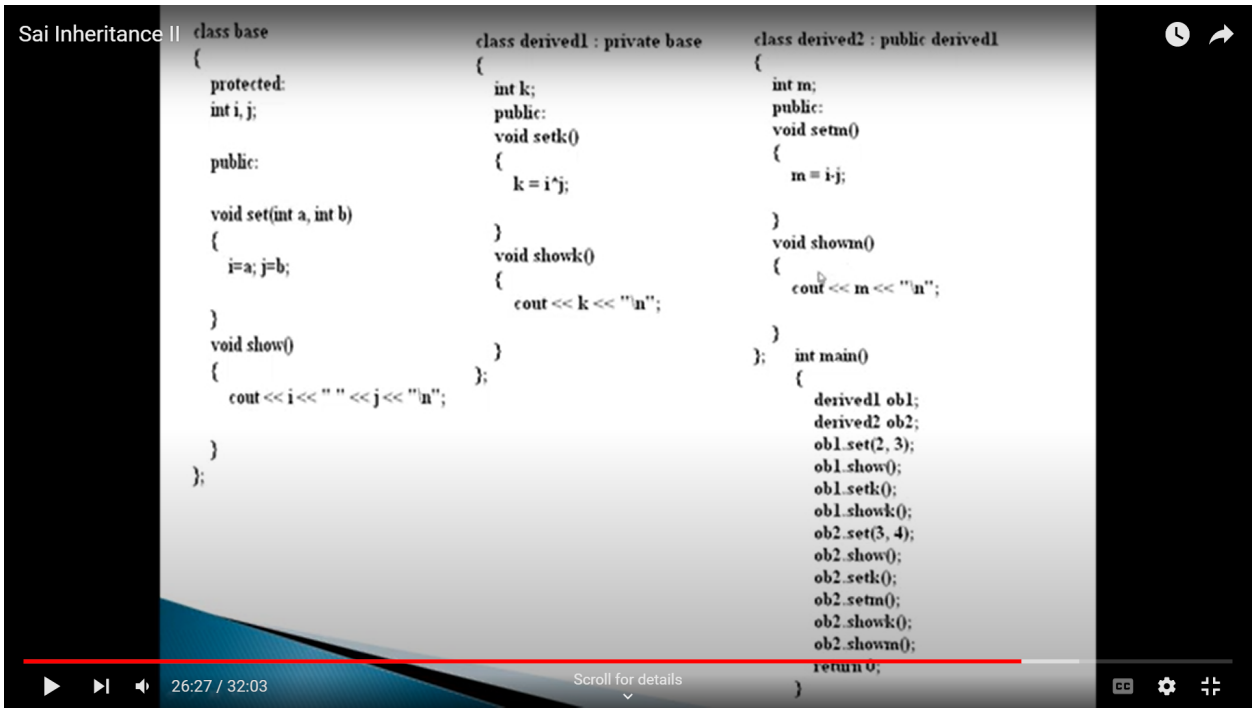

Sai Inheritance II
Press Esc to exit full screen
What to do?
Suppose, I want a member to be private to the base class,
But, still be available to the derived class.
Then, that member should be made as protected.
13:44 / 32:03
Scroll for details

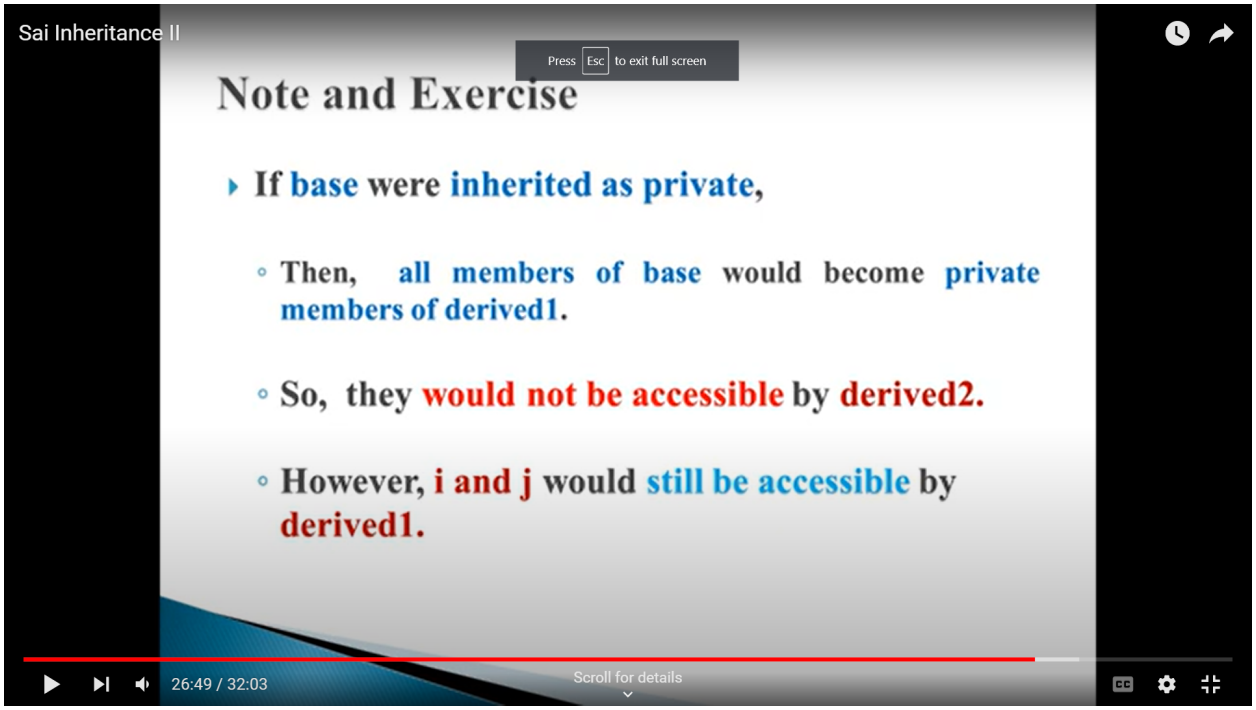

Sai Inheritance II
class base
{
int i, j;
public:
void set(int a, int b)
{
i=a;
j=b;
}
void show()
{
cout << i << " " << j << "\n";
}
};
class derived : public base
{
int k;
public:
void setk()
{
k = i * j;
}
void showk()
{
cout << k << "\n";
}
};
int main()
{
derived ob;
ob.set(2, 3);
ob.show();
ob.setk();
ob.showk();
return 0;
}
18:34 / 32:03
Scroll for details

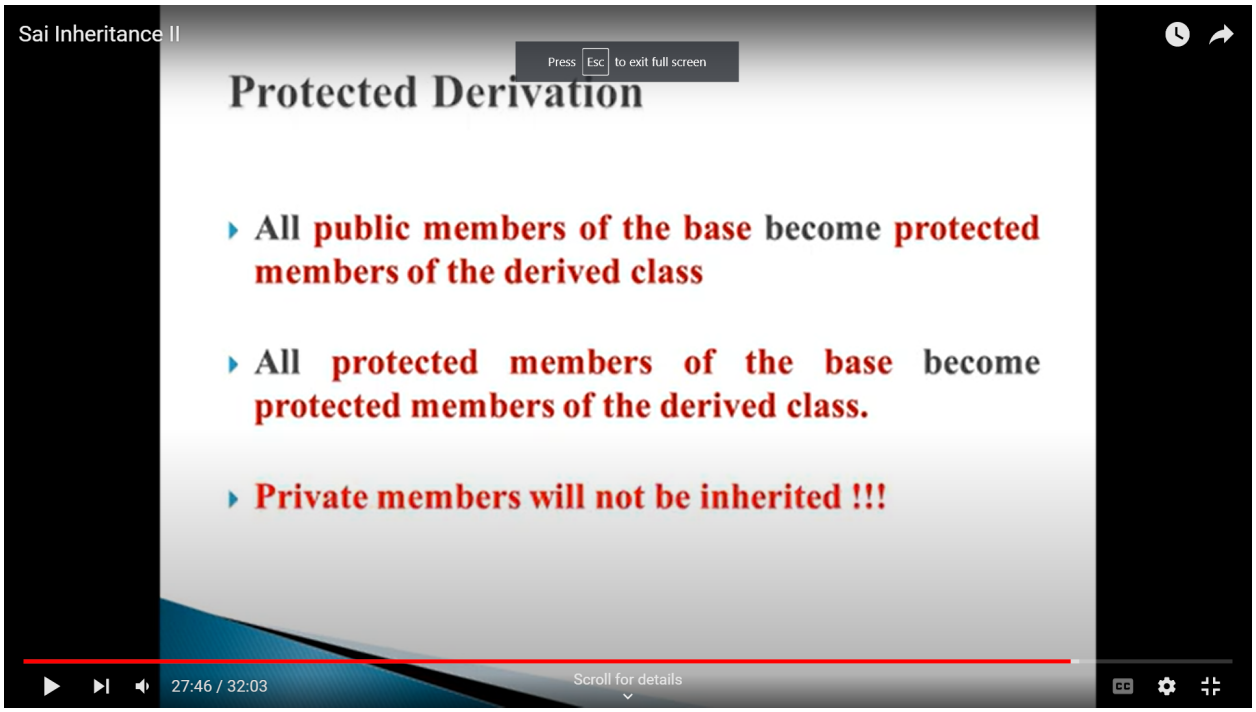

Sai Inheritance II
Press Esc to exit full screen
Note
When a derived class is used as a base class for another derived class,
Any protected member of the initial base class
That is inherited (as public) by the first derived class
May also be inherited as protected again by a second derived class.
23:09 / 32:03
Scroll for details

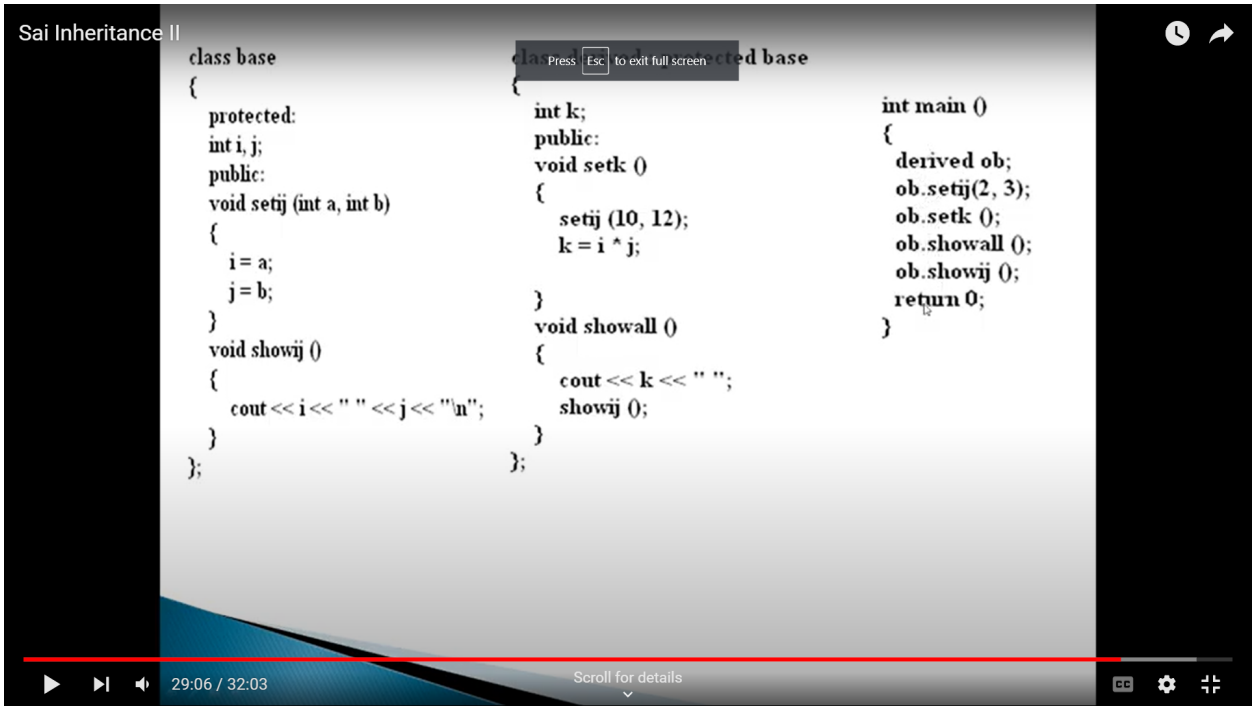
d

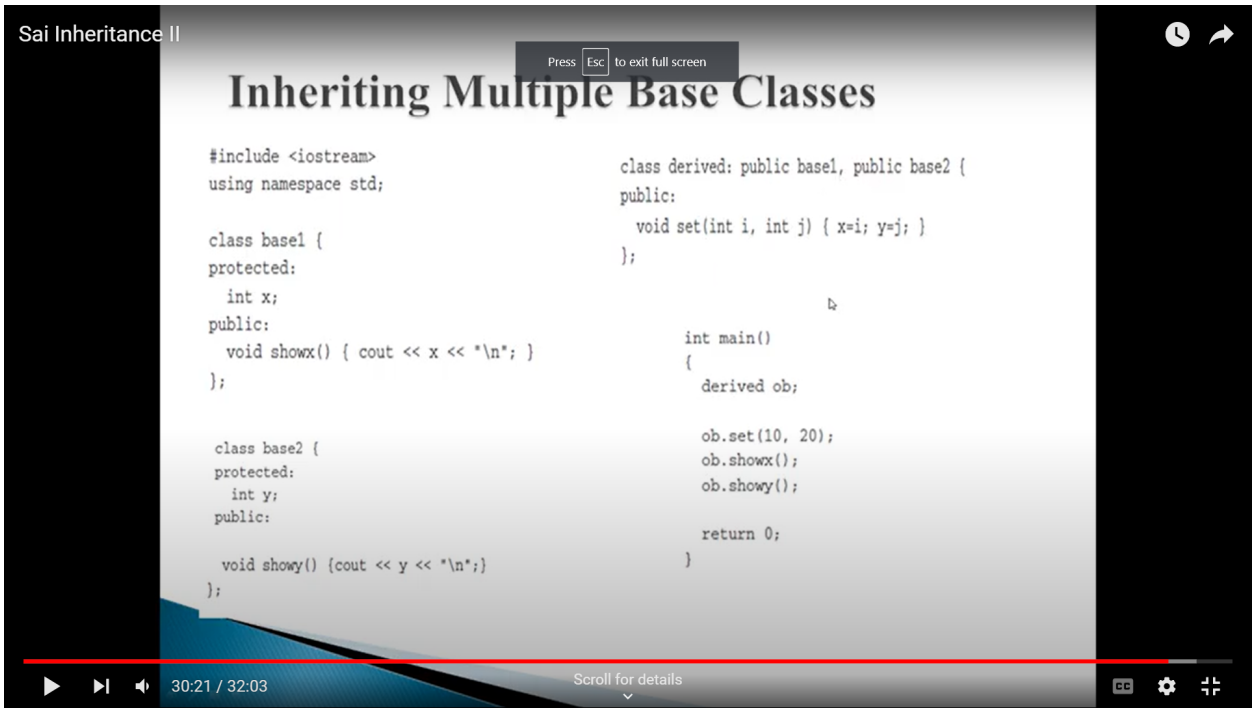

Sai Inheritance II
class base
{
protected:
int i, j;
public:
void set(int a, int b)
{
i=a; j=b;
}
void show()
{
cout << i << " " << j << "\n";
}
};
class derived1 : public base
{
int k;
public:
void setk()
{
k = i*j;
}
void showk()
{
cout << k << "\n";
}
};
Press Esc to exit full screen
class derived2 : public derived1
{
int m;
public:
void setm()
{
m = i-j;
}
void showm()
{
cout << m << "\n";
}
};
int main()
{
derived1 ob1;
derived2 ob2;
ob1.set(2, 3);
ob1.show();
ob1.setk();
ob1.showk();
ob2.set(3, 4);
ob2.show();
ob2.setk();
ob2.setm();
ob2.showk();
ob2.showm();
return 0;
}
24:35 / 32:03
Scroll for details


Sai Inheritance II
class base
{
protected:
int i, j;
public:
void set(int a, int b)
{
i=a; j=b;
}
void show()
{
cout << i << " " << j << "\n";
}
};
class derived1 : private base
{
int k;
public:
void setk()
{
k = i*j;
}
void showk()
{
cout << k << "\n";
}
};
class derived2 : public derived1
{
int m;
public:
void setm()
{
m = i-j;
}
void showm()
{
cout << m << "\n";
}
};
int main()
{
derived1 ob1;
derived2 ob2;
ob1.set(2, 3);
ob1.show();
ob1.setk();
ob1.showk();
ob2.set(3, 4);
ob2.show();
ob2.setk();
ob2.setm();
ob2.showk();
ob2.showm();
return 0;
}
26:27 / 32:03
Scroll for details


Sai Inheritance II
Press Esc to exit full screen
Note and Exercise
If base were inherited as private,
Then, all members of base would become private members of derived1.
So, they would not be accessible by derived2.
However, i and j would still be accessible by derived1.
26:49 / 32:03
Scroll for details


Sai Inheritance II
Press Esc to exit full screen
Protected Derivation
All public members of the base become protected members of the derived class
All protected members of the base become protected members of the derived class.
Private members will not be inherited !!!
27:46 / 32:03
Scroll for details


Sai Inheritance II
Press Esc to exit full screen
class base
{
protected:
int i, j;
public:
void setij (int a, int b)
{
i = a;
j = b;
}
void showij ()
{
cout << i << " " << j << "\n";
}
};
int k;
public:
void setk ()
{
setij (10, 12);
k = i * j;
}
void showall ()
{
cout << k << " ";
showij ();
}
};
int main ()
{
derived ob;
ob.setij(2, 3);
ob.setk ();
ob.showall ();
ob.showij ();
return 0;
}
29:06 / 32:03
Scroll for details


Sai Inheritance II
Press Esc to exit full screen
Inheriting Multiple Base Classes
#include <iostream>
using namespace std;
class base1 {
protected:
int x;
public:
void showx() { cout << x << "\n"; }
};
class base2 {
protected:
int y;
public:
void showy() {cout << y << "\n";}
};
class derived: public base1, public base2 {
public:
void set(int i, int j) { x=i; y=j; }
};
int main()
{
derived ob;
ob.set(10, 20);
ob.showx();
ob.showy();
return 0;
}
30:21 / 32:03
Scroll for details