FUNDAMENTALS OF OBJECT ORIENTED PROGRAMMING

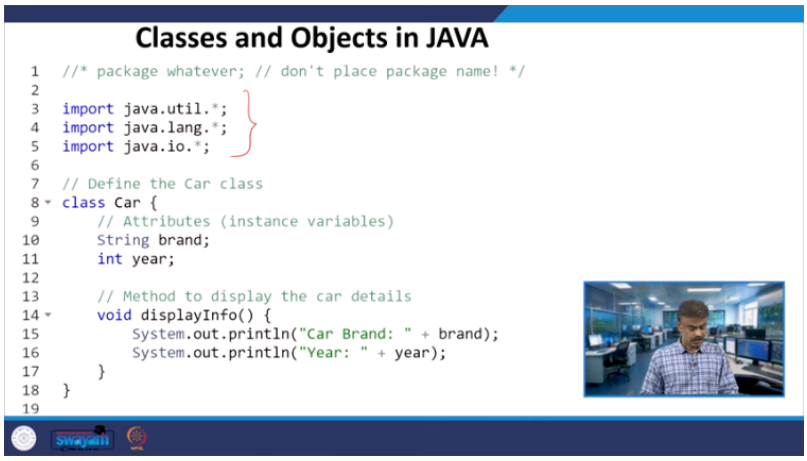
Lecture04

Lecture 4: Introduction to Classes and Objects in Java

So, welcome to lecture 4 introduction to object oriented programming. So, in the last few lectures, so we talked about the classes and objects in C++. So, we have run several codes and now what is the syntax in Java, right? Another object oriented programming is called Java. So, we are going to see the syntax.

As I already defined, when you define a class, the only difference is in C++ you have a semicolon. So if I am writing the C++ same the class I have to put a semicolon over here. So whereas you should not put a semicolon in Java. So that is the first difference in the class. Rest of them almost similar.

So you can see I have imported several functions utility, lang and input output. So here you have class car. Car is the name of the class. So, under this you have two data members. So, here you have a data member one is brand under string another one is year under integer.



So, you have two data members one is brand another one is year one is string another one is integer. So here we call this as a method. So this is a method. In C++ we call it as member function. So in Java it is nothing but a method.

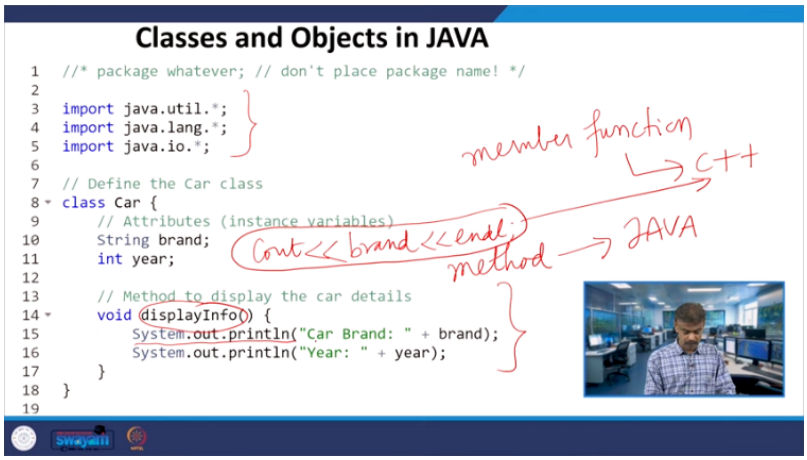
So this display info is a method. Whereas in the last few lectures in C++ we call this as a member function. Both are equivalent. So I call this in C++. Alright.

So here you have a method called display info. More or less similar. Alright. So syntax is C out. We used to write C out in C++.

And in Java it is system.out.println. In stands for new line. Right. System.out.println. So there you use C out.

The same syntax in C++. If we write like this C out brand. And then you put endl. Exactly the same syntax. This is the similar syntax in C++.

Similar syntax in C++. And in Java you have system.out.println. This is like a quote you are writing. We can write in C++ also. That is understood.



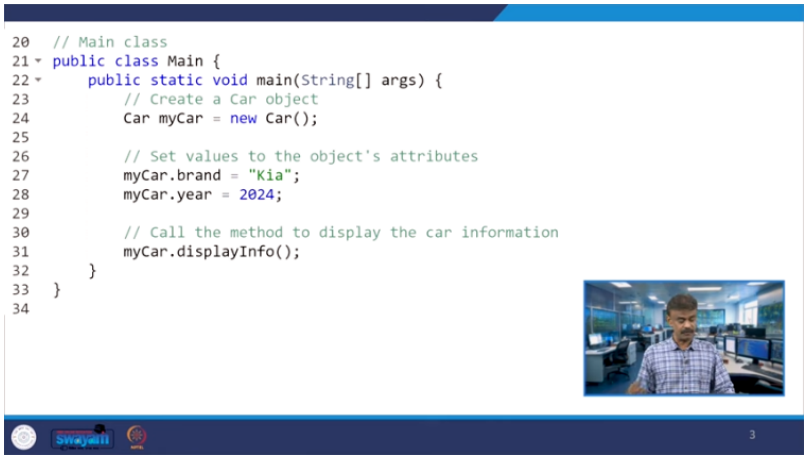
I did not write it. In the code you are writing. Whatever you are writing in the code will be printed. And then you have plus symbol and brand. All right.

So, exactly similar. We have already seen several C++ syntax. So, you are now familiar. So, cout, brand, endl, equivalent. And similarly, we are printing system.out.println year.

All right. System.out.println year. And what is the year? Correct. So, we have two member data.

And you have one member function. The class is over. The class is called car. Right. The car class has two member data.

In fact, I have to call data member in Java and method display info, which is equivalent to member function in C++. So now. This is the driver class. Right. Which is having main.



Whenever you are having a main in Java, you call this as a driver class. Right. So you can find the main. Right. So you can find the main.

Correct. So this main is name of the class, capital M. And here, like a void main or int main when you are writing in C++. So here you have void main. Rest of the thing we will see in the due course. Right now no need to worry because our focus is on class and object.

So here in Java because it is dynamic in nature. So when you are creating an object. When you are creating an object. So my car is an object. And then dynamically we have to allocate a memory.

Or automatically it will be allocated when you write new. And the next one is slowly I am introducing this constructor. Right. So here my car is object. Right.

And here you have class. The new operator will take one example in the case of array. Right. And here this is first time we are seeing this is called constructor. Anyway, we will define constructor extensively, right.

Suppose I declare, let us assume in C++, int a of 10, right, array of 10 elements, static array. So, what about a dynamic array? If I ask you to create a dynamic array, what will you do? int star a, right, a equal to new int of 10. So you might have used this new operator while creating a dynamic array.

This is a dynamic array in C++. So this is a static array. So in the case of Java, everything is dynamic. Suppose I want to write the same code. So this is a C++ code.

So in Java, what you will do? You use like this int. Array notation instead of pointer I will put this int a. And then you are writing a equal to new int of 10. Same. Almost the same code except the star.

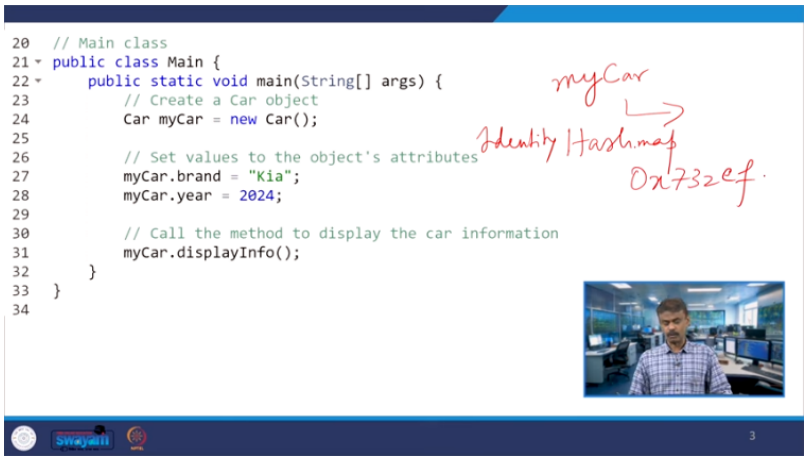
Right. So this is in the case of Java. So in Java everything we have to talk in terms of dynamic. Right. This is ruled out.

I cannot declare like this in Java. Right. So similarly when I have this the object my car. You have object. and car is the class.

Now I have given some example of new in C++. So dynamically create a memory. The new will dynamically create a memory. So my car my capital C here. So object.

So this object will be located in some address. Right. So, here in Java we call that is a identity hash map. I will write completely. Identity hash map.

You call it as address in C++. Right. So, here is a identity hash map. So, some address let us say 0x732ef something like that. So, in this address your my car object



Alright. Will be stored. Right. And line number 27 and 28. We have the data members.

Brand. My car dot brand. So I give some brand name. And my car dot year 2024. So now my car is invoking display info.

My car object. Which is invoking. So dot operator same. Exactly same what we have studied in C++. my car dot operator display info right so when it calls so you can see that brand is kia and year is 2024 right so that will be printed system.out.println we want to print these two values brand and year so when i run the code i will get this as a output

All right. So car brand is Kia and year is 2024. So we will see. Now I introduce. I'll use escape.

I'll go to the Eclipse program. If you open Eclipse. Right. So Java. This is called Java Eclipse.

This environment is called Java Eclipse. So here you go. The same program that I have written. So here you have class car. Right.

So here you have class car. Class car has two member data or data member brand and year. Right. And then you have a method called display info. Correct.

Line number 15 to 18. Then when you come to the main program, we call it as main one. Right. So we have car. Car is the class and my car is the object.

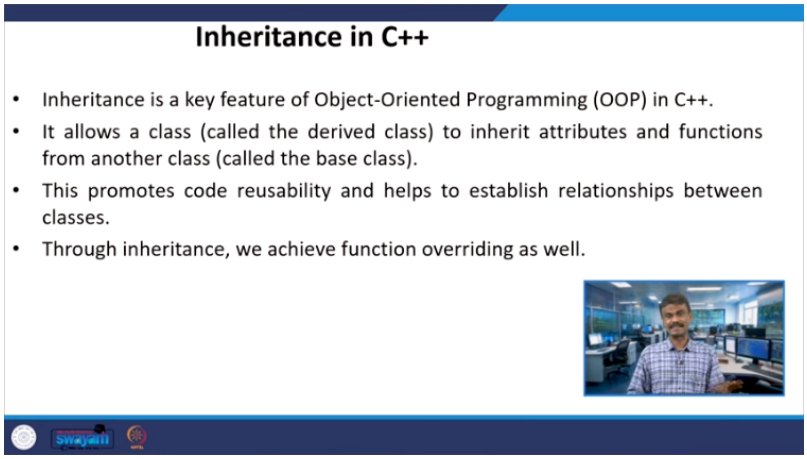
Right. So we are creating, we are instantiating an object called my car. under class car and a new operator and you have a constructor. So line number 28 and 29, the my car, which is having two data member is getting the value, right? My car dot brand is Kia and my car dot year is 2024.

So now with line number 32, my car is invoking display info. My car is invoking display info. When it is invoking display info, it goes line number 15 and there are two printout statements. Two print statements. System.out.println car brand, so brand value.

What is the brand name? Kia. And system.out.println year is 2024. So when I run this code, So here you have, you can go here and then run this code.

You can see the output in the bottom. Car brand is Kia and year is 2024. So this is under Java Eclipse. So you can install Java Eclipse and then it's a good compiler for Java compiler. So you can run the code.

So this is in-house. You can install this in your machine or in your computer and then you start running this code, right. So, more or less we had seen the objects and classes. Extensively we saw in C++, I have introduced Java. So, you can write the basic programs in Java as well, right.



So, at one point of time, I will introduce all the programs, right or all the concepts, with the help of both C++ and Java and at what point of time right. So, we will follow only C++. So, it is understood both are almost same right. So, once you get that feeling.

So, you can start writing the code only in C++, but whenever there is a difference or some of the functionalities available only in C++ not in Java or vice versa. So, that point of time I will talk about. that right so now the next concept is inheritance in C++ so extensively I talked about the code reusability in the last few classes right so that means somebody has written class A right and I am the new user I have written class B so can I use some properties of class A yes you can do that

that is called as inheritance that means B is taking the properties of A Right. So one of the key feature in OOP, inheritance is one of the key feature in OOP.

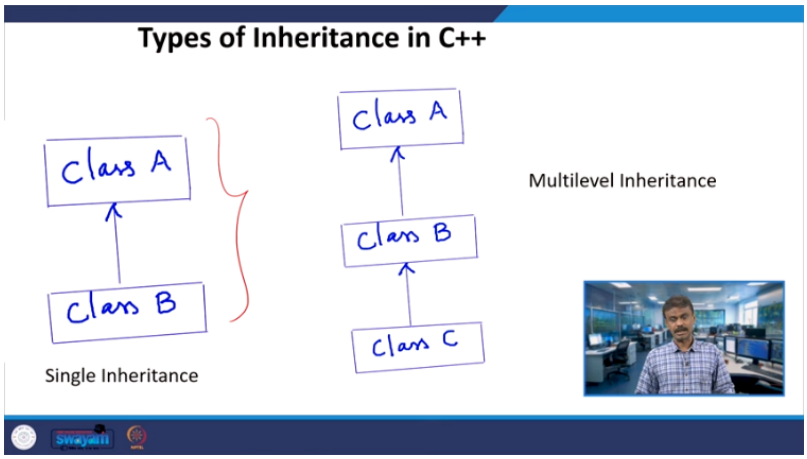
So in this case, A will be called as a base class. A will be called as a base class and B will be called as a derived class. So B is using all the properties of A. Right. Unless otherwise I talked about this access modifier. When we, I mean slowly I introduce the private and public.

When everything is public, assume that all the member data and member functions are public, B can use. If it is private, only A can use. So you cannot do the derivation. So those things we will talk once I introduce the access modifiers. Like public, private, protected, etc.

So right now we know what is when by inheritance. So B can use all the properties of A. right if it is publicly available so one you call it as a base class another one is the derived class so that means we can reuse the code that means that may contain a member function that may contain a member data so you are establishing a relationship between the classes right so also you might have studied function overloading right in the basic programming like a procedural oriented programming function overloading when i introduce the class class on objects so we have another concept called function overriding so this can be achieved with the help of inheritance so these are all the basic features of inheritance in c++ So what are all the different types of inheritance in C++?

So this I have told, right? So class B is using the property of class A. Class A and class B. Class B is using the property of class A. This is a notation. You have an arrow. So that means class B is using, let us say, the data member and member functions. Or member data and member functions.

So that inheritance is termed as single inheritance. Or you can also call it as a simple inheritance. Now you are extending. So can I have three classes? Can class B inherit class A and class C inherit class B?



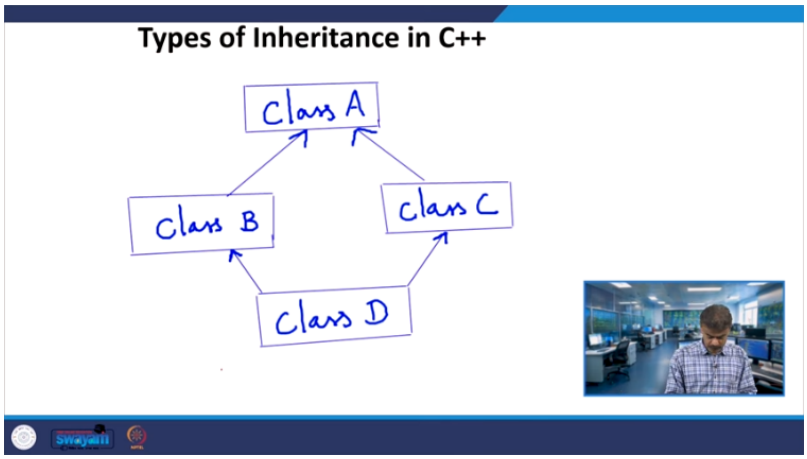
The answer is yes. You can do, you call this as a multi-level inheritance. You can have A, B, C, D, E, F up to that, right? So here B is inheriting class A and class C is inheriting class B. You call this as, this concept as multi-level inheritance. And similarly, now assume that you have three classes, class A, class B and class C.

So, class C can use the properties of class A and class B. Class C, they can use the properties of class A and class B. Two classes. You call it as multiple inheritance. This is the concept of multiple inheritance. We have a notation. We will see.

So, this is called the multiple inheritance. And similarly, you have hierarchical inheritance. Class B is using the property of class A and class C can also use the properties of class, right? Or other words, class B can be derived from class A and class C can also be derived from class A. In the previous case, multiple inheritance, class C is derived from both class A and class B. That is the meaning, right? So, you can say other way round, it can use the properties.

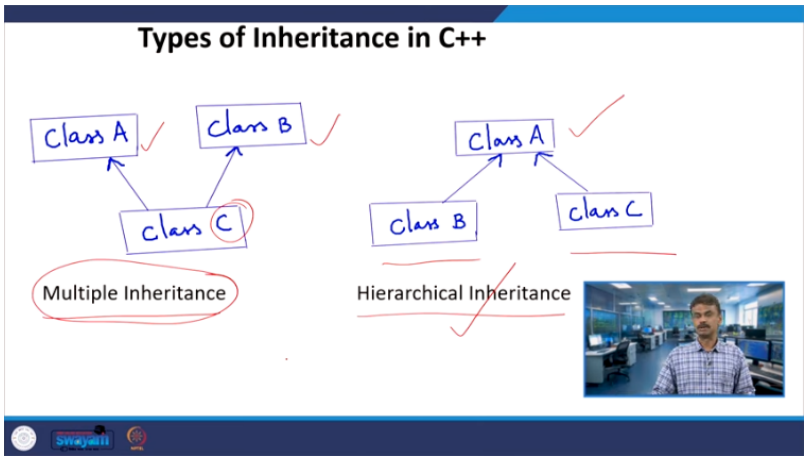
Otherwise, when I talk in terms of base class and derived class, The multiple inheritance class A and class B are the base classes and class C is the derived class, right. So, class C can be derived from class A and class B and you call this as a multiple inheritance, right. Here class A is a base class and class B and class C are the derived classes. So, that is also possible.

right. Class B and class C are derived from class A. You call this as a hierarchical. And the last one is you have this we call it as hybrid inheritance, right. This is called hybrid inheritance, right. So, you can see both hierarchical and multiple.



The combination of hierarchical and multiple, right. The multiple plus hierarchical you call this as a hybrid inheritance, So, for example, here class B and class C, they are derived from class A. So, class A is the base class. So, these are all I put D1 and D2 derived classes. D1 is a derived class 1 and D2 is a derived class 2.

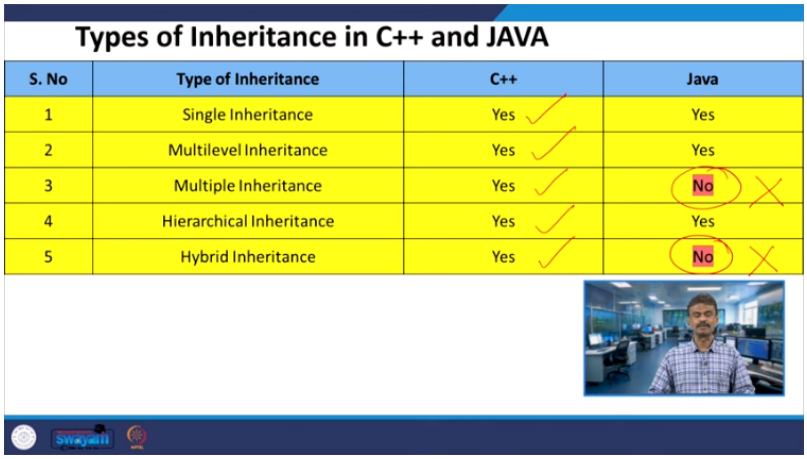
And now class D, I will put D1, 2. So, now this will become derived class. Which are all the base classes? Class B and class C. So, when you do all together whatever the inheritance that we had seen in the previous slide that are nothing but multiple inheritance and hierarchical inheritance. So, the combination is hybrid right.



So, in C++ we have all these features in the case of inheritance. So you may have single inheritance like this. You may have multi-level inheritance like this. You have multiple inheritance. You have hierarchical.

The combination of multiple plus hierarchical, you will have multiple plus hierarchical. You have hybrid inheritance. So these are all five different inheritances which is available in C++. Whereas when I am comparing with Java, Java does not have this multiple inheritance. C++ you can see all are there.

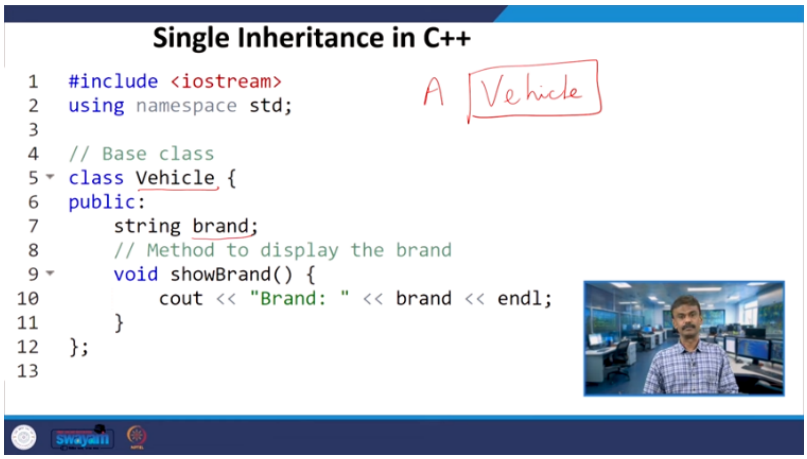
C++ you have all are there. Whereas in Java you do not have multiple inheritance. The moment you do not have multiple inheritance we know multiple plus hierarchical is hybrid right. So which obviously hybrid also will not be available right naturally right. So we have seen the last diagram of in C++.



In C++ all the inheritance are available whereas in Java we do not have multiple inheritance in terms of class right. So, right now we have class only. So, in Java we have a concept called interface right. So, once we study in fact we are going to study interface. So, when I introduce interface we will have the concept of multiple inheritance right.

So, right now in terms of class we do not have the concept of multiple inheritance in Java. So, naturally you do not have hybrid inheritance also in Java. So, C++ point of view, we have all these five inheritance. So, now we will see an example in both C++ and Java, all right. So, you have in the base class, you have, right, we call it as a base class.

So, we call it as A, right. Because we discussed about A and B. Right. So this class has one member data. Right. We are in C++ now.



One member data called brand. And we have one member function called show brand. Right. The show brand is just displaying see out brand. Brand.

Right. So this is under vehicle. So this contains brand. That is a member data. And it has another one that is called member function which is called show brand.

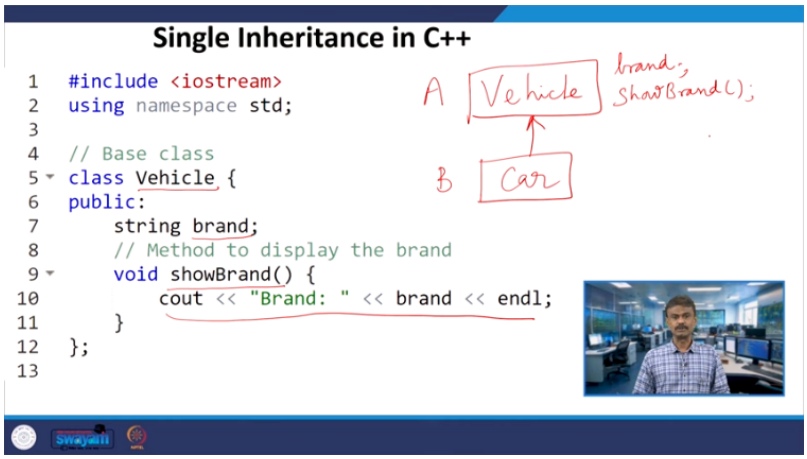
So you can write like this. So one is brand, another one is show brand. One is a data member, another one is a function, member function, right? So what is the derived class? We'll see another class.

So another class is car, class car. So this is the syntax now, how you are going to derive. Class car, we are going to publicly derive. Still, I have not introduced this access modifier. We will see slightly later.

So, right now, we have to worry about how it is being inherited, right? So, vehicle, now it is a base class. This is a way, class car, colon, right? You put public, private or protected. These are all the access modifier.

Right now, without loss of generality, we put public. We will see the meaning slightly later, right? So, this is how the syntax go for inheritance. So car is the derived class. Vehicle is the base class.

So I'll just go back to the previous slide. I'll write car here, right? So that is B. So since I put the syntax like this, so now I am eligible to draw an arrow. That means car can inherit the properties of vehicle, right? So that means it can use brand because it is in public.



It can use brand. It can use show brand. Suppose you are right under car, you have an object. Let us say car C1. So this C1 can use rights or can access brand and show brand.

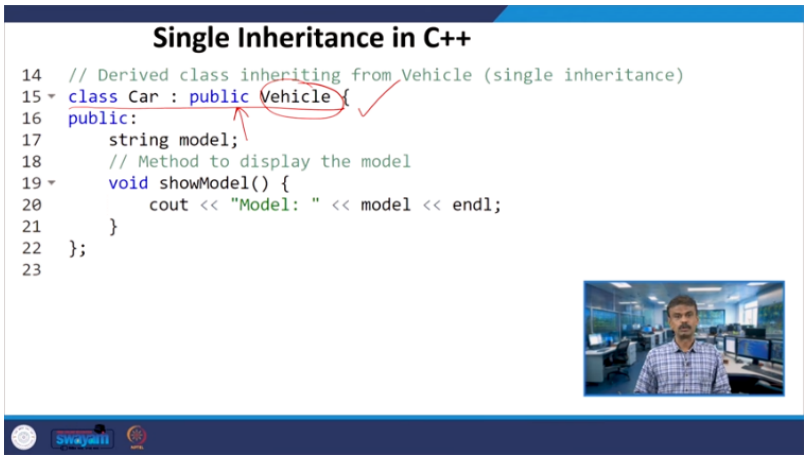
That is a meaning. Right. Anyway, we'll see that how we are going to do that. So right now, the diagram is clear. A is the base class and B is the derived class.

A is the vehicle, car is the derived class. B is the derived class. That means car is the derived class. So, that means car can access both brand and show brand. So, now this car class that itself is having one member data which is called model.

Whereas here you have brand, here you are naming as model. So, string model. So, model that belongs to car. This member data belong to car and you have one more member function. Right.

That is under car. We have used the member function under vehicle. One more member function line number 19. It is under car. You have show model.

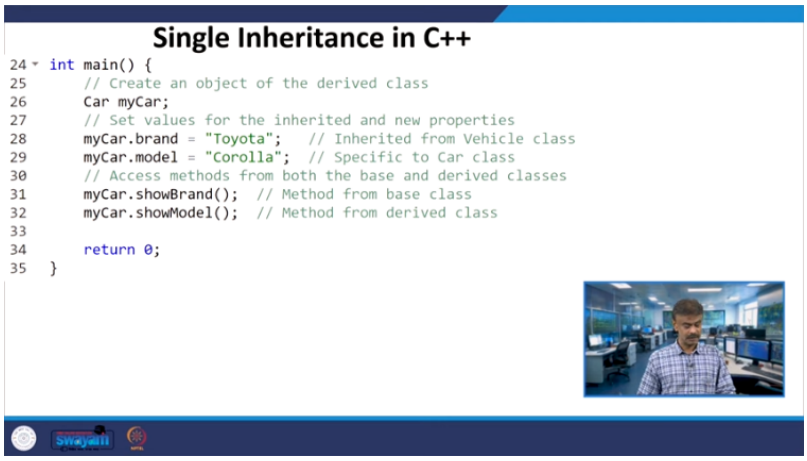
So show model will display model. Right. What is the model? Right. Number.



Right. So here your model is a string. Model is a string. Line number 17. Your model.

So string will be displayed. So we will go to the main program. So main program you have my car which is an object. Right. My car is an object.

So this is coming under object car. Right. So my car is an object. This is coming under class car. All right.



So I will go back again in a written scenario. Car is the derived class, which is the base class vehicle. Right. So under this class car, I have an object called my car. Right.

So now my car dot brand. Right. So my car dot brand. Where is brand is available in base class. Yes, this can access.

So now under this I have an object called my car. So this my car can access both brand and show brand. Right. So that is why you have line number 28. So this is a valid statement.

There won't be any error. My car dot brand. Brand is available in base class. Yes. And next one is my car dot model.

Model is already available in derived class. So it is not a problem. The usual class and object that we had seen. Right. So this is also clear.

That's a usual one. Now, my car dot show brand. Show brand is available in base class, right? So, this can access your my car, the object my car under car can access show brand. No problem.

Because it's concept of inheritance, right? So, show brand and the next one is my car dot show model is already available in the derived class, right? So, now, so I have My car dot brand is Toyota. My car dot model is Corolla.

All right. So now what will happen if I invoke? I mean to say my car, the object my car is invoking show brand. Right. What is happening?

Show brand. You go to the show brand. Right. So that will display brand. See out brand.

So what is brand I passed? So brand is nothing but Toyota. My car dot brand as Toyota. So the Toyota will be printed as a first input. Right.

Underline number thirty one. If you look at line number 32, my car dot show model, right? So show model, model, what is model? Corolla, right? So when you go over here, the show model function, because it is invoking.

If you look at line number 32, it is invoking. My car object is invoking show model. So show model, when you look over here, model will be displayed. What is your model? Corolla.

So Corolla will be displayed, right? So both Toyota and Corolla will be displayed. displayed as output and if you look at if you run the code right so you have to get these two as a output so we will see the c plus plus code right for whatever we have written we will see the c plus plus code now go to the cpp code code 7 dot cpp right the same code that i have written all right so here you can have class

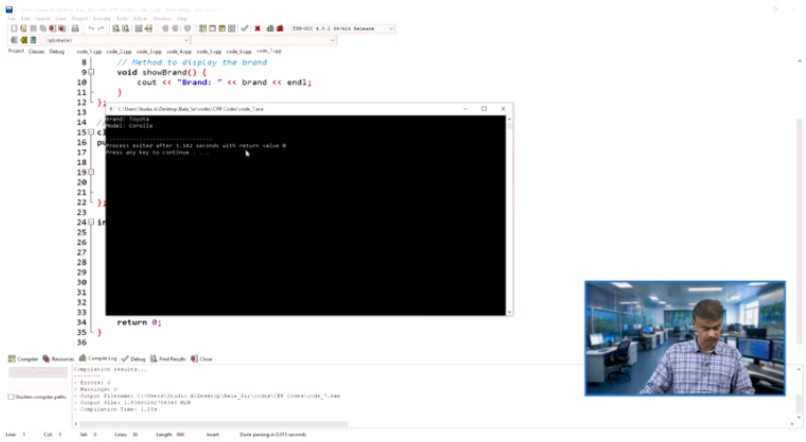
vehicle which is the base class by number five You have class vehicle, which is a base class, right? So you have string brand and void show brand, right?

And similarly, you have class car, which is a derived class. Line number 15, you have derived class. So, under line number 15, again you have one string called model and you have one member function. So, base class is also having one data member and member function. So, here also you are having one member data and member function.

So, these things we had already seen. So, now if you go to the main program, my car is an object and car is the class. So, my car.brand is Toyota and my car.model is Corolla. And my car is invoking show brand and my car is invoking show model. Right.

Which is possible. Right. So those things we have seen in the theory. And now if I run the code, compile and run. Right.

So you can see the output. So we are getting brand Toyota and model Corolla. Right. So this is how your CPP program is running for the single inheritance. So, this is the output in fact we got the same output you can see in the program also.



So, now in the next class the same single inheritance program we will run in Java ok. So, with this I am concluding this lecture. Thank you very much.