REPORT ZYNY\_CLASS\_07.  
\*----------------------------------------------------------------------\*  
\*       CLASS vehicle DEFINITION  
\*----------------------------------------------------------------------\*  
\*  
\*----------------------------------------------------------------------\*  
CLASS vehicle DEFINITION.  
  PUBLIC SECTION.  
    METHODS: gofaster.  
  
  PROTECTED SECTION.  
    DATA speed TYPE i.  
ENDCLASS.                    "vehicle DEFINITION  
  
\*----------------------------------------------------------------------\*  
\*       CLASS car DEFINITION  
\*----------------------------------------------------------------------\*  
\*  
\*----------------------------------------------------------------------\*  
CLASS car DEFINITION INHERITING FROM vehicle.  
  PUBLIC SECTION.  
    METHODS: gofaster REDEFINITION,  
             writespeed.  
ENDCLASS.                    "car DEFINITION  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  
CLASS vehicle IMPLEMENTATION.  
  METHOD gofaster.  
    speed = speed + 1.  
    WRITE: / 'The VEHICLE Speed is: ', speed LEFT-JUSTIFIED.  
  ENDMETHOD.                    "gofaster  
ENDCLASS.                    "vehicle IMPLEMENTATION  
  
\*----------------------------------------------------------------------\*  
\*       CLASS car IMPLEMENTATION  
\*----------------------------------------------------------------------\*  
\*  
\*----------------------------------------------------------------------\*  
CLASS car IMPLEMENTATION.  
  METHOD gofaster.  
    speed = speed + 10.  
    WRITE: / 'The CAR Speed is: ', speed LEFT-JUSTIFIED.  
  ENDMETHOD.                    "gofaster  
  
  METHOD writespeed.  
    WRITE: / 'My CAR is Grease Lightning'.  
  ENDMETHOD.                    "writespeed  
ENDCLASS.                    "car IMPLEMENTATION  
  
\* Our program starts here.  
  
START-OF-SELECTION.  
  
  DATA vehicle1 TYPE REF TO vehicle.  
  DATA car1     TYPE REF TO car.  
  
\* VEHICLE object without any NARROWING CAST  
  WRITE: / 'VEHICLE object without any NARROWING CAST'.  
  CREATE OBJECT vehicle1.  
  vehicle1->gofaster( ). "calling gofaster method from vehicle class  
  CLEAR vehicle1.  
  
\* VEHICLE object with NARROWING CAST  
  ULINE.  
  WRITE: / 'VEHICLE - NARROWING CAST from CAR'.  
  CREATE OBJECT car1.  
  vehicle1 = car1.  
  vehicle1->gofaster( ). "calling gofaster method from car class  
  skip 3.  
\*  vehicle1->writespeed( ). "write is not visible here  
  
\* Demonstrate WIDENING CAST  
\* At this point we have used a NARROWING cast on VEHICLE1  
\* Create an obj ref to catch the error. You don't need to do this. You can just  
\* use CATCH SYSTEM-EXCEPTIONS  move\_cast\_error = 4.  
  
  DATA: my\_cast\_error TYPE REF TO cx\_sy\_move\_cast\_error,  
        car2          TYPE REF TO car.  
\* Note, we haven't created an object for car2... it is an empty ref variable  
  
  TRY.  
\* Do a WIDENING CAST to move the ref from VEHICLE TO CAR (more specific)  
      car2 ?= vehicle1. " above did a narrow cast car=>vehicle, now vehicle=>car , so we are first passing car to vehicle, then car back to car  
\*      so it is calling car class methods  
    CATCH cx\_sy\_move\_cast\_error INTO my\_cast\_error.  
      WRITE: / 'The WIDENING CAST failed FIRST'.  
  ENDTRY.  
  IF car2 IS NOT INITIAL.  
    car2->gofaster( ). "method from car class will be called  
    car2->writespeed( ). " method from car class will be called  
  ENDIF.  
  SKIP 3.  
  
\* Now, lets generate an error  
  
  CLEAR: car1, car2, vehicle1, my\_cast\_error.  
  CREATE OBJECT vehicle1.  
  TRY.  
      car1 ?= vehicle1. "since we cleared the vehicle1 ref, so there is no ref in it, this will generate error  
    CATCH cx\_sy\_move\_cast\_error INTO my\_cast\_error.  
      WRITE: / 'The WIDENING CAST failed SECOND'.  
  ENDTRY.  
  IF car1 IS NOT INITIAL.  
    car1->gofaster( ).  
    car1->writespeed( ).  
  ENDIF.

So now we've gone over narrowing casts and a widening cast, let's have a look at some code and let's

see in action, let's see a good example to try and really explain what I think are quite complicated

topics.

So let me bring up the op ed.

What I will do is create a brand new program so we can have a look and debug some new code to see all

this in action, so I'll tell you what I'll do.

I'll pulls the video here and I'll come back once I've written the program.

There's no need for you to watch me type it line by line.

OK, I've got the code and I'll scroll to the top and we'll walk through it, and then once we've been

through the code and explain what's going on, we'll then run in debug mode and see in action.

So to start with, I'm using our.

Vehicle subclass that we've used before, so we have the vehicle class definition and I've defined to

go faster method and there's some protected data called speed.

Then we have a car class that inherits from vehicle, and you can see we're redefining the go faster

method and adding a right speed method.

So now down to the implementation, our vehicle method, really simple stuff, we're just adding the

value of one to speed.

I'm writing a message to the screen showing the vehicle speed is whatever it should be.

And the car class that redefines to go first a method, as you can see, it's redefining it, adding

10 instead of one and then outputting the car speed instead.

And then the new right speed method is just a simple one line output to the screen showing my car is

greased lightning.

So let me scroll down.

So now we start with the starts selection.

So this is where our calling program starts and I'm declaring to object reference variables.

We're going to be using vehicle one, which points to our super class vehicle and car one, which points

to our car subclass.

And the first example we've got is I'm just showing how we create the vehicle object and call to go

faster method without implementing any of this narrowcasting or widening casts.

So very basic stuff here just to show that it's working within the object reference variable out.

So it's not pointing to anything.

And then we do a narrowing cast.

So I just don't put that to the screen and then we create the subclass object car one.

Once that's created, we then do the narrowcast.

So we assign the car one object reference to our vehicle, one super class object reference variable.

So remember, vehicle one only knows about the high level common methods and attributes that were declared

in the vehicle class itself.

It doesn't know about that extra method of the car subclass.

So when we do this, when we assign the Chowan subclass object to vehicle one object reference variable

sounds complicated and that means we can only call the methods that are defined in the superclass.

We can't call the methods defined in the subclass.

So we have a narrow view.

And to show this, working with them perform, to go faster method, and that will show us when we're

debugging exactly which method is being executed.

So as we come down, we then demonstrate widening casts, so I declare.

A new object reference variable to catch the system exceptions, the runtime errors are going to occur

in our example below.

You don't have to do this.

You can just use the catch system exceptions by assigning the value for to move cast error or something

like that.

But this is just the way I'm doing it.

And then I declare a new object reference variable.

Based on the subclause now, remember, with a widening cast, I did say you've got to watch out for

these runtime errors.

So we'll start with this try block.

And the first thing we'll do is a widening cast where I'm using this special casting operator to assign

the object reference helpful vehicle one to our object reference variable car two, which was defined

based on the subclass.

So that has a wide view.

We can actually perform more methods and we have the statement just to catch any system errors, which

there shouldn't be at this point.

Then as long as there's no system errors, runtime errors is going to perform this.

He'll say if Kata is not initial, which means the object reference variable has been filled, then

it's going to perform the go faster method and the right speed method.

And then just to prove that sometimes widening casts won't work if you haven't got it set up correctly.

We've got an example here where I'm clear in our object reference variables out.

Creating a new object for vehicle one.

We got to try again, and then when we do the casting here, it's going to fail our vehicle.

One object reference variable is actually empty at this point, so it's not going to be able to move

an object reference into the car.

One object reference variable.

So we will get a runtime error and this will not be performed.

Let's see in action so we can go through it step by step and hopefully understand a little bit more.

So I'll set breakpoints and the first line and let's get going.

OK, here we go.

So we'll go line by line, so write this out to the screen, which is fine, a normal crate's object

will call the method and we can see it's going to rinse out.

The vehicle's speed is one.

No more clear, the object reference variable, so if we give it a click, we can see it said two initial.

Now we're going to do an narrowcast.

So create the object Kangwon one, which is the subclass.

A double click and we can see we have an entry here in Kawan.

Now, when we perform the narrowcast.

We can see how a vehicle one here.

Is now pointing to exactly the same object about Concours.

A double click, we can see it's pointing to the car.

Not the vehicle superclass, which is how it was defined for this shows it's a dynamic type reference

variable.

Then we can call the go faster method and as you can see, because we've done the narrowcast, we're

now performing the go faster method from the car subclass.

Now we come to the widening cast, so things to keep in mind here is our vehicle.

One is pointing to the object car, so that's the subclause.

Since we stepped through.

We're then going to do a widening cast to then say, set our new car to object, reference variable

to whatever was held in vehicle one, and because we have previously done a narrowcast, we know that

it's holding a reference to the subclass.

So our car to subclass should then point to the same thing.

And then we get access to the wider scope of elements for the subclass, which means we can perform

not only to go faster method on a car subclass, but we can perform the right speed method as well.

If we try to perform the right speed just on a vehicle superclass, it would break.

Well, we probably couldn't even compile our program because the right speed method doesn't exist for

vehicle one.

The superclass.

Let's continue on.

So we do the widening cast, no error existed.

So he's going to perform the go faster method for the car, so class.

And then the right speed.

Once again, we'll clear our objects out.

So now everything has been initialized again will create a brand new object for vehicle one.

So now Vehicle one is actually holding an object reference to a super class object.

And now when we're trying to do a widening cast, assigning the super class object to the car object

reference variable, it's going to break because a car, one object reference variable can only receive

the object type reference for subclasses, which we had here.

Vehicle one is now holding an object reference for a super class instead.

So we'll go through it and then we have an exception, an exception.

Ba ba occurred.

So the exception is called.

Then when we try to check.

For the Kawan is not an issue, obviously it's false, so we can't perform these methods.

And there's the output.

So that's it.

Back to the coach, so hopefully you got to grips with the narrowcast and widening cash there and you

could understand a little bit better than me just explaining on the slides of exactly what it all meant.

Seeing it in action normally helps an awful lot to explain these types of concepts.

Well, I hope it does anyway.

If not, read, watch the videos and practice yourself.

Let's move on.