Inheritance

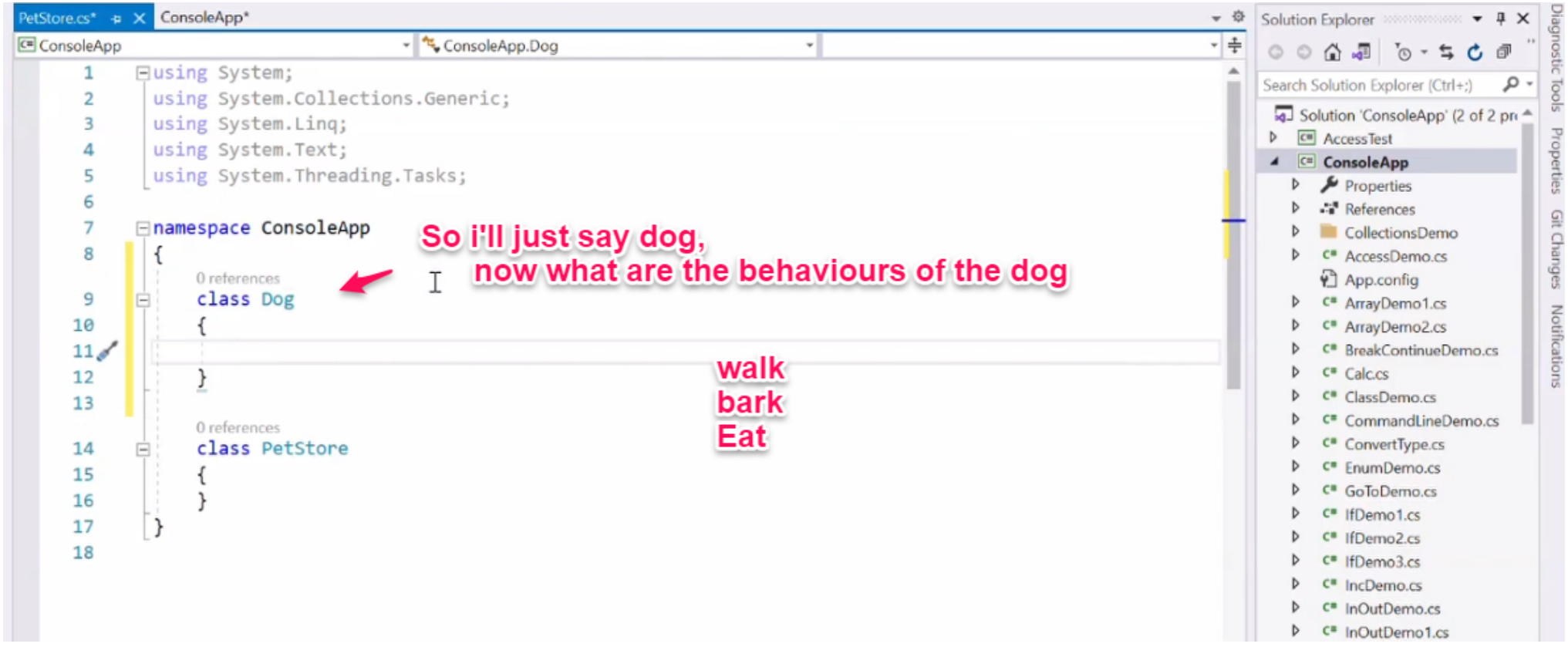
Inheritance in C#

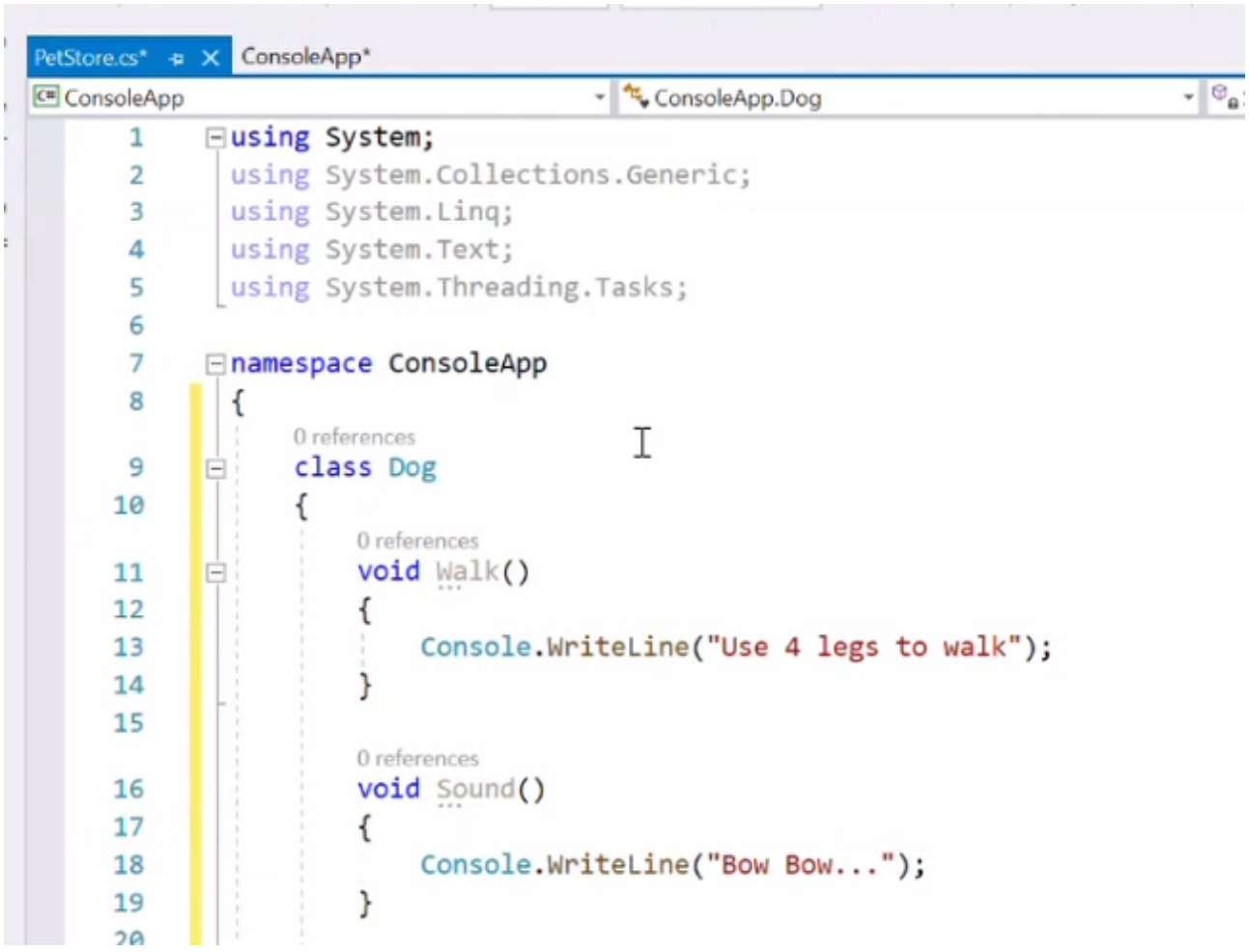
- *Inheritance* is the ability to derive new classes from existing ones. A derived class ("subclass") inherits the instance variables and methods of the base class (“parent class"), and may add new instance variables and methods.
- Inheritance defines a hierarchical relationship among classes wherein one class shares the attributes and methods defined in one or more classes.
- Inheritance is a relationship among classes in which one class shares the structure and behavior of another. A subclass inherits from a base class.

Its a ability of the derived class that's a new class to extent to information of the base class, Base class is something like a parent class
For Example :- a phone has came into market it is fabulous but its too costly so i want to take the look and feel
so i can borrow the look and feel of some other phone with same color same big screen size
Whenever i want to duplicate my phone i can copy
now what you want to borrow that is something which you want to question
now when i want to buy new phone so i only want to borrow the look and feel of i phone 13 its only the Abstraction which i want to borrow, if so then what you can do is We just create a instance of it and then we borrow it
to get same look and feel i'll just inherit it
when i inherit what happens all the function parent has, we'll have it in child but internally i can put all the chinese chip to make it cheaper so internally it will be different
But if i'm not bothered of look and feel but i want to copy inly the internal details so then i'll create the instance of these and i can access that information
So it all depends on the design that what you want to borrow
With inheritance you can borrow all the functionality, all the features of the class
Now we'll see same program in 6 different ways

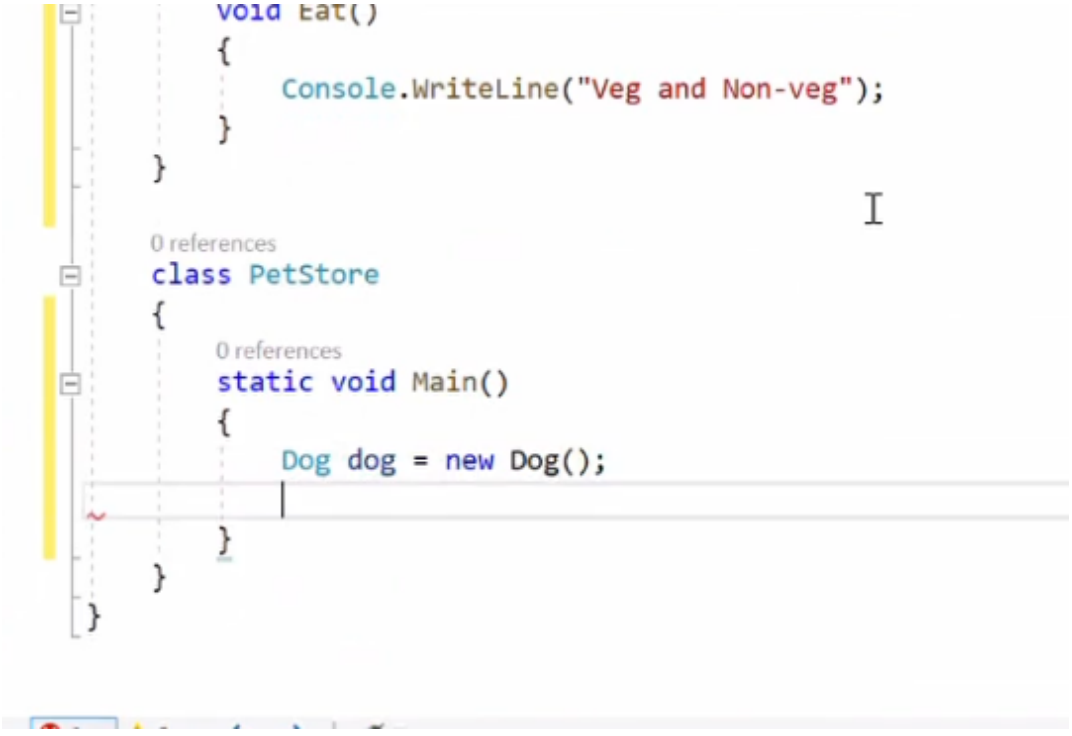
1. Take a Example we have a particular client who want to create a PetStore

there where lot of dogs in the pet store so i want to maintain the dog information

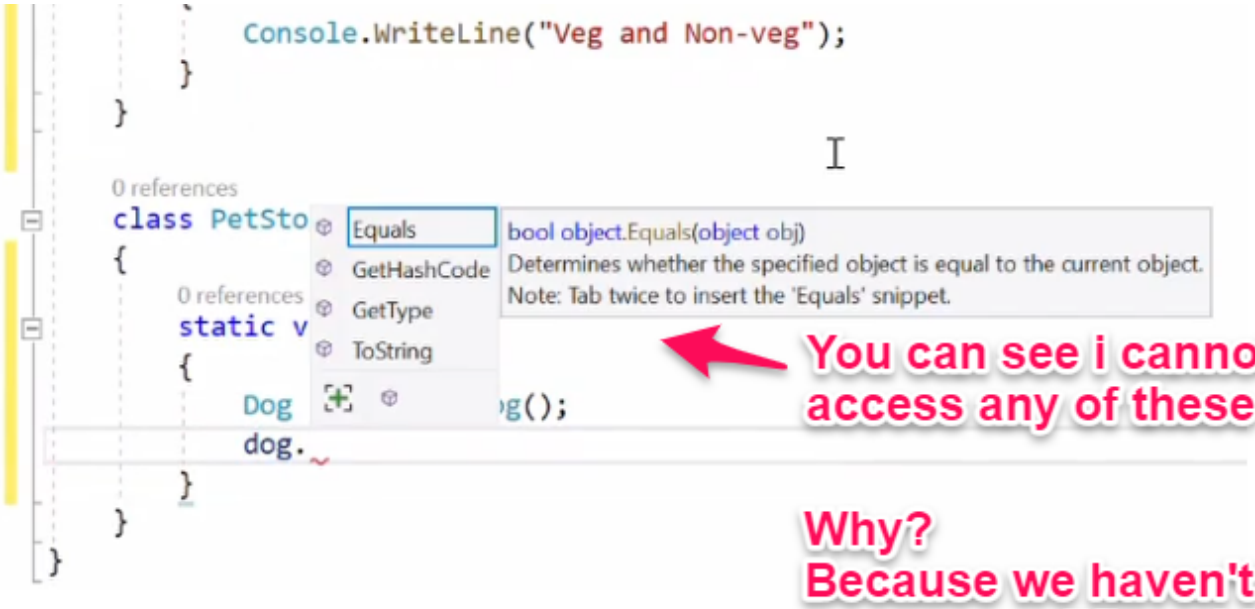

PetStore.cs*
ConsoleApp*
ConsoleApp
ConsoleApp.Dog
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace ConsoleApp
{
0 references
class Dog
{
}
0 references
class PetStore
{
}
}
So i'll just say dog,
now what are the behaviours of the dog
walk
bark
Eat
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'ConsoleApp' (2 of 2 pr
AccessTest
ConsoleApp
Properties
References
CollectionsDemo
AccessDemo.cs
App.config
ArrayDemo1.cs
ArrayDemo2.cs
BreakContinueDemo.cs
Calc.cs
ClassDemo.cs
CommandLineDemo.cs
ConvertType.cs
EnumDemo.cs
GoToDemo.cs
IfDemo1.cs
IfDemo2.cs
IfDemo3.cs
IncDemo.cs
InOutDemo.cs
InOutDemo1.cs
Diagnostic Tools
Properties
Git Changes
Notifications


PetStore.cs*
ConsoleApp*
ConsoleApp
ConsoleApp.Dog
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace ConsoleApp
{
0 references
class Dog
{
0 references
void Walk()
{
Console.WriteLine("Use 4 legs to walk");
}
0 references
void Sound()
{
Console.WriteLine("Bow Bow...");
}

```
15
                0 references
16              void Sound()
17              {
18                  Console.WriteLine("Bow Bow...");
19              }
20
                0 references
21              void Eat()
22              {
23                  Console.WriteLine("Veg and Non-veg");
24              }
25          }
26
            0 references
27          class PetStore
28          {
29          }
30      }
31
```

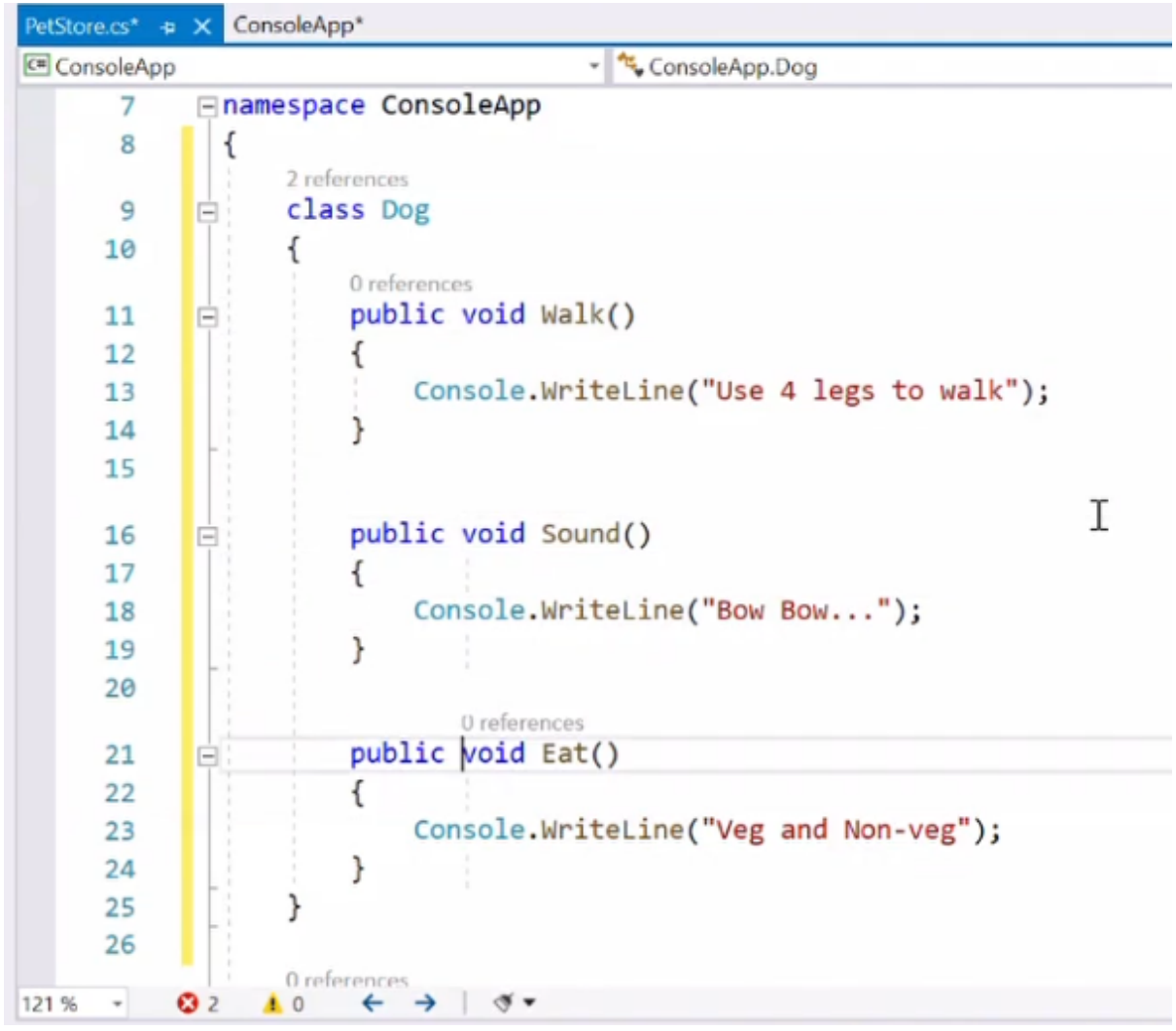


So what are the things we can access by creating instance of dog

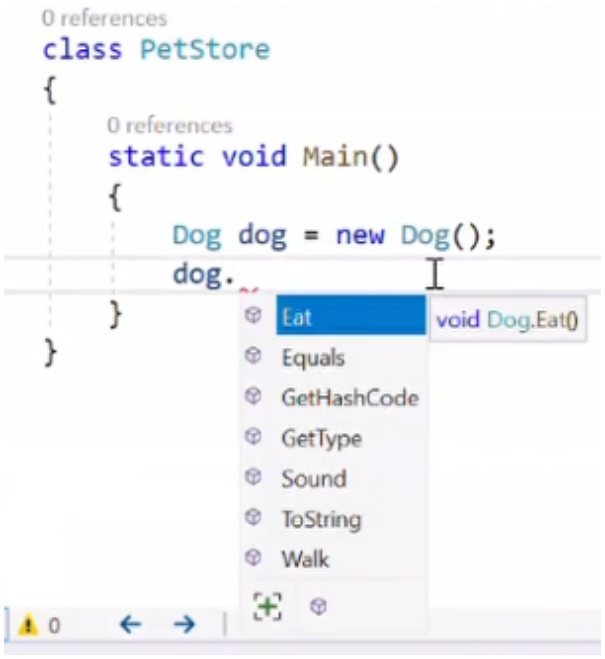


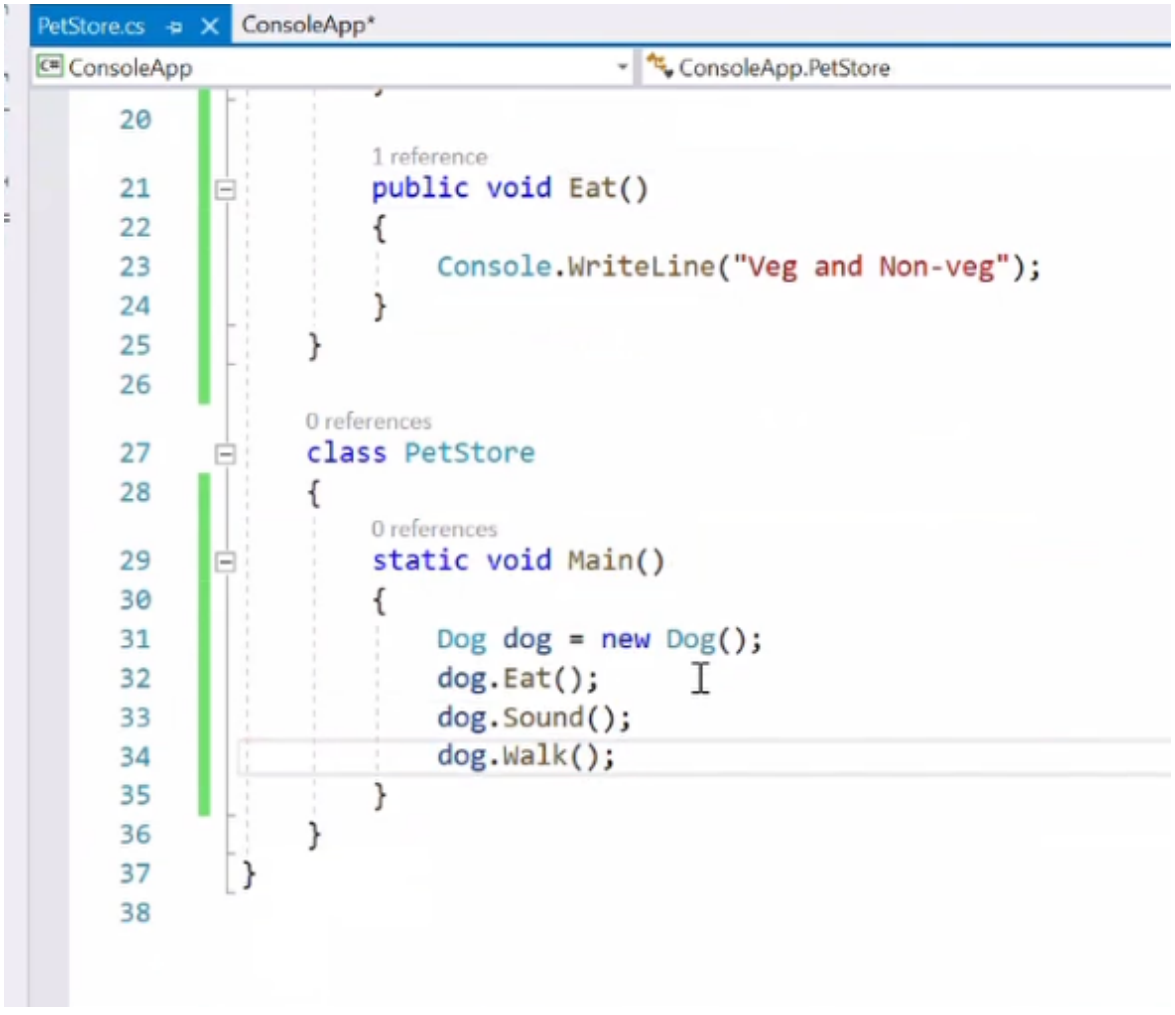
You can see i cannot access any of these

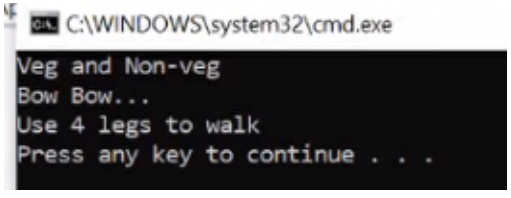
Why?
Because we haven't use any access modifire while creating the method so by default they are private so we have to make them public



now you can see when i say dog. i'll get all the methods listed


0 references
class PetStore
{
0 references
static void Main()
{
Dog dog = new Dog();
dog.
}
}
Eat
Equals
GetHashCode
GetType
Sound
ToString
Walk
void Dog.Eat()
0





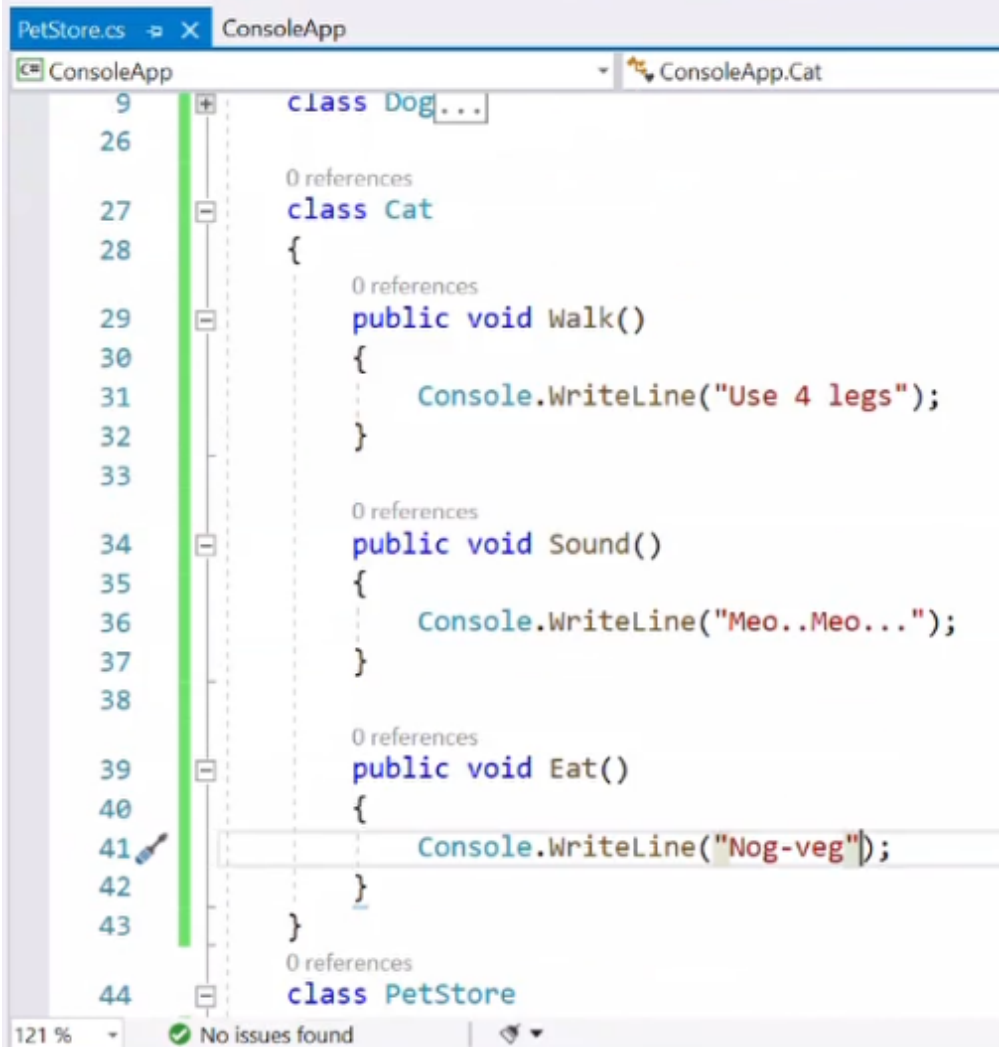
so in the 1st program what we have done ?

We have seen a crowd for dog so we are just creating a class called dog and dog will have certain features like walk sound and eat, so when i went to doctor and showed them, So they are quite happy about this but they say shashi you know what with the Dog even get the cat also, so cat also does the same

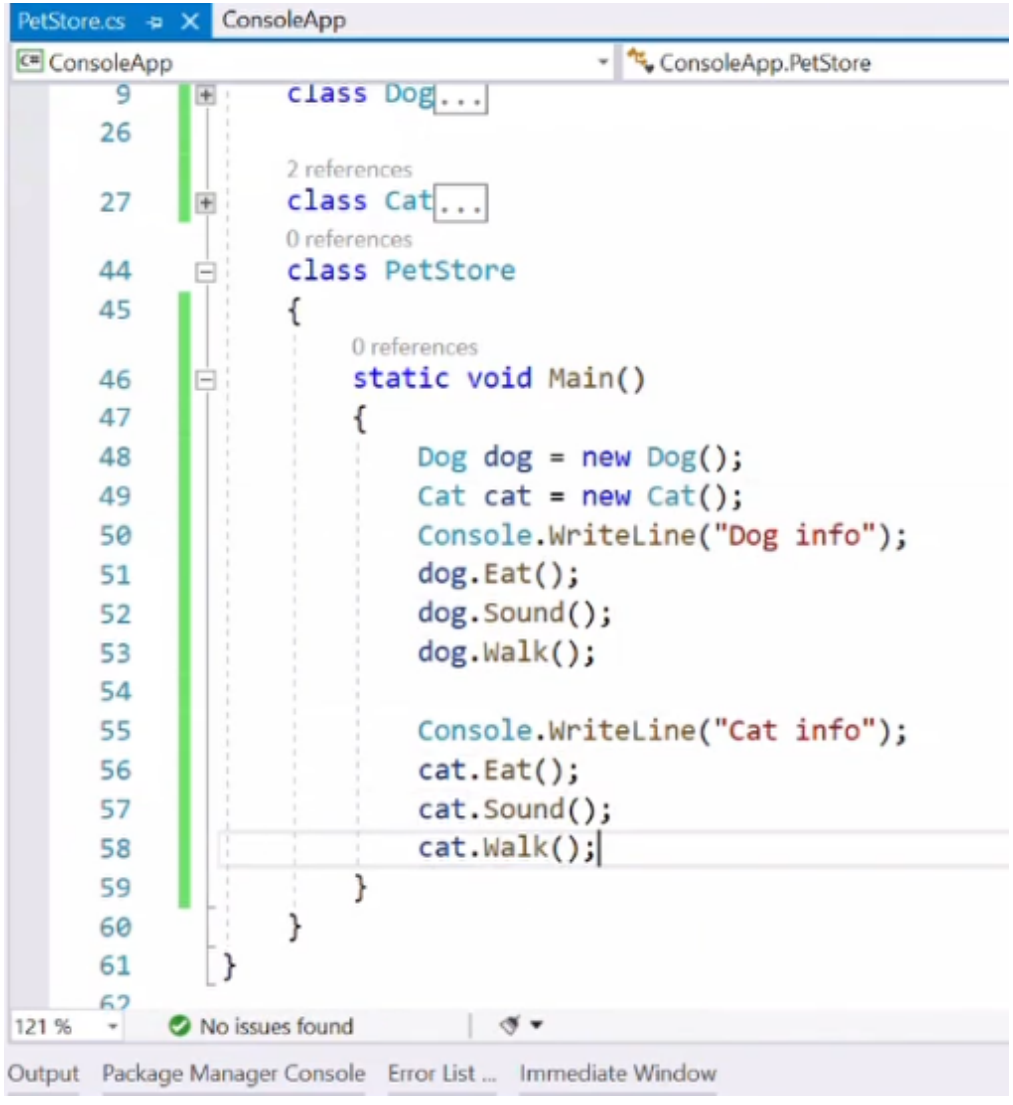
So now we are in the second sprint and we want to enhance now so we are going with the class called cat

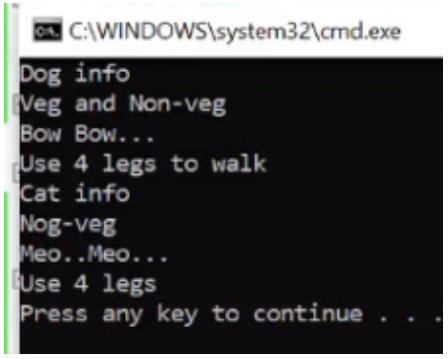
So when i want to doctor and ask like what are the feature my cat will have

SO doctor said the same thing it can walk sound eat



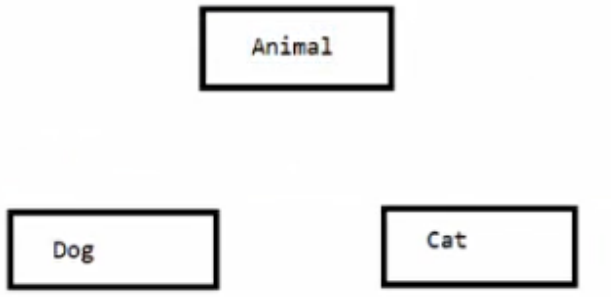
so we need implementation of cat so we create instance of it





so we have 2 different classes cat and dog

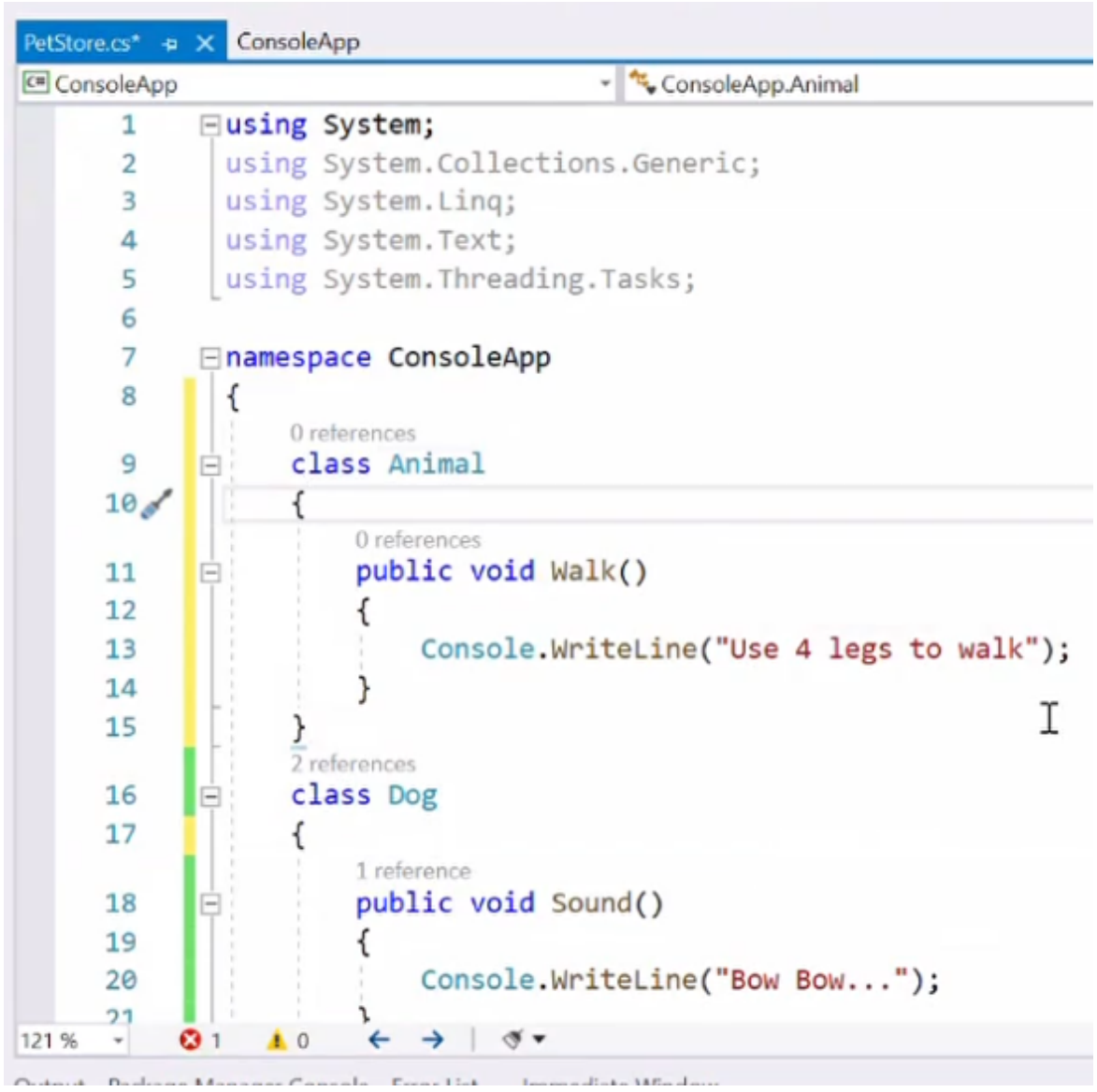
now cat has its own implementation do has its own implementation
so when i look into this particular program so we will have something which is called retrospective done
In Retrospective what we do we just check what is the code we have written this is something which should be carry forward or not when i open this particular project and see, walk are present in both cat and dog and both are using 4 legs to walk
but the implementation of sound and eat is different in dog and cat , So i'll discuss with the project lead and we came out with the new class called Animal

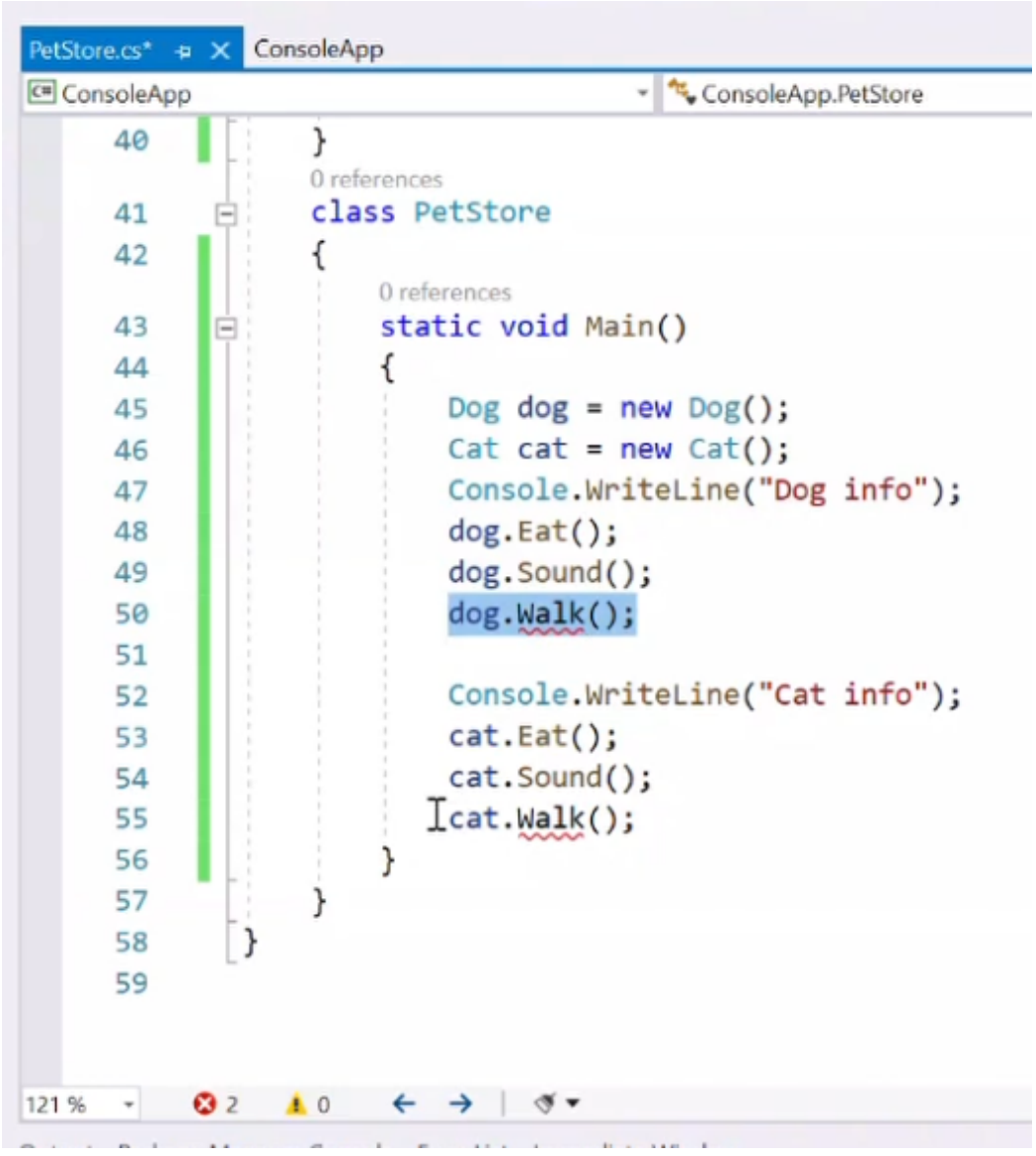


Why are we bringing a new class animal here ?

because i have some features which i can put it in my base class which i can always inherit from here

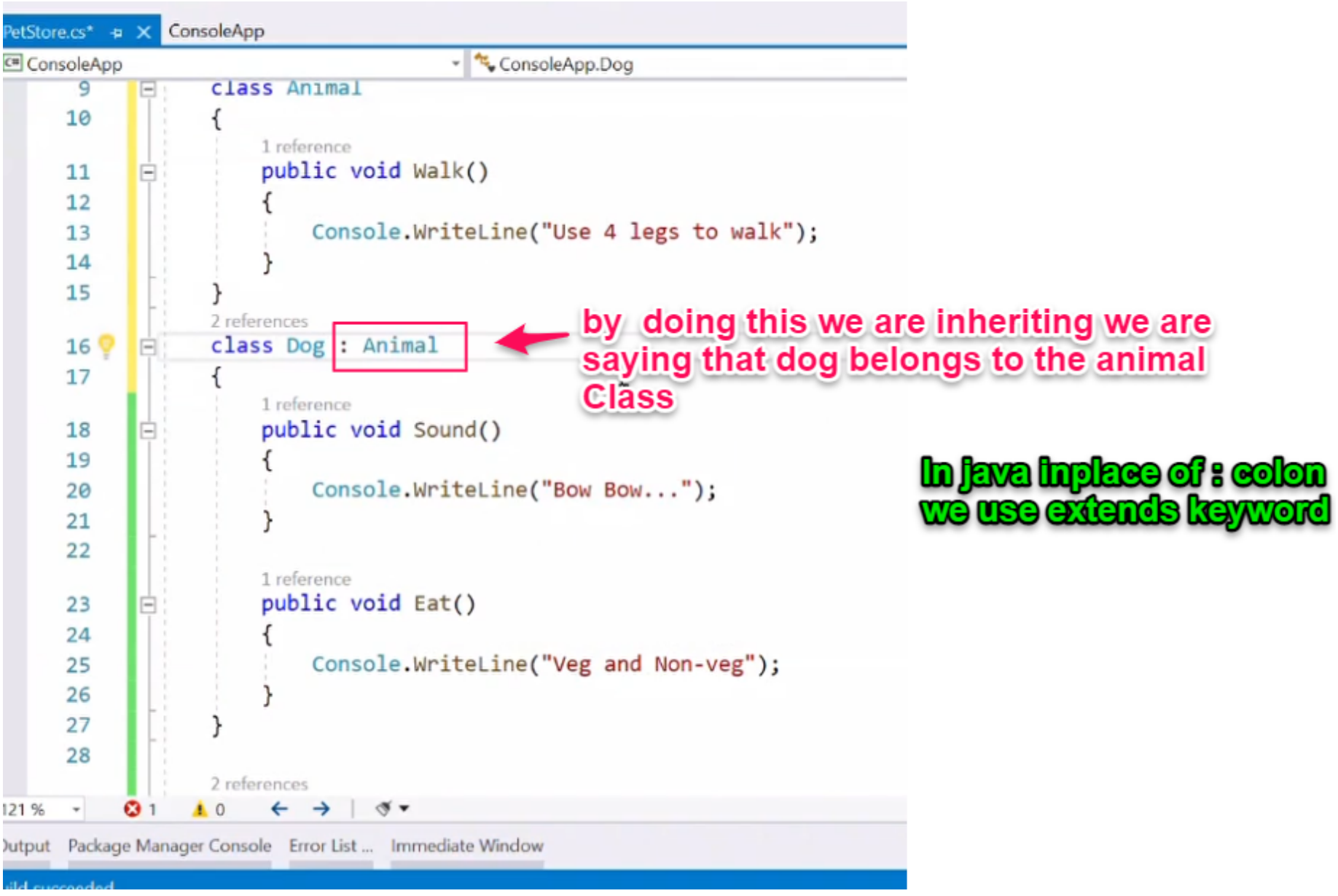
So from dog and cat i remove walk method which is repeated and i'll put it in the class called animal


PetStore.cs*
ConsoleApp
ConsoleApp
ConsoleApp.Animal
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace ConsoleApp
{
0 references
class Animal
{
0 references
public void Walk()
{
Console.WriteLine("Use 4 legs to walk");
}
}
2 references
class Dog
{
1 reference
public void Sound()
{
Console.WriteLine("Bow Bow...");
121 %
1
0



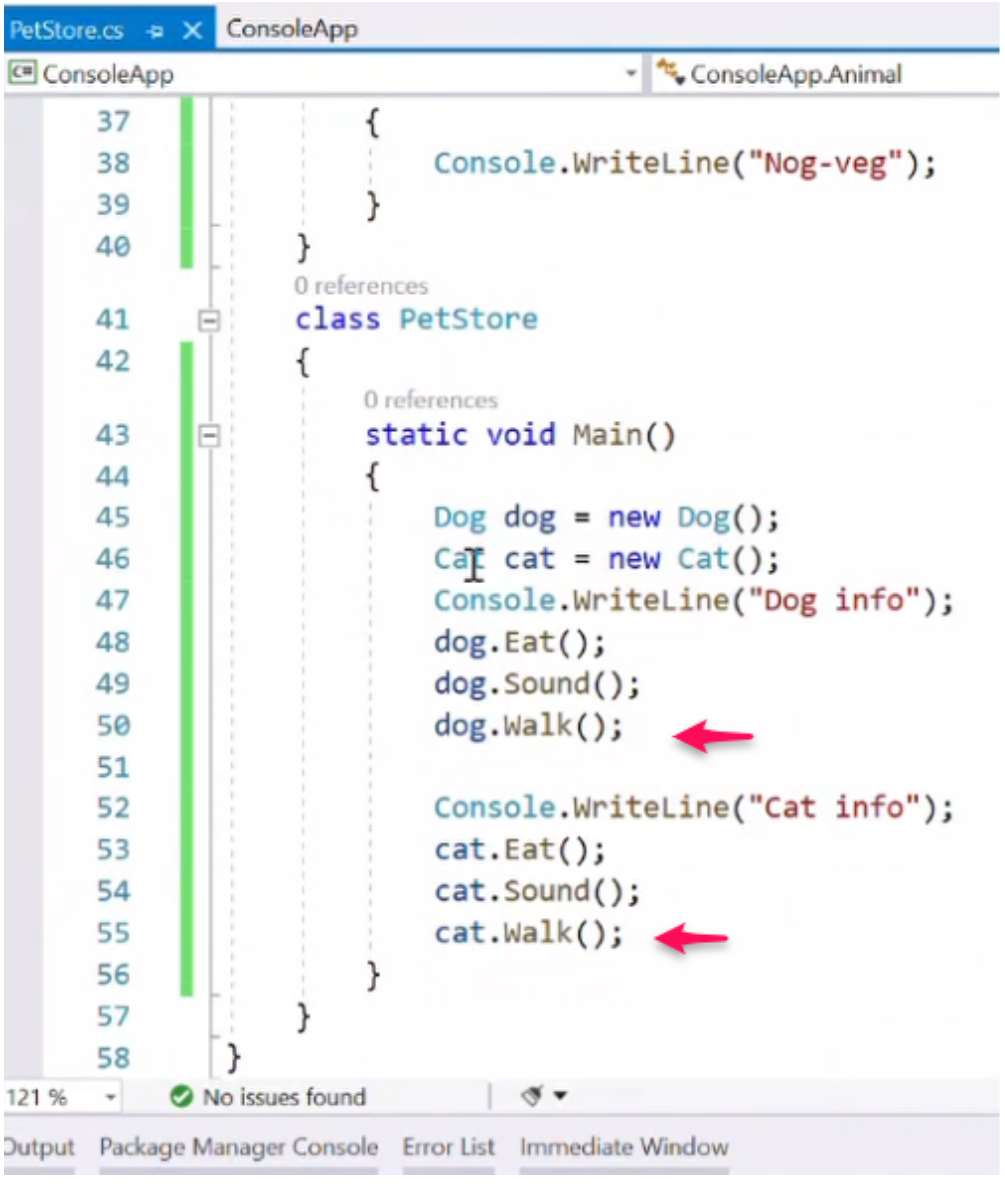
Why we create Animal class?

Because we saw some common implementation so we keep that in Animal Class



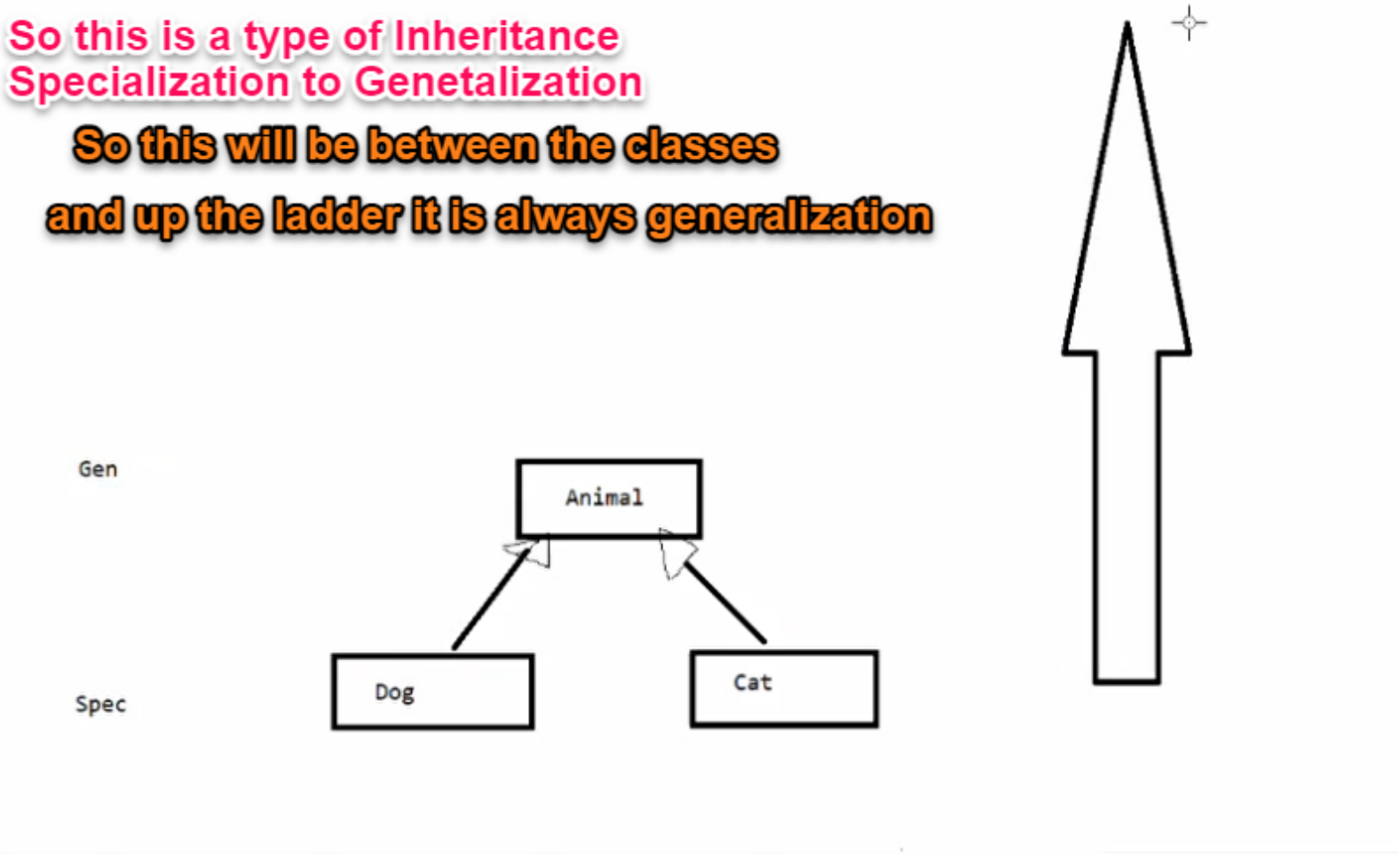
this is a **IS A** Relation, so by doing this we are saying dog is a animal cat is a animal and it should be represent as arrow headed

So i don't have the implementation of walk in the dog or cat



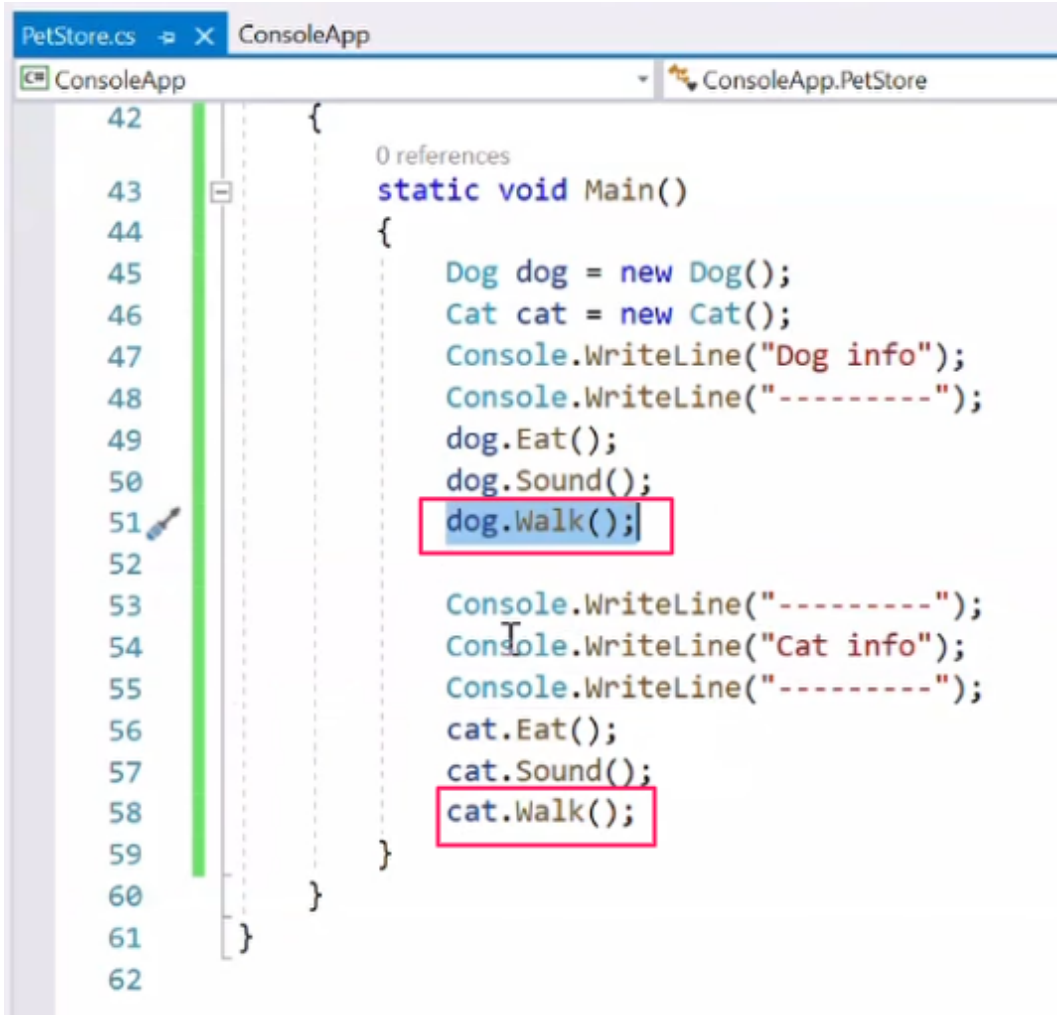
But still we are able to access that methods


Animal
Here we have Generic Data
Dog
Cat
So here we have the Specific Data

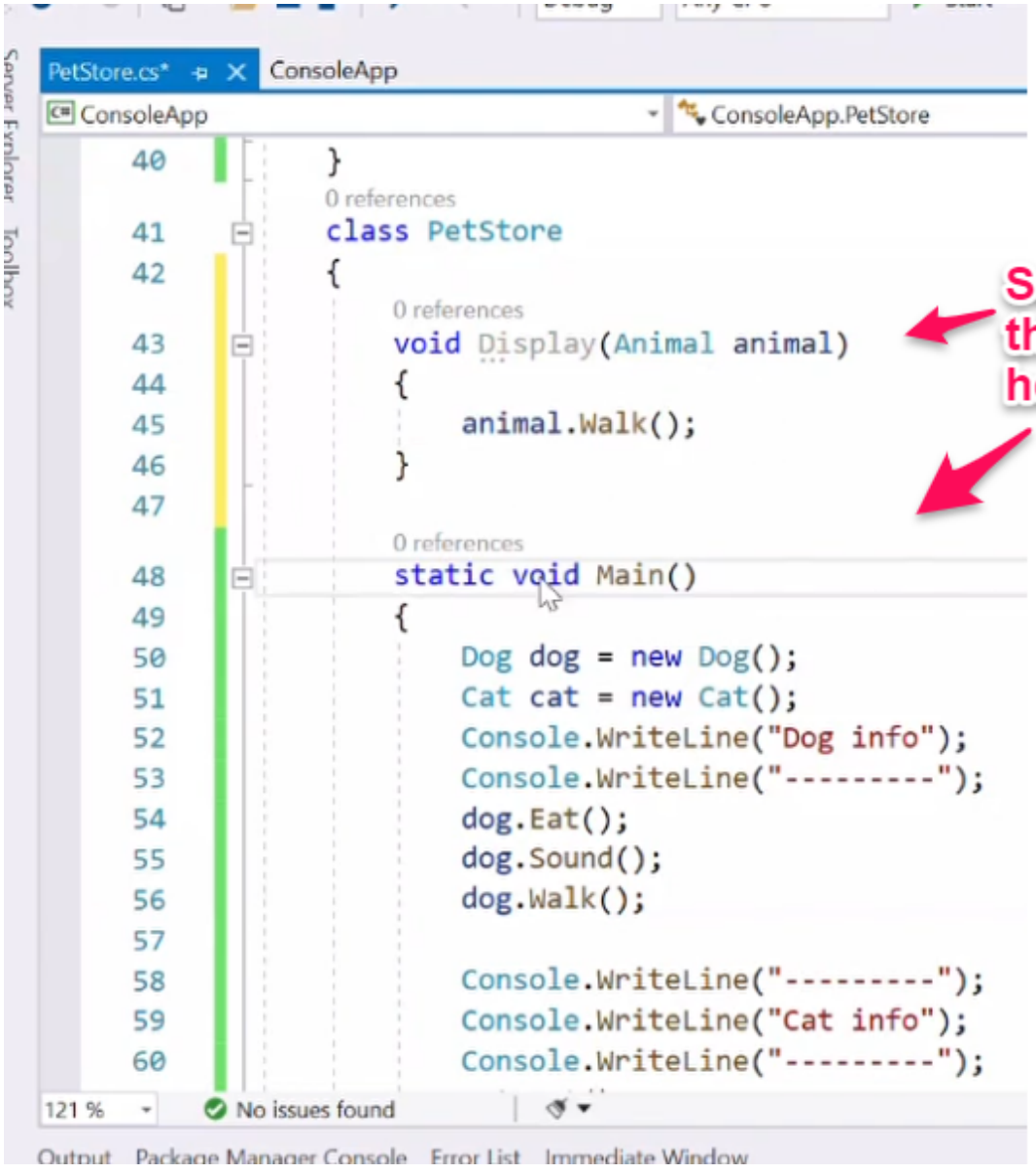


So this is Fabulous but still i'm thinking of Optimization

3. program no 3



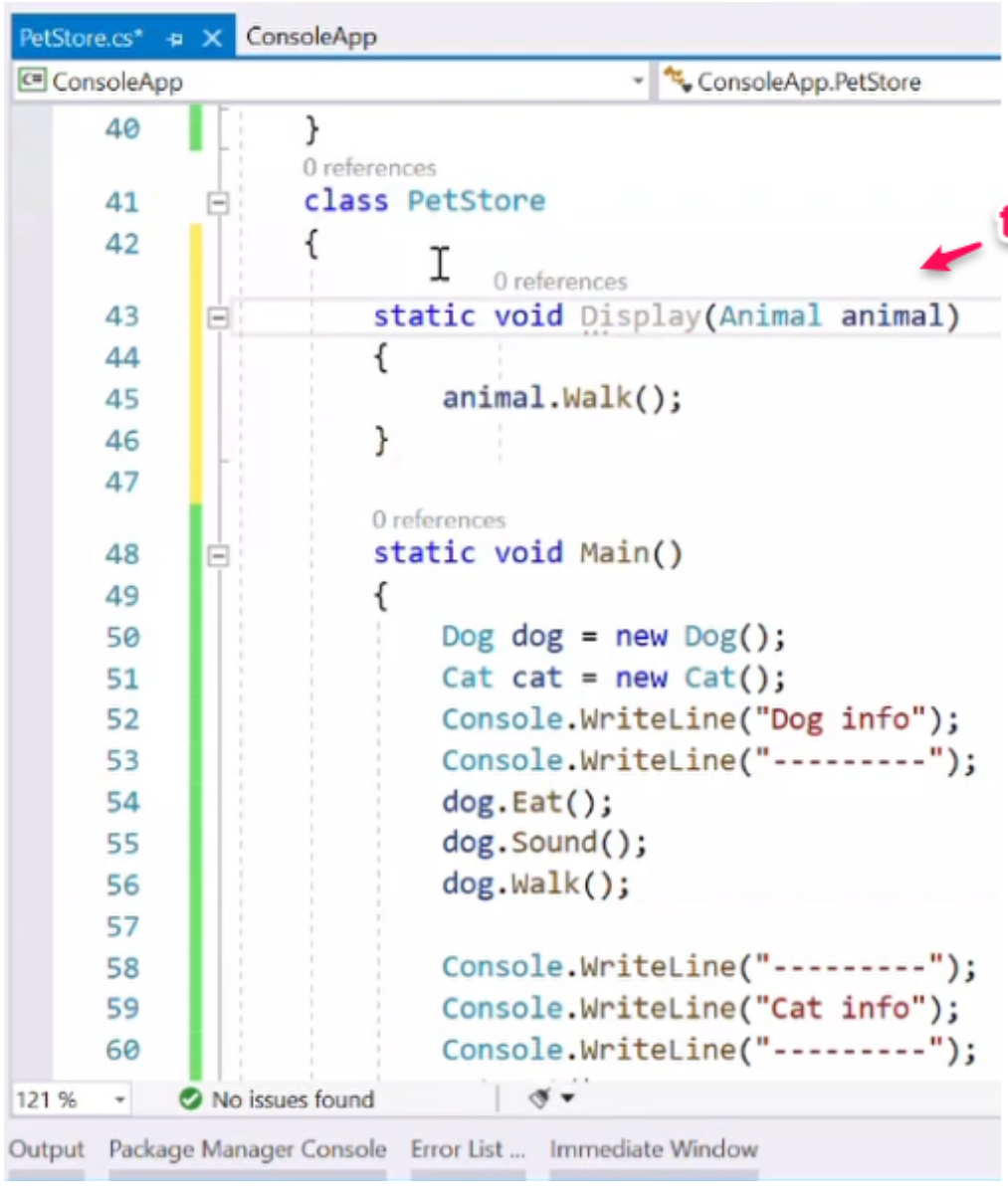
can we generalize as we have walk walk repeated, so that we have something that repeated so instead of doing that we have



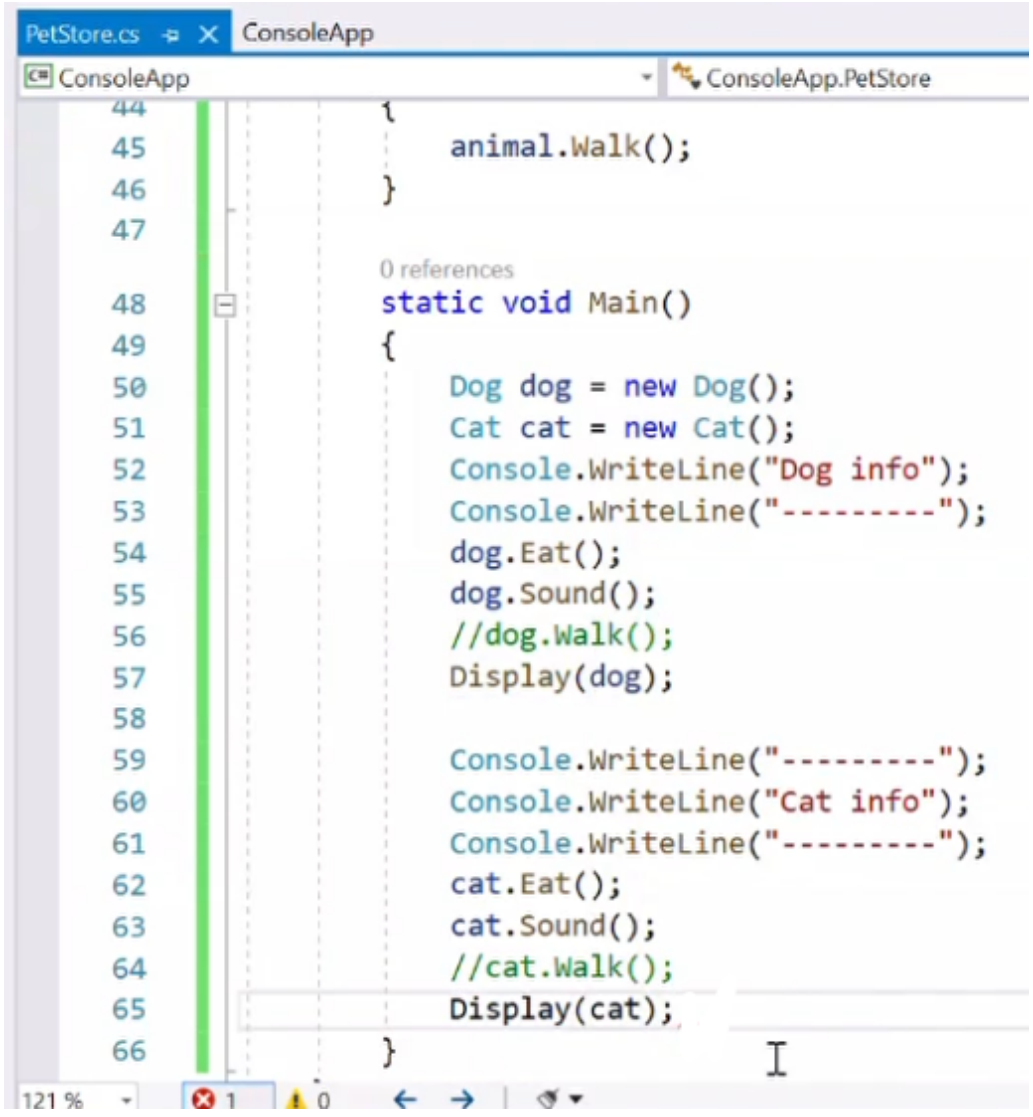
So can we access this display method here in main

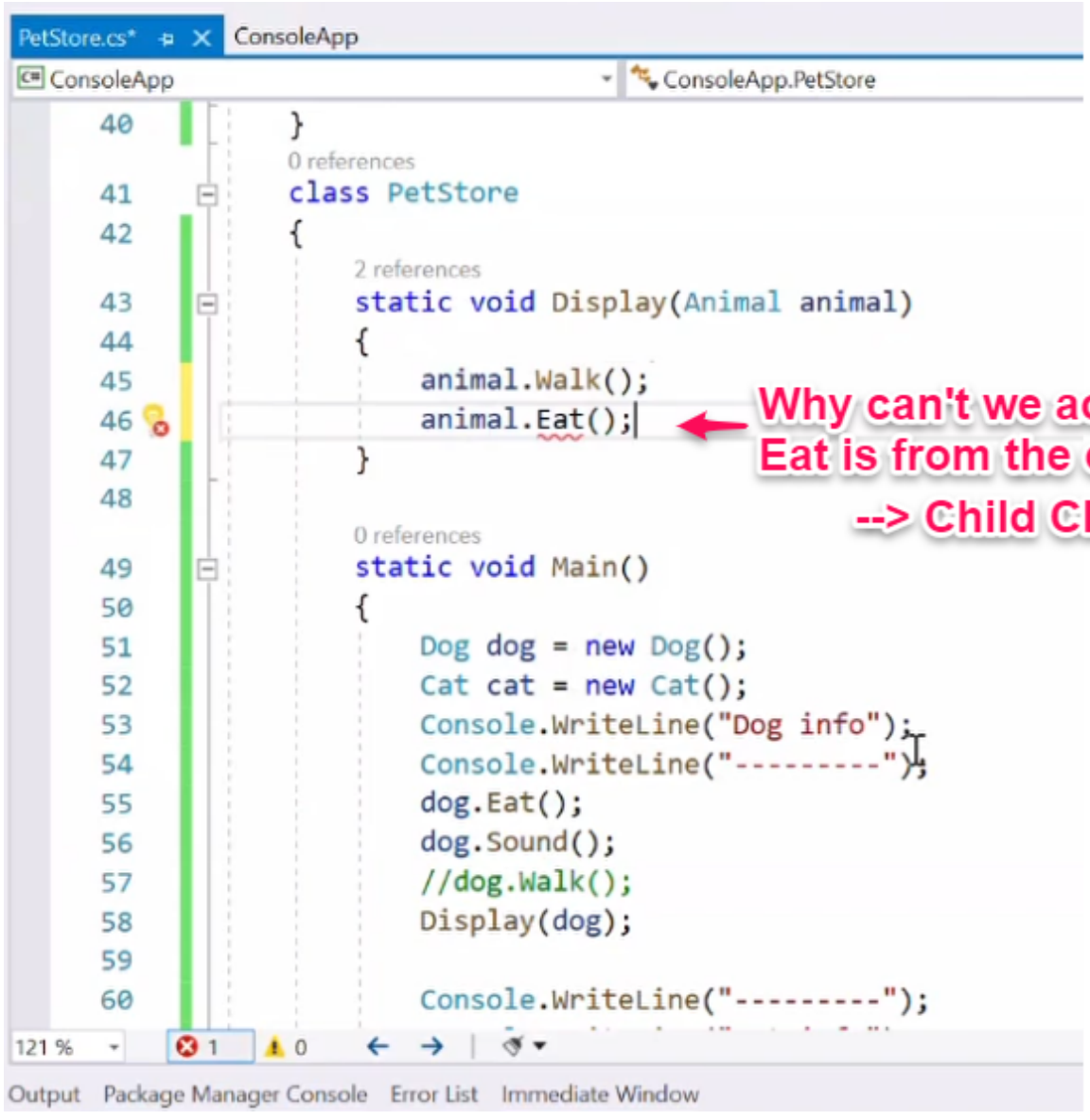
No, because we have to create instance of this petstore in main to access the Display()

Or we can change this to Static

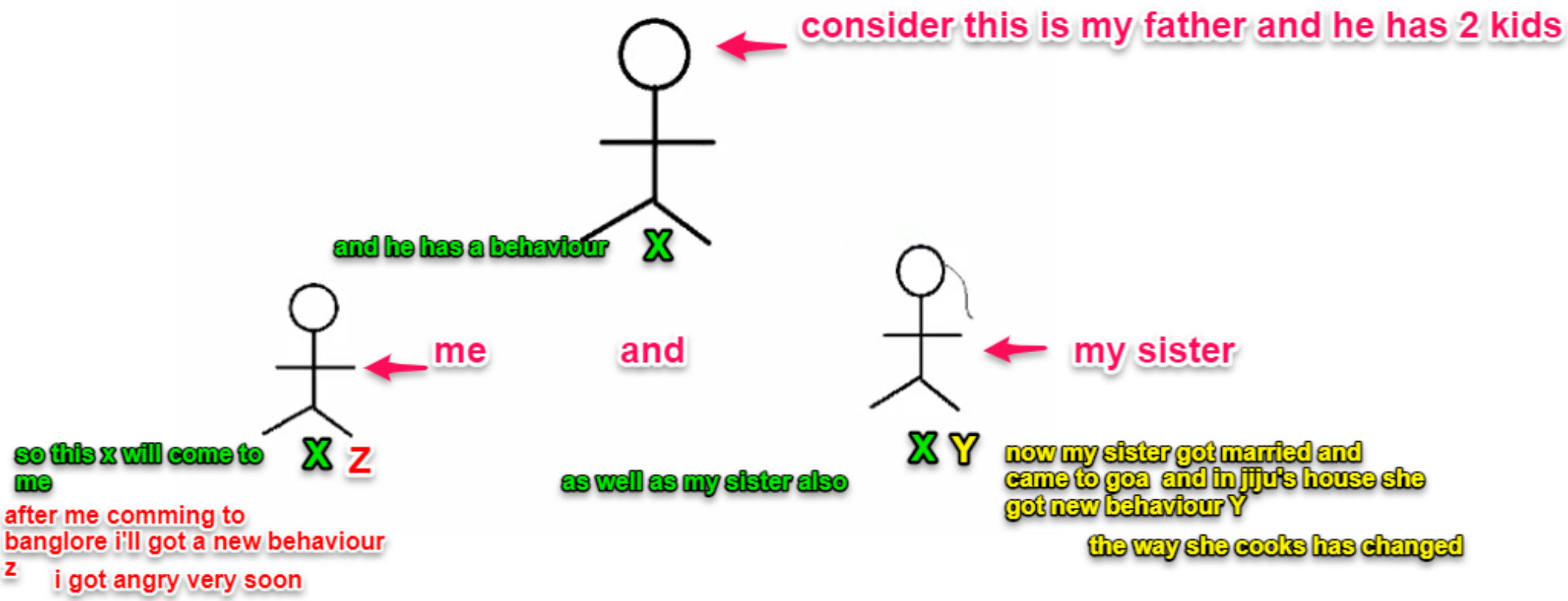


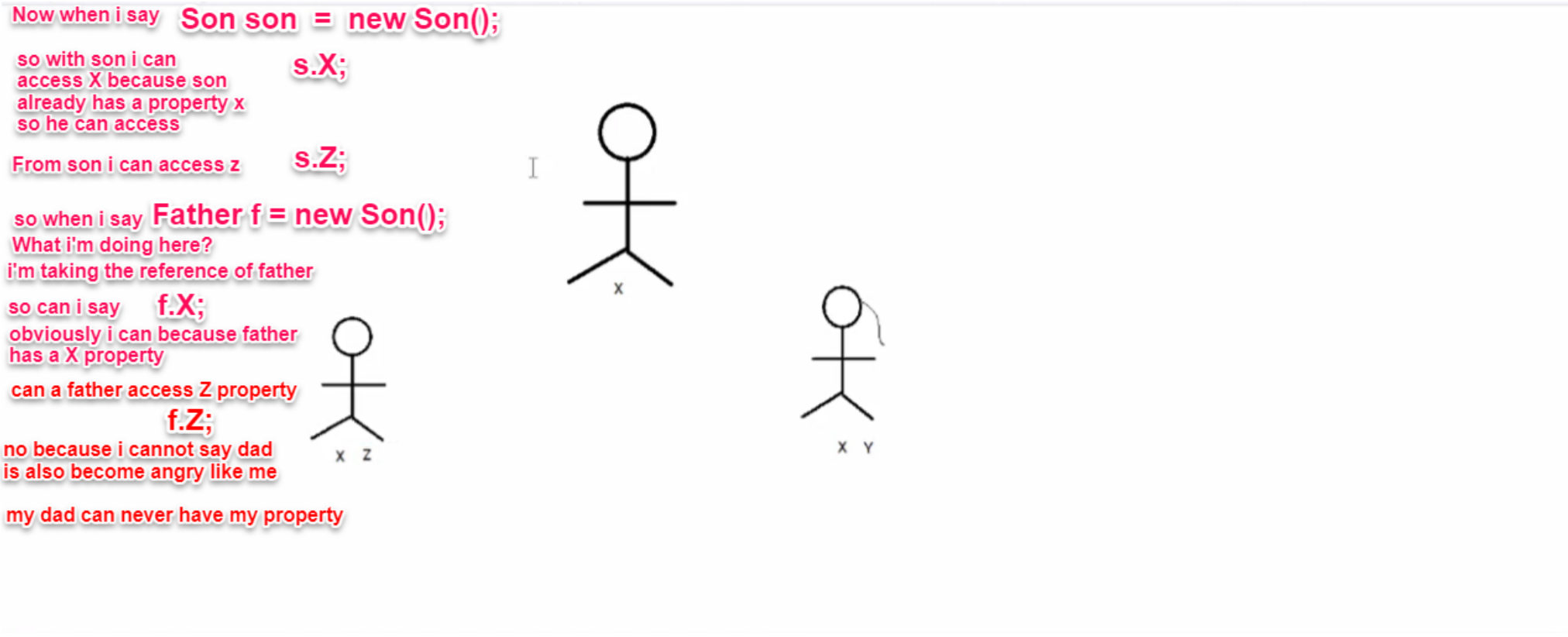
this is just a reference


PetStore.cs
ConsoleApp
ConsoleApp
ConsoleApp.PetStore
{
animal.Walk();
}
0 references
static void Main()
{
Dog dog = new Dog();
Cat cat = new Cat();
Console.WriteLine("Dog info");
Console.WriteLine("---------");
dog.Eat();
dog.Sound();
//dog.Walk();
Display(dog);
Console.WriteLine("---------");
Console.WriteLine("Cat info");
Console.WriteLine("---------");
cat.Eat();
cat.Sound();
//cat.Walk();
Display(cat);
}
121 %
1
0



Why can't we access eat here
Eat is from the child class or parent class?
--> Child Class
it is a behaviour

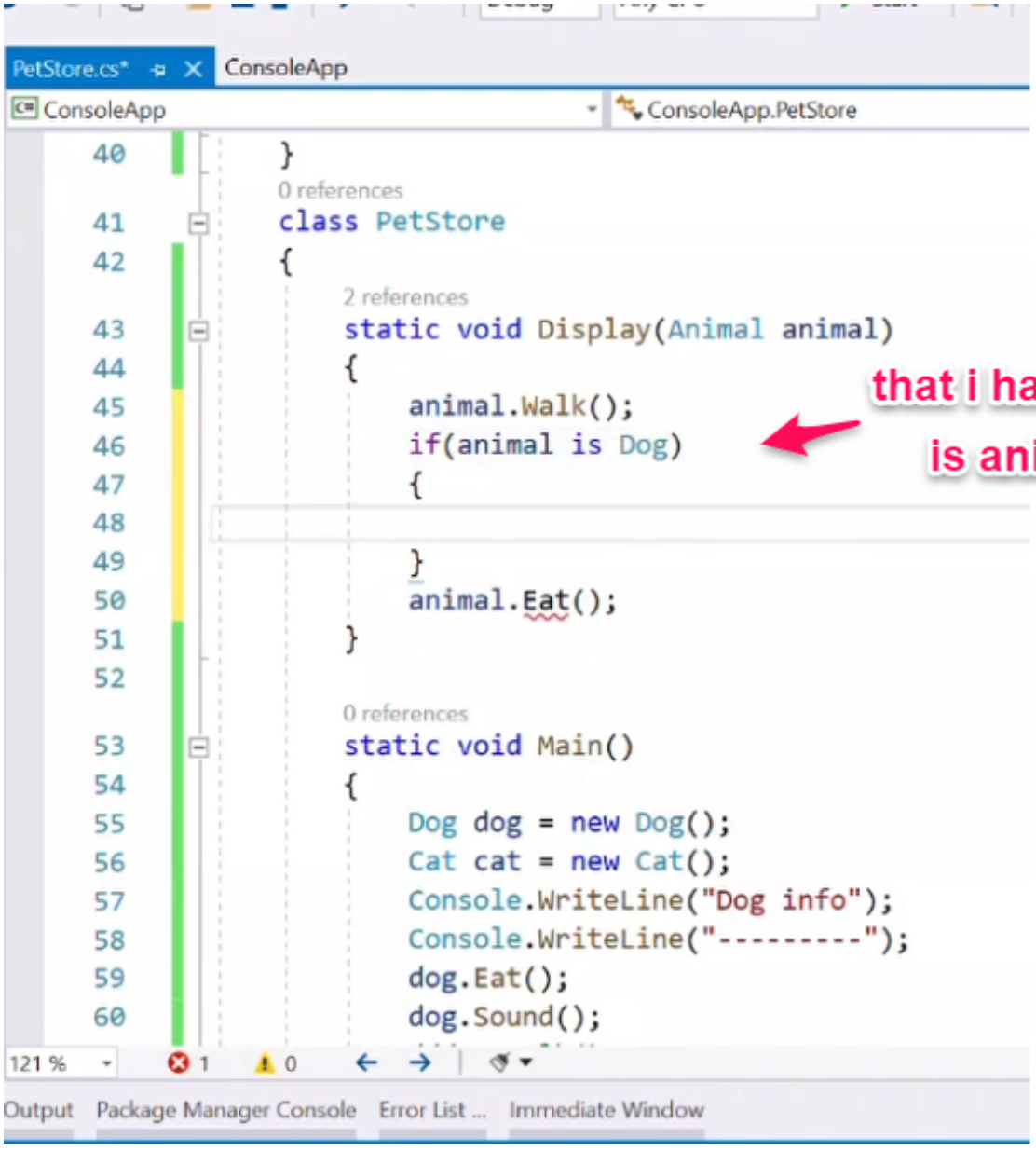

consider this is my father and he has 2 kids
and he has a behaviour X
me
and
my sister
so this x will come to me
X Z
after me comming to banglore i'll got a new behaviour z
i got angry very soon
as well as my sister also
X Y
now my sister got married and came to goa and in jiju's house she got new behaviour Y
the way she cooks has changed



So to achieve this i'll Down Caste it

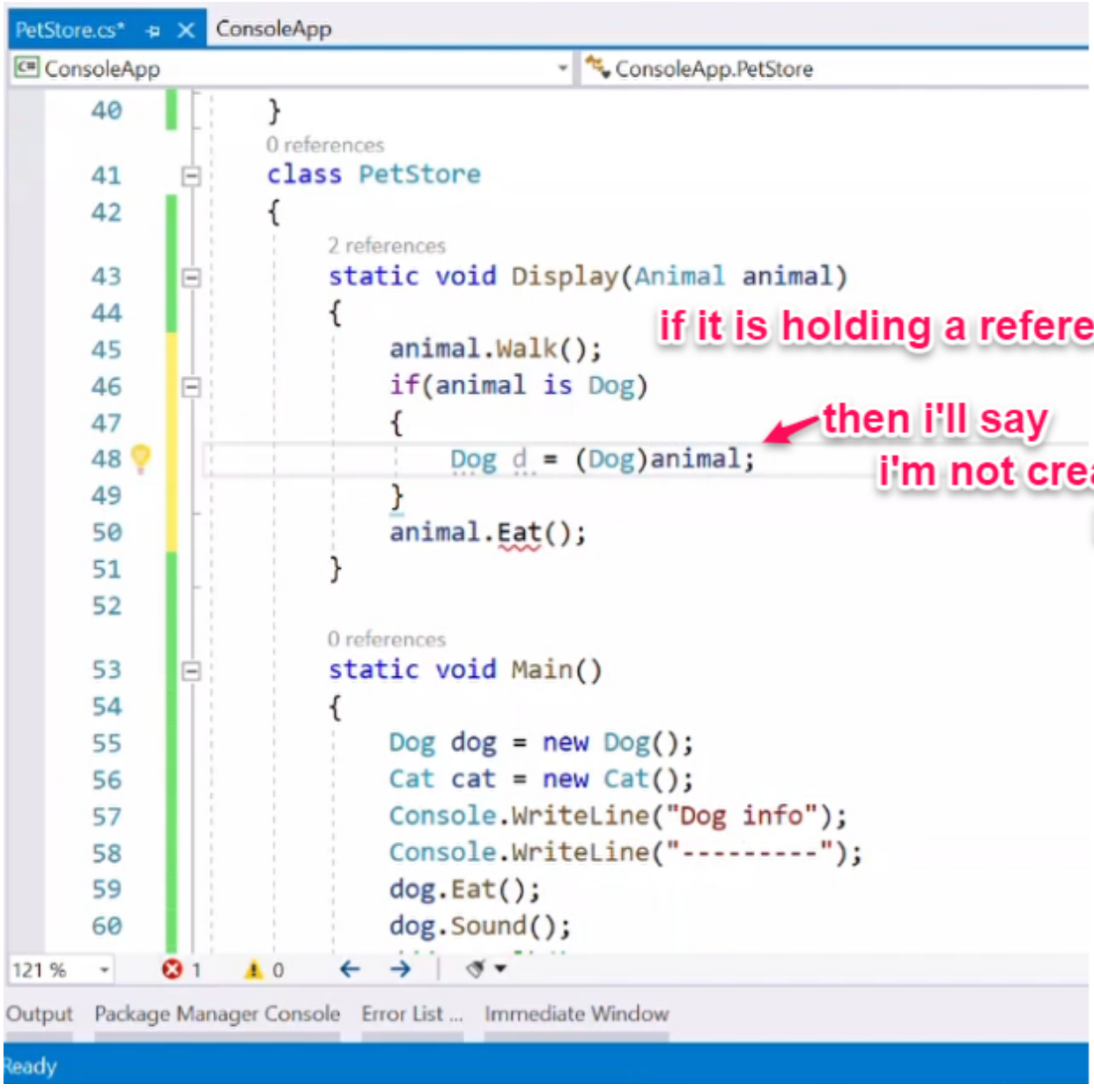
Down Casting

So to Down Caste it i don't know weather my animal is now handelling with the dog instance or with the cat instance



that i have to check here

is animal a dog

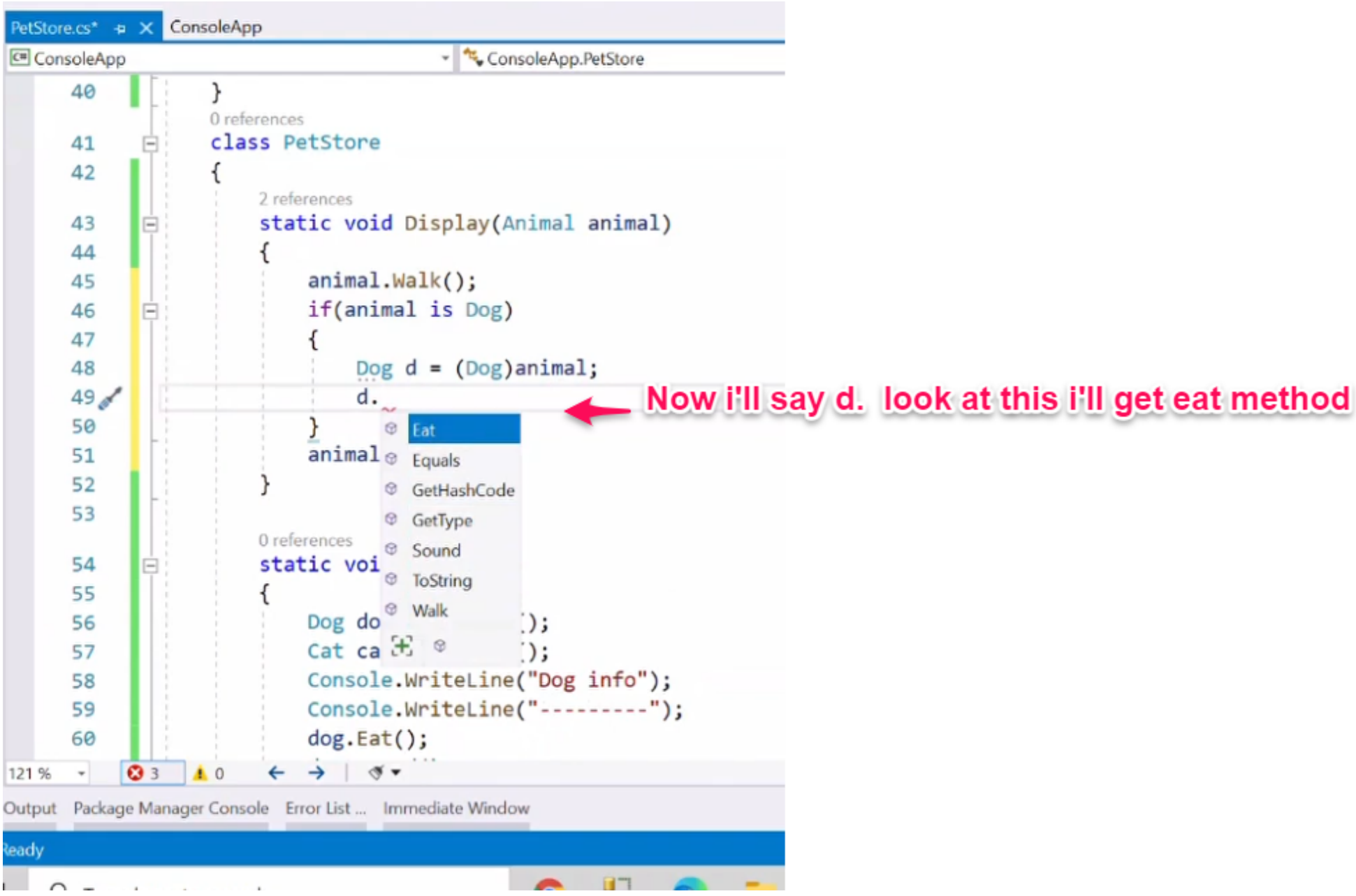


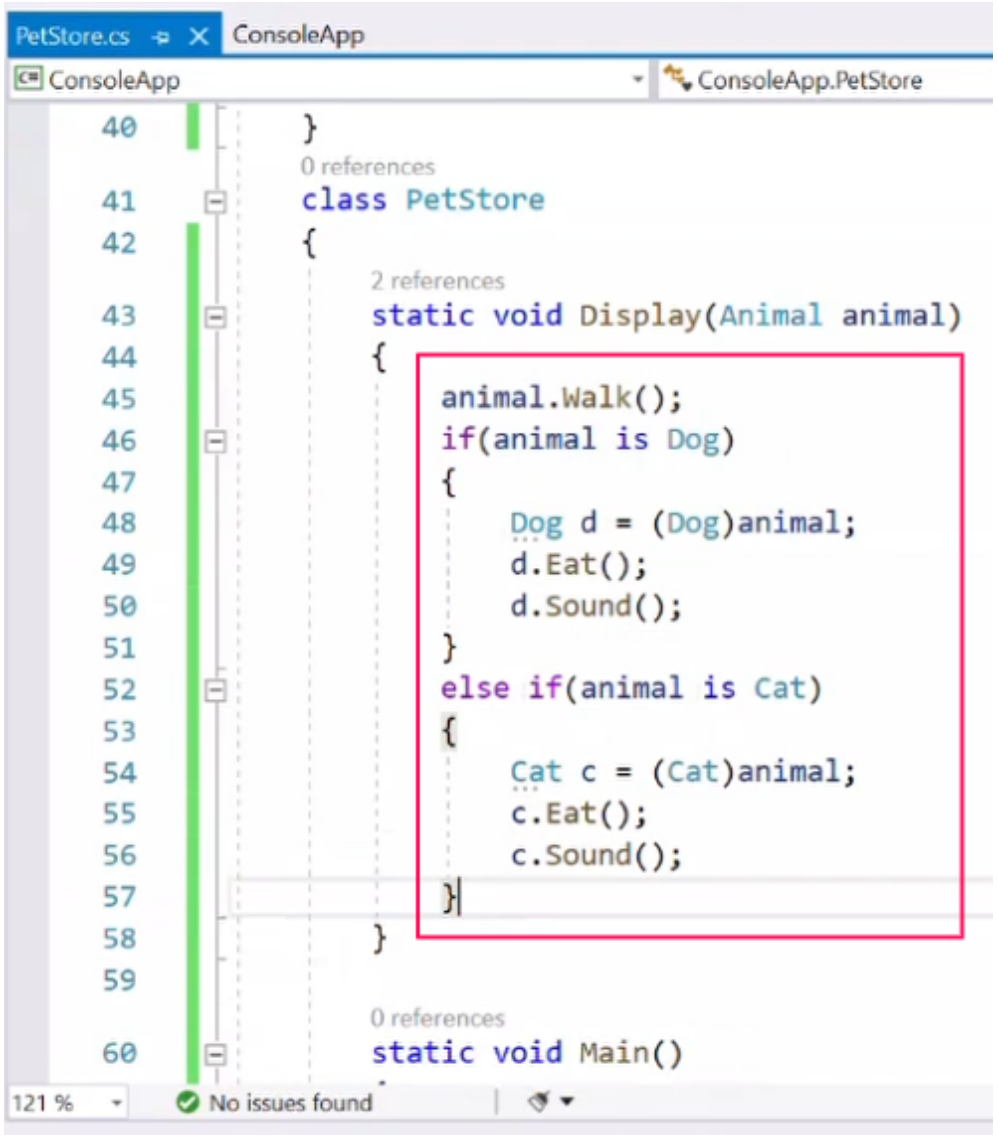
if it is holding a reference of dog

then i'll say

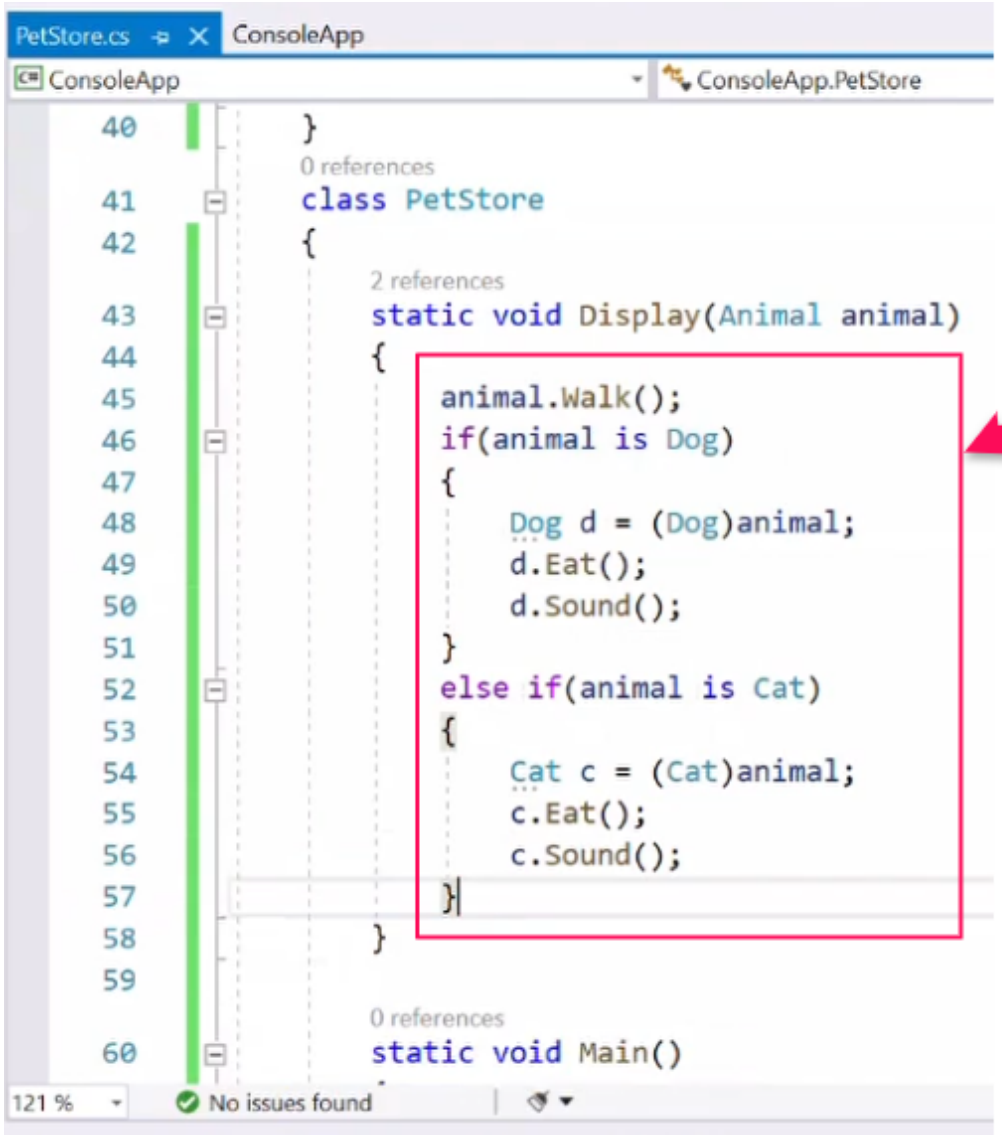
i'm not creating a new object here

i'm down casting my animal here

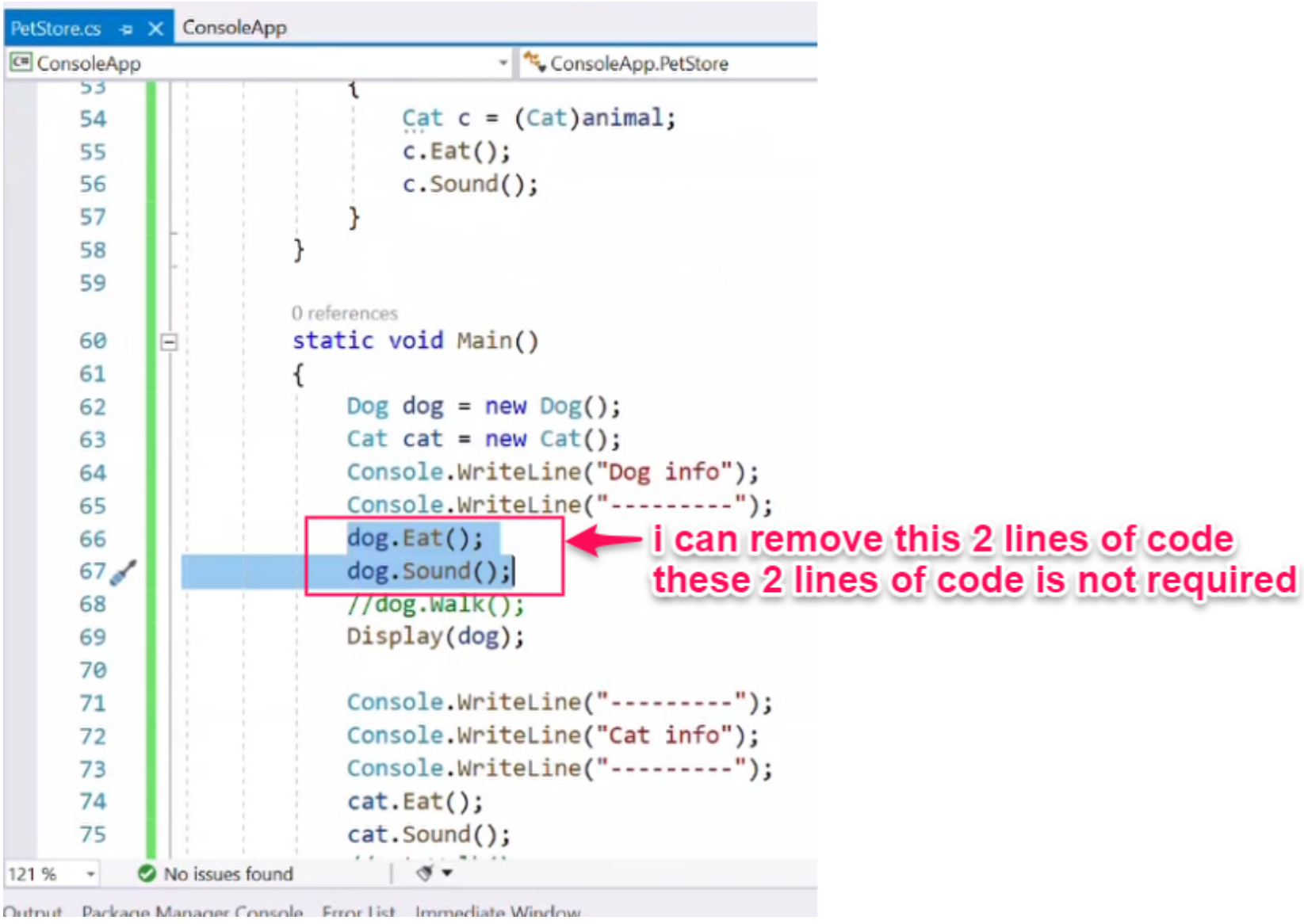

PetStore.cs*
ConsoleApp
ConsoleApp
ConsoleApp.PetStore
}
0 references
class PetStore
{
2 references
static void Display(Animal animal)
{
animal.Walk();
if(animal is Dog)
{
Dog d = (Dog)animal;
d.
}
animal
}
0 references
static voi
{
Dog do
Cat ca
Console.WriteLine("Dog info");
Console.WriteLine("---------");
dog.Eat();
Eat
Equals
GetHashCode
GetType
Sound
ToString
Walk
Now i'll say d. look at this i'll get eat method
121 %
3
0
Output
Package Manager Console
Error List ...
Immediate Window
Ready

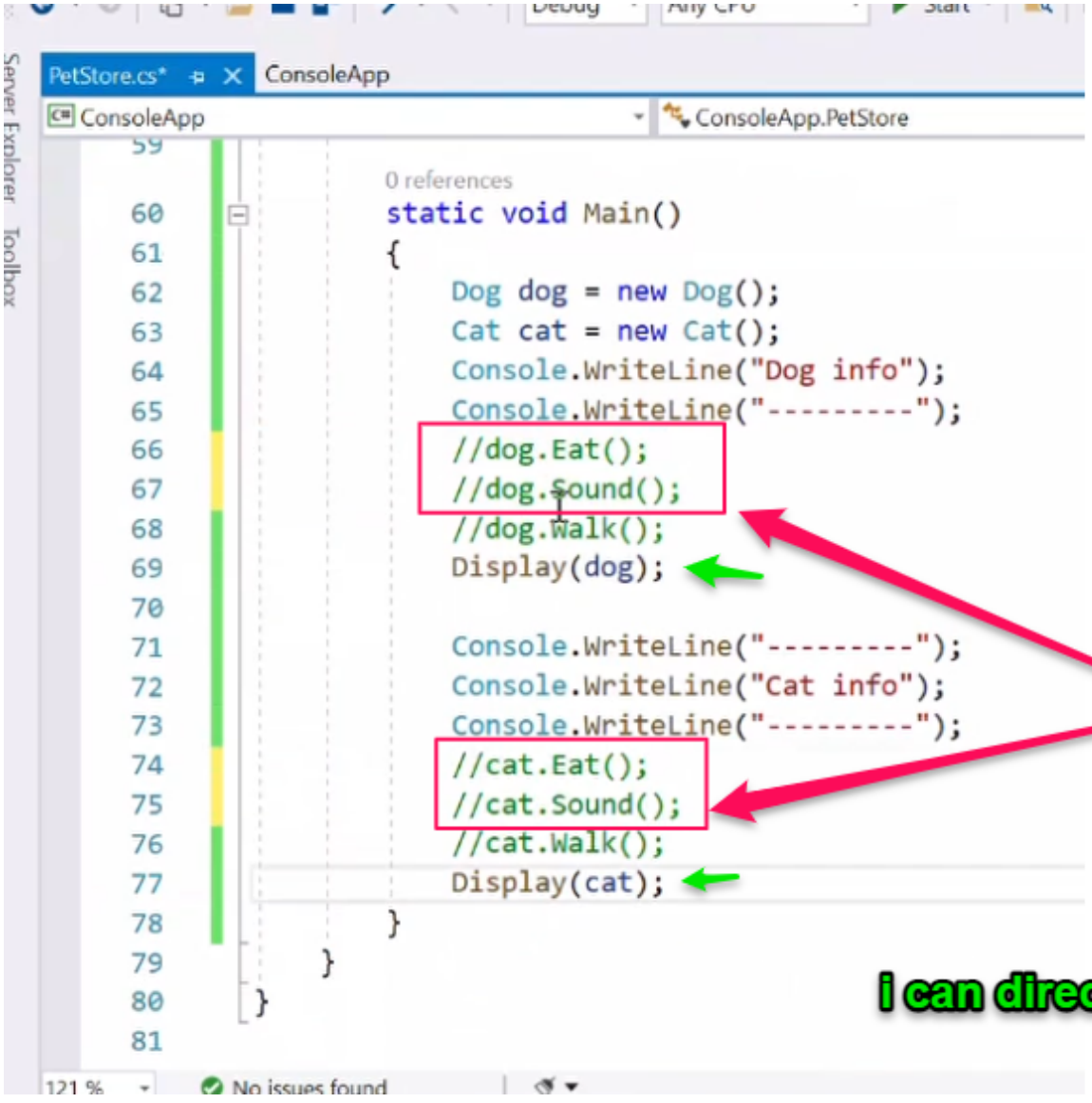


From the parent if you want to access the child information you should always Down Caste it



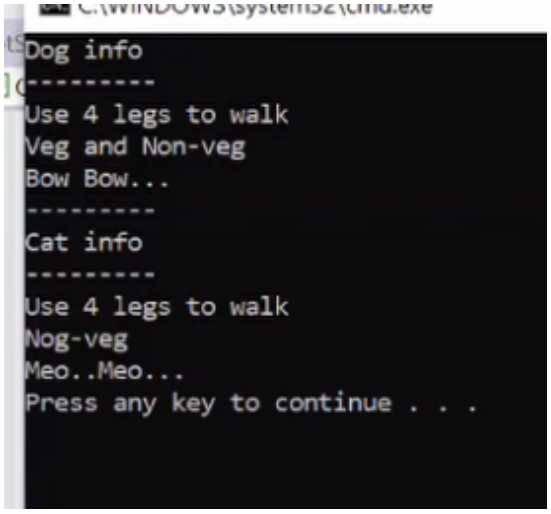
Now as i'm writing
this particular piece of code


PetStore.cs
ConsoleApp
ConsoleApp
ConsoleApp.PetStore
{
Cat c = (Cat)animal;
c.Eat();
c.Sound();
}
}
0 references
static void Main()
{
Dog dog = new Dog();
Cat cat = new Cat();
Console.WriteLine("Dog info");
Console.WriteLine("---------");
dog.Eat();
dog.Sound();
//dog.Walk();
Display(dog);
Console.WriteLine("---------");
Console.WriteLine("Cat info");
Console.WriteLine("---------");
cat.Eat();
cat.Sound();
i can remove this 2 lines of code
these 2 lines of code is not required
121 %
No issues found
Output
Package Manager Console
Error List
Immediate Window



Look at this
How beautifully we have
eliminated these lines

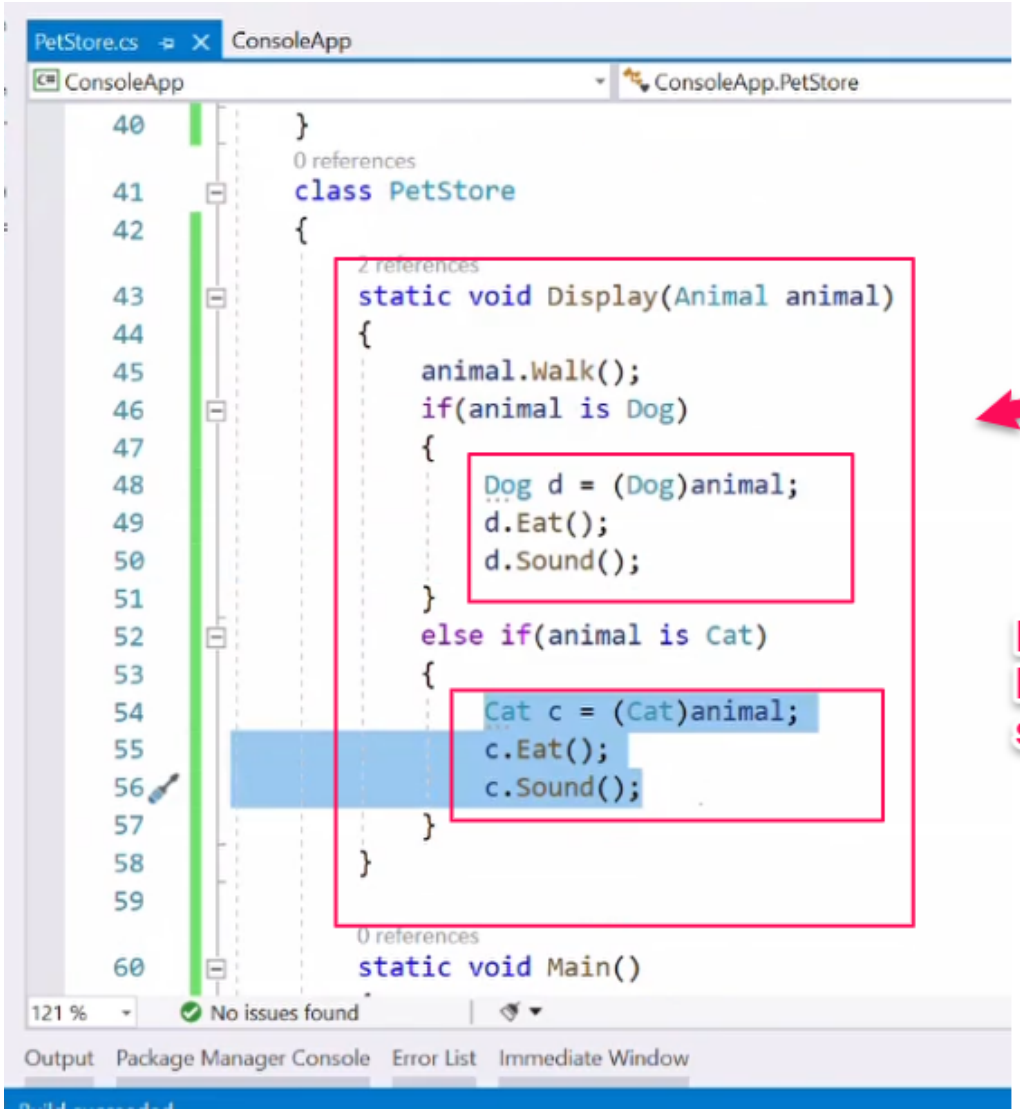
i can directly call display Dog and display cat


C:\WINDOWS\system32\cmd.exe
Dog info

Use 4 legs to walk
Veg and Non-veg
Bow Bow...

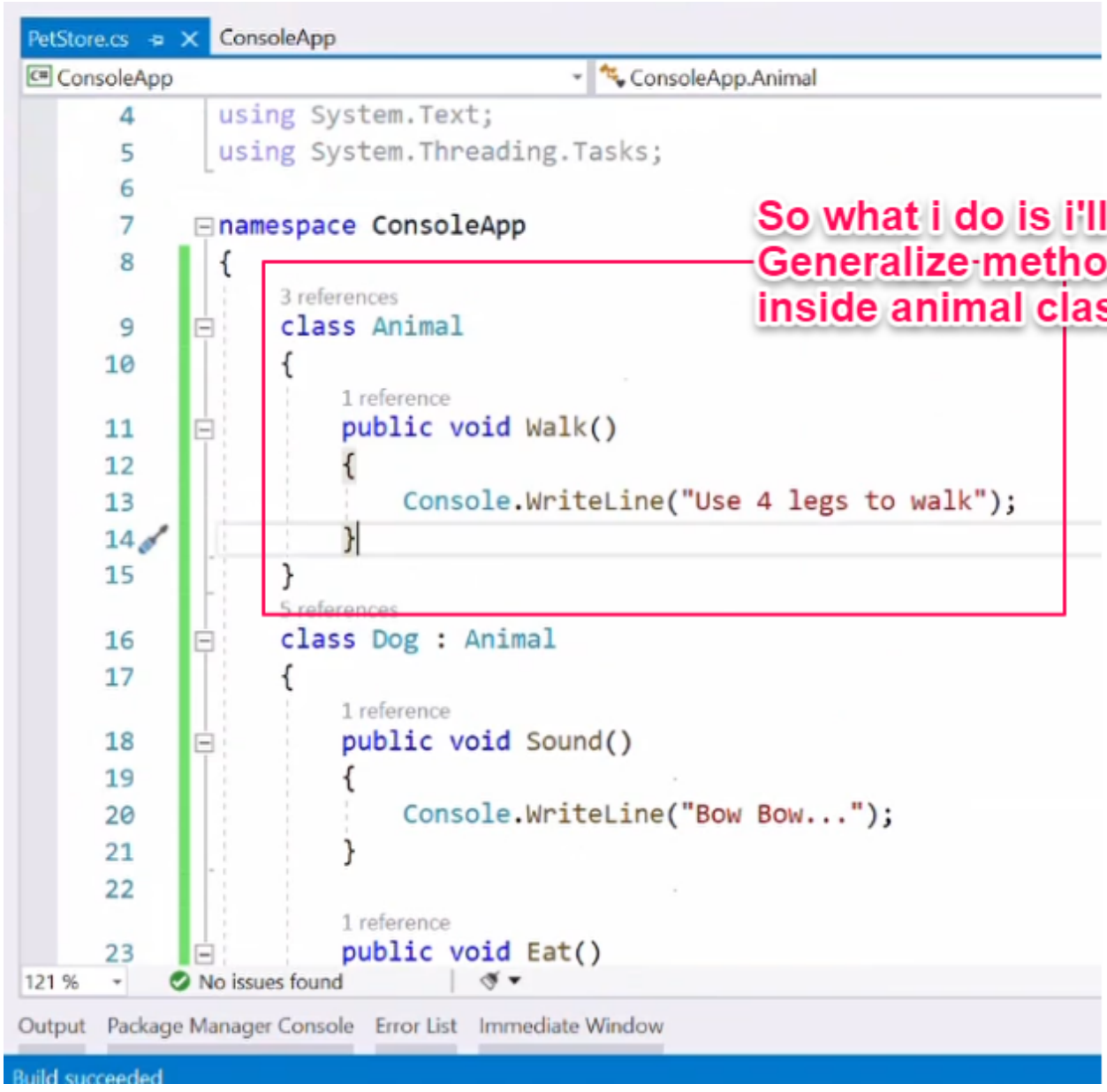
Cat info

Use 4 legs to walk
Nog-veg
Meo..Meo...
Press any key to continue . . .



Now
Me being developer
i was thinking aree yaar
this is bakwas code

I cannot keep doing if and else
because tomorrow if i'll going to add new animal
so i cannot keep doing this if else if else



So what i do is i'll write all the Generalize methods here inside animal class

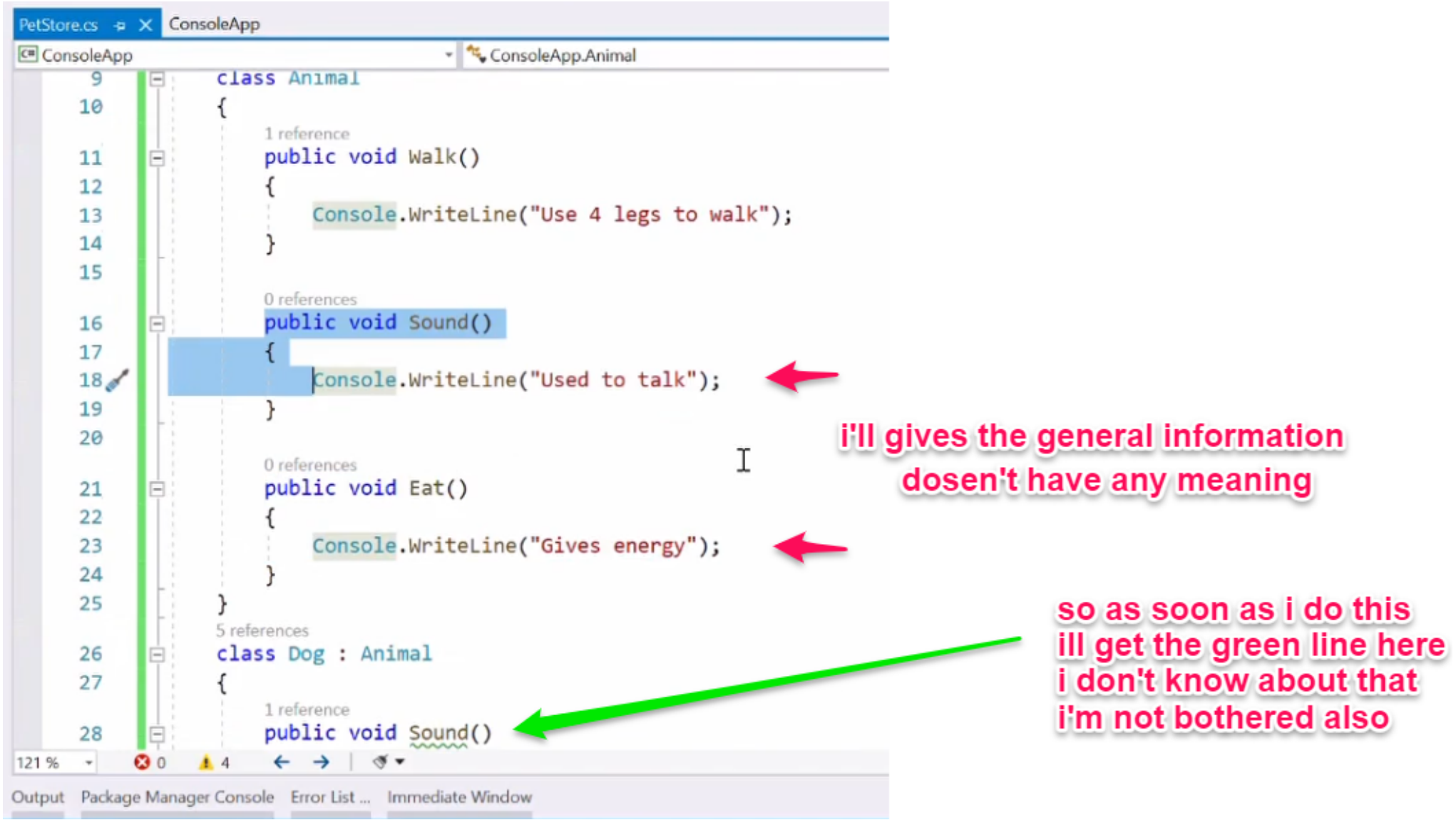
So by doing this we are saying father has property Z and Y and son and daughter will also have Z and Y so no need of me doing Down Casting it

i can directly say

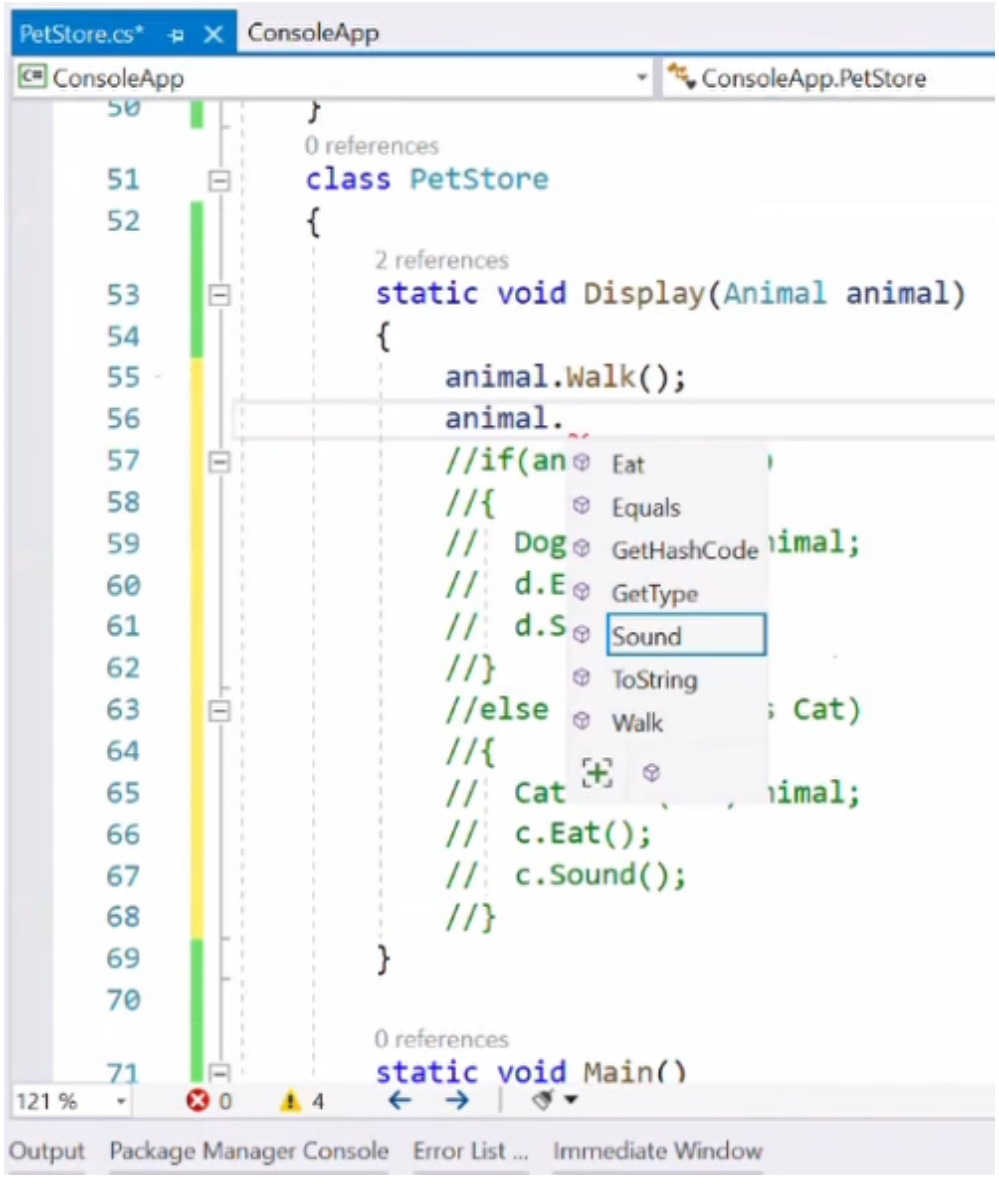
f.X;

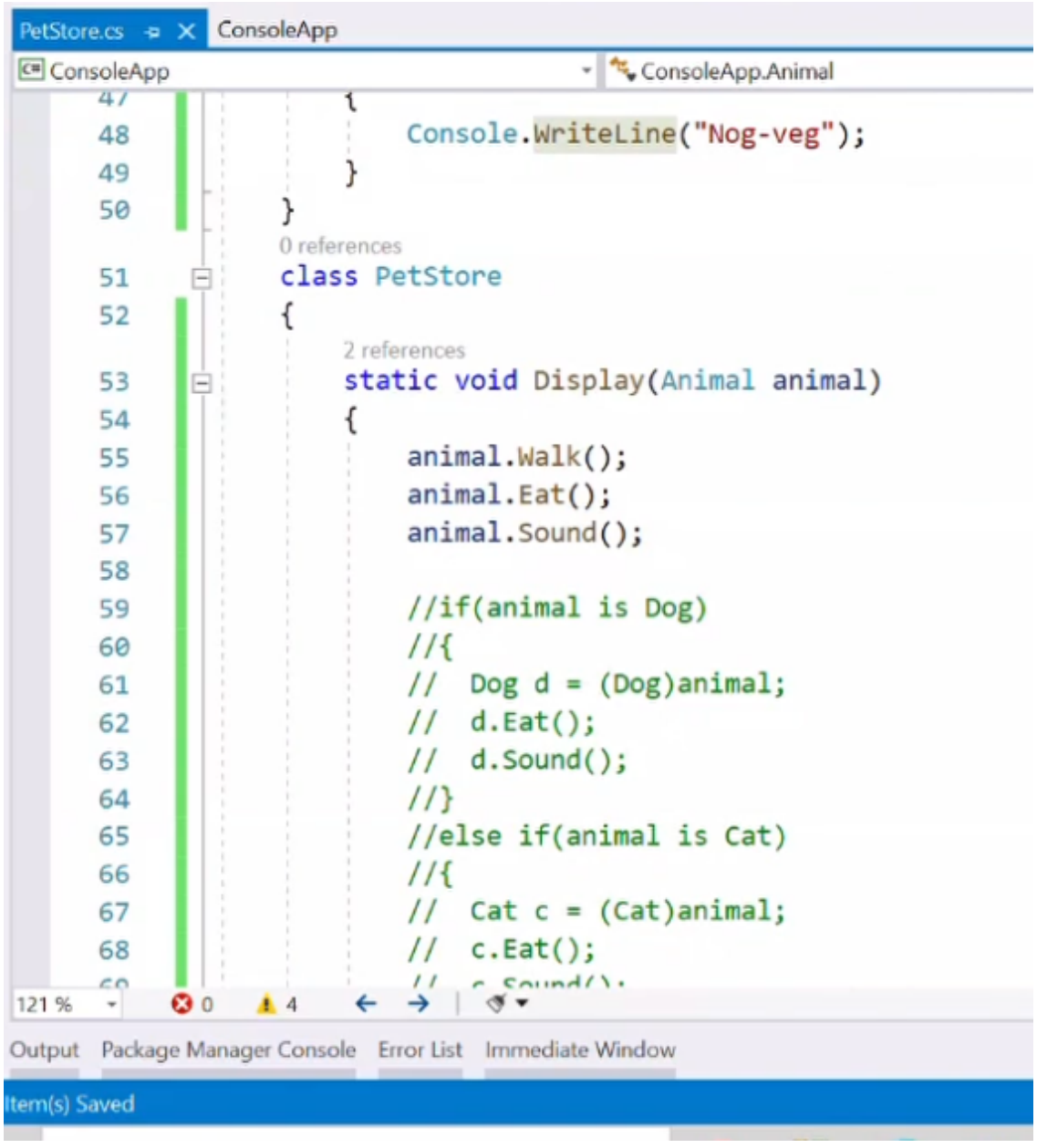
f.Y;

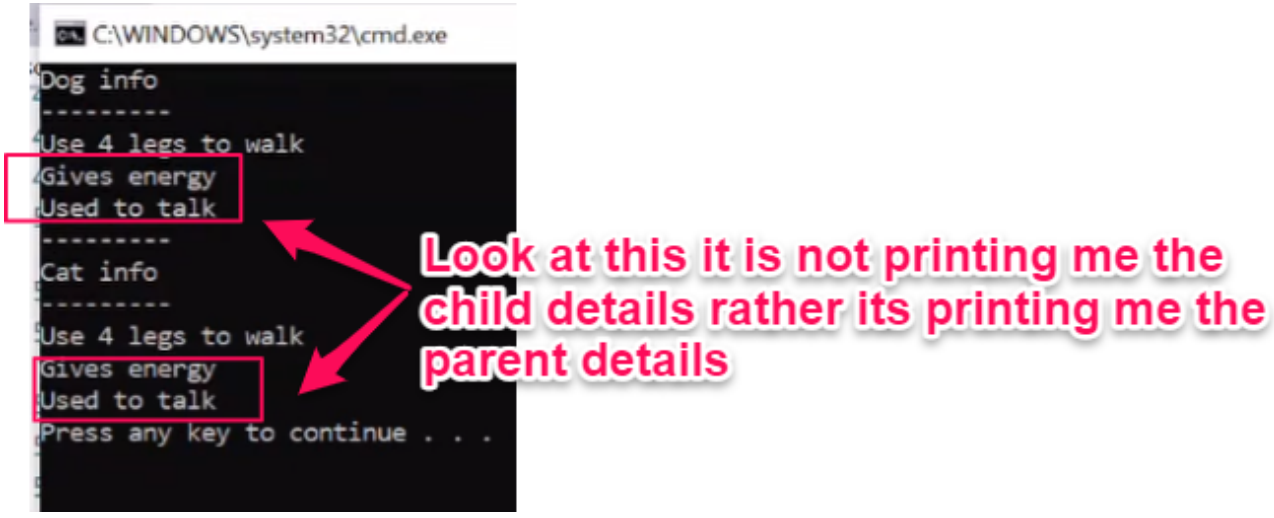
f.Z;



Now you can see Animal has Eat walk ..... all the methods


PetStore.cs*
ConsoleApp
ConsoleApp
ConsoleApp.PetStore
}
0 references
class PetStore
{
2 references
static void Display(Animal animal)
{
animal.Walk();
animal.
//if(an
//{
// Dog
// d.E
// d.S
//}
//else
//{
// Cat
// c.Eat();
// c.Sound();
//}
}
0 references
static void Main()
Eat
Equals
GetHashCode
GetType
Sound
ToString
Walk
121 %
0
4
Output
Package Manager Console
Error List ...
Immediate Window


PetStore.cs
ConsoleApp
ConsoleApp
ConsoleApp.Animal
{
Console.WriteLine("Nog-veg");
}
}
0 references
class PetStore
{
2 references
static void Display(Animal animal)
{
animal.Walk();
animal.Eat();
animal.Sound();
//if(animal is Dog)
//{
// Dog d = (Dog)animal;
// d.Eat();
// d.Sound();
//}
//else if(animal is Cat)
//{
// Cat c = (Cat)animal;
// c.Eat();
121 %
0
4
Output
Package Manager Console
Error List
Immediate Window
Item(s) Saved



SO this is wrong this is not what i want

but in Java if i write this particular code it will print me the information of child Because all methods in java are virtual by default and that's the biggest blunder of java that's why java is so slow

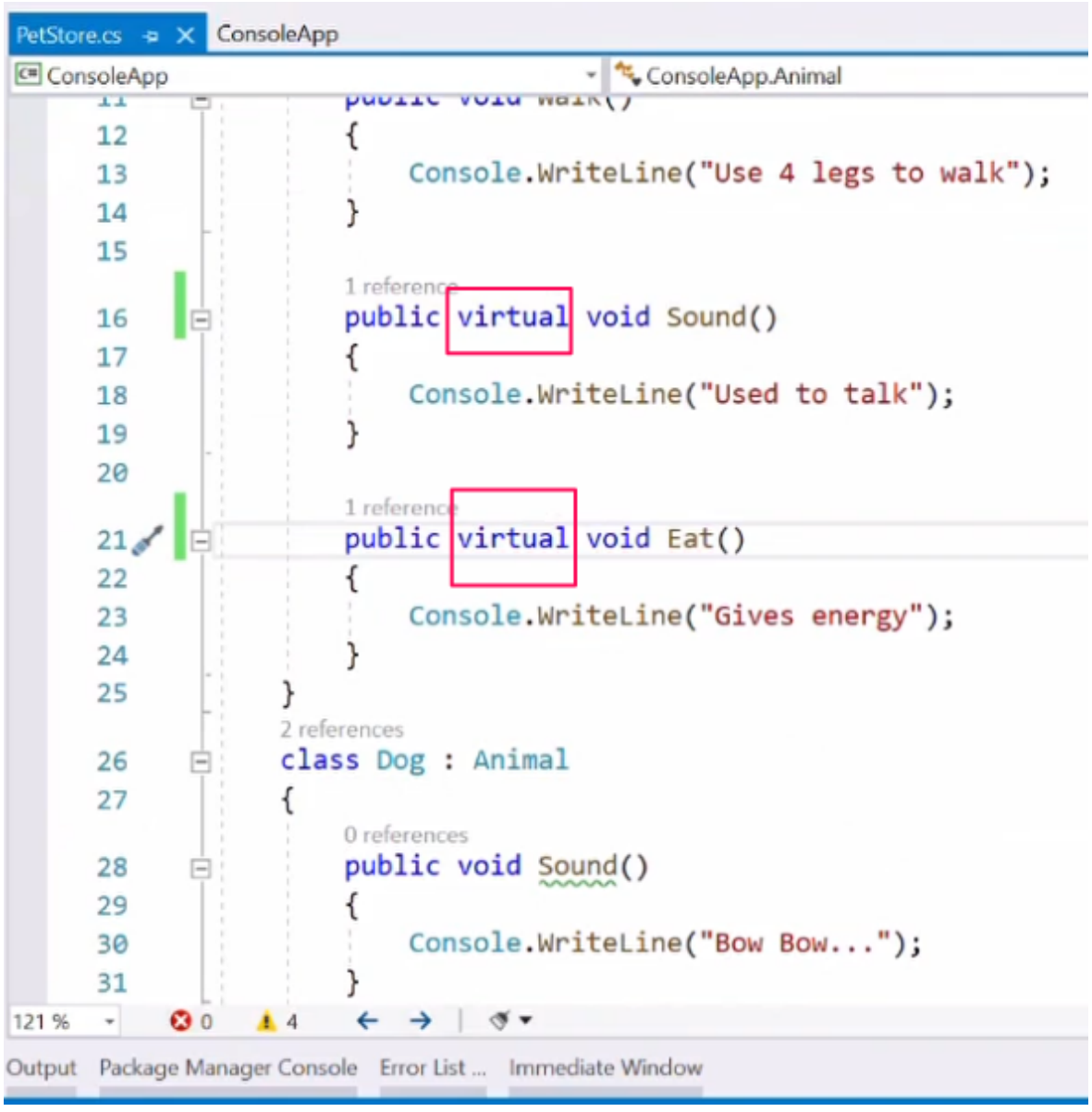
So this is wrong output

So what want is if i'm passing Dog here so i should get dog information, if i pass cat i should get cat information

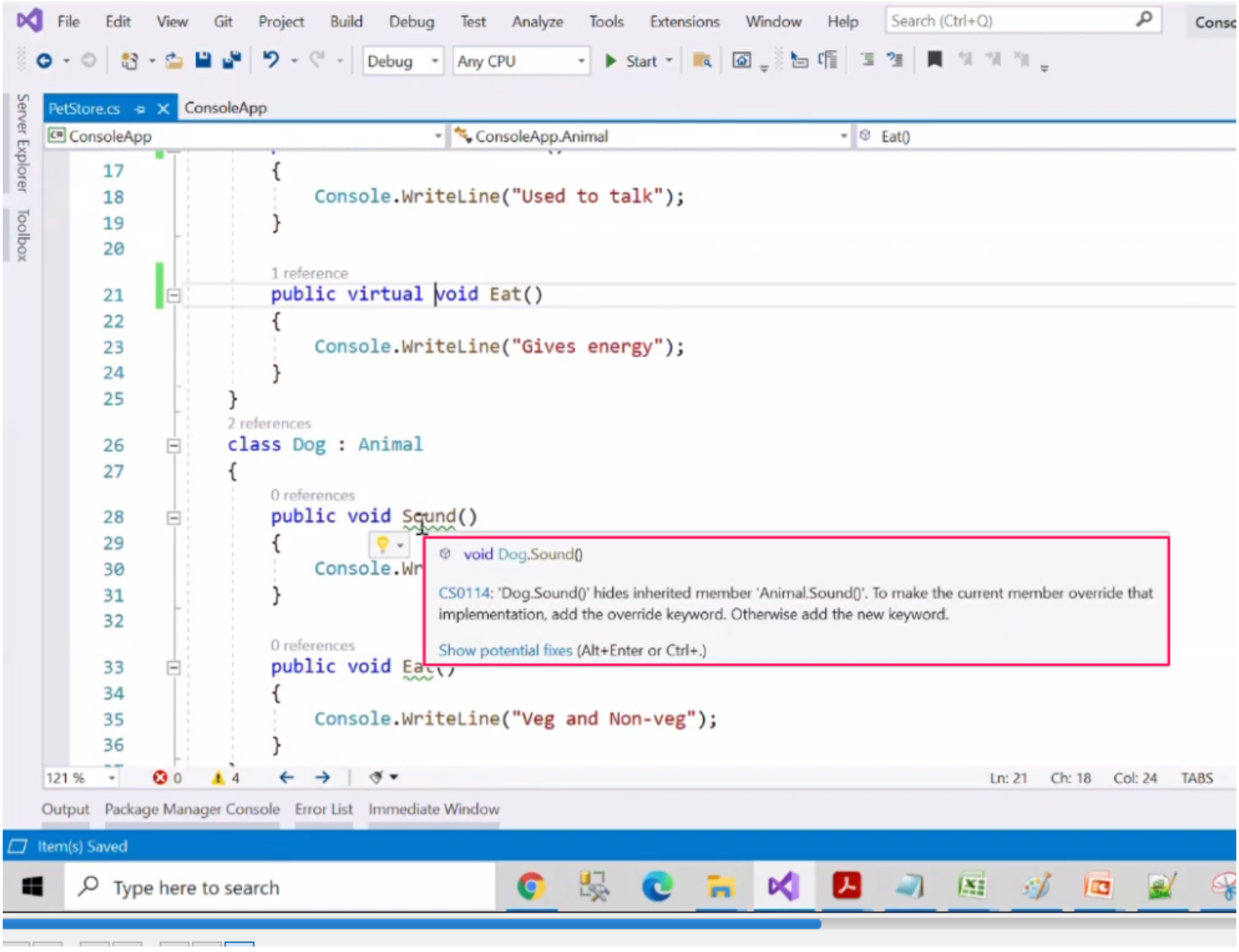
meaning based on the reference what we are passing here in display, the respective information of that class gets printed

So How will i achieve that

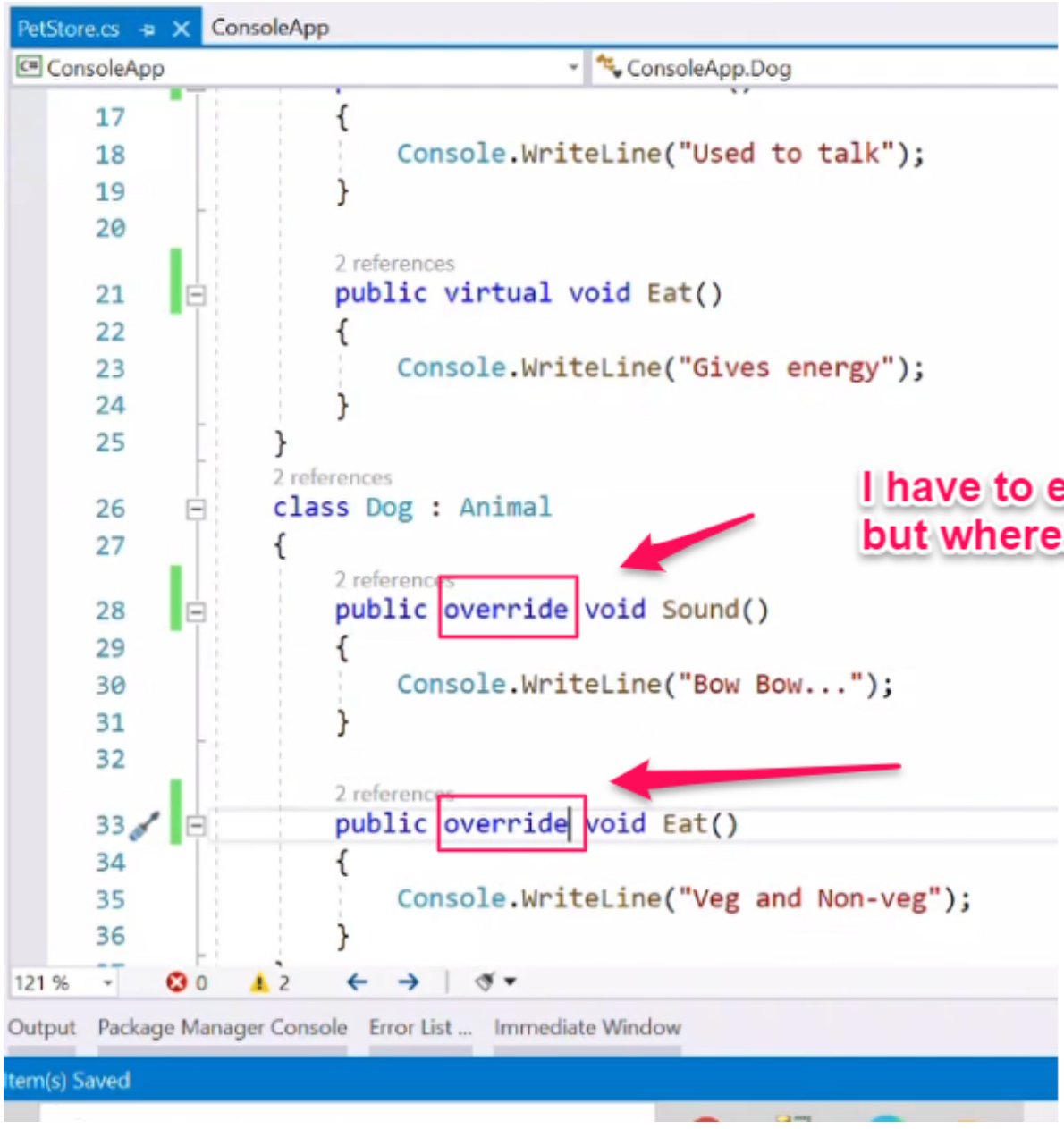
So i'll just add **Virtual** KeyWord here



So as soon as i add virtual here it will say boss is this a overriding method

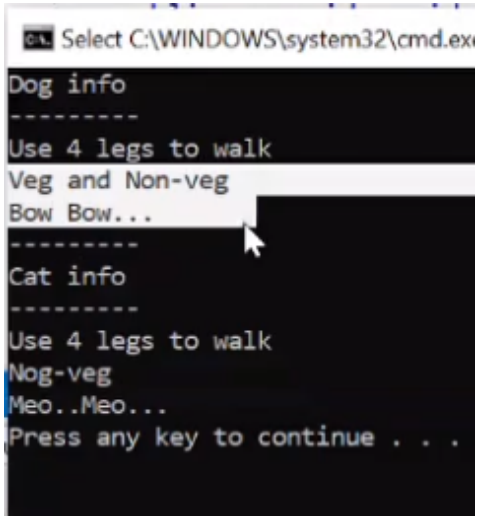

File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search (Ctrl+Q)
Debug
Any CPU
Start
PetStore.cs
ConsoleApp
ConsoleApp
ConsoleApp.Animal
Eat()
Server Explorer
Toolbox
{
Console.WriteLine("Used to talk");
}
1 reference
public virtual void Eat()
{
Console.WriteLine("Gives energy");
}
}
2 references
class Dog : Animal
{
0 references
public void Sound()
{
Console.Wr
}
0 references
public void Eat()
{
Console.WriteLine("Veg and Non-veg");
}
void Dog.Sound()
CS0114: 'Dog.Sound()' hides inherited member 'Animal.Sound()'. To make the current member override that implementation, add the override keyword. Otherwise add the new keyword.
Show potential fixes (Alt+Enter or Ctrl+.)
121 %
0
4
Ln: 21
Ch: 18
Col: 24
TABS
Output
Package Manager Console
Error List
Immediate Window
Item(s) Saved
Type here to search

So i'll say yes this is a overriding method



I have to explicitly mention this in .Net
but whereas java does this automatically

so i'm just saying that it's a overriding method in my dog class as well as in the cat class



So even thoe you pass the reference of the child it automatically picks the information of the child not the parent because we are using virtual here, so internally a virtual table is created where these particular methods are registered and whenever you say override it will gone to replace dynamically

this is what we called as **Dynamic Binding**

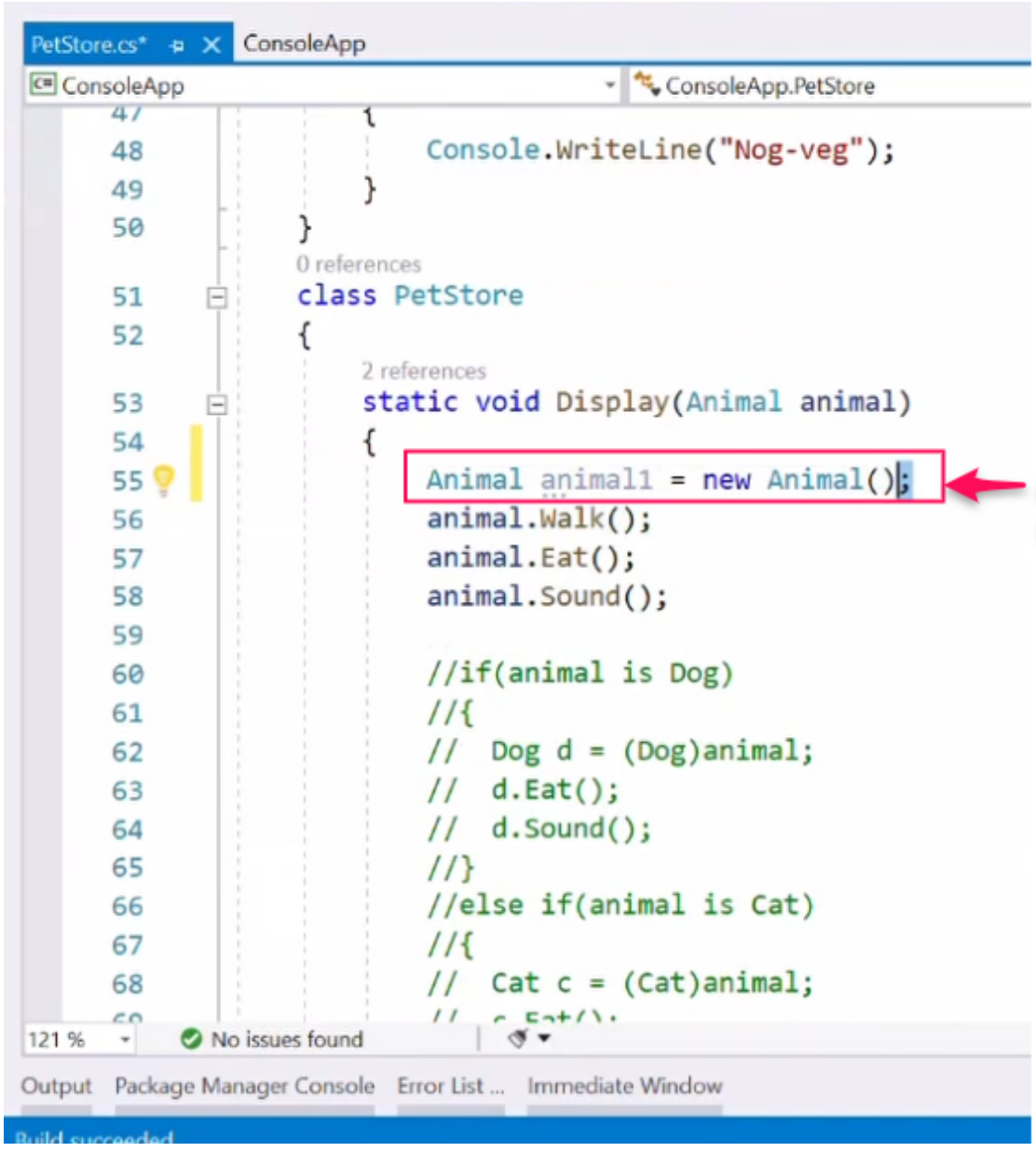
this is a concept of Over Riding

we are overriding the method on the fly

Now based on the object what we are passing weather it is a dog or a cat class whatever you are passing based on that the respective information gets printed

So whenever you see such code you should relate like this look like animal class this looks like cat class this looks like dog class

Now there is a possibility that we can create object of Animal class



Any way we can create object of animal class

But Do Animal Exist in Real World

------No

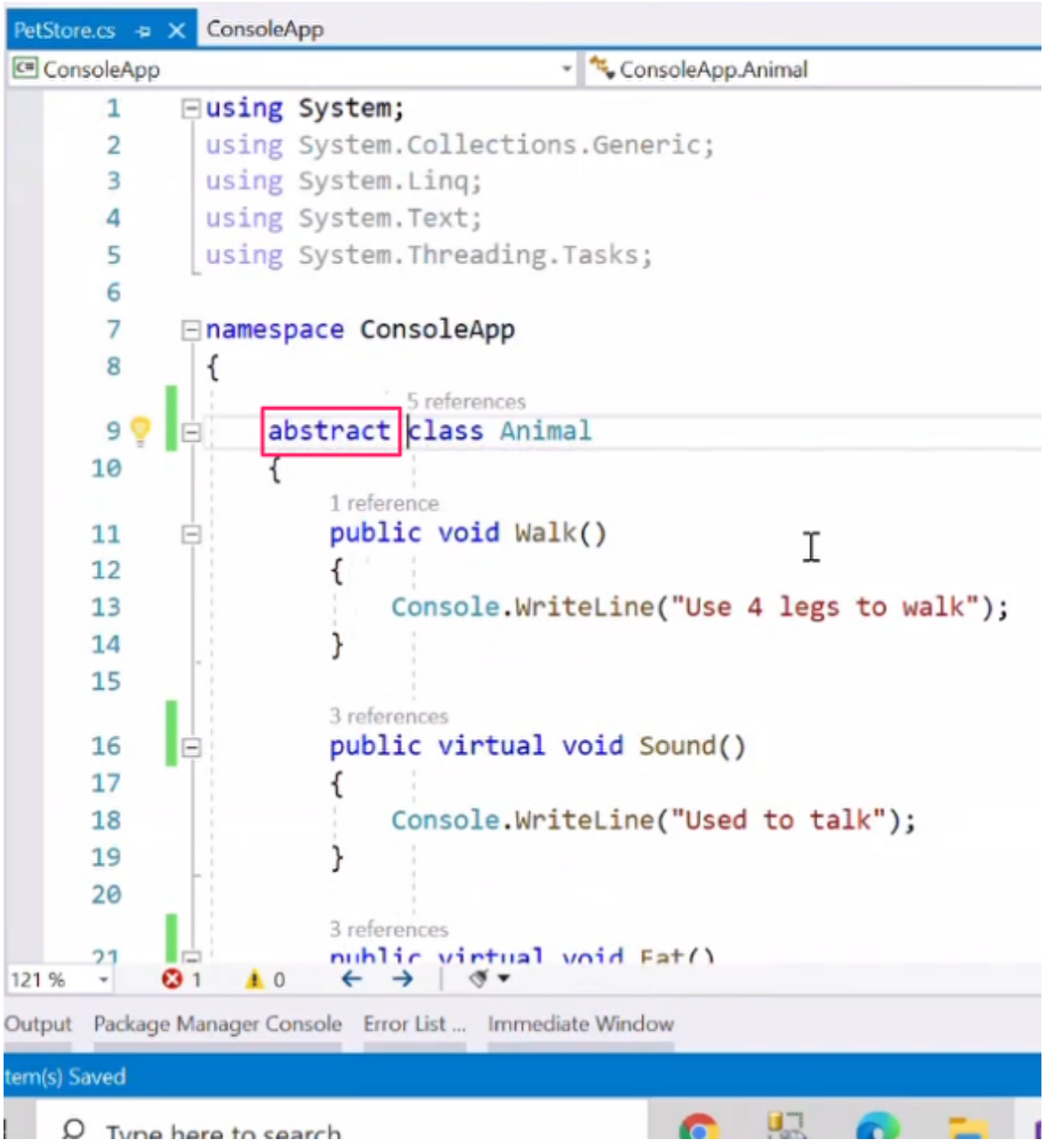
So i want that nobody will creates object of animal class

i don't want anybody to say

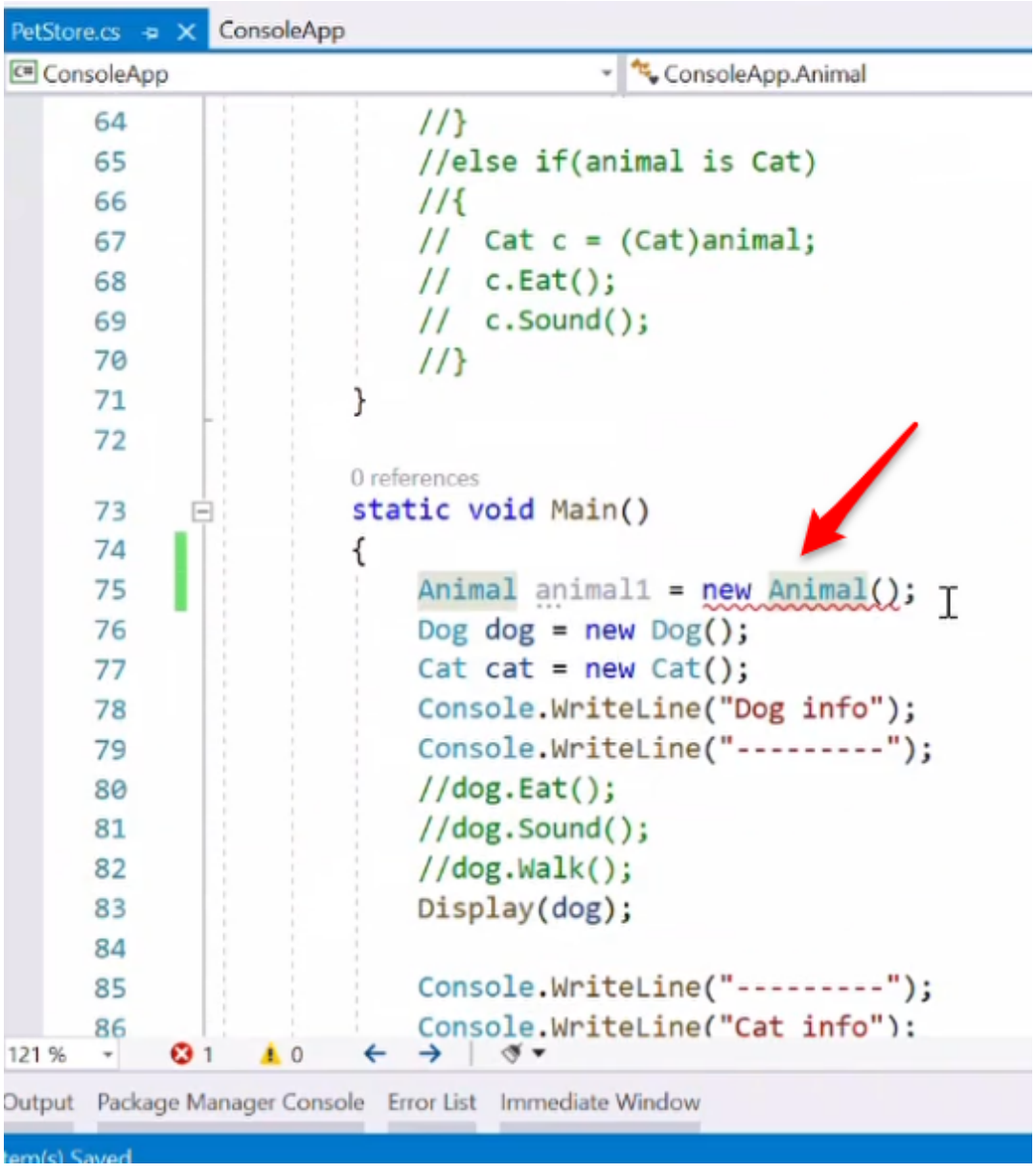
Animal animal1 = new Animal();

because this is bad, animal doesn't exist in real world So, How will i restrict them ?

By Making the animal class as Abstract we are ensuring that no one can create the object of that animal class

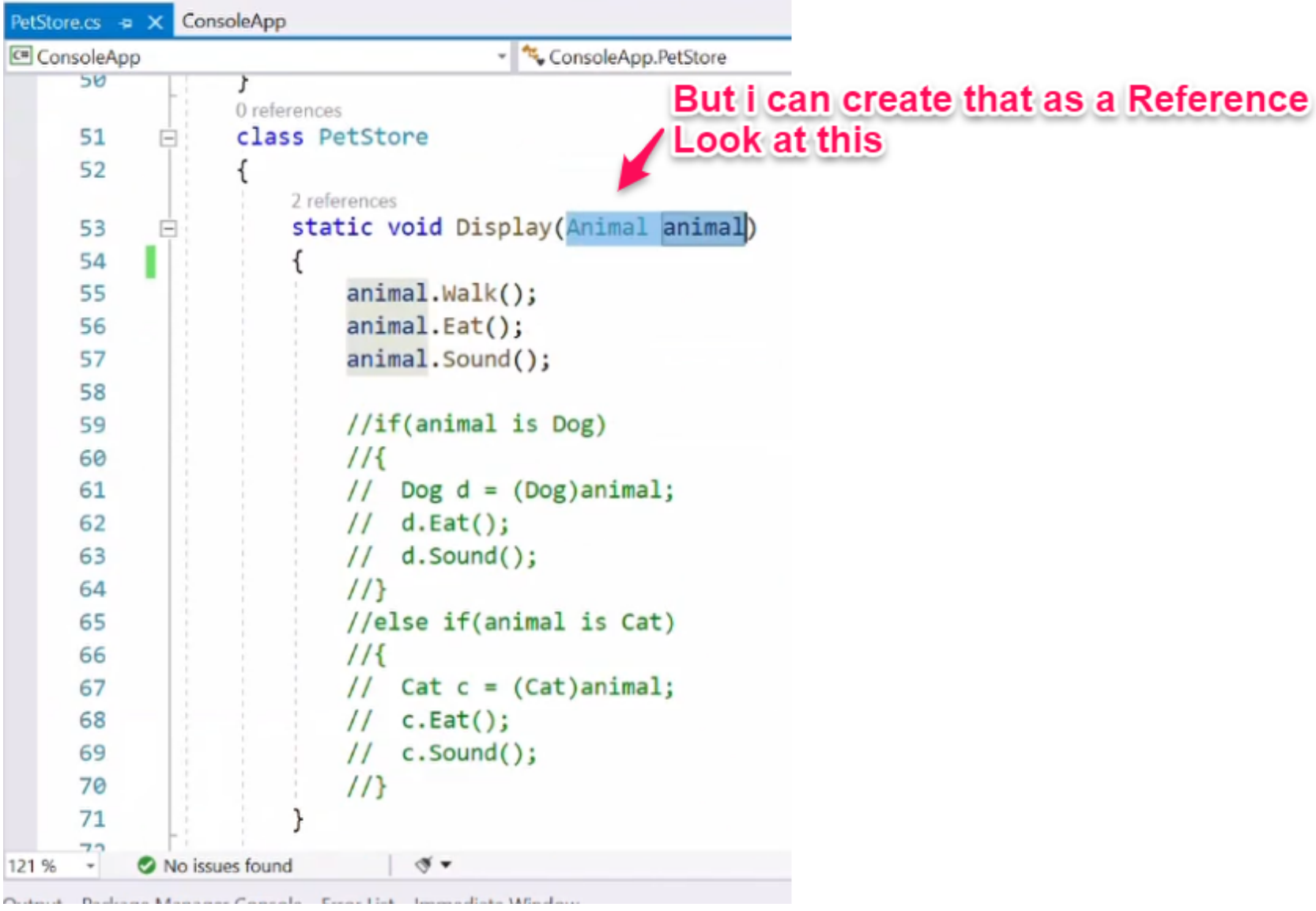


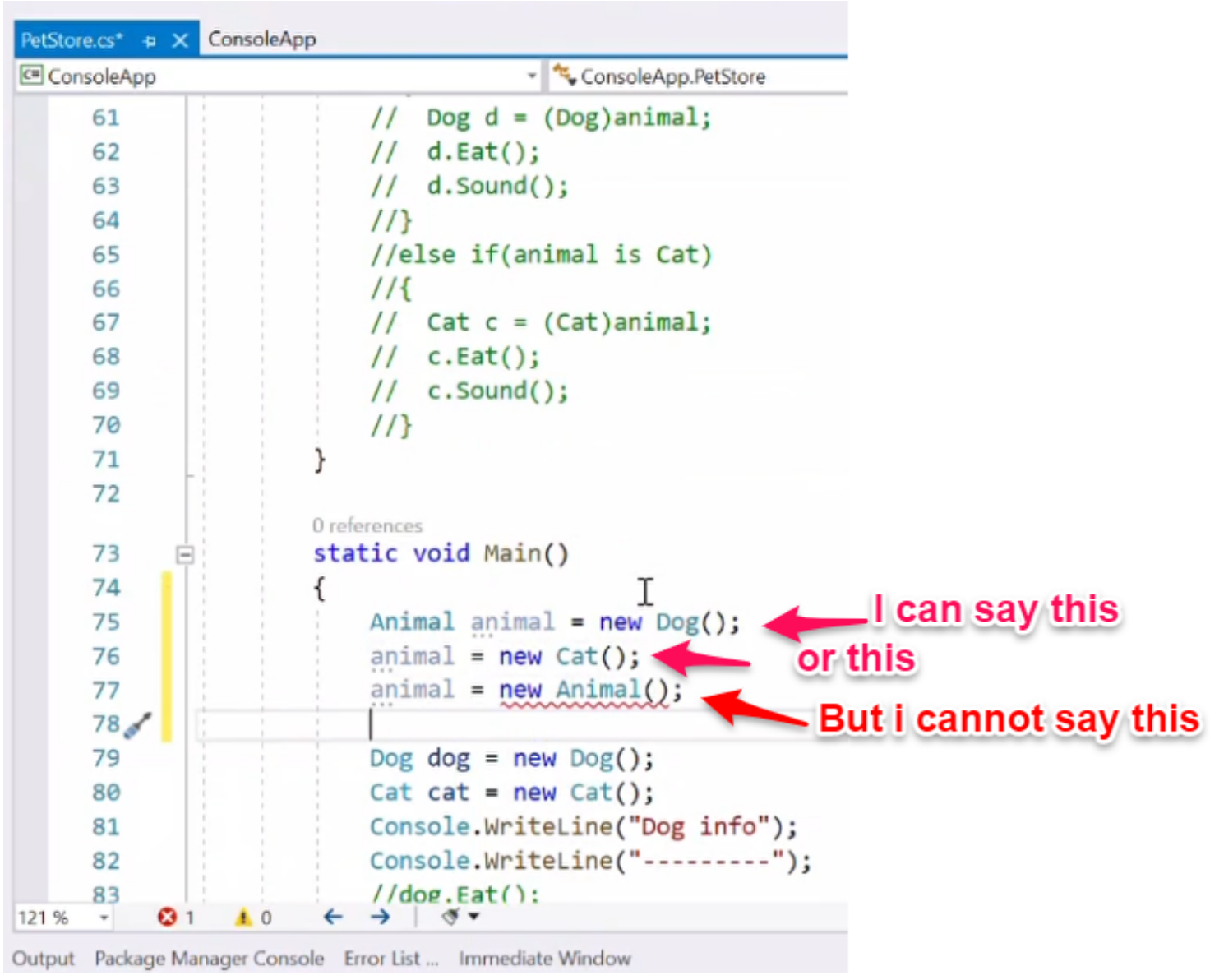
As soon as i do this you can see i'll get Error here


PetStore.cs
ConsoleApp
ConsoleApp
ConsoleApp.Animal
64 //}
65 //else if(animal is Cat)
66 //{
67 // Cat c = (Cat)animal;
68 // c.Eat();
69 // c.Sound();
70 //}
71 }
72
0 references
73 static void Main()
74 {
75 Animal animal1 = new Animal();
76 Dog dog = new Dog();
77 Cat cat = new Cat();
78 Console.WriteLine("Dog info");
79 Console.WriteLine("---------");
80 //dog.Eat();
81 //dog.Sound();
82 //dog.Walk();
83 Display(dog);
84
85 Console.WriteLine("---------");
86 Console.WriteLine("Cat info");
121 %
1
0
Output
Package Manager Console
Error List
Immediate Window

What are Abstract Classes ?

Abstract Classes are those classes of which we cannot create a Object or Instance

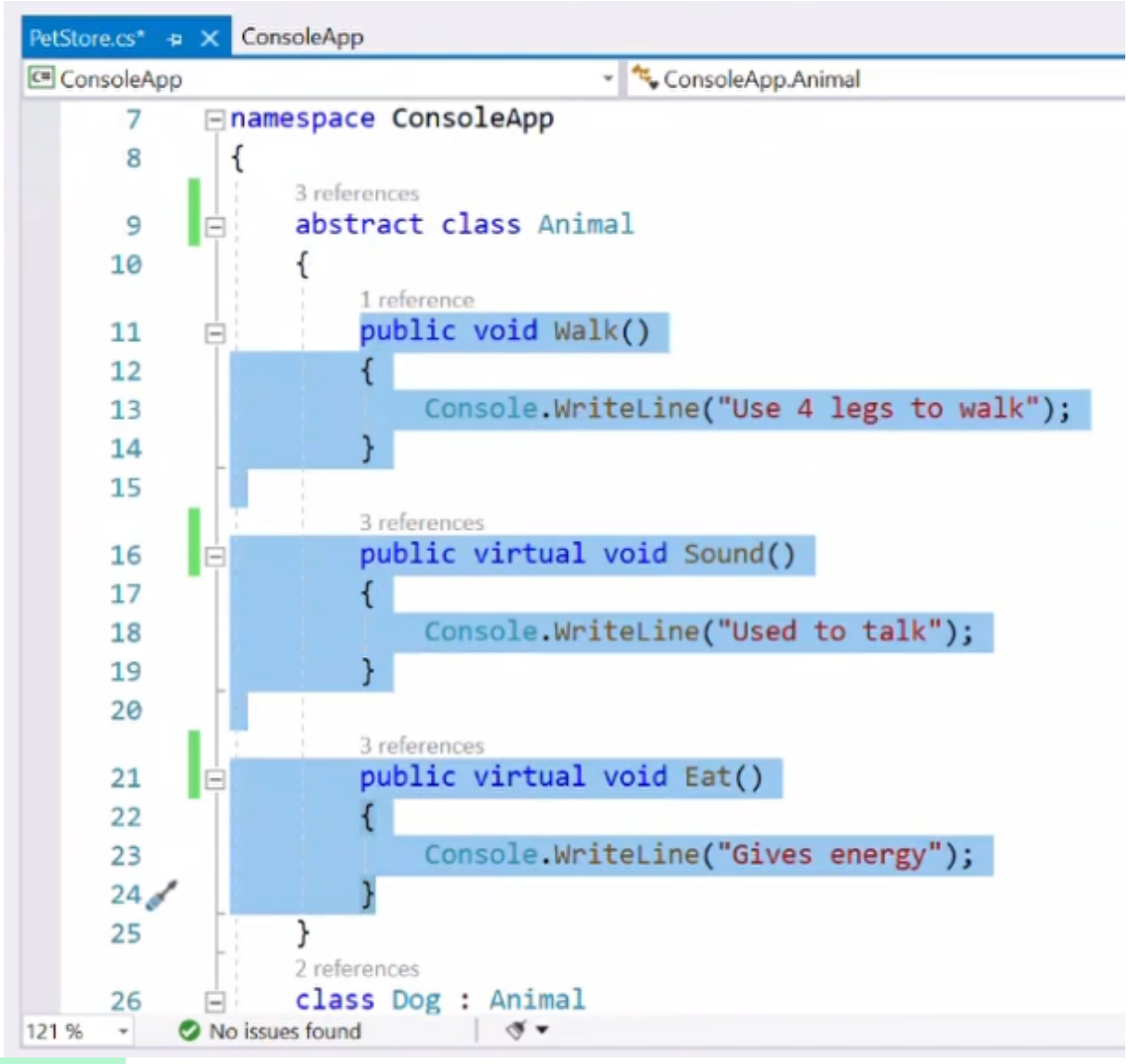




So whenever you have abstract classes you can create a reference of it, you cannot create a object

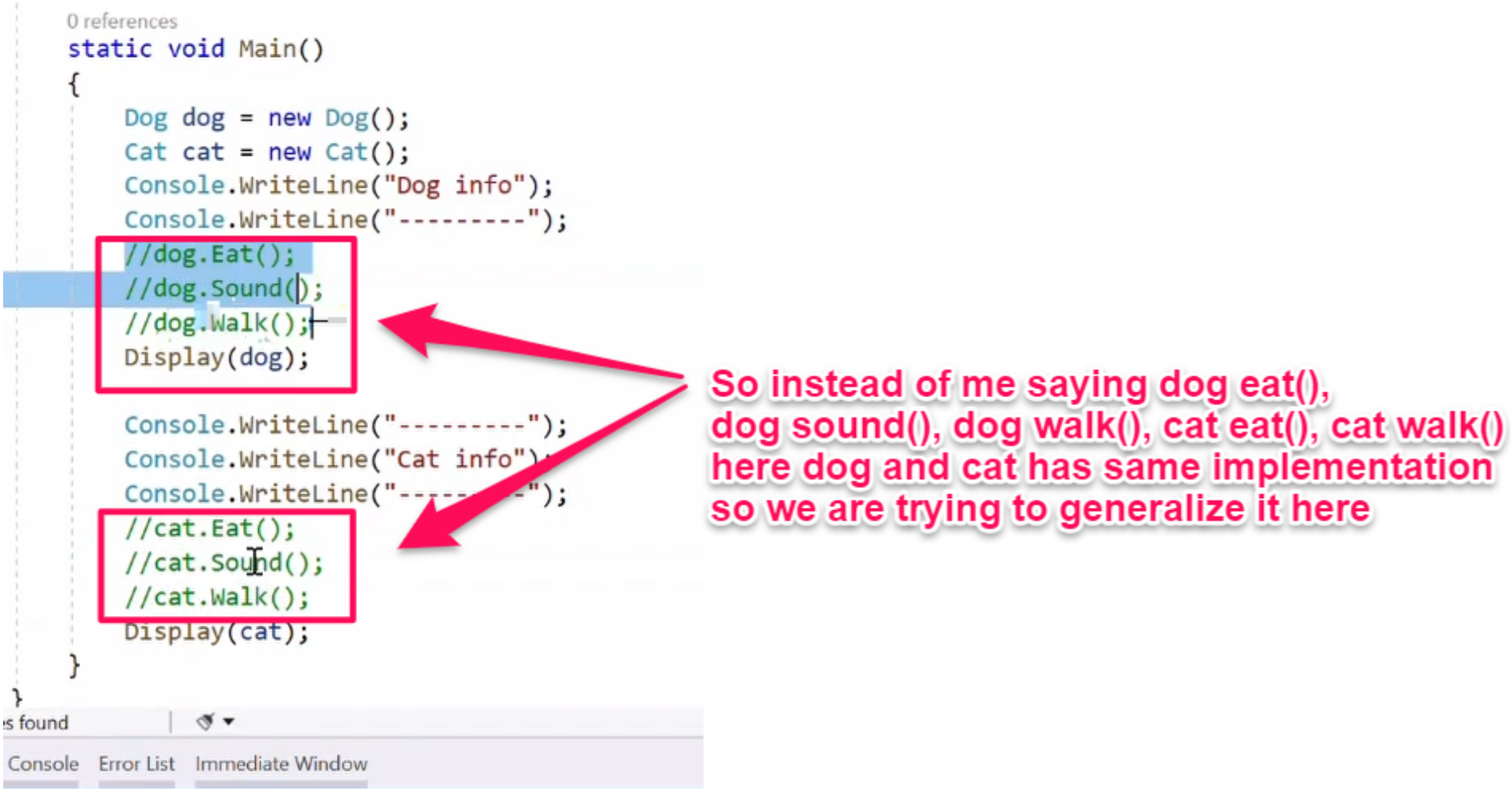
Behaviours of Abstract Class

1. **Abstract Classes May or May not have Abstract methods**

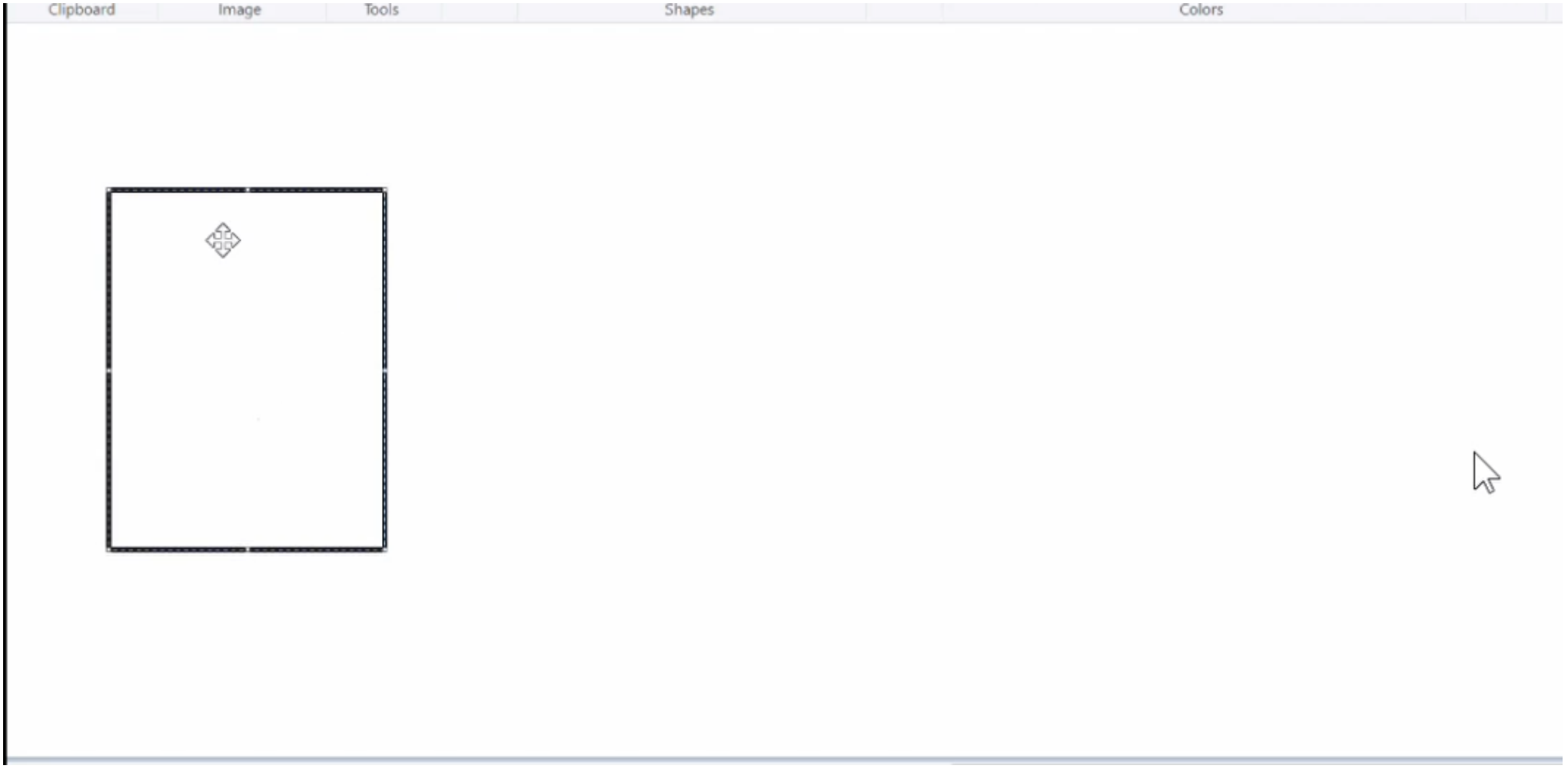


Summary: Why I use Animal as a reference If I want to create object of dog we can create reference of dog, If I want to create object of cat we can create reference of cat, Why i use reference of animal ?

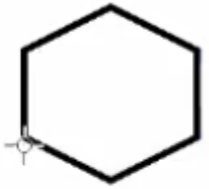
So



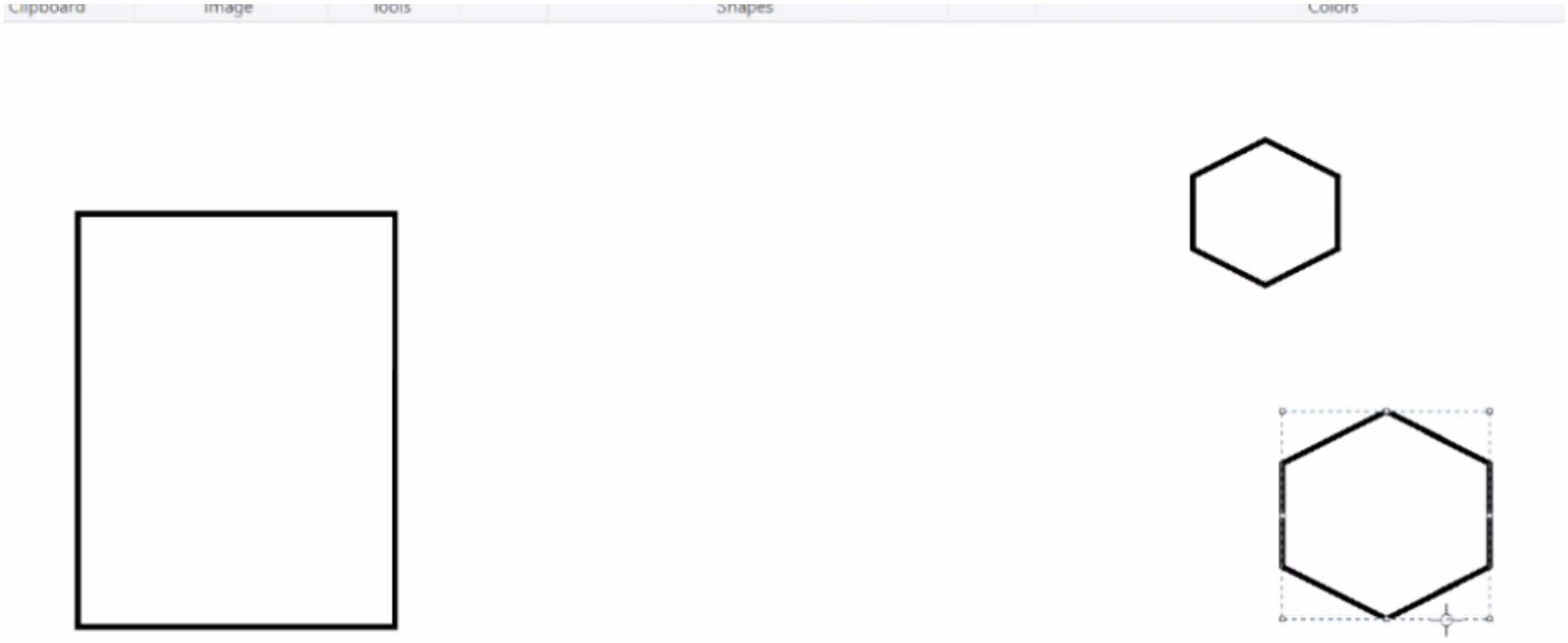
Now take a example that you are creating a huge and bulky application with different type of databases you have a piece of code and this piece of common code irrespective of database



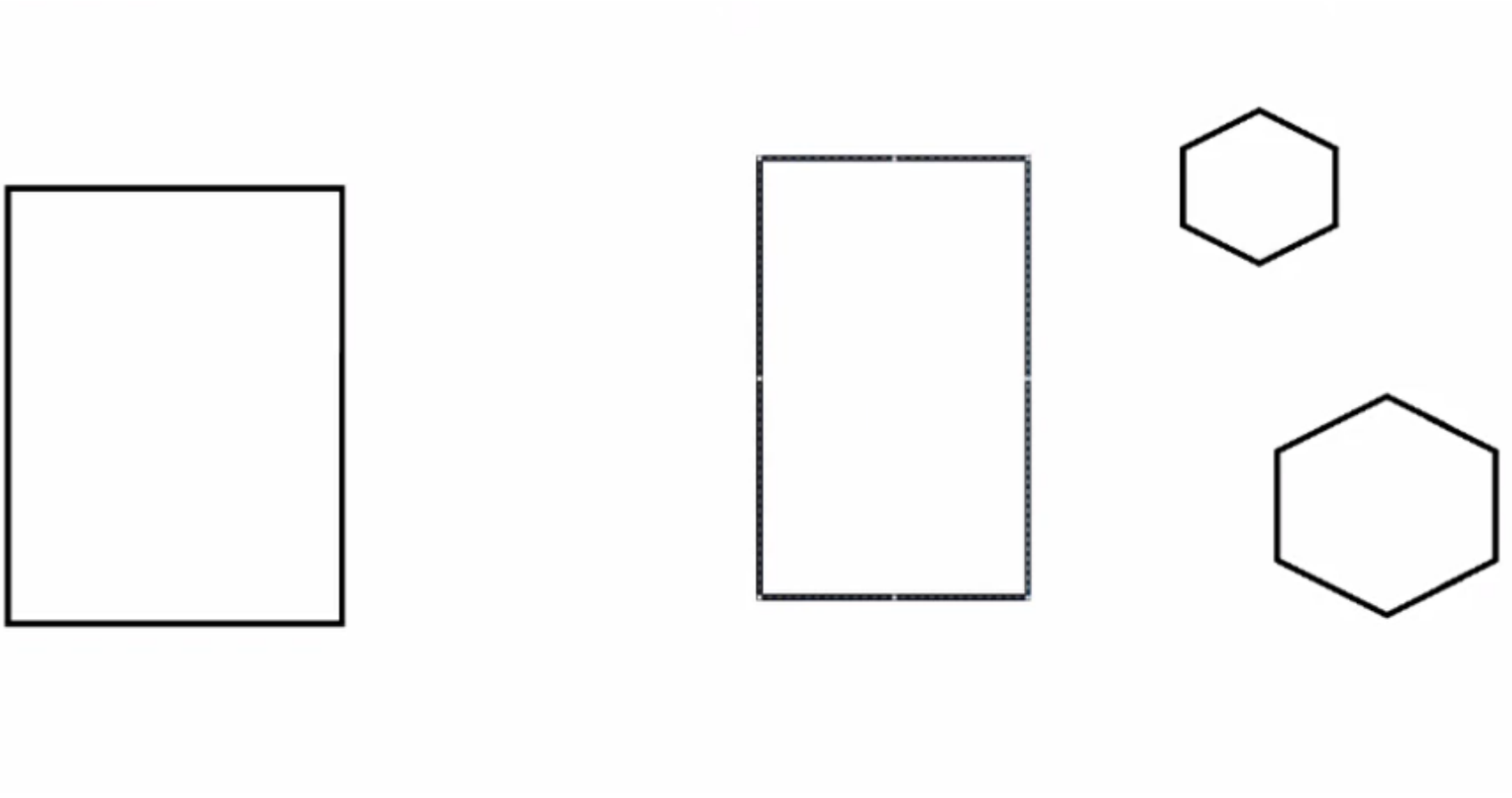
and 1 u have a access database, and it was a prety old database,



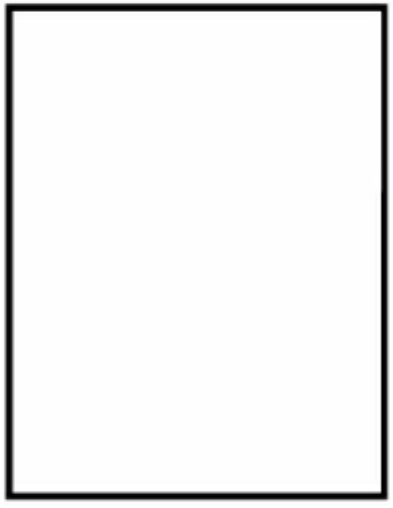
now we are trying to migrate the data from access database to new database i.e my sql database

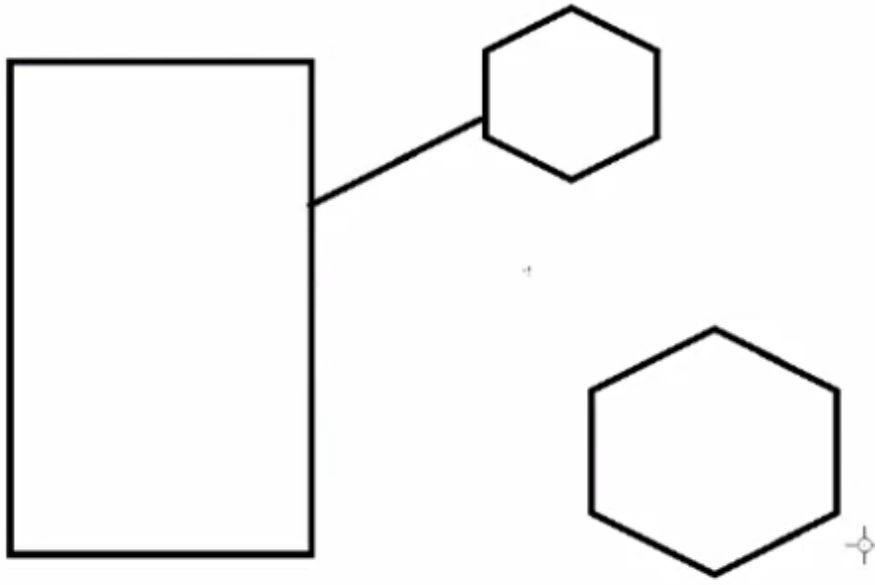


now while doing so you cannot directly do this as it has huge amount of data, we cannot directly dump all this data at day 1, so what we are doing is, we are writing a particular program



ad this particular database fetching the data from this database as of today



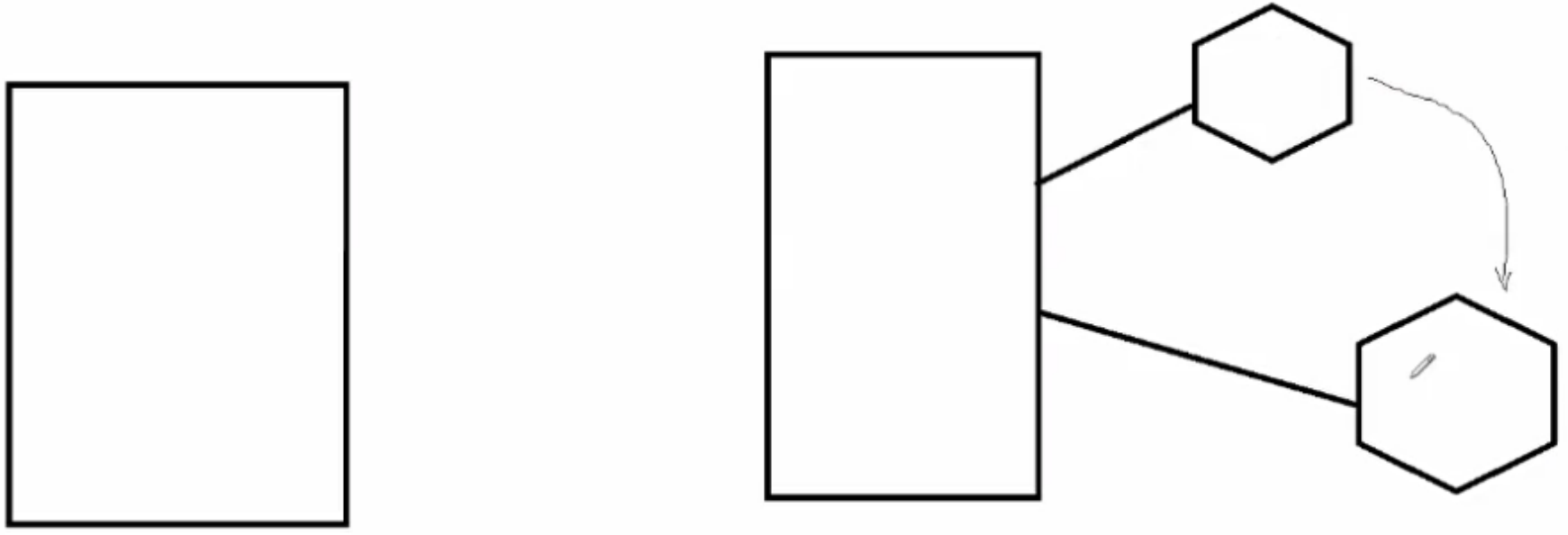


but client will say you know what deepak it is taking lot of time to fetch it is taking 2 min 6 min because access database will not support so many data

so looking that we thought that we'll migrate that to new database

so i want to make sure that what ever the screen we are getting here either from the access database or from the new database

When can it come to the new database? only for those data which is already been transfered here



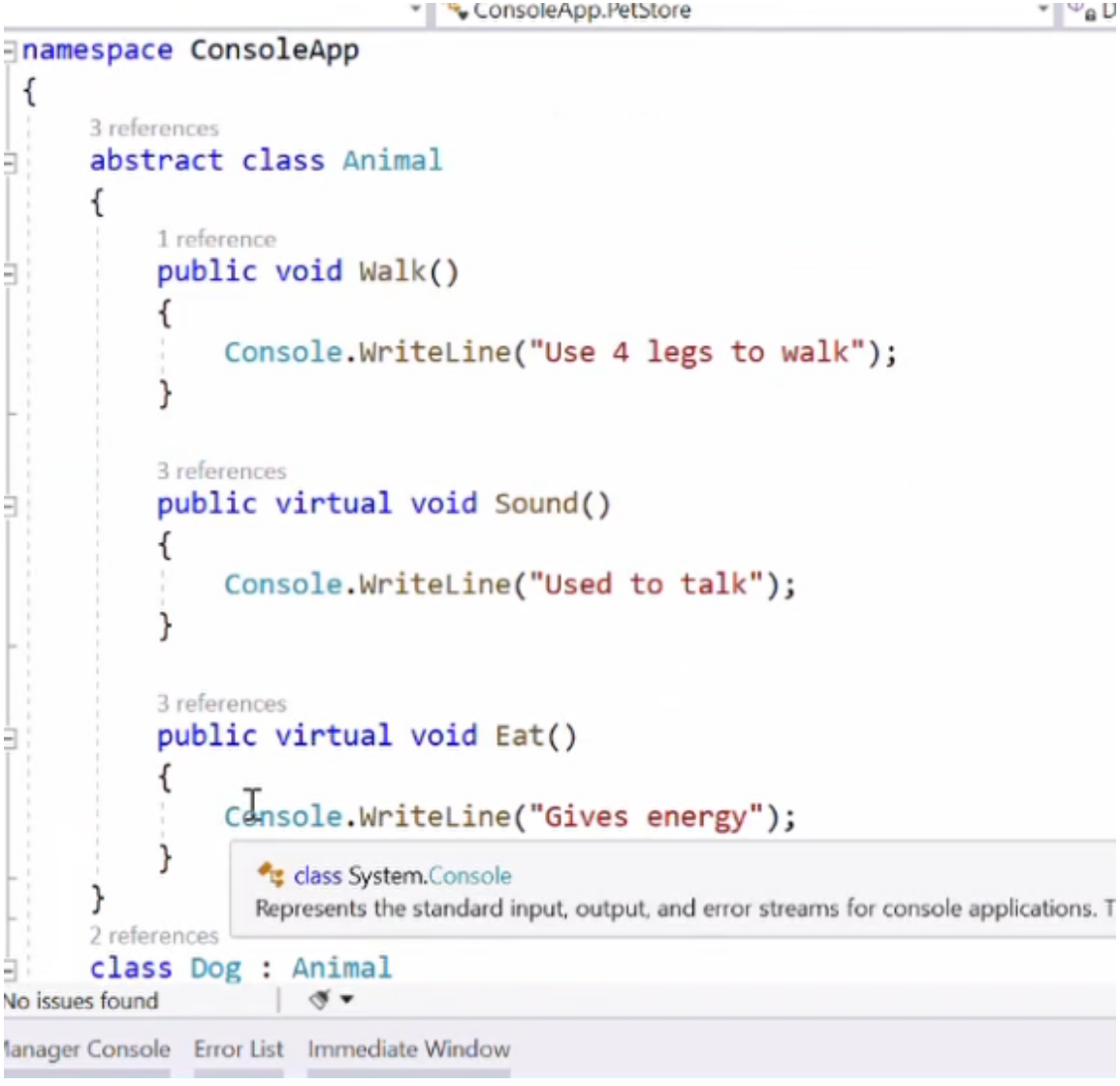
only for those data it should get the data from new database not from the old one, and it is happening at the fly , meaning my application is running and I'm doing all this things
Dynamically I want to take a call weather weather I should go to this one or to this one
How do you do this If I'm creating Instance of both separately So How do you switch, Dynamically you cannot do this right

So what we are doing is based on what we are passing, dog or cat, The Implementation gets changed So dynamically based on some condition I can get the data from access database or from mysql database
This kind of configuration based working, I'll write configuration for both , I will not touch the code, inside the code I will not say go connect to this database or to this database

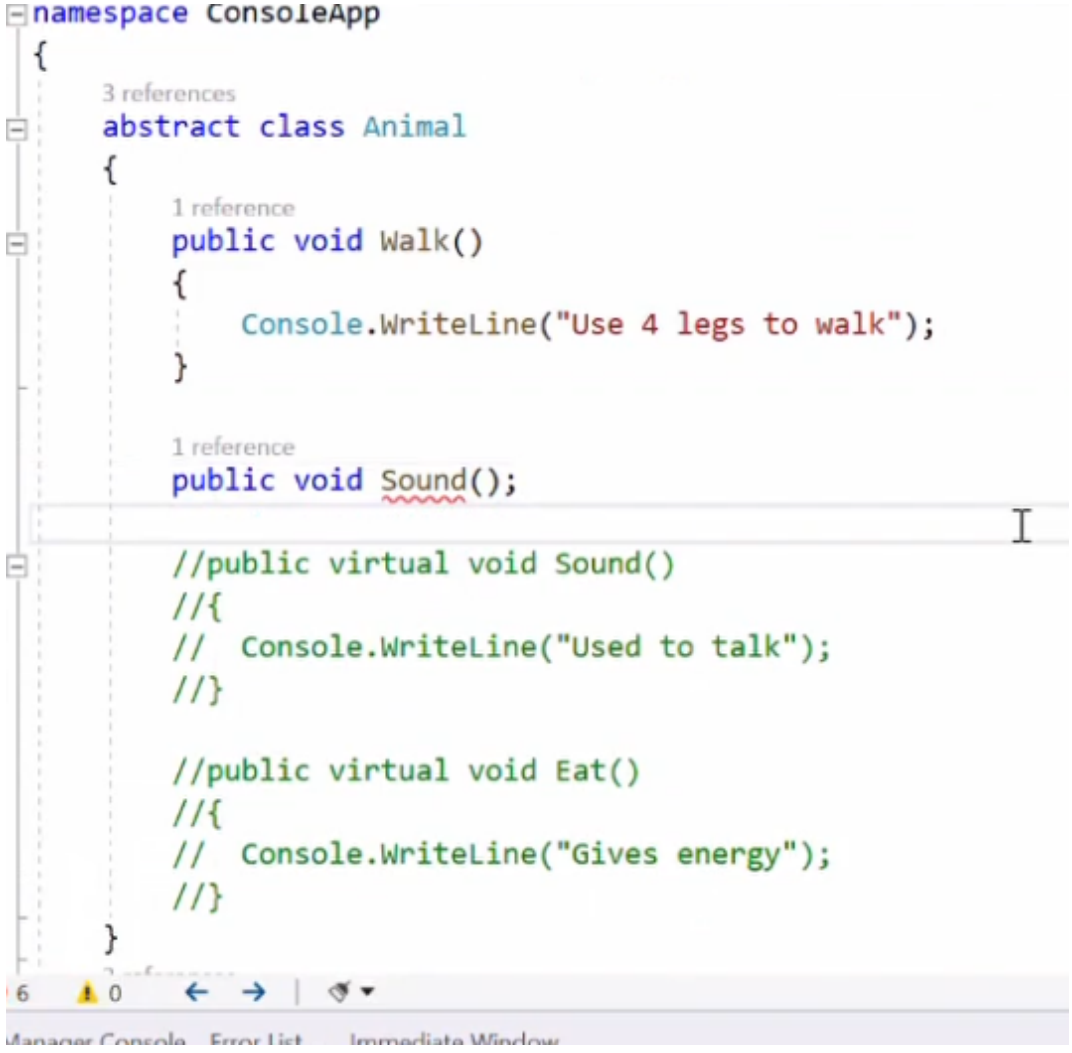
How to switch based on config
Like suppose I have a ADO Implementation and tomorrow I want to go with EF Implementation, So Implementations can be different but the methods what we are using will be common, So I will write a particular configuration to switch between both the implementation, that's where

your dynamic Overriding will come into pitcher, Dynamic Polymorphism will come into pitcher, Late Binding will come into pitcher

Now you can see that we are creating virtual methods here and now one are using it

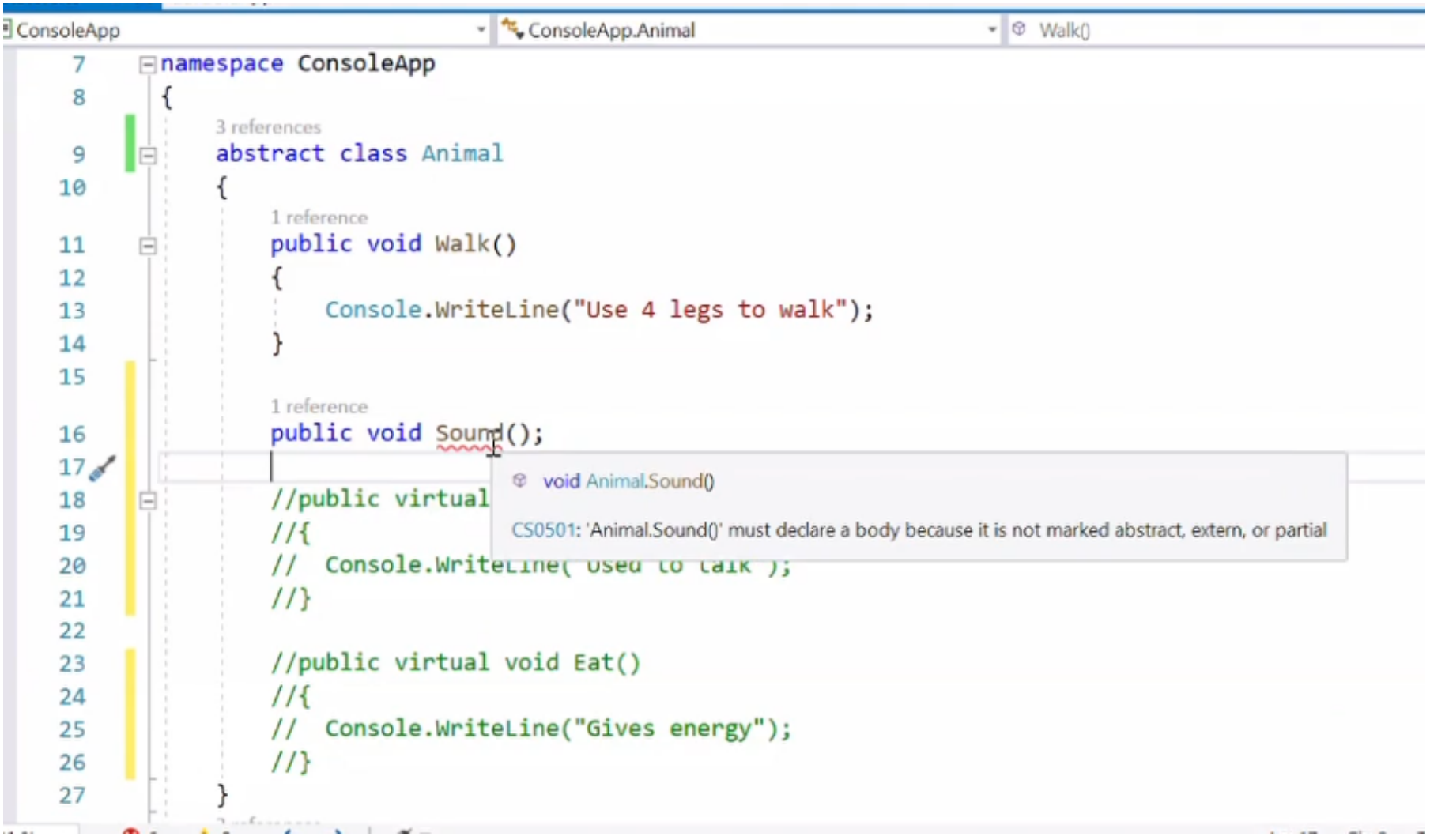


So i'll Eliminate this methods and I'll make this methods as Abstract

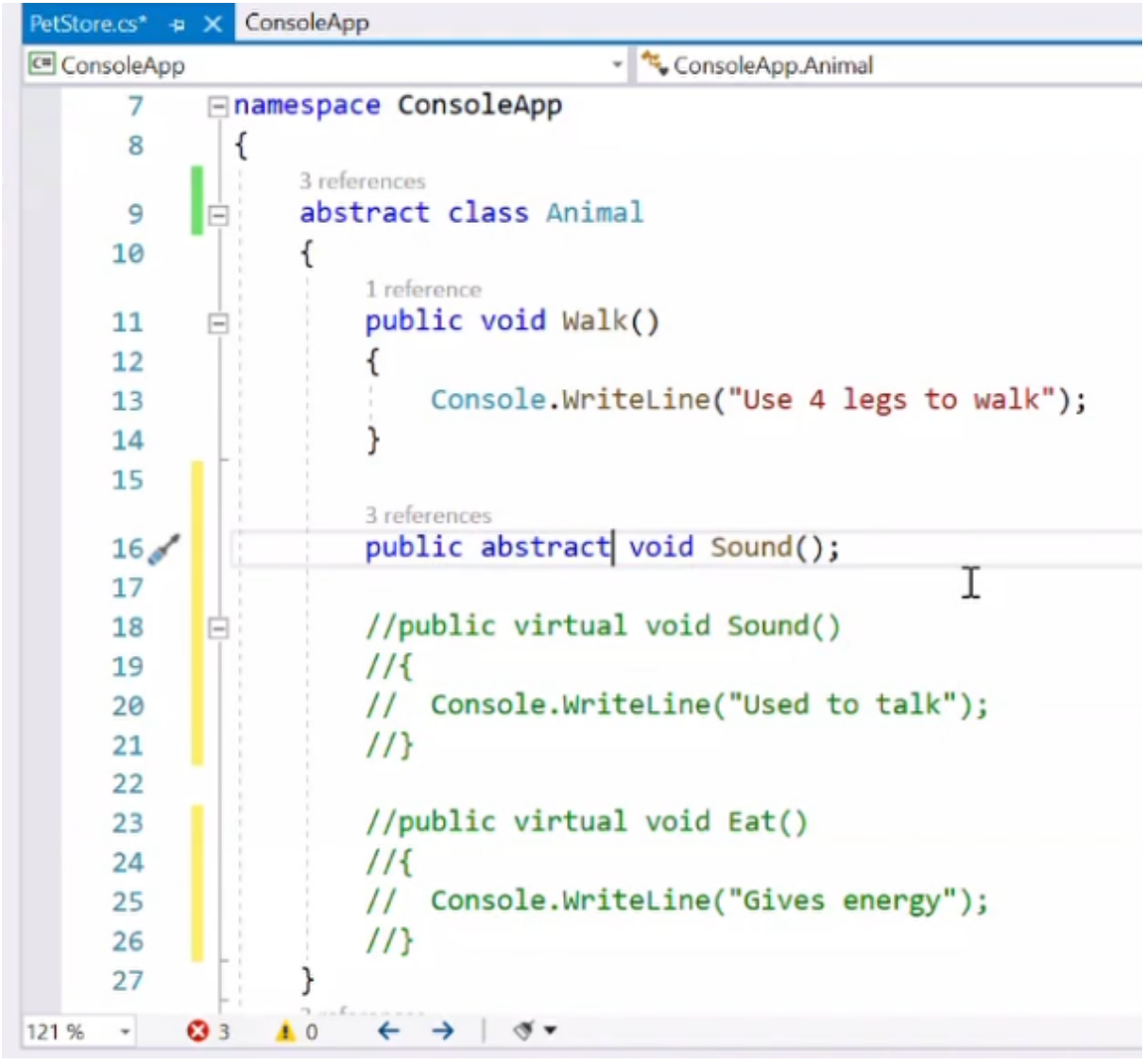


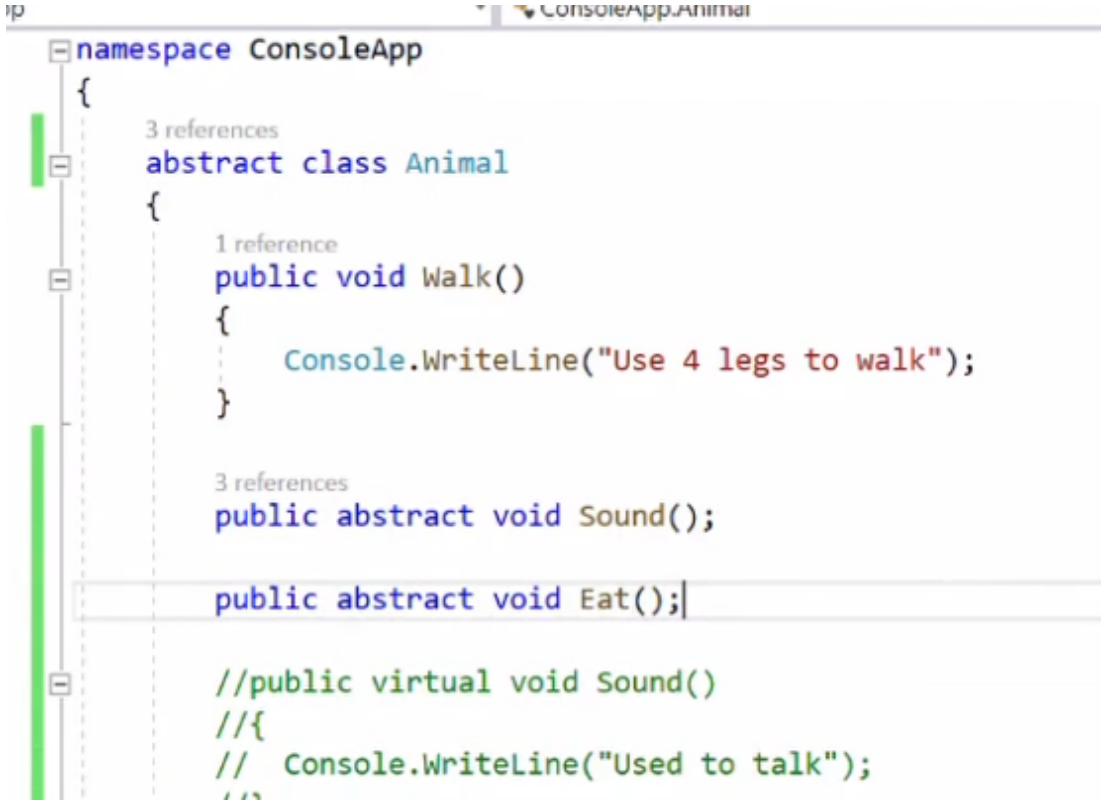
```
namespace ConsoleApp
{
    3 references
    abstract class Animal
    {
        1 reference
        public void Walk()
        {
            Console.WriteLine("Use 4 legs to walk");
        }

        1 reference
        public void Sound();

        //public virtual void Sound()
        //{
        //  Console.WriteLine("Used to talk");
        //}

        //public virtual void Eat()
        //{
        //  Console.WriteLine("Gives energy");
        //}
    }
```

AS soon as I put semicolon it means i don't want to write any sorts of dummy code here because it will also take some memory, Implementation someone will do



But you get error saying that it always expects that to marked as Abstract , External or partial
because in .net every method should have body and if it is not having the body then
you should always mark method as Abstract , External or partial
You cannot just put the semicolon at the end, it is not possible
So I'll make it as abstract


PetStore.cs*
ConsoleApp
ConsoleApp
ConsoleApp.Animal
namespace ConsoleApp
{
3 references
abstract class Animal
{
1 reference
public void Walk()
{
Console.WriteLine("Use 4 legs to walk");
}
3 references
public abstract void Sound();
//public virtual void Sound()
//{
// Console.WriteLine("Used to talk");
//}
//public virtual void Eat()
//{
// Console.WriteLine("Gives energy");
//}
}
121 %
3
0



meaning i don't have Impl for this, Impl was done by this peoples below the classes

```
35          }
36
            3 references
37          public override void Eat()
38          {
39              Console.WriteLine("Veg and Non-veg");
40          }
41      }
42
        2 references
43      class Cat : Animal
44      {
            3 references
45          public override void Sound()
46          {
47              Console.WriteLine("Meo..Meo...");
48          }
49
            3 references
50          public override void Eat()
51          {
52              Console.WriteLine("Nog-veg");
53          }
54      }
        0 references
```

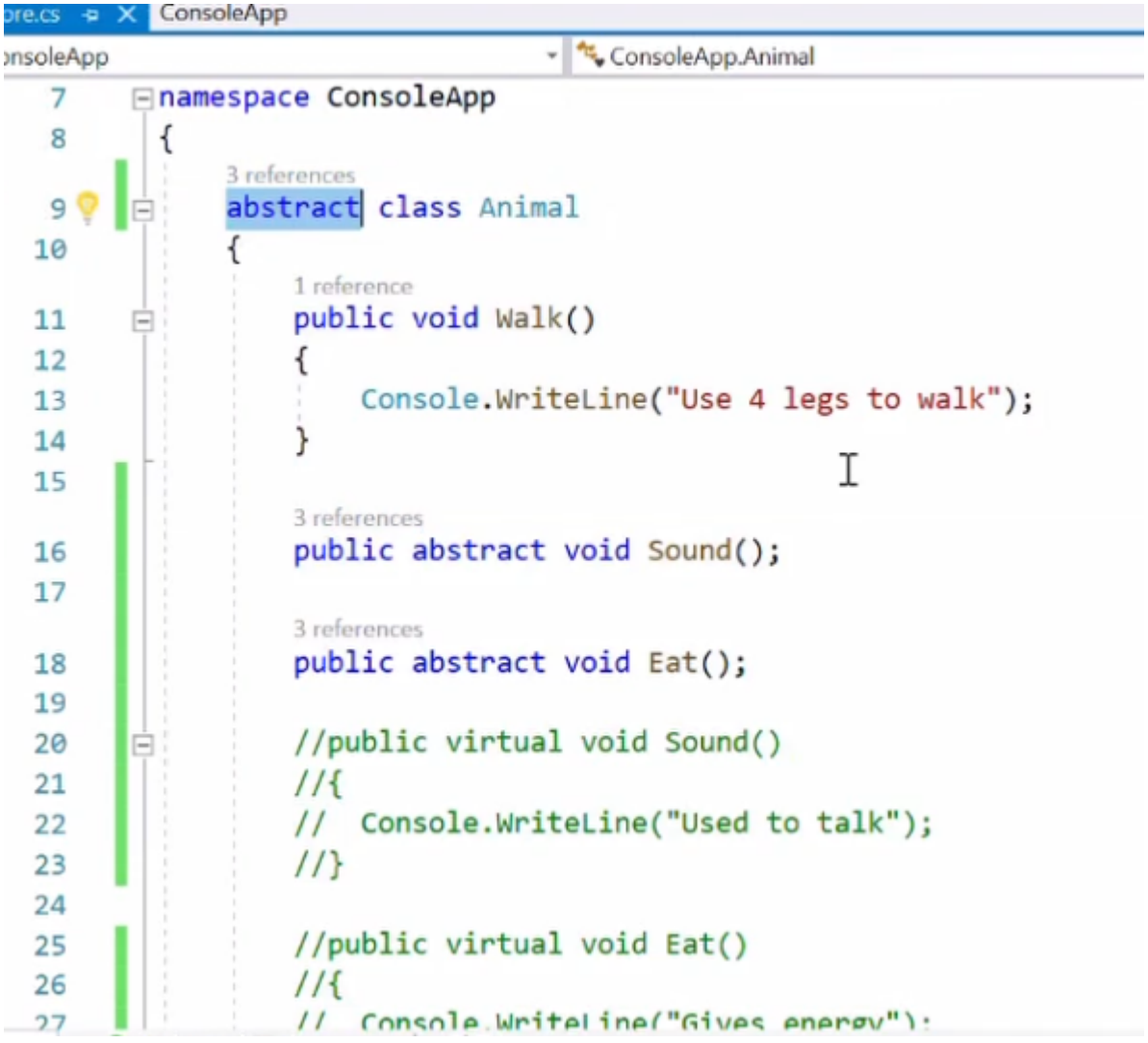
I have a parent, So parent is having some kind of abstract behaviour

for ex parent is taking some king of loan and now he is not paying it, the loan what ever he has taken

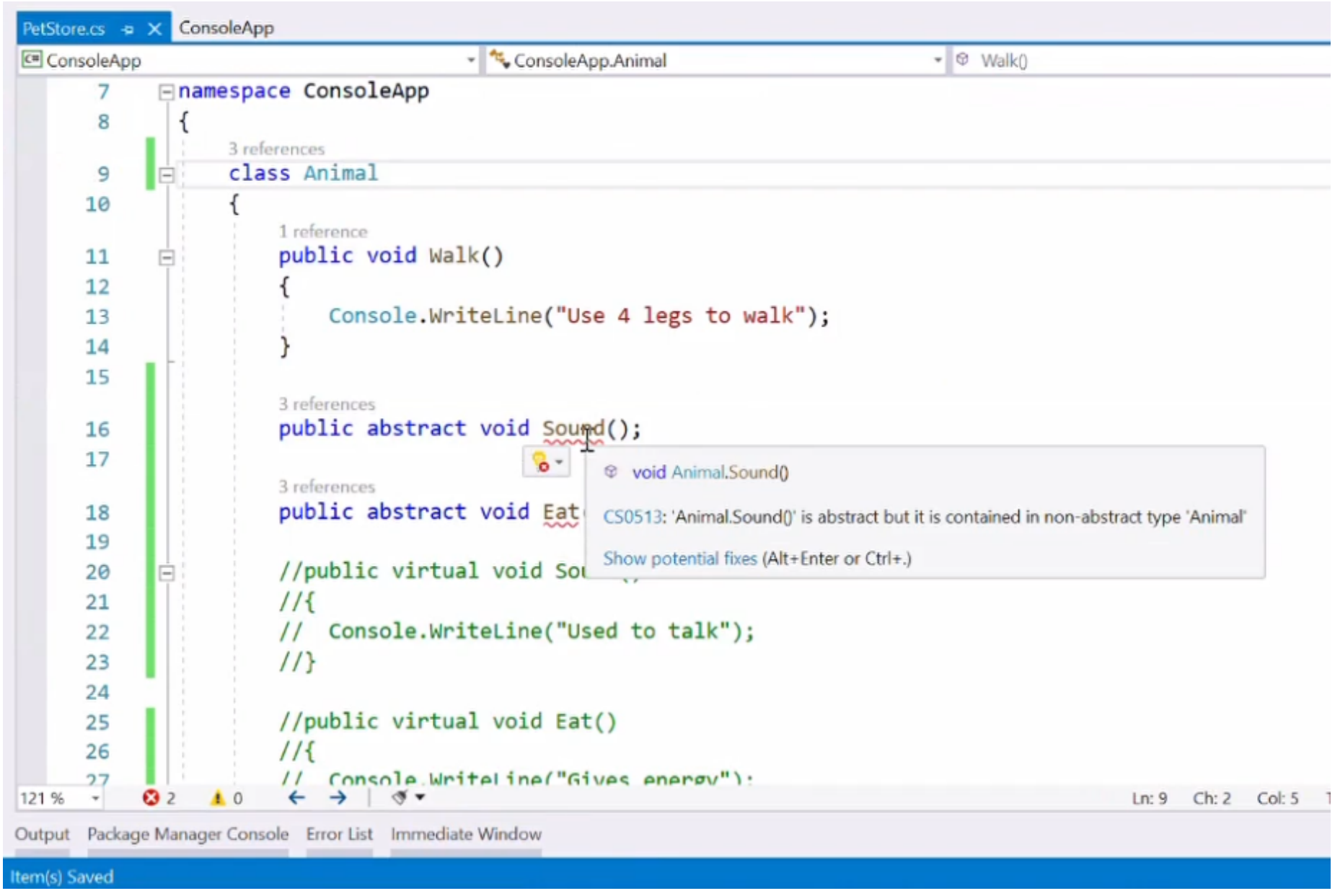
```
3 references
abstract class Animal
{
    1 reference
    public void Walk()
    {
        Console.WriteLine("Use 4 legs to walk");
    }

    3 references
    public abstract void Sound();

    3 references
    public abstract void Eat();
```

So the loan will be paid by his son or daughter , His son or daughter will override that particular method

```
    //public virtual void Eat()
    //{
    //  Console.WriteLine("Gives energy");
    //}
}
2 references
class Dog : Animal
{
    0 references
    public override void Sound()
    {
        Console.WriteLine("Bow Bow...");
    }

    3 references
    public override void Eat()
    {
        Console.WriteLine("Veg and Non-veg");
    }
}
```

2. 

I'll remove this abstract



you can see we got err, it is saying that it is abstract but it is present in non abstract class called animal

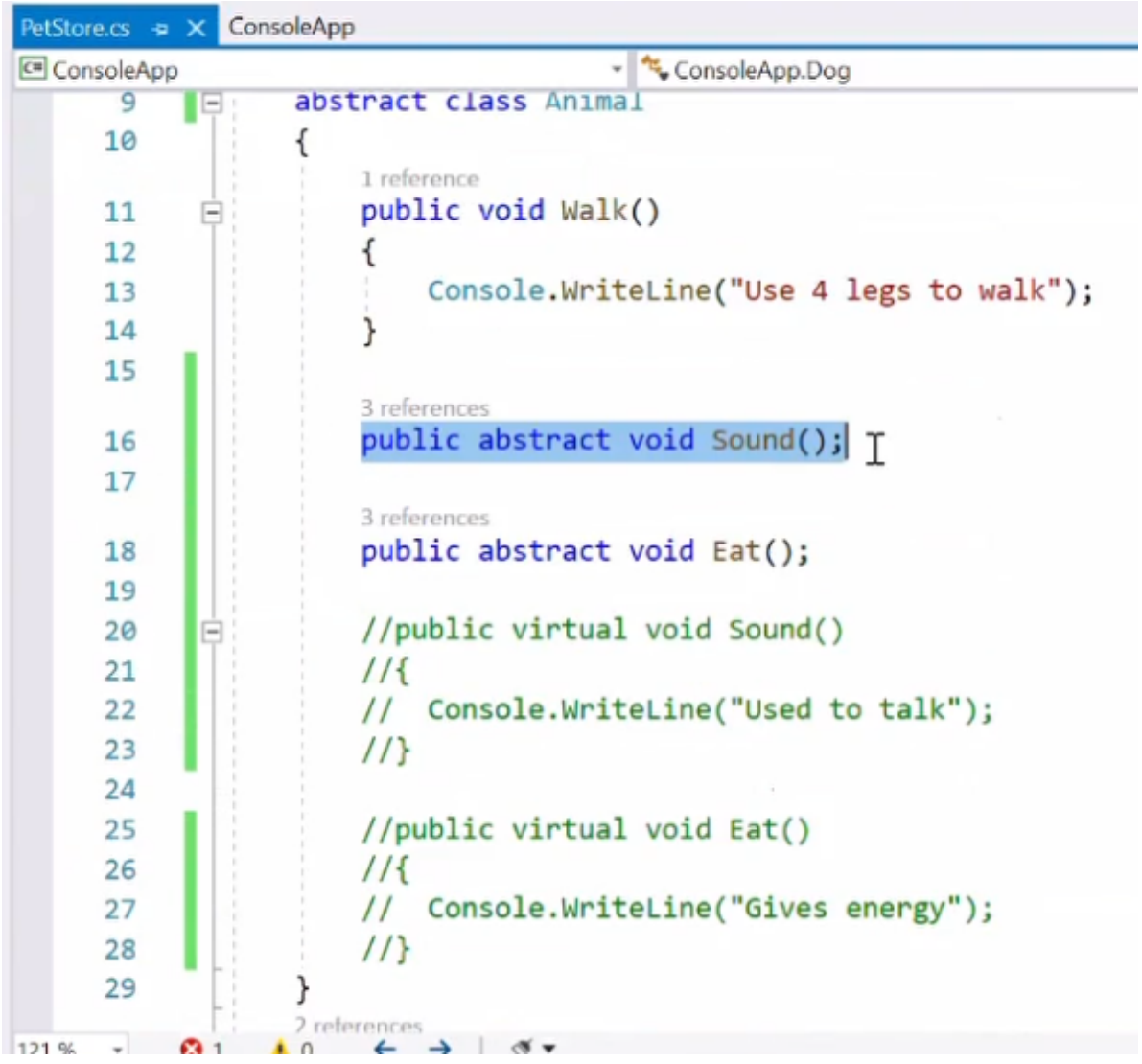
2. **Even if you have 1 method as Abstract the class should and must be abstract**

When ever you have abstract members, Class should and must be defined as abstract

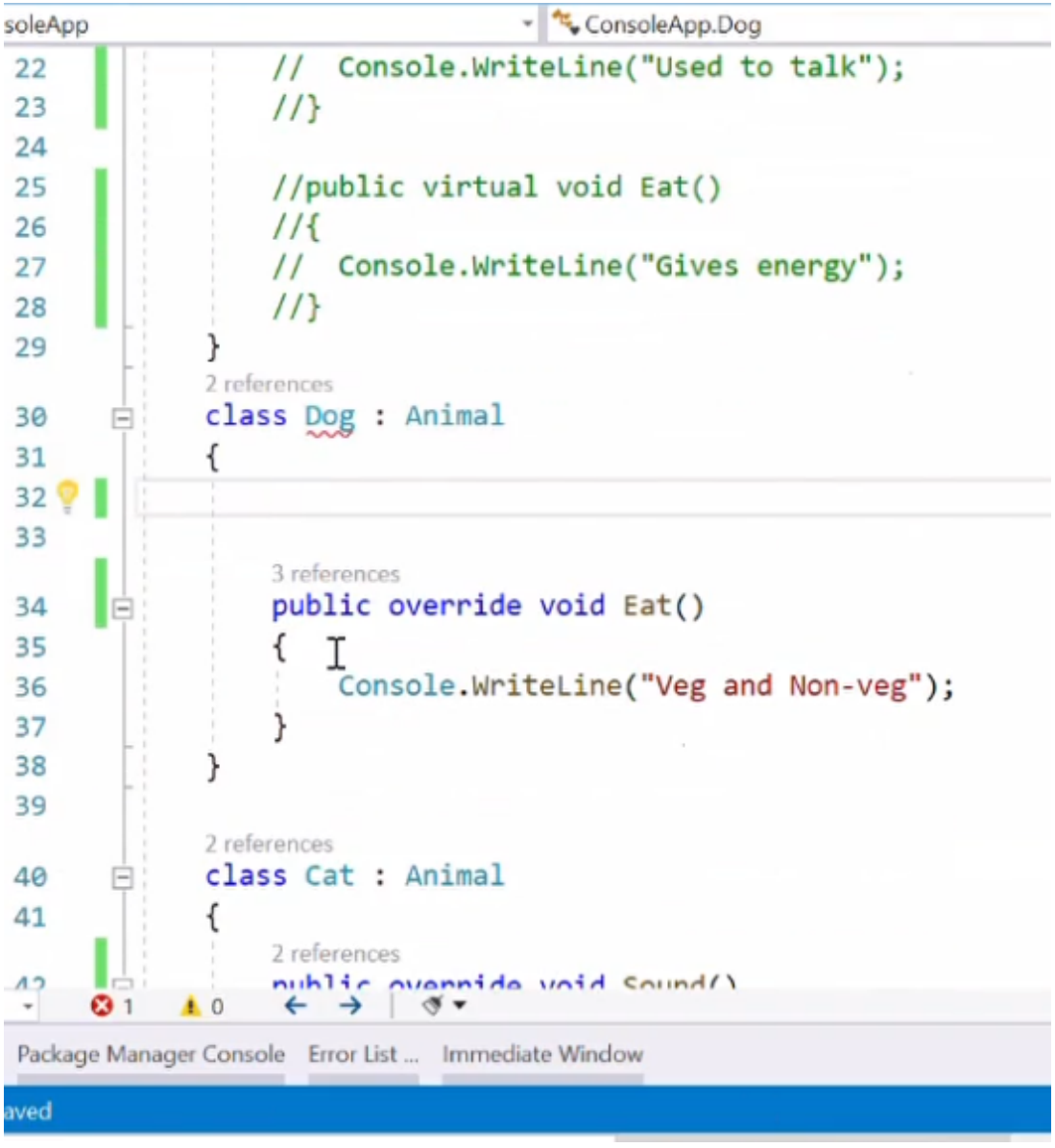
What about reverse, If the class is abstract can it have a abstract method or it can have even non abstract method as well

==> It may or may not have

3. I have a abstract method here



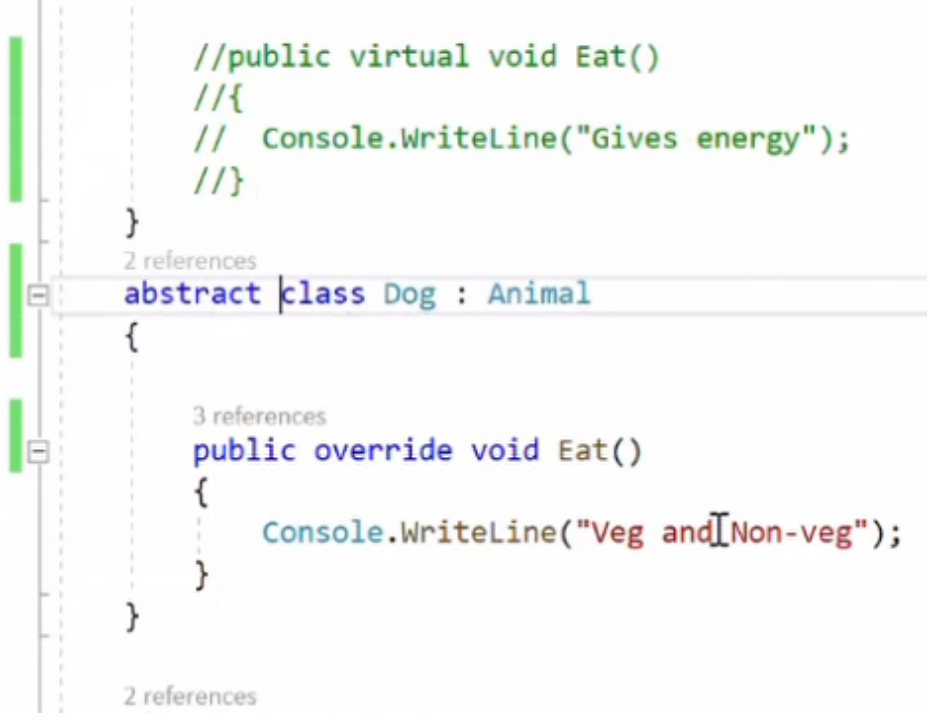
and i'll not Implement it here



It simply means your father has taken a loan and neither he is paying the loan nor you
So bank wale err marenge☺

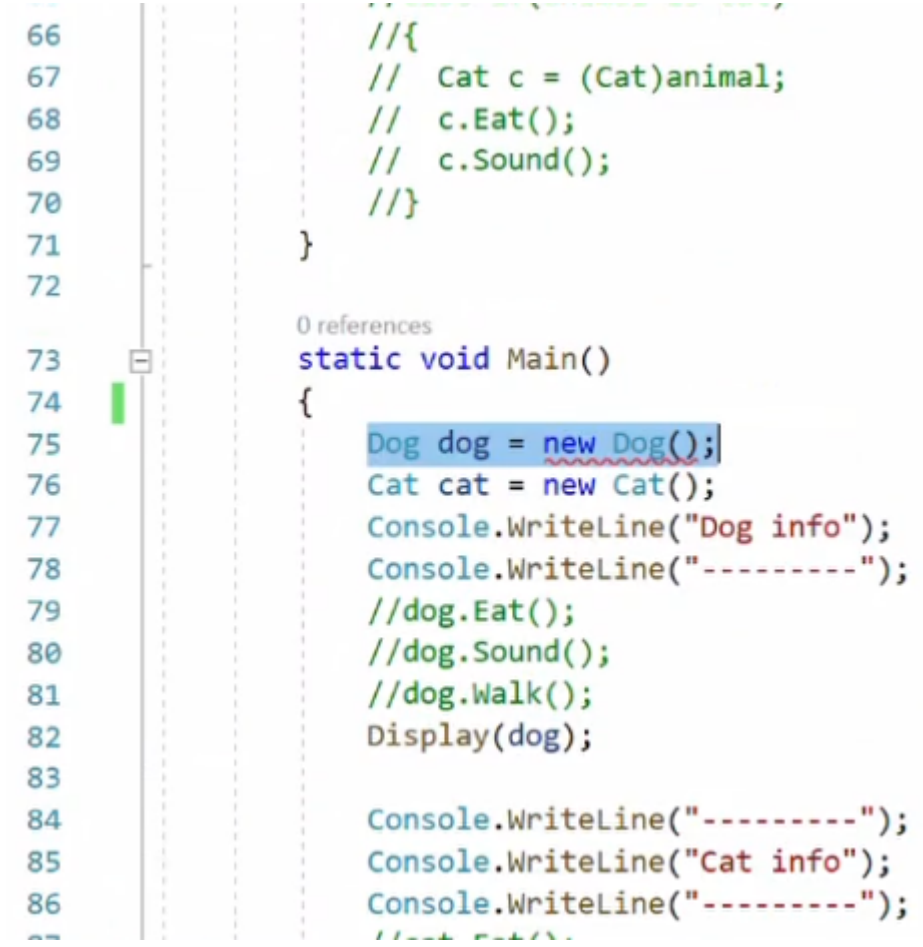
So this will give me err saying

So I'll say boss i'll not implement it, so remove this you have to make this class also as abstract



meaning I'm saying that dog class is a incomplete class
now meaning animal and dog both are incomplete class

that means now I cannot create instance of dog class



```
66          //{
67          //  Cat c = (Cat)animal;
68          //  c.Eat();
69          //  c.Sound();
70          //}
71      }
72
        0 references
73      static void Main()
74      {
75          Dog dog = new Dog();
76          Cat cat = new Cat();
77          Console.WriteLine("Dog info");
78          Console.WriteLine("---------");
79          //dog.Eat();
80          //dog.Sound();
81          //dog.Walk();
82          Display(dog);
83
84          Console.WriteLine("---------");
85          Console.WriteLine("Cat info");
86          Console.WriteLine("---------");
```

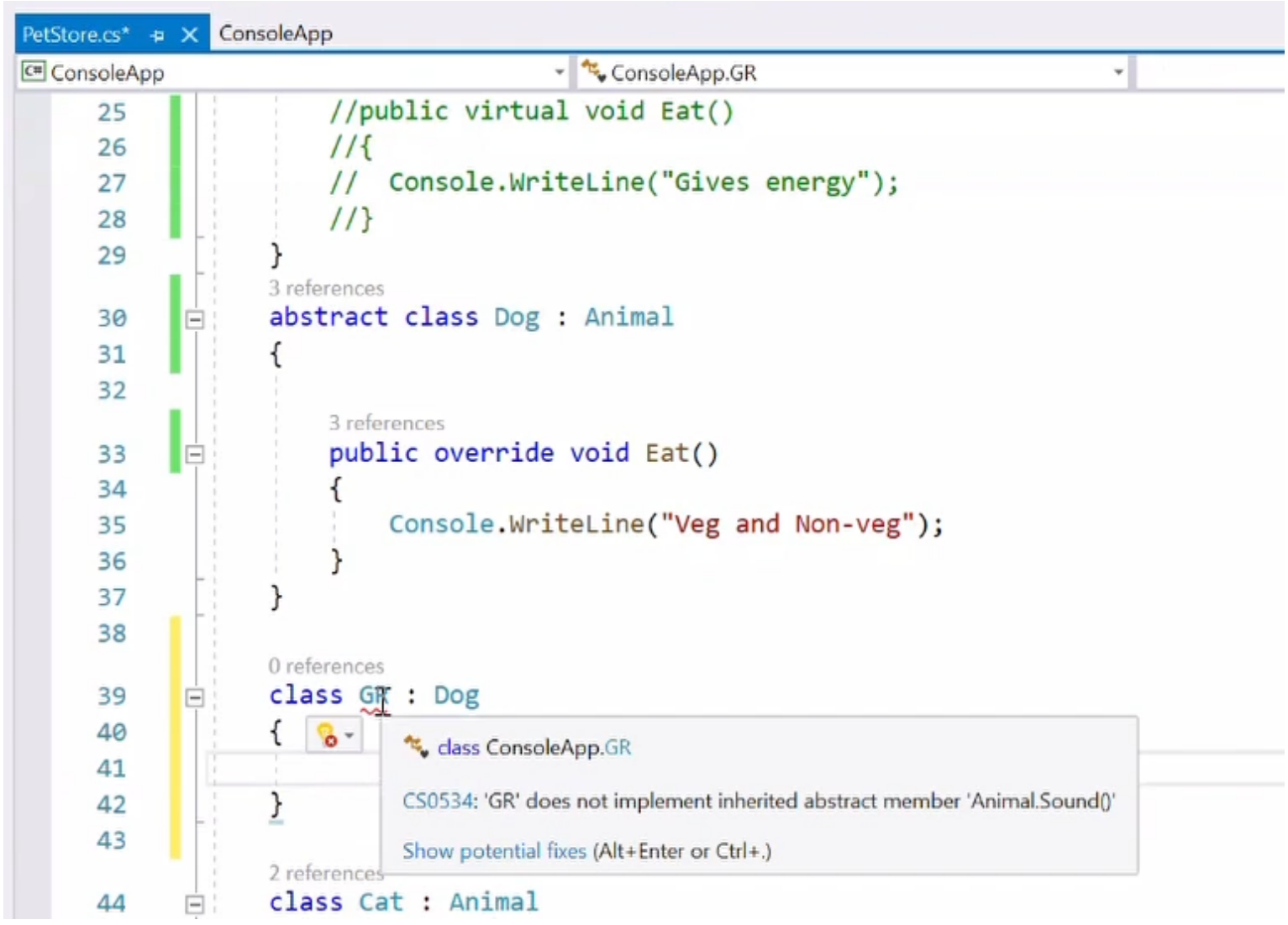
Do we have a dog in the real world

So when i say dog so which pitcher is coming in your mind ?

=> There are so many dogs are there right, street dog, bull dog, labra .....

So dog is not there in the world, When you say dog so dog is generalized

So I'll say Golden Retriever (GR) is a dog

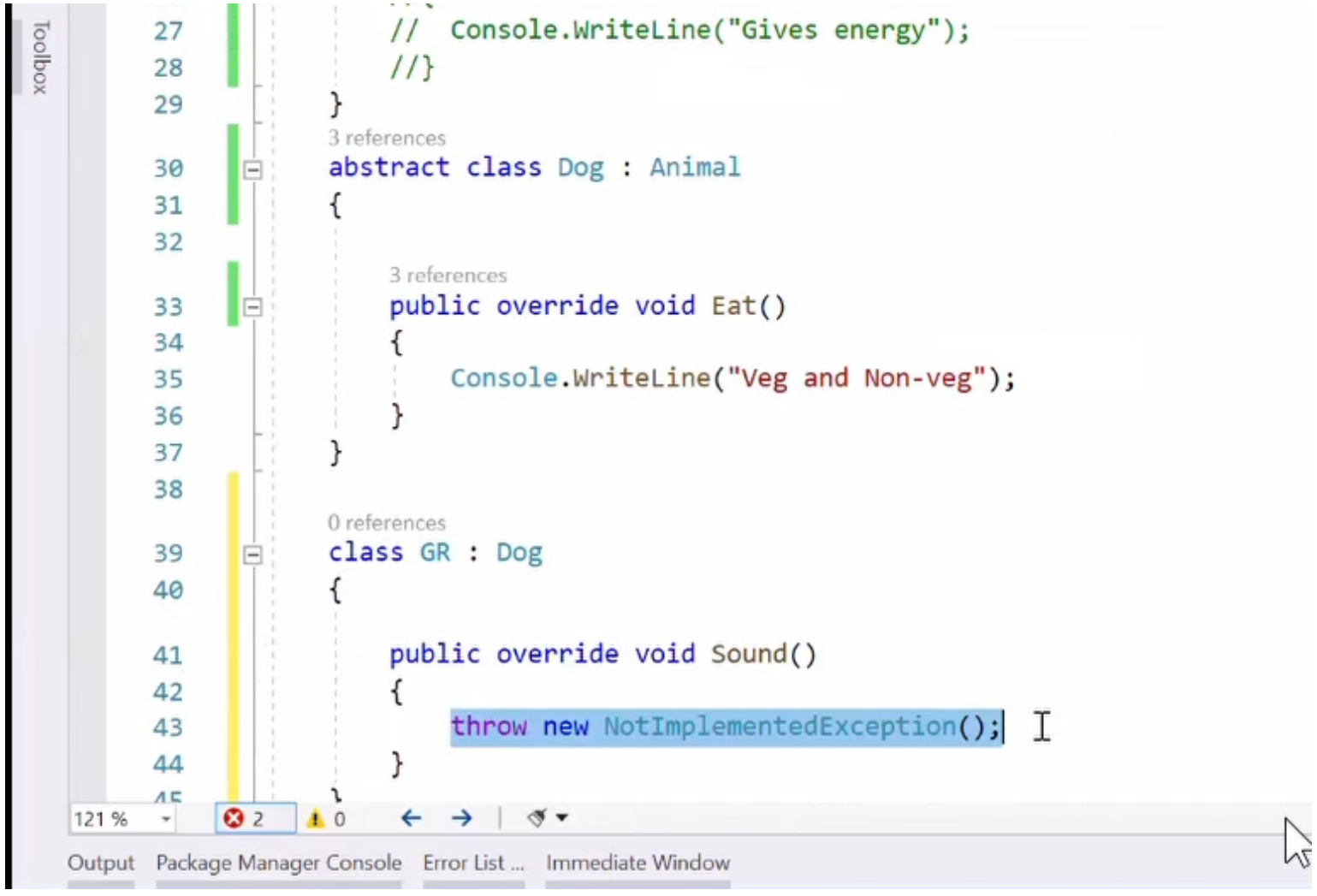


Now look at this I got a err msg here, saying boss you should inherit

As soon as a new baby takes a birth, so new baby get a msg on his mobile that you have to pay this particular amount

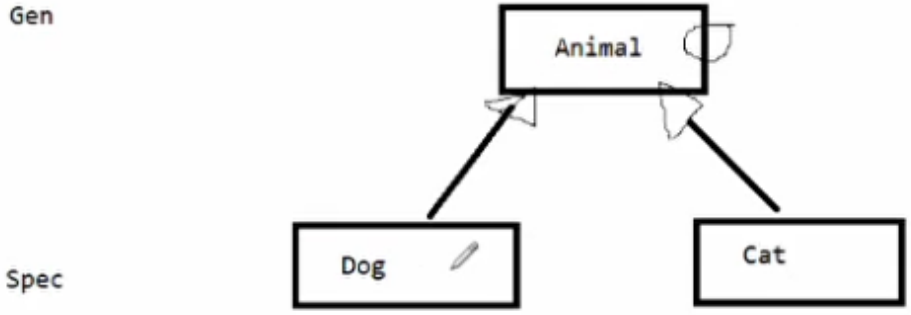
So what i do id I press Ctrl + . enter

So that particular method is present here, You just write a impl here



So somewhere down the line you should handle this abstraction otherwise all the intermediate methods you have to keep as abstract

Summary of Inheritance: the commonly written code can always be push to a common class and that can be inherited
why we are doing this ---- Code Reuse ability
So this is Specialization to Generalization



While doing this if we inherit animal to dog and cat, so a copy of animal from the memory (it will take the complete animal class) and it is pasted here in cat and dog class

that's why when we say dog. so it will access the whole animal class info as well

But what if I wrote my own information(method) in Dog which is similar to parent i.e Animal, Then we saw the information of parent was getting printed

So how did we overcome this, I wan't my child information to get printed not the parent

So to overcome this we have to use kewords Virtual and Overriding

We mention Parent to Virtual

and Child to Overriding

as soon as we did this, the child based on the reference, the respective method gets called

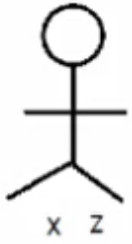
Doing this we saw that we want to access the child info from the animal reference

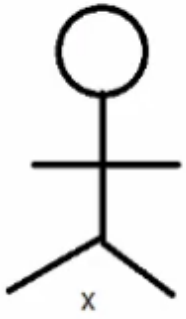
CAN I access child data from animal

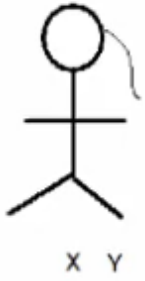
No, because copy of child is not present in the parent

So How can i achieve this, With the F I want to access the Z

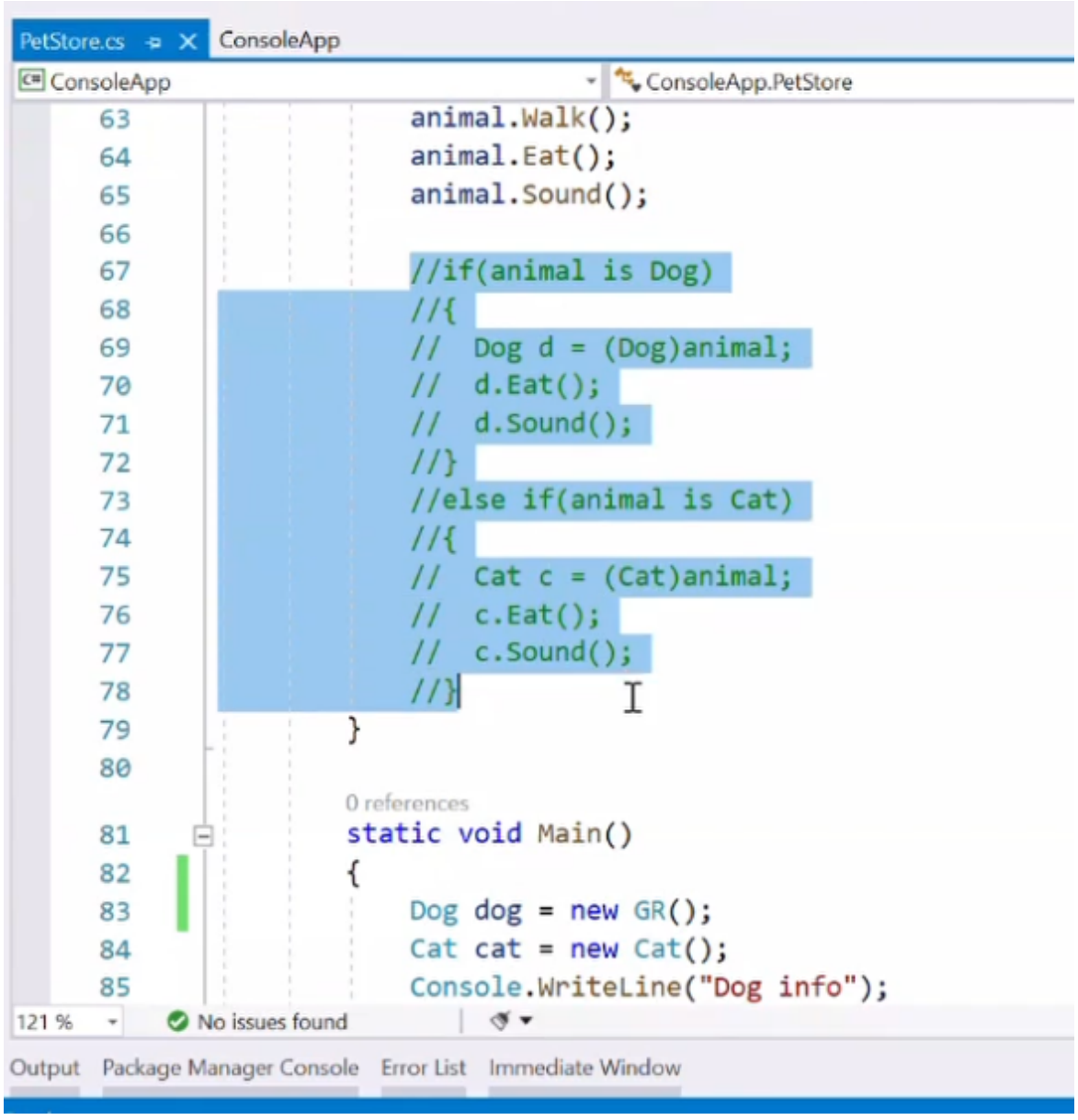
```
Son s = new Son();
s.x;
s.z;

Father f = new Son();
f.x;
f.z;
```







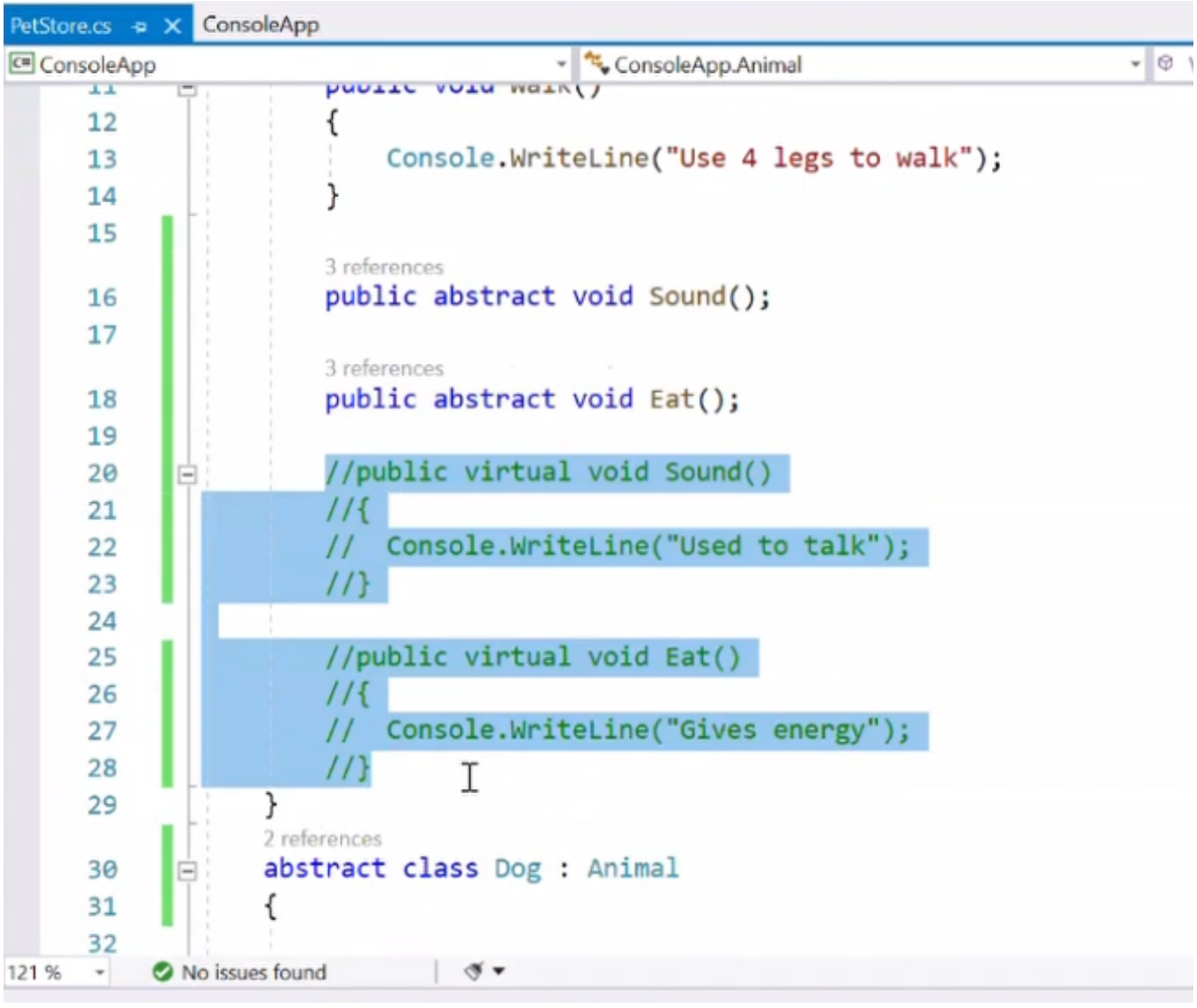
I.e means I want to do **Down casting**



so we are down casting it and after down casting, we are trying to access it .

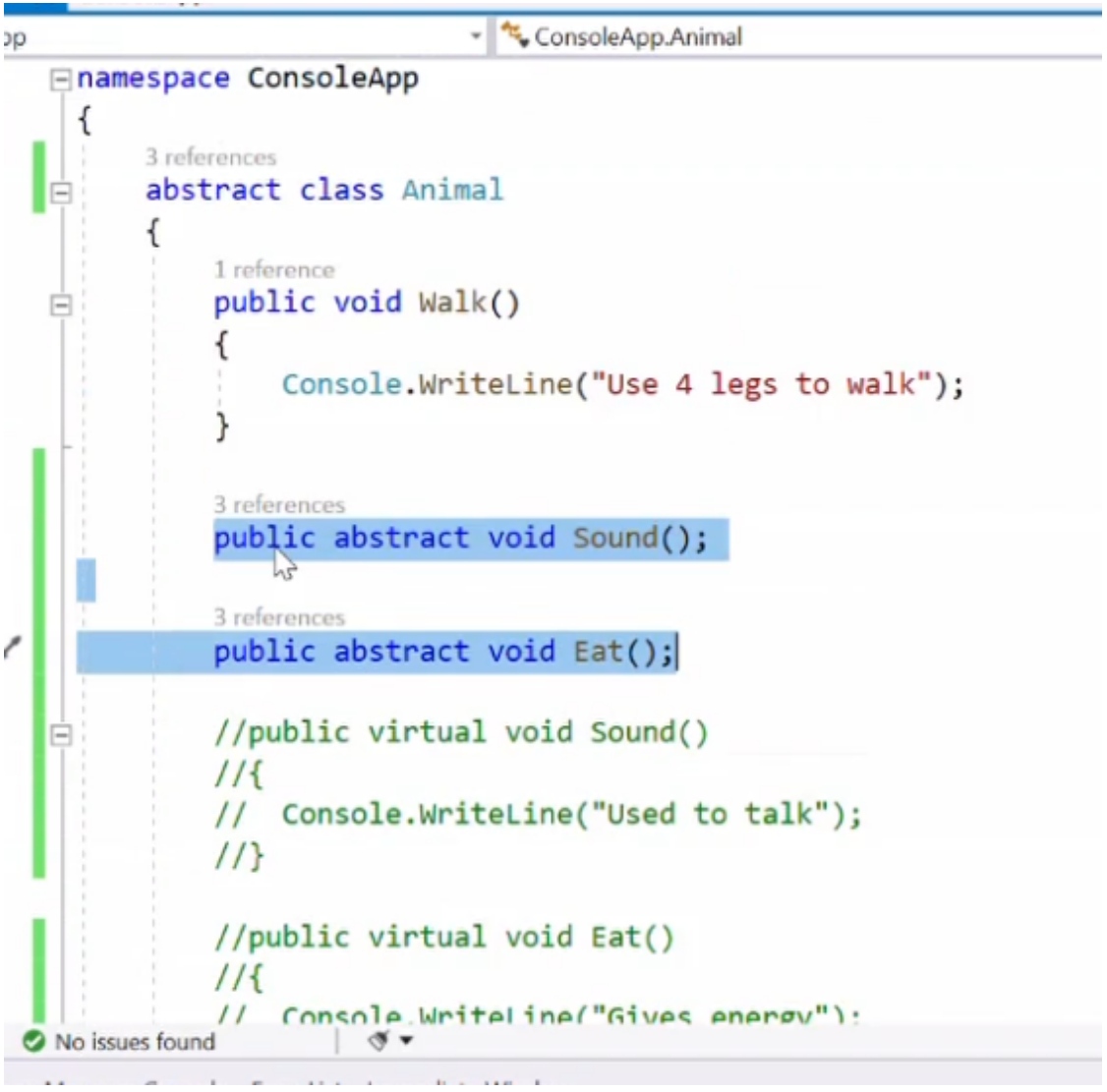
So we understood that it is a costly process, to access it you have to write again and again if else if else soo many times with 2 class its ok but what if i keep adding multiple animal class, so this becomes very huge

So to overcome this we introduce 2 dummy method in the parent



then we could directly access it
to access that we say virtual and override

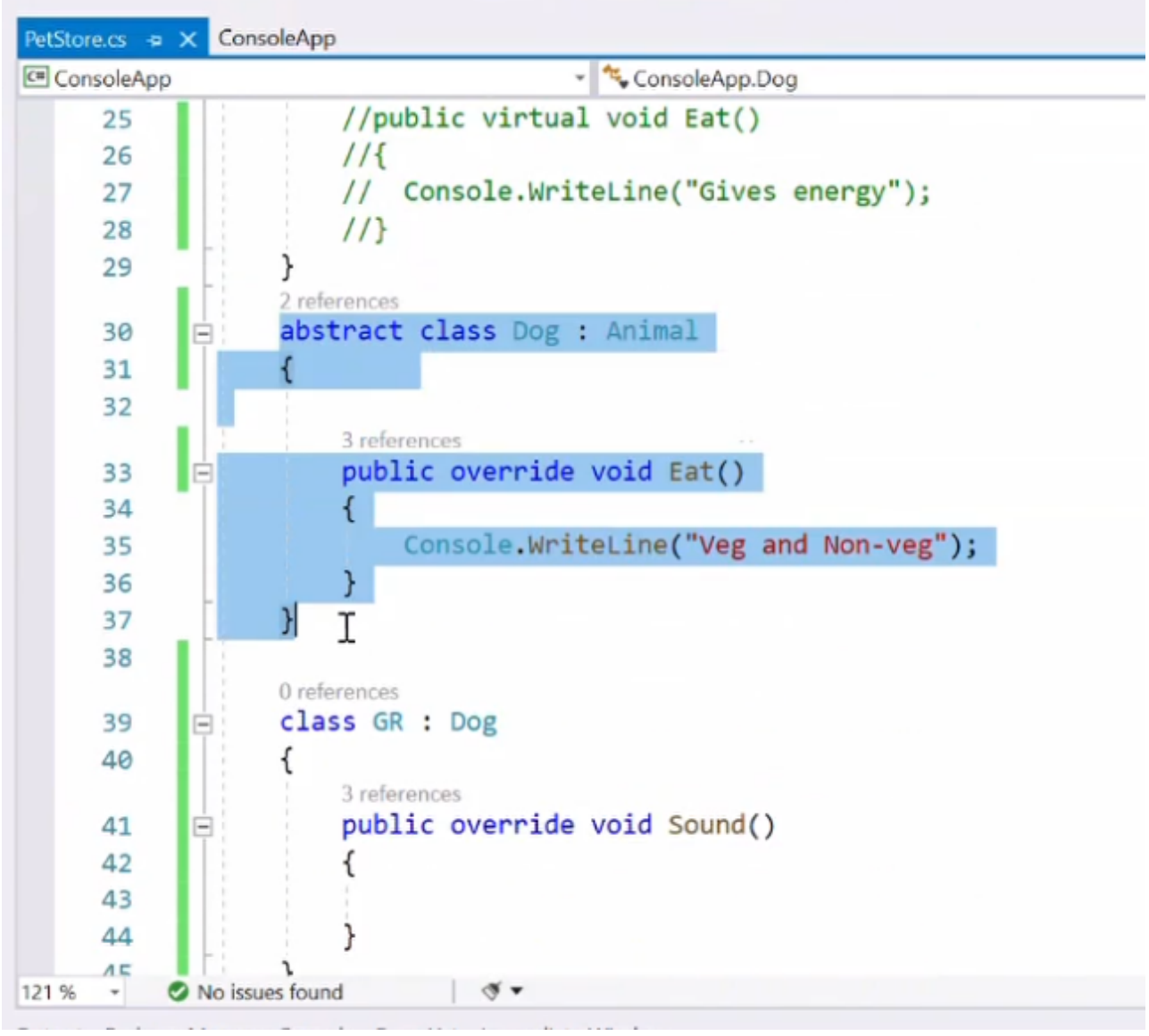
but we understood that Implementation of this will also be costly, it will also take memory,
So that's a reason we created it as a ABSTRACT method



so when ever you create a abstract method then you have to create class also as abstract

when ever you do this, you cannot create a object/Instance of abstract class

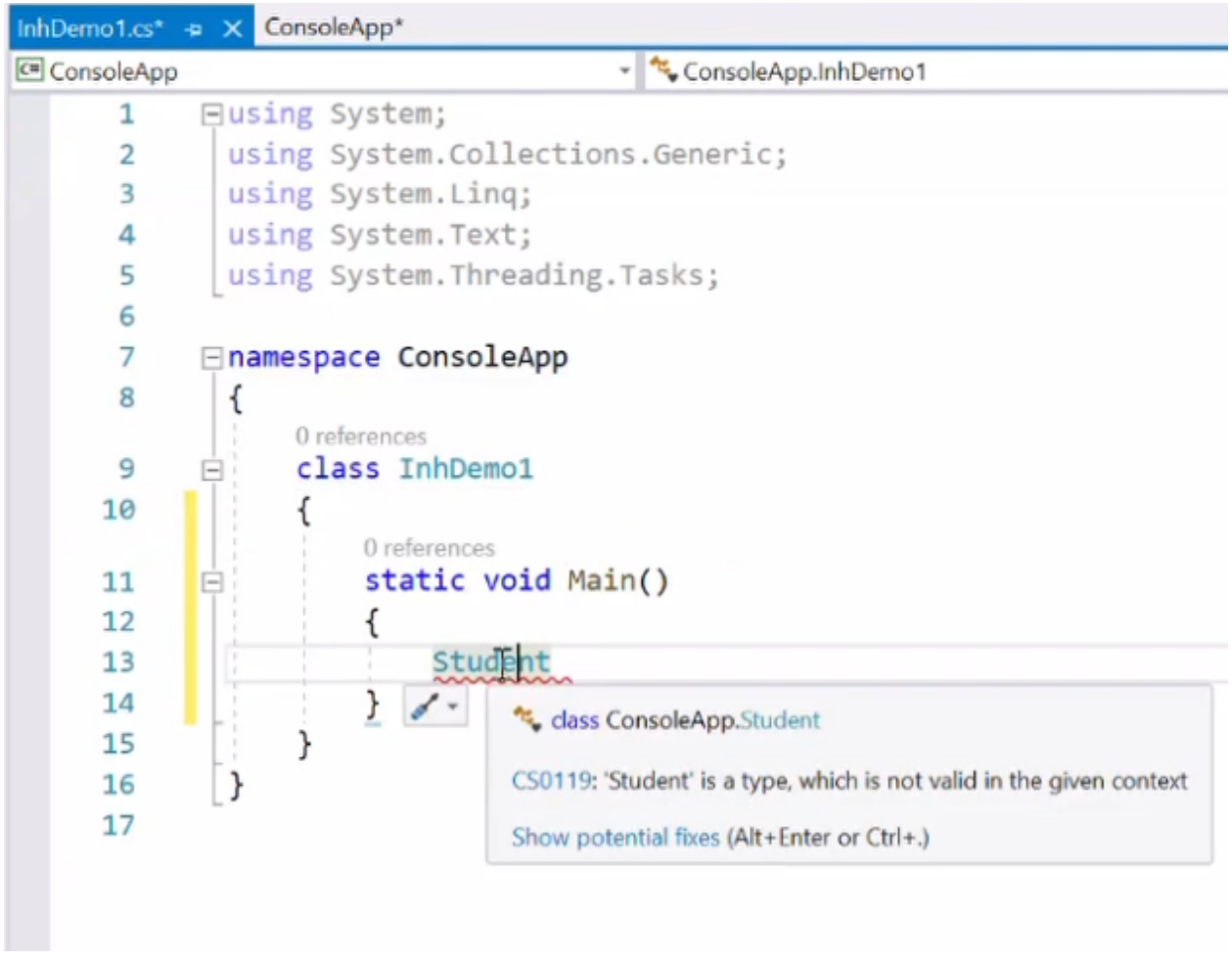
you cannot say
animal a = new animal
that is not possible
and abstract method you have should and must be implemented in its child class



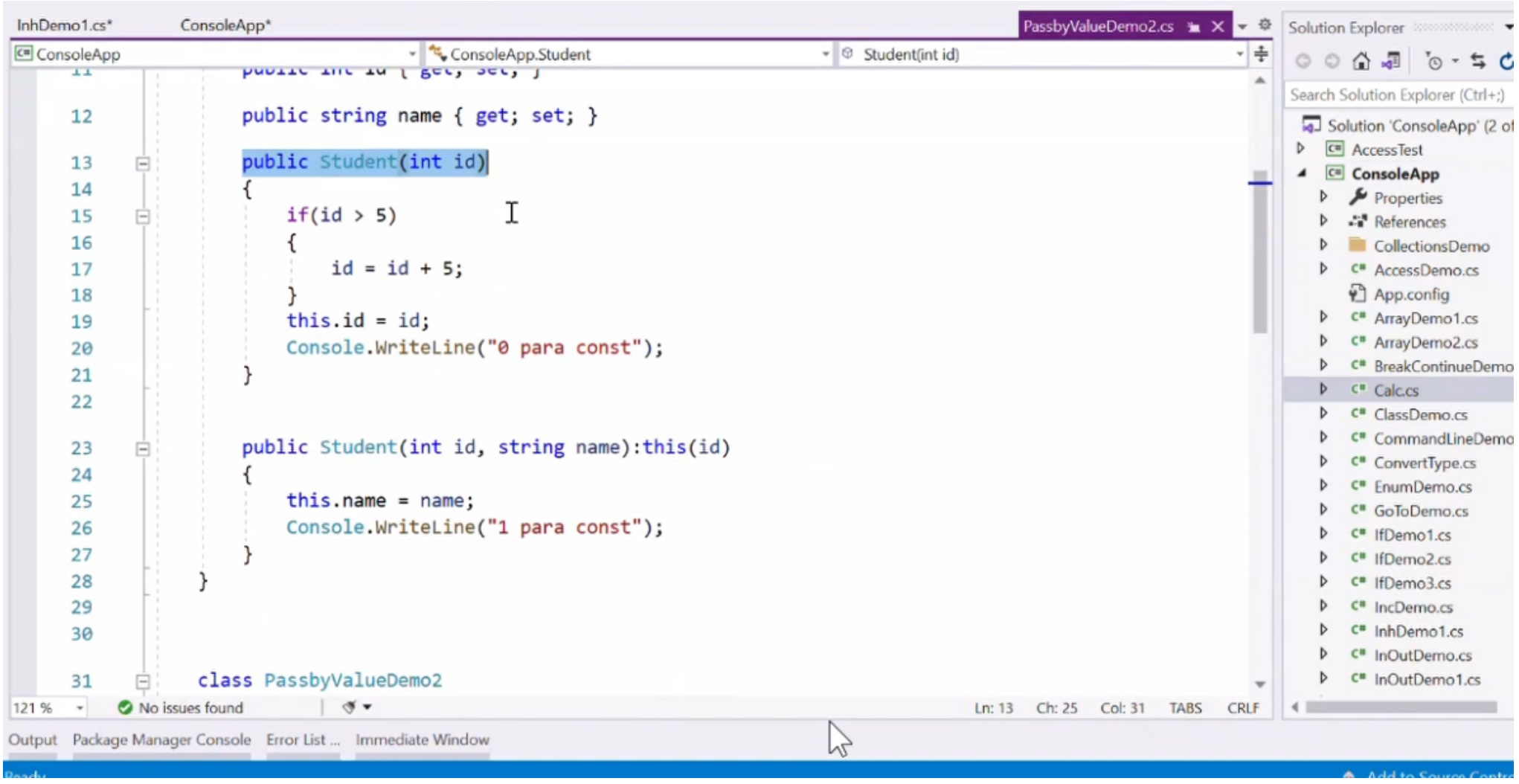
otherwise you should make even the child class as abstract

In a project Naming is very important

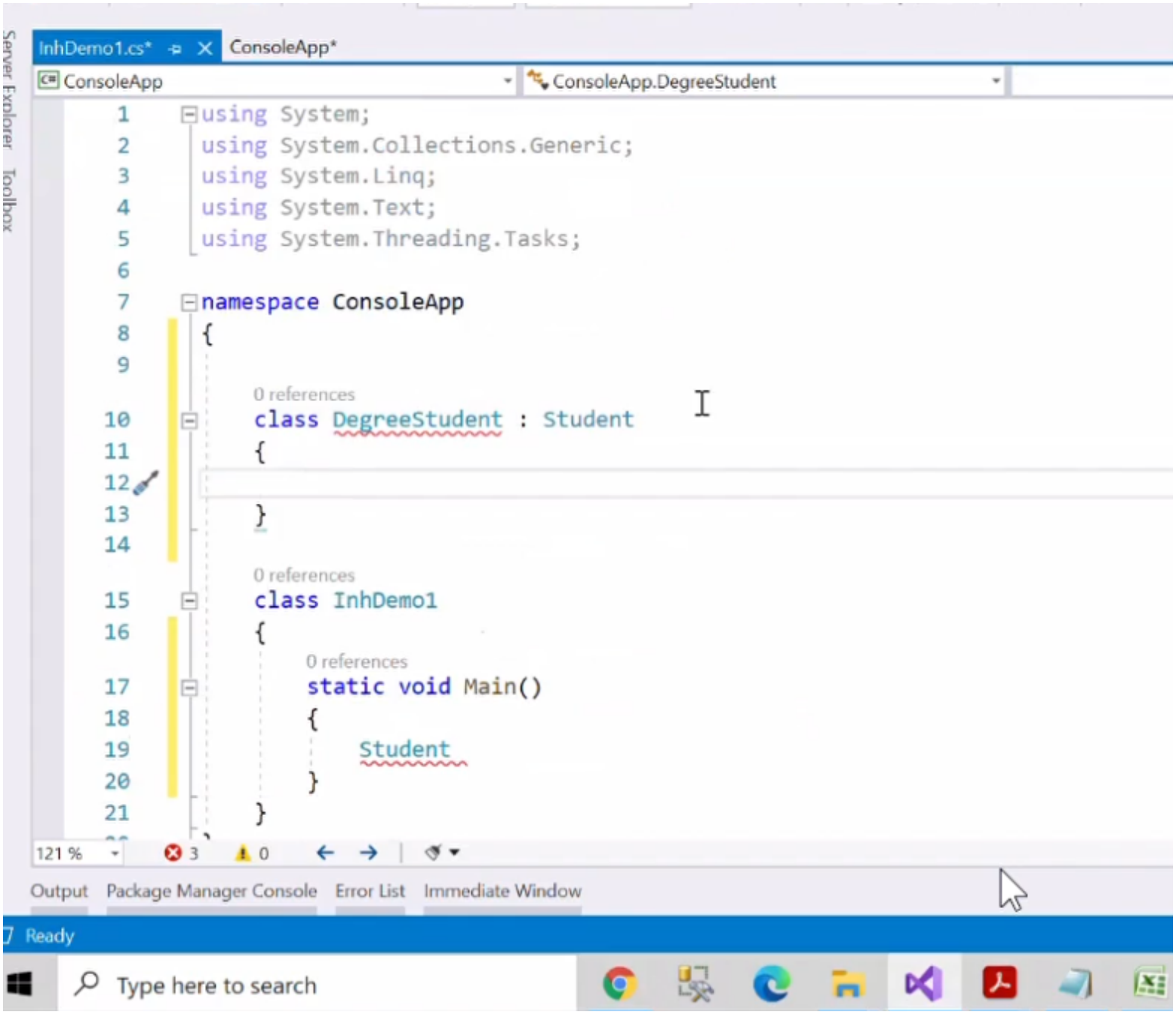
student if you look into, we have already created a constructor in this



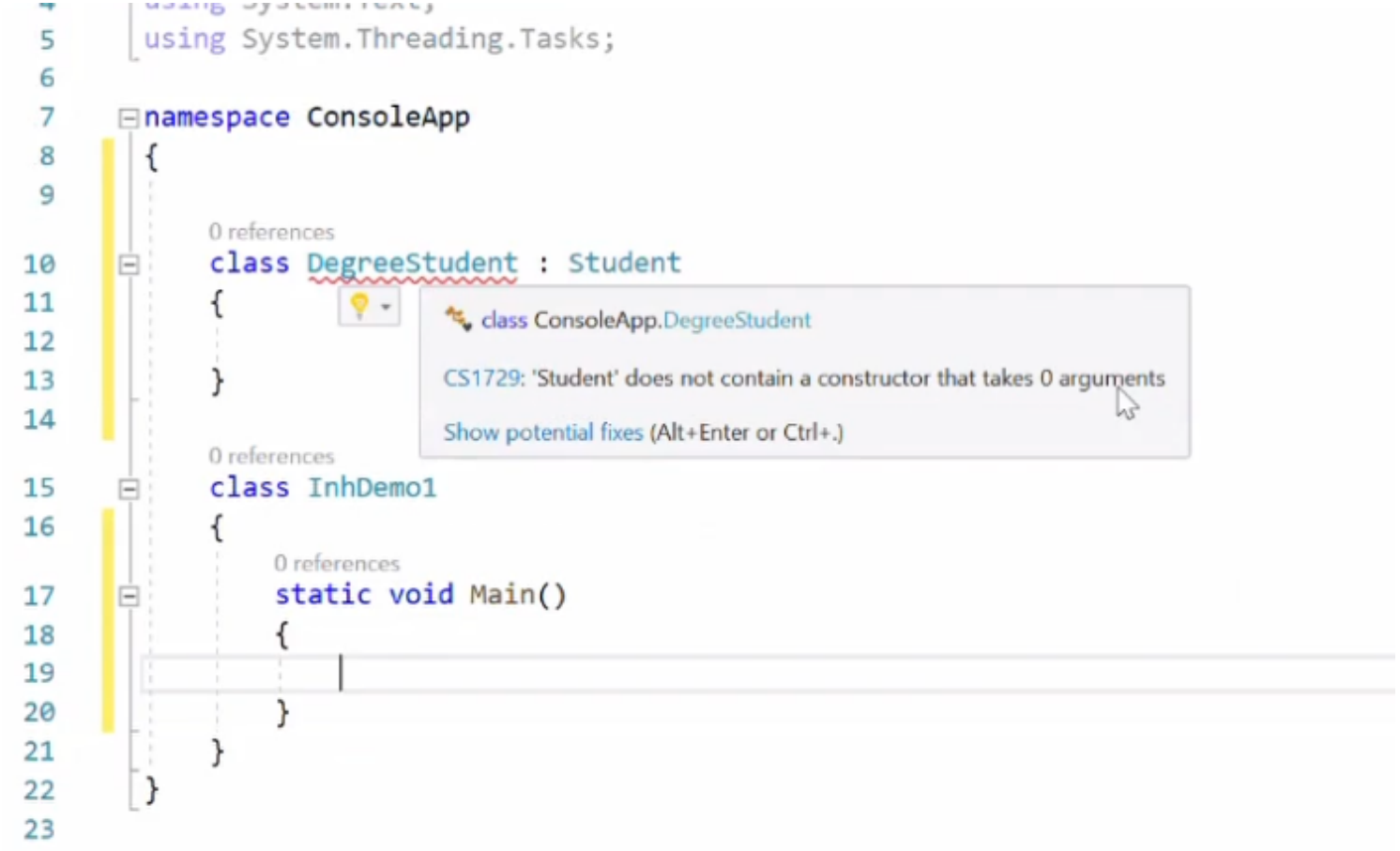
look at this we already have a constructor



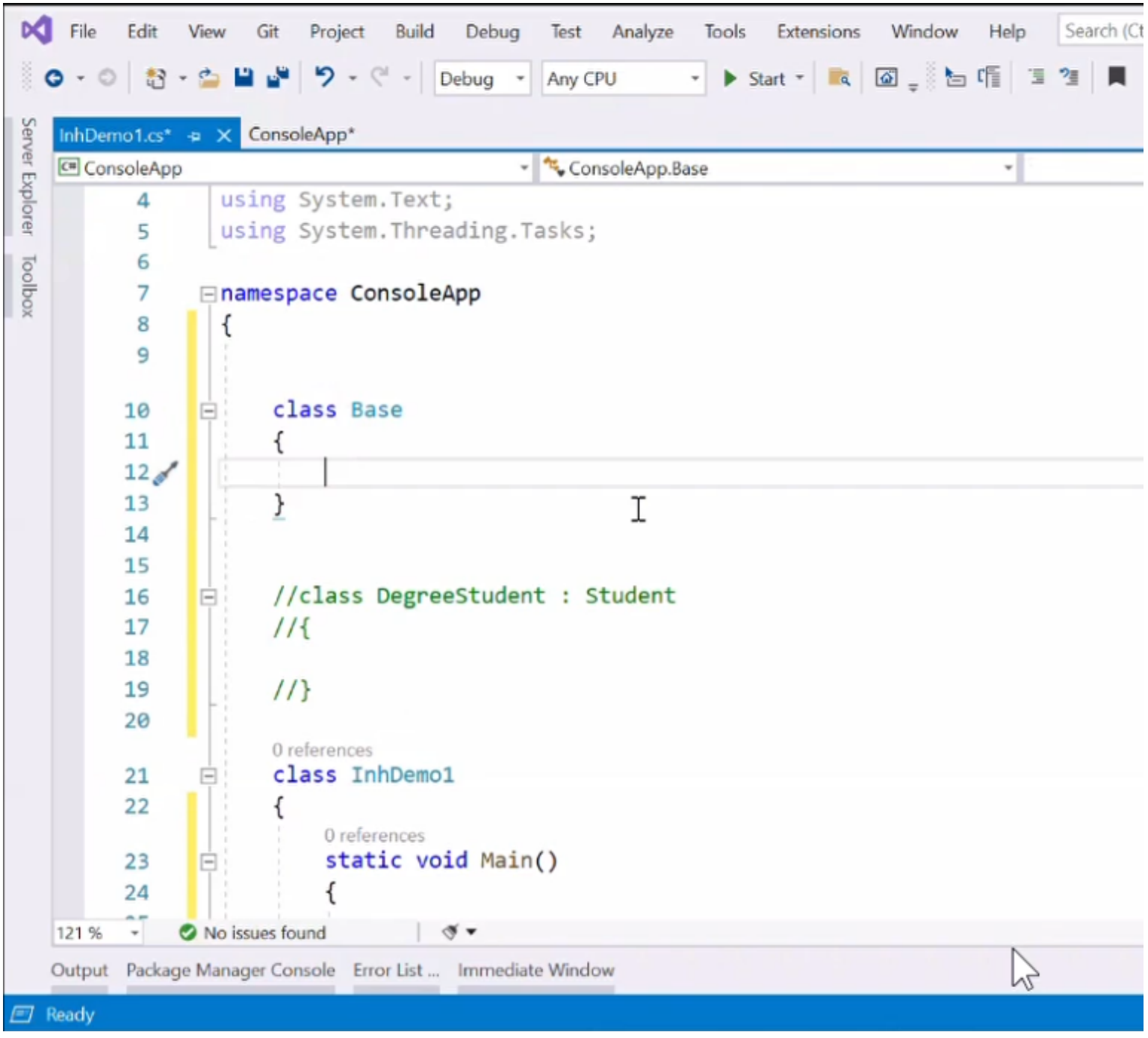
Is it ok if i inherit that class here

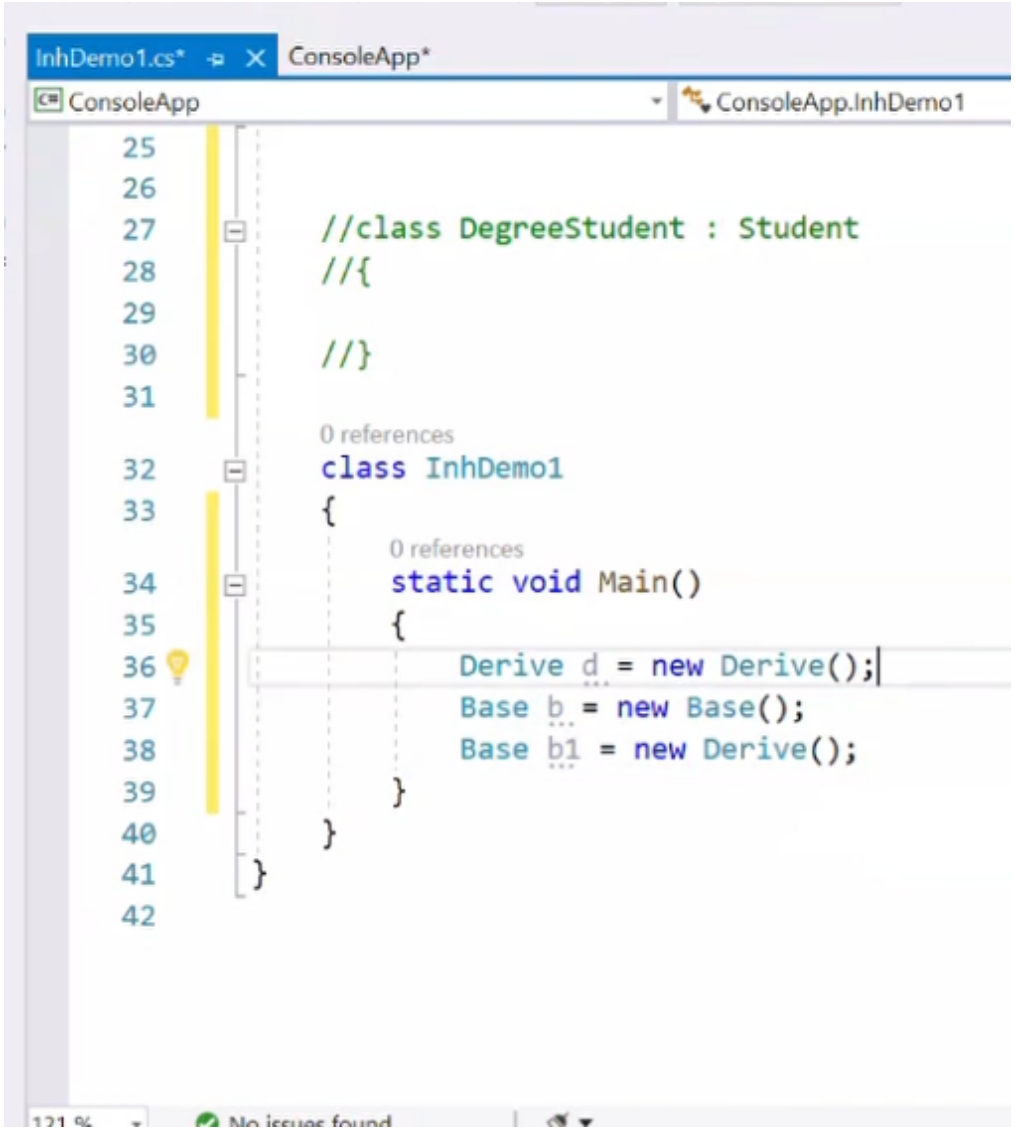

InhDemo1.cs*
ConsoleApp*
ConsoleApp
ConsoleApp.DegreeStudent
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace ConsoleApp
{
0 references
class DegreeStudent : Student
{
}
0 references
class InhDemo1
{
0 references
static void Main()
{
Student
}
}
121 %
3
0
Output
Package Manager Console
Error List
Immediate Window
Ready
Type here to search

As soon as i say degree student is a Student here , I'm getting here, Why??

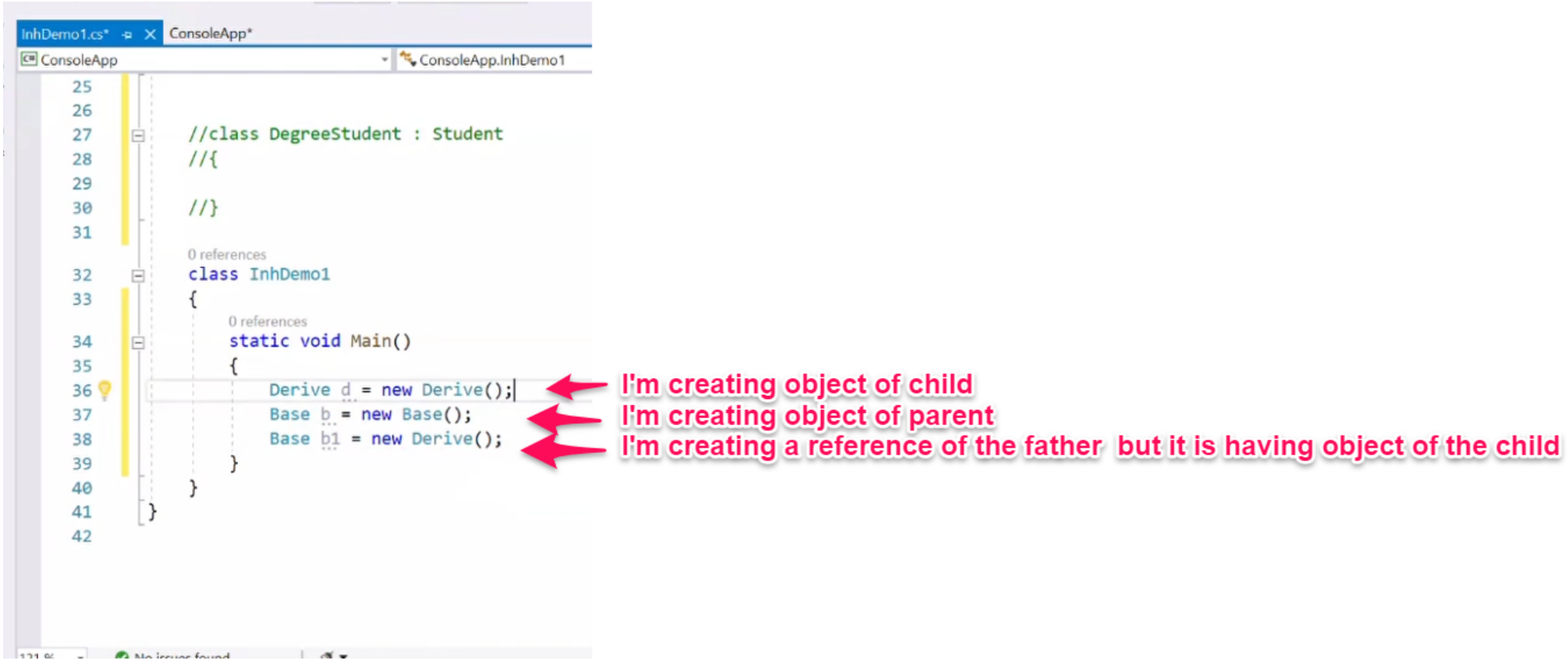


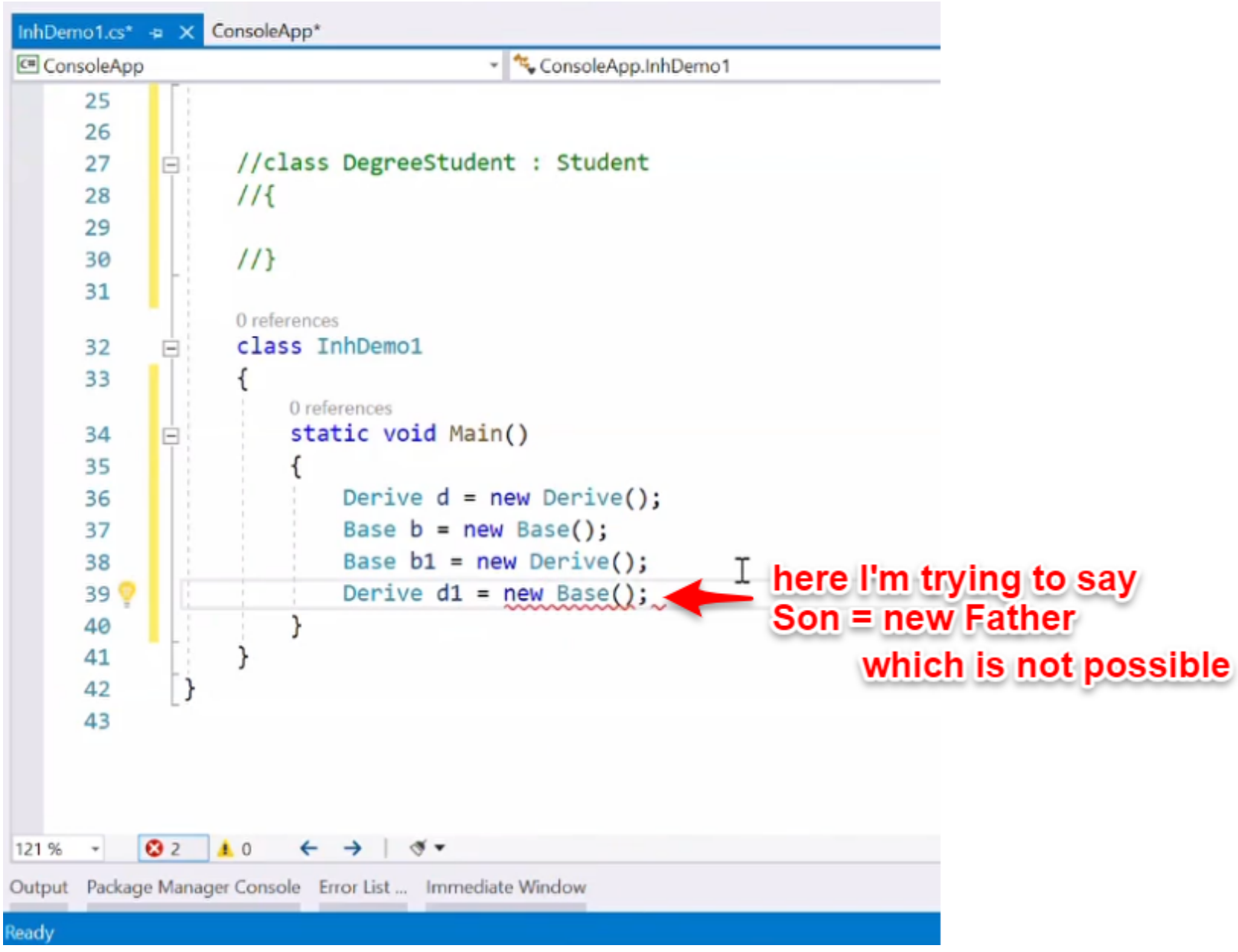
Lets get back to this later
Suppose I have a class called Base
my Base class is having some kind of property


File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Debug
Any CPU
Start
InhDemo1.cs*
ConsoleApp*
ConsoleApp
ConsoleApp.Base
Server Explorer
Toolbox
using System.Text;
using System.Threading.Tasks;
namespace ConsoleApp
{
class Base
{
}
//class DegreeStudent : Student
//{
//}
0 references
class InhDemo1
{
0 references
static void Main()
{
121 %
No issues found
Output
Package Manager Console
Error List ...
Immediate Window
Ready

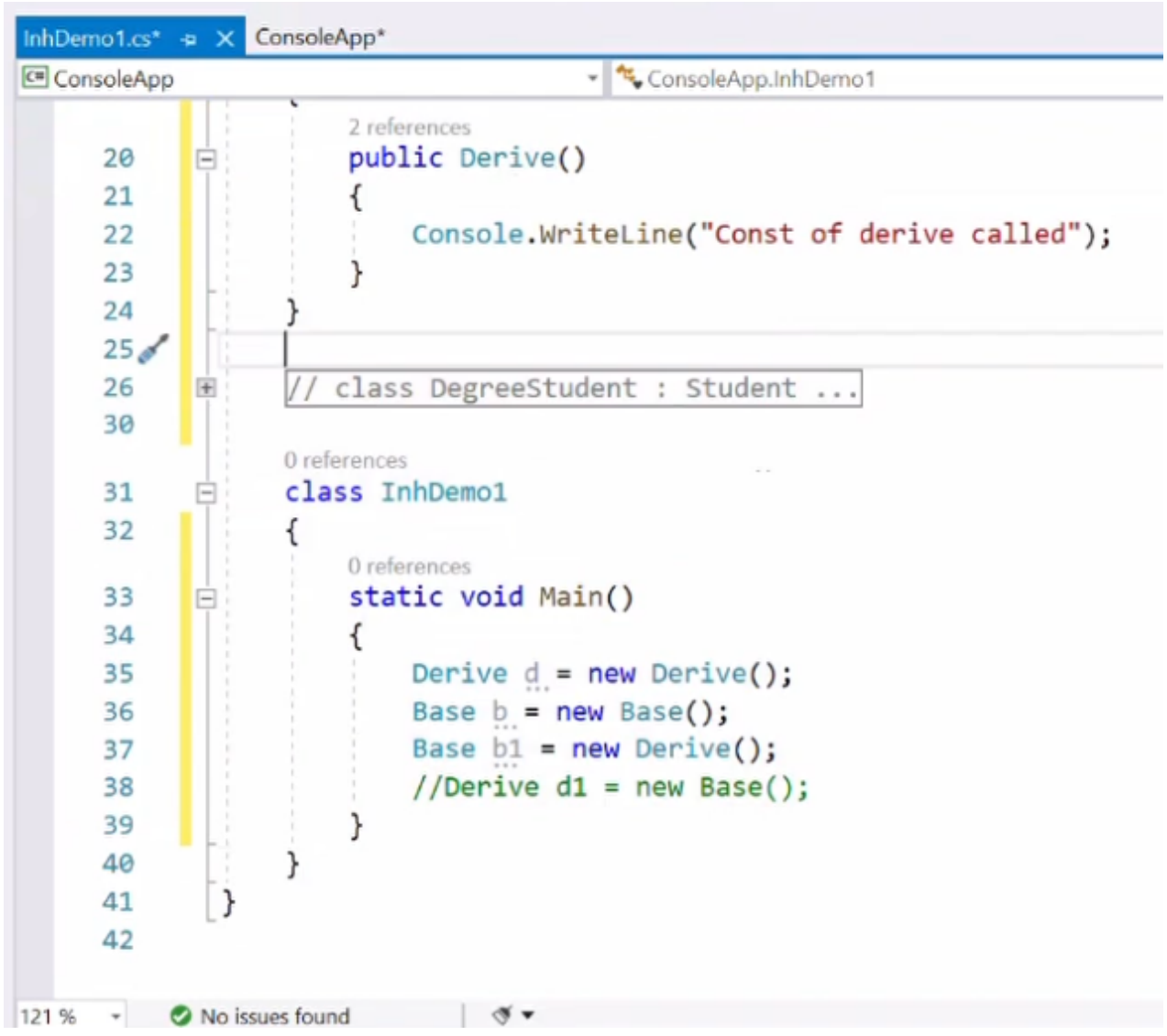


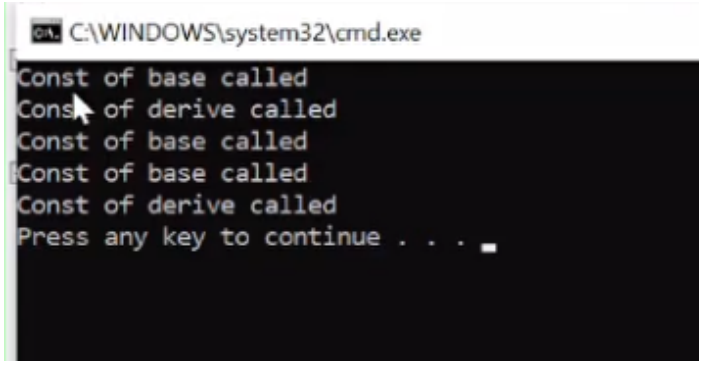
in Main() i'll say

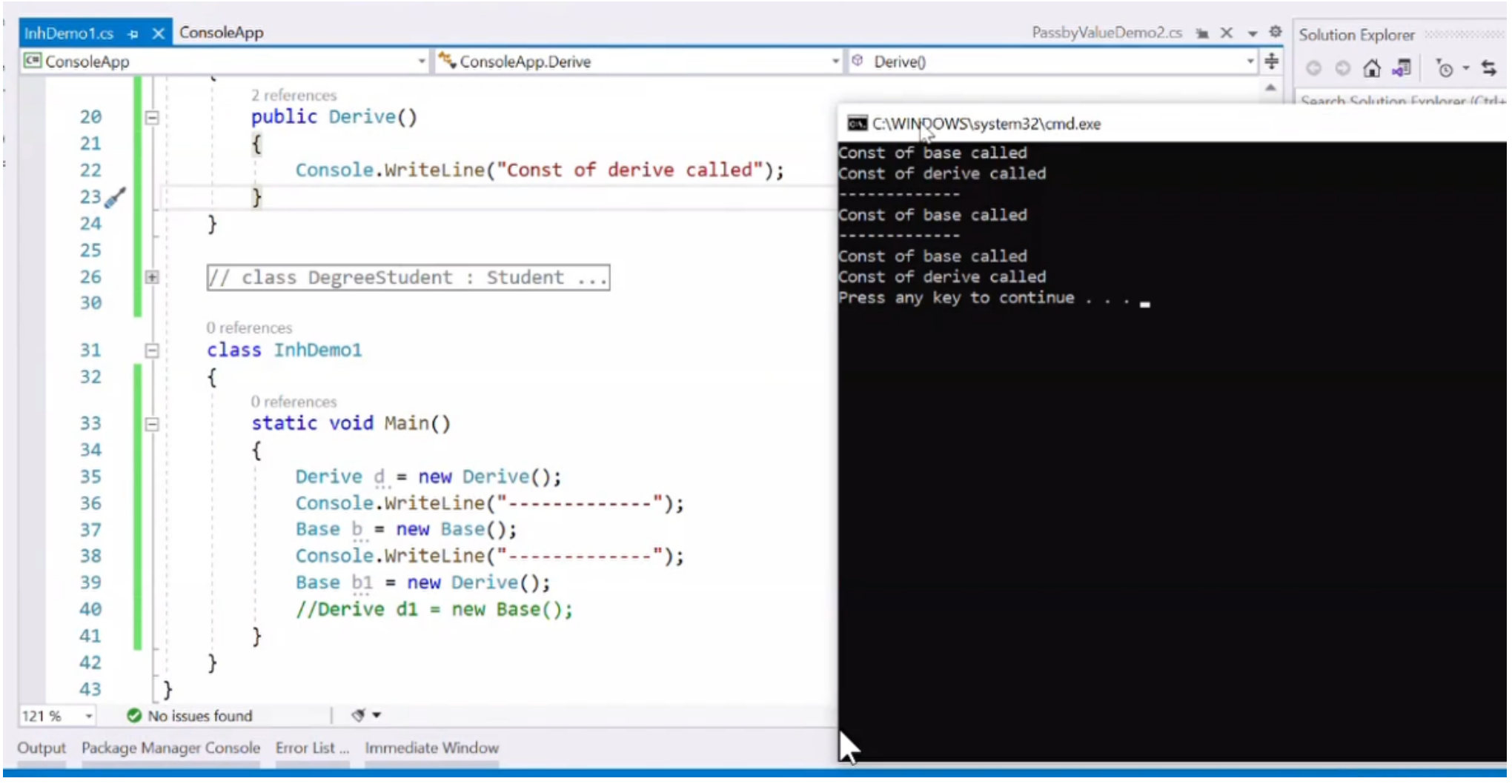

InhDemo1.cs*
ConsoleApp*
ConsoleApp
ConsoleApp.InhDemo1
25
26
27 //class DegreeStudent : Student
28 //{
29
30 //}
31
0 references
32 class InhDemo1
33 {
0 references
34 static void Main()
35 {
36 Derive d = new Derive();
37 Base b = new Base();
38 Base b1 = new Derive();
39 }
40 }
41 }
42
I'm creating object of child
I'm creating object of parent
I'm creating a reference of the father but it is having object of the child



What will be the output of this program


InhDemo1.cs*
ConsoleApp*
ConsoleApp
ConsoleApp.InhDemo1
2 references
public Derive()
{
Console.WriteLine("Const of derive called");
}
}
// class DegreeStudent : Student ...
0 references
class InhDemo1
{
0 references
static void Main()
{
Derive d = new Derive();
Base b = new Base();
Base b1 = new Derive();
//Derive d1 = new Base();
}
}
}
121 %
No issues found


C:\WINDOWS\system32\cmd.exe
Const of base called
Const of derive called
Const of base called
Const of base called
Const of derive called
Press any key to continue . . .



my derive class is inheriting the base class that's how first it has to create the object of base class
like son cannot be existing without the father,
so first father should be there
that's why first it create object of base then derived

Now move to 2nd Can a father be there without child ------ YES

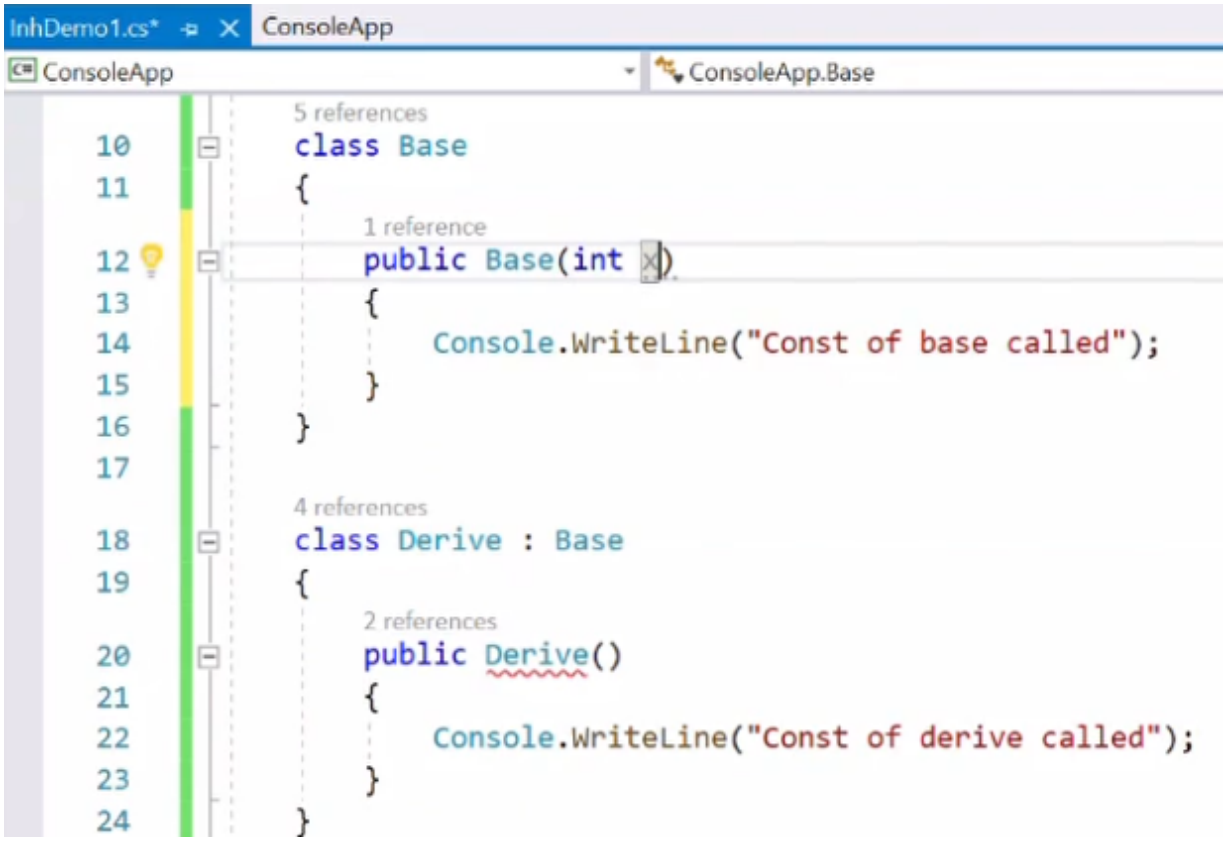
---

now when i say Base b1 = new Derive();
So obviously it will go to the base create the object of base class then it create the object of derived class

So whenever you create a object of derive it will always make sure 1st it calls the base class constructor and then it creates the object of base class and the it creates the object of self

when ever you create a object of the derive it will always check weather the base class already has the information or it has a constructor or not

I don't have anything any constructor



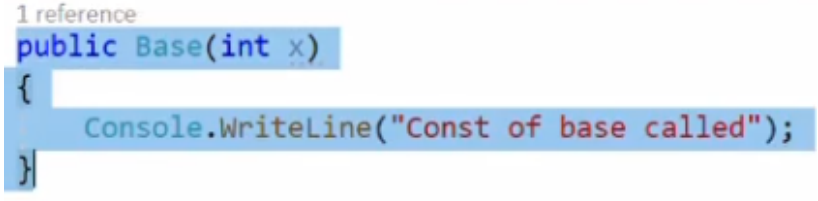
Now I'll pass int X here, see I got Error

now this derive is trying to create a object of base, but as my base dosen't have a default constructor or 0 parameterize constructor in this case

So it will say Boss you know what I cannot do this because you don't have a constructor which will not give me any parameter

So I need to support this constructor

I need to create a Instance of this particular constructor



this base is expecting me to pass x

So i need to ask user to pass x here or in this particular constructor I have to call base passing some value (dummy value)