[What is Polymorphisms?](http://www.dotnetfunda.com/interviews/show/64/what-is-polymorphisms" \o "What is Polymorphisms?" \t "_blank)

Polymorphism means one interface and many forms. Polymorphism is a characteristics of being able to assign a different meaning or usage to something in different contexts specifically to allow an entity such as a variable, a function or an object to have more than one form.   
  
There are two types of Polymorphism.   
**Compile time:** function or operator overloading   
**Runtime:** Inheritence & virtual functions

[What is Abstract method?](http://www.dotnetfunda.com/interviews/show/65/what-is-abstract-method" \o "What is Abstract method?" \t "_blank)

Abstract method doesn't provide the implementation & forces the derived class to override the method.

[What is Virtual method?](http://www.dotnetfunda.com/interviews/show/66/what-is-virtual-method" \o "What is Virtual method?" \t "_blank)

Virtual Method has implementation & provide the derived class with the option to override it.

[Can Struct be inherited?](http://www.dotnetfunda.com/interviews/show/67/can-struct-be-inherited" \o "Can Struct be inherited?" \t "_blank)

No, Struct can't be inherited as this is implicitly sealed.

[What is Object?](http://www.dotnetfunda.com/interviews/show/82/what-is-object" \o "What is Object?" \t "_blank)

Object is anything that is identifiable as a single material item.

[What is Class?](http://www.dotnetfunda.com/interviews/show/83/what-is-class" \o "What is Class?" \t "_blank)

A Class is the generic definition of what an object is a template.   
  
The keyword class in C# indicates that we are going to define a new class (type of object)

[What is Static field?](http://www.dotnetfunda.com/interviews/show/84/what-is-static-field" \o "What is Static field?" \t "_blank)

To indicate that a field should only be stored once no matter how many instance of the class we create.

[What is Static Method?](http://www.dotnetfunda.com/interviews/show/85/what-is-static-method" \o "What is Static Method?" \t "_blank)

It is possible to declare a method as Static provided that they don't attempt to access any instance data or other instance methods.

[What is Inheritance?](http://www.dotnetfunda.com/interviews/show/87/what-is-inheritance" \o "What is Inheritance?" \t "_blank)

It provides a convenient way to reuse existing fully tested code in different context thereby saving lot of coding.   
  
Inheritance of classes in C# is always implementation Inheritance.

[What is Virtual keyword?](http://www.dotnetfunda.com/interviews/show/88/what-is-virtual-keyword" \o "What is Virtual keyword?" \t "_blank)

This keyword indicates that a member can be overridden in a child class. It can be applied to methods, properties, indexes and events.

[What is New modifiers?](http://www.dotnetfunda.com/interviews/show/89/what-is-new-modifiers" \o "What is New modifiers?" \t "_blank)

The new modifiers hides a member of the base class. C# supports only hide by signature.

[What is Abstract Class?](http://www.dotnetfunda.com/interviews/show/90/what-is-abstract-class" \o "What is Abstract Class?" \t "_blank)

Abstract class is a class that can not be instantiated, it exists extensively for inheritance and it must be inherited. There are scenarios in which it is useful to define classes that is not intended to instantiate; because such classes normally are used as base-classes in inheritance hierarchies, we call such classes abstract classes.   
  
Abstract classes cannot be used to instantiate objects; because abstract classes are incomplete, it may contain only definition of the properties or methods and derived classes that inherit this implements it's properties or methods.   
  
Static, Value Types & interface doesn't support abstract modifiers. Static members cannot be abstract. Classes with abstract member must also be abstract.   
  
**For detailed example, read this article**<http://www.dotnetfunda.com/articles/article467-abstract-class--explained.aspx>

[What is Sealed modifiers?](http://www.dotnetfunda.com/interviews/show/91/what-is-sealed-modifiers" \o "What is Sealed modifiers?" \t "_blank)

Sealed types cannot be inherited & are concrete.   
Sealed modifiers can also be applied to instance methods, properties, events & indexes. It can't be applied to static members.   
  
Sealed members are allowed in sealed and non-sealed classes.

[What is an Interface?](http://www.dotnetfunda.com/interviews/show/92/what-is-an-interface" \o "What is an Interface?" \t "_blank)

An interface is a contract & defines the requisite behavior of generalization of types.   
  
An interface mandates a set of behavior, but not the implementation. Interface must be inherited. We can't create an instance of an interface.   
  
An interface is an array of related function that must be implemented in derived type. Members of an interface are implicitly public & abstract.   
  
An interface can inherit from another interface.

[When to use Interface over abstract class?](http://www.dotnetfunda.com/interviews/show/332/when-to-use-interface-over-abstract-class" \o "When to use Interface over abstract class?" \t "_blank)

Abstract Classes: Classes which cannot be instantiated. This means one cannot make a object of this class or in other way cannot create object by saying ClassAbs abs = new ClassAbs(); where ClassAbs is abstract class.   
Abstract classes contains have one or more abstarct methods, ie method body only no implementation.   
Interfaces: These are same as abstract classes only difference is we can only define method definition and no implementation.   
When to use wot depends on various reasons. One being design choice.   
One reason for using abstarct classes is we can code common   
functionality and force our developer to use it. I can have a complete   
class but I can still mark the class as abstract.   
Developing by interface helps in object based communication.

[What is pure virtual function?](http://www.dotnetfunda.com/interviews/show/333/what-is-pure-virtual-function" \o "What is pure virtual function?" \t "_blank)

When you define only function prototype in a base class without and do the complete implementation in derived class. This base class is called abstract class and client won’t able to instantiate an object using this base class.   
  
A pure virtual function is a function that must be overridden in a derived class and need not be defined. A virtual function is declared to be "pure" using the curious "=0"   
syntax:   
class Base {   
public:   
void f1(); // not virtual   
virtual void f2(); // virtual, not pure   
virtual void f3() = 0; // pure virtual   
};

[Can we specify the access modifier for explicitly implemented interface method?](http://www.dotnetfunda.com/interviews/show/419/can-we-specify-the-access-modifier-for-explicitly-implemented-interfac" \o "Can we specify the access modifier for explicitly implemented interface method?" \t "_blank)

No, we can't specify the access modifier for the explicitly implemented interface method. By default its scope will be internal.

[What is Protected access modifier in C#?](http://www.dotnetfunda.com/interviews/show/420/what-is-protected-access-modifier-in-csharp" \o "What is Protected access modifier in C#?" \t "_blank)

The protected keyword is a member access modifier. It can only be used in a declaring a function or method not in the class ie. a class can't be declared as protected class.   
  
A protected member is accessible from within the class in which it is declared, and from within any class derived from the class that declare this member. In other words access is limited to within the class definition and any class that inherits from the class   
  
A protected member of a base class is accessible in a derived class only if the access takes place through the derived class type.   
  
For more details see http://msdn.microsoft.com/en-us/library/bcd5672a(VS.71).aspx

[What is Public access modifier in C#?](http://www.dotnetfunda.com/interviews/show/421/what-is-public-access-modifier-in-csharp" \o "What is Public access modifier in C#?" \t "_blank)

The public keyword is an access modifier for types and type members ie. we can declare a class or its member (functions or methods) as Public. There are no restrictions on accessing public members.

[What is Private access modifier in C#?](http://www.dotnetfunda.com/interviews/show/422/what-is-private-access-modifier-in-csharp" \o "What is Private access modifier in C#?" \t "_blank)

The private keyword is a member access modifier ie. we can't explicitly declare a class as Private, however if do not specify any access modifier to the class, its scope will be assumed as Private. Private access is the least permissive access level of all access modifiers.   
  
Private members are accessible only within the body of the class or the struct in which they are declared. This is the default access modifier for the class declaration.   
  
For more details, see http://msdn.microsoft.com/en-us/library/st6sy9xe(VS.71).aspx

[What is Internal access modifier in C#?](http://www.dotnetfunda.com/interviews/show/423/what-is-internal-access-modifier-in-csharp" \o "What is Internal access modifier in C#?" \t "_blank)

The internal keyword is an access modifier for types and type members ie. we can declare a class as internal or its member as internal. Internal members are accessible only within files in the same assembly (.dll). In other words, access is limited exclusively to classes defined within the current project assembly.   
  
For more details see http://msdn.microsoft.com/en-us/library/7c5ka91b(VS.71).aspx

[What is Protected Internal access modifier in C#?](http://www.dotnetfunda.com/interviews/show/424/what-is-protected-internal-access-modifier-in-csharp" \o "What is Protected Internal access modifier in C#?" \t "_blank)

Protected Internal is a access modifiers for the members (methods or functions) ie. you can't declare a class as protected internal explicitly. The members access is limited to the current assembly or types derived from the containing class.   
  
Protected Internal means the method is accessible by anything that can access the protected method UNION with anything that can access the internal method.   
  
For more details read http://haacked.com/archive/2007/10/29/what-does-protected-internal-mean.aspx

[Default Access modifiers in C#?](http://www.dotnetfunda.com/interviews/show/425/default-access-modifiers-in-csharp" \o "Default Access modifiers in C#?" \t "_blank)

An **enum** has default modifier as ***public***  
  
A **class** has default modifiers as ***Internal***. It can declare members (methods etc) with following access modifiers:   
public   
internal   
private   
protected internal   
  
An **interface**has default modifier as ***public***   
  
A **struct**has default modifier as ***Internal***and it can declare its members (methods etc) with following access modifiers:   
public   
internal   
private   
  
A **methods, fields, and properties** has default access modifier as "Private" if no modifier is specified.

[What is method overloading?](http://www.dotnetfunda.com/interviews/show/453/what-is-method-overloading" \o "What is method overloading?" \t "_blank)

Method overloading allows us to write different version of the same method in a class or derived class. Compiler automatically select the most appropriate method based on the parameter supplied. 

public class MultiplyNumbers

{

public int Multiply(int a, int b)

{

return a \* b;

}

public int Multiply(int a, int b, int c)

{

return a\*b\*c;

}

}

To call the above method, you can use following code. 

MultiplyNumbers mn = new MultiplyNumbers();

int number = mn.Multiply(2, 3) // result = 6

int number1 = mn.Multiply(2, 3, 4) // result = 24

You can't have a overload method with same number parameters but different return type. In order to create overload method, the return type must be the same and parameter type must be different or different in numbers.

[What is Overriding?](http://www.dotnetfunda.com/interviews/show/456/what-is-overriding" \o "What is Overriding?" \t "_blank)

Method overriding is a feature that allows to invoke functions (that have the same signatures) and that belong to different classes in the same hierarchy of inheritance using the base class reference. In C# it is done using keywords **virtual** and **overrides**.   
  
For more information visit <http://www.codeproject.com/KB/cs/cs_methodoverride.aspx>

[What is Method overloading?](http://www.dotnetfunda.com/interviews/show/488/what-is-method-overloading" \o "What is Method overloading?" \t "_blank)

Method overloading occurs when a class contains two methods with the same name, but different signatures.

[What is Method Overriding? How to override a function in C#?](http://www.dotnetfunda.com/interviews/show/489/what-is-method-overriding-how-to-override-a-function-in-csharp" \o "What is Method Overriding? How to override a function in C#?" \t "_blank)

Use the override modifier to modify a method, a property, an indexer, or an event. An override method provides a new implementation of a member inherited from a base class. The method overridden by an override declaration is known as the overridden base method. The overridden base method must have the same signature as the override method.   
  
You cannot override a non-virtual or static method. The overridden base method must be virtual, abstract, or override.

[Can we call a base class method without creating instance?](http://www.dotnetfunda.com/interviews/show/490/can-we-call-a-base-class-method-without-creating-instance" \o "Can we call a base class method without creating instance?" \t "_blank)

Yep. But ..   
  
\* Its possible If its a static method.   
  
\* Its possible by inheriting from that class also.   
  
\* Its possible from derived classes using base keyword.

[In which cases you use override and new base?](http://www.dotnetfunda.com/interviews/show/491/in-which-cases-you-use-override-and-new-base" \o "In which cases you use override and new base?" \t "_blank)

Use the new modifier to explicitly hide a member inherited from a base class. To hide an inherited member, declare it in the derived class using the same name, and modify it with the new modifier.

[Difference between new and override keyword?](http://www.dotnetfunda.com/interviews/show/647/difference-between-new-and-override-keyword" \o "Difference between new and override keyword?" \t "_blank)

Let me explain this through code. 

using System;

using System.Data;

using System.Text;

using System.Windows.Forms;

namespace BaseDerive

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

BaseClass b = new BaseClass();

b.func1();

DeriveClass d = new DeriveClass();

d.func1();

//Calls Base class function 1 as new keyword is used.

BaseClass bd = new DeriveClass();

bd.func1();

//Calls Derived class function 2 as override keyword is used.

BaseClass bd2 = new DeriveClass();

bd2.func2();

}

}

public class BaseClass

{

public virtual void func1()

{

MessageBox.Show("Base Class function 1.");

}

public virtual void func2()

{

MessageBox.Show("Base Class function 2.");

}

public void func3()

{

MessageBox.Show("Base Class function 3.");

}

}

public class DeriveClass : BaseClass

{

public new void func1()

{

MessageBox.Show("Derieve Class fuction 1 used new keyword");

}

public override void func2()

{

MessageBox.Show("Derieve Class fuction 2 used override keyword");

}

public void func3()

{

MessageBox.Show("Derieve Class fuction 3 used override keyword");

}

}

}

This is a window application so all the code for calling the function through objects is written in Form\_Load event.   
As seen in above code, I have declared 2 classes. One works as a Base class and second is a derieve class derived from base class.   
  
Now the difference is   
  
**new: hides the base class function.   
Override: overrides the base class function.** 

BaseClass objB = new DeriveClass();

If we create object like above notation and make a call to any function which exists in base class and derive class both, then it will always make a call to function of base class. If we have overidden the method in derive class then it wlll call the derive class function.   
  
For example… 

objB.func1(); //Calls the base class function. (In case of new keyword)

objB.func2(); //Calls the derive class function. (Override)

objB.func3(); //Calls the base class function.(Same prototype in both the class.)

Note:   
// This will throw a compile time error. (Casting is required.) 

DeriveClass objB = new BaseClass();

//This will throw run time error. (Unable to cast) 

DeriveClass objB = (DeriveClass) new BaseClass();

Hope this helps!!!!!

[What is a private constructor? Where will you use it?](http://www.dotnetfunda.com/interviews/show/661/what-is-a-private-constructor-where-will-you-use-it" \o "What is a private constructor? Where will you use it?" \t "_blank)

When you declare a Constructor with Private access modifier then it is called Private Constructor. We can use the private constructor in singleton pattern.   
  
If you declare a Constructor as private then it doesn’t allow to create object for its derived class, i.e you loose inherent facility for that class.   
  
Example: 

Class A

{

// some code

Private Void A()

{

//Private Constructor

}

}

Class B:A

{

//code

}

B obj = new B();// will give Compilation Error

Because Class A constructor declared as private hence its accessibility limit is to that class only, Class B can't access. When we create an object for Class B that constructor will call constructor A but class B have no rights to access the Class A constructor hence we will get compilation error.

[Can we declare private class in a Namespace?](http://www.dotnetfunda.com/interviews/show/662/can-we-declare-private-class-in-a-namespace" \o "Can we declare private class in a Namespace?" \t "_blank)

No. If you try to create a private class in a Namespace, Compiler will throw a compile time error “Namespace elements cannot be explicitly declared as private, protected, or protected internal”.   
  
Reason: The message says it all. Classes can only be declared as private, protected or protected internal when declared as nested classes, other than that, it doesn't make sense to declare a class with a visibility that makes it unusable, even in the same module. Top level classes cannot be private, they are "internal" by default, and you can just make them public to make them visible from outside your DLL.

[What is Polymorphism?](http://www.dotnetfunda.com/interviews/show/854/what-is-polymorphism" \o "What is Polymorphism?" \t "_blank)

In OPP’S, polymorphism(Greek meaning “having multiple forms”) is the ablity of being able to assign a different meaning or usage to something in different contexts - specifically, to allow an entity such as a a function, or an object to have more than one forms.   
  
In C# :   
Parent classes may define and implement “virtual” methods(Which is done using the “virtual” keyword), and derived classes can override them(using the “override” keyword), which means they provide their own definition and implementation.At run-time, when user’s code calls the method, the CLR looks up the run-time type of the object, and invokes that override of the virtual method. Thus in your source code when a method of the base class is called it executes the overriden method.   
====================================================   
Regards Hefin Dsouza.

[What Are Attributes in DotNet?](http://www.dotnetfunda.com/interviews/show/855/what-are-attributes-in-dotnet" \o "What Are Attributes in DotNet?" \t "_blank)

An Attribute is a declarative tag which can be used to provide information to the compiler about the behaviour of the C# elements such as classes and assemblies.   
C# provides convenient technique that will handle tasks such as performing compile time operations , changing the behaviour of a method at runtime or maybe even handle unmanaged code.   
C# Provides many Built-in Attributes   
  
Some Popular ones are   
  
- Obsolete   
- DllImport   
- Conditional   
- WebMethod   
  
and Many more.   
Members please keep on posting more responses providing more In-Built attributes.   
  
Regards Hefin Dsouza

[What can you do to make class available for inheritance but you need to prevent it's method to come in inheritance chain?](http://www.dotnetfunda.com/interviews/show/1619/what-can-you-do-to-make-class-available-for-inheritance-but-you-need-t" \o "What can you do to make class available for inheritance but you need to prevent it's method to come in inheritance chain?" \t "_blank)

Well, Declare a class with **public**access specifier and mark all it's method to **private** . As anything which is declared with sealed keyword cannot be inherited.

[What's the Difference between Interface and Abstract Class](http://www.dotnetfunda.com/interviews/show/1715/whats-the-difference-between-interface-and-abstract-class" \o "What's the Difference between Interface and Abstract Class" \t "_blank)

Abstract Class:   
Have constructors.   
Not necessarily for the class inheriting it to Implement all the Methods.   
Doesn't Support Multiple Inheritance.   
  
Where everything is Opposite in the Interfaces.

[What are the various types of Constructors](http://www.dotnetfunda.com/interviews/show/1721/what-are-the-various-types-of-constructors" \o "What are the various types of Constructors" \t "_blank)

**Public :** Accessible to All   
**Private:**Those classes in which only static members are there and you don't want there objects to be created in any class.   
**Static:** Used for initializing only the static members of the class. These will be invoked for the very first time the class is being loaded on the memory. They cannot accept any arguments. Static Constructors cannot have any access modifiers.   
**Intern:** implementations of the abstract class to the assembly defining the class. A class containing an internal constructor cannot be instantiated outside of the assembly (Namespace).   
and **External**

[What are Constructors ?](http://www.dotnetfunda.com/interviews/show/1723/what-are-constructors" \o "What are Constructors ?" \t "_blank)

Constructors are used for initializing the members of a class whenever an object is created with the default values for initialization.   
  
If no constructor defined then the CLR will provide an implicit constructor which is called as Default Constructor.   
  
A class can have any number of constructors provided they vary with the number of arguments that are passed, which is they should have different signatures.   
  
Constructors do not return a value   
Constructors can be overloaded

[When to Use Abstract Classes and When Interfaces.](http://www.dotnetfunda.com/interviews/show/1729/when-to-use-abstract-classes-and-when-interfaces" \o "When to Use Abstract Classes and When Interfaces." \t "_blank)

If you anticipate creating multiple versions of your component, create an abstract class. Abstract classes provide a simple and easy way to version your components. By updating the base class, all inheriting classes are automatically updated with the change. Interfaces, on the other hand, cannot be changed once created. If a new version of an interface is required, you must create a whole new interface.   
  
If the functionality you are creating will be useful across a wide range of disparate objects, use an interface. Abstract classes should be used primarily for objects that are closely related, whereas interfaces are best suited for providing common functionality to unrelated classes.   
  
If you are designing small, concise bits of functionality, use interfaces. If you are designing large functional units, use an abstract class.   
  
If you want to provide common, implemented functionality among all implementations of your component, use an abstract class. Abstract classes allow you to partially implement your class, whereas interfaces contain no implementation for any members.

[Diversities between an abstract method & virtual method ?](http://www.dotnetfunda.com/interviews/show/1899/diversities-between-an-abstract-method-virtual-method" \o "Diversities between an abstract method & virtual method ?" \t "_blank)

An Abstract method does not provide an implementation and forces overriding to the deriving class (unless the deriving class also an abstract class), where as the virtual method has an implementation and leaves an option to override it in the deriving class. Thus Virtual method has an implementation & provides the derived class with the option of overriding it. Abstract method does not provide an implementation & forces the derived class to override the method.

[What is Early binding and late binding?](http://www.dotnetfunda.com/interviews/show/1916/what-is-early-binding-and-late-binding" \o "What is Early binding and late binding?" \t "_blank)

Calling a non-virtual method, decided at a compile time is known as early binding. Calling a virtual method (Pure Polymorphism), decided at a runtime is known as late binding.

[Difference between ASP Session and ASP.NET Session?](http://www.dotnetfunda.com/interviews/show/1918/difference-between-asp-session-and-aspnet-session" \o "Difference between ASP Session and ASP.NET Session?" \t "_blank)

Asp.net session supports cookie less session & it can span across multiple servers.

[Illustrate Server.Transfer and Response.Redirect?](http://www.dotnetfunda.com/interviews/show/1927/illustrate-servertransfer-and-responseredirect" \o "Illustrate Server.Transfer and Response.Redirect?" \t "_blank)

Server.Transfer, transfers the control of a web page, posting a form data, while Response.Redirect simply redirects a page to another page, it can not post a form data to another page. Server.Transfer is more efficient over the Response.Redirect, because Response.Redirect causes a round trip to server as the page is processed once again on the client and a request is made to server there after.   
But the browser url is not changed in case of Server.Transfer i.e. Browser history is not modified in using it.

[How's method overriding different from overloading?](http://www.dotnetfunda.com/interviews/show/1939/hows-method-overriding-different-from-overloading" \o "How's method overriding different from overloading?" \t "_blank)

When overriding, you change the method behavior for a derived class. Overloading simply involves having a method with the same name within the class.

[What does the keyword virtual mean in the method definition?](http://www.dotnetfunda.com/interviews/show/1940/what-does-the-keyword-virtual-mean-in-the-method-definition" \o "What does the keyword virtual mean in the method definition?" \t "_blank)

The method can be over-ridden.

[Can you declare the override method static while the original method is non-static?](http://www.dotnetfunda.com/interviews/show/1941/can-you-declare-the-override-method-static-while-the-original-method-i" \o "Can you declare the override method static while the original method is non-static?" \t "_blank)

No, you can't, the signature of the virtual method must remain the same, only the keyword virtual is changed to keyword override.

[Can you override private virtual methods?](http://www.dotnetfunda.com/interviews/show/1942/can-you-override-private-virtual-methods" \o "Can you override private virtual methods?" \t "_blank)

No, you cannot access private methods in inherited classes.

[Can you prevent your class from being inherited and becoming a base class for some other classes?](http://www.dotnetfunda.com/interviews/show/1943/can-you-prevent-your-class-from-being-inherited-and-becoming-a-base-cl" \o "Can you prevent your class from being inherited and becoming a base class for some other classes?" \t "_blank)

Yes, that's what keyword sealed in the class definition is for. The developer trying to derive from your class will get a message: cannot inherit from Sealed class WhateverBaseClassName. It's the same concept as final class in Java.

[Can you allow class to be inherited, but prevent the method from being over-ridden?](http://www.dotnetfunda.com/interviews/show/1944/can-you-allow-class-to-be-inherited-but-prevent-the-method-from-being" \o "Can you allow class to be inherited, but prevent the method from being over-ridden?" \t "_blank)

Yes, just leave the class public and make the method sealed.

[Why can't you specify the accessibility modifier for methods inside the interface?](http://www.dotnetfunda.com/interviews/show/1945/why-cant-you-specify-the-accessibility-modifier-for-methods-inside-the" \o "Why can't you specify the accessibility modifier for methods inside the interface?" \t "_blank)

you are not allowed to specify any accessibility, it's public by default.

[Static datamembers should be initialized inside the constructor. True or False.](http://www.dotnetfunda.com/interviews/show/2035/static-datamembers-should-be-initialized-inside-the-constructor-true-o" \o "Static datamembers should be initialized inside the constructor. True or False." \t "_blank)

False. Static datamembers should not be initialised inside constructor.

[Static methods can not use non static members. True or False.](http://www.dotnetfunda.com/interviews/show/2036/static-methods-can-not-use-non-static-members-true-or-false" \o "Static methods can not use non static members. True or False." \t "_blank)

True

[A constructor can be private. True or False](http://www.dotnetfunda.com/interviews/show/2037/a-constructor-can-be-private-true-or-false" \o "A constructor can be private. True or False" \t "_blank)

True. A constructor can be private. We can declare a constructor as private.

[What is the work of a constructor?](http://www.dotnetfunda.com/interviews/show/2074/what-is-the-work-of-a-constructor" \o "What is the work of a constructor?" \t "_blank)

Constructor creates and initialises the objects in an application.

[Name the operators that cannot be overloaded.](http://www.dotnetfunda.com/interviews/show/2104/name-the-operators-that-cannot-be-overloaded" \o "Name the operators that cannot be overloaded." \t "_blank)

sizeof   
.   
.\*   
.->   
::   
?:

[OOps Interview](http://www.dotnetfunda.com/interviews/show/2133/oops-interview" \o "OOps Interview" \t "_blank)

ok

[What is "this" pointer?](http://www.dotnetfunda.com/interviews/show/2462/what-is-this-pointer" \o "What is \"this\" pointer?" \t "_blank)

This pointer is a pointer which points to the current object of a class. this is actually a keyword which is used as a pointer which differentiate the current object with global object.

[public class Base { public virtual void foo(int x) { Console.WriteLine("Base foo(int)"); } } public class Derived: Base { public void foo(int x) { Console.WriteLine("Derived foo(int)"); } } class Program { static void Main(string[] args) { Derived d = new Derived(); int i = 10; d.foo(i); } }](http://www.dotnetfunda.com/interviews/show/3748/public-class-base-public-virtual-void-fooint-x-consolewritelinebase-fo" \o )

NOTE: This is objective type question, Please click question title for correct answer.

[Difference between sealed and static classes](http://www.dotnetfunda.com/interviews/show/3881/difference-between-sealed-and-static-classes" \o "Difference between sealed and static classes" \t "_blank)

sealed classes:   
  
1)we can create their instances, but cannot inherit them   
  
ex:   
  
sealed class demo   
{   
  
}   
  
class abc:demo   
{   
--Wrong   
}   
2)They can contain static as well as nonstatic members.   
  
static classes:   
  
1)we can neither create their instances, nor inherit them   
  
ex:   
static class Program   
{   
  
}   
  
2)They can have static members only.

[Differences between a structure and class](http://www.dotnetfunda.com/interviews/show/3884/differences-between-a-structure-and-class" \o "Differences between a structure and class" \t "_blank)

Structure:   
  
1)It is a Value Type   
2)Its variable directly contains the data on the stack   
3)Each structure variable has its independent copy   
of data.   
4)One structure cannot inherit other   
5)They do not have destructors   
6)They do no have explicit parameterless constructors   
7)we cannot put sealed /abstract modifiers before the structures.   
8)Easier memory management   
9)examples:   
int, short,long,DateTime,   
Point   
(predefined)   
  
Uses:   
Structures are typically used for handling   
small amounts of data or where inheritance, overriding is not required   
example: int a=100;   
  
  
Class   
  
1)It is a reference Type   
2)Its variable has references to the data(data is stored in the object created in the heap) .   
3)Two Class variables can refer to the same object   
4)One class can inherit the other(unless the class is sealed/static)   
5)Classes have destructors   
6)They can have explicit parameterless constructors   
7)Sealed/abstract modifers can be put before classes.   
8) Comparitively Difficult memory management   
9)example: SqlConnection,DataView(predefined classes)   
  
Classes are typically used where inheritance, overriding is required   
or we need to create objects capable of handling large data   
example: DataSet,ArrayList can handle large data.

[What are the different ways a method can be overloaded?](http://www.dotnetfunda.com/interviews/show/3888/what-are-the-different-ways-a-method-can-be-overloaded" \o "What are the different ways a method can be overloaded?" \t "_blank)

Different parameter data types, different number of parameters, different order of   
parameters.   
  
example: int area(int a, int b)   
{   
return a\*b; --different number of parameters   
}   
int area(int b)   
{   
return a\*a;   
}   
  
  
--parameter return types   
  
int calc(int a)   
{   
return a;   
}   
  
double calc(double b)   
{   
return b\*5;   
}

[Difference between a Class and an object](http://www.dotnetfunda.com/interviews/show/3953/difference-between-a-class-and-an-object" \o "Difference between a Class and an object" \t "_blank)

Class:   
  
1)It is a datatype that contains the programming logic.   
  
2)Class is visible in the source code and resides in hard disk.   
  
3)Class is like a template or blueprint of the object. It implements reusability,   
  
encapsulation, inheritance   
  
example:Button is a class with properties like Text,BackColor,   
events like click, methods like Focus   
  
Object:   
  
1)it is a chunk of memory that implements the class logic.   
  
2)Object is in the RAM and not visible in source code.   
  
3)It is the real world implementation of the class.   
Each object has its own copy of data.   
example: Button1, Button2 are the objects of Button class.

[Define OOPS. What are its benefits?](http://www.dotnetfunda.com/interviews/show/3961/define-oops-what-are-its-benefits" \o "Define OOPS. What are its benefits?" \t "_blank)

**OOPS**   
  
Object Oriented Programming Stuctures:   
  
It is a programming methodology in which the programs are organized as collections of objects.Each object represents an instance of some class.The classes can be interrelated to each other through inheritance   
  
OOPS revolves around these concepts:   
  
1)Class   
2)Object   
3)Inheritance   
4)Polymorphism   
5)Abstraction   
6)Encapsulation   
-----------------------------------------------------   
Advantages:   
**1)Code reusability:**define a class and n number of objects implement the class   
logic: example: Button class and n number of Button objects   
  
  
**2)Inheritance** : Eliminates redundant code and extend the use of existing classes.   
  
example: 1)we can create our own TextBox by inheriting from the TextBox class.   
2)We can inherit one Windows Frorm into another.   
  
**3)Encapsulation:** The programmer can hide the data and functions in a class from other classes.It is accomplished through modifiers like private, protected,   
protected internal.   
  
 **4)Easy to Maintain and Upgrade:**   
If we want to make changes in a class, we can make them and save the changes in the .dll This .dll can easily be updated in the client by using Update Reference.   
  
**5)Polymorphism** provides us with methods extensibility.   
we can have different methods with same name, but with different kinds of behavior   
  
ex:   
Console.WriteLine("welcome");   
Console.WriteLine(100);   
  
 **6)Abstraction** allows us to define a common definition of a base class   
that multiple derived classes can share.   
For example,   
create an abstract class bank with simple interest function   
This function will be implemented in the derived classes of   
the bank class.   
ex: 1)icici class will have its own simple interest.   
2)ABC class will have its own simple interest.   
  
icici and ABC both are child classes of bank class.

[Can we have Sealed Method in abstarct class ?](http://www.dotnetfunda.com/interviews/show/4013/can-we-have-sealed-method-in-abstarct-class" \o "Can we have Sealed Method in abstarct class ?" \t "_blank)

Looking at first site the The Keyword Sealed & Abstract are contradictory to each other..In simple terms we can Say Answer is NO...   
  
Look the code below   
  
using System;   
  
abstract class A   
{   
public abstract void Hello();   
public sealed void Hi();   
}   
  
when we will complie the code.. we will get the Compile time Error as below   
  
'A.Hi()' cannot be sealed because it is not an override..   
  
But the Crux is We can have Sealed methods in abstract class when the abstract class is Dervided class .. for Eg.   
  
using System;   
  
class A   
{   
public virtual void Hello()   
{   
Console.WriteLine(" Say Hello");   
}   
  
}   
  
abstract class B : A   
{   
public sealed override void Hello()   
{   
Console.WriteLine(" Say Hi");   
}   
  
}   
  
class C : B   
{   
  
}   
  
  
class Demo   
{   
public static void Main()   
{   
C c1 = new C();   
c1.Hello();// Output is Say Hi   
}   
}   
  
  
// Thanks

[Can we have an Abstract class without having any abstract method ??](http://www.dotnetfunda.com/interviews/show/4014/can-we-have-an-abstract-class-without-having-any-abstract-method" \o "Can we have an Abstract class without having any abstract method ??" \t "_blank)

Yes we can have Abstract class without having any abstract method ..   
  
See the code below   
using System;   
  
abstract class A   
{   
public void Hello()   
{   
Console.WriteLine(" Say Hi");   
}   
}   
  
class B:A   
{   
}   
  
class Demo   
{   
public static void Main()   
{   
B b1 = new B();   
b1.Hello();   
}   
}   
// Output is Say HI   
  
  
the class A is abstract class.. but it does not have any abstract methods..   
  
Thanks

[Can we have Multiple Main Methods in one .cs file](http://www.dotnetfunda.com/interviews/show/4027/can-we-have-multiple-main-methods-in-one-cs-file" \o "Can we have Multiple Main Methods in one .cs file" \t "_blank)

Yes we can Have multiple Main methods in one .cs file.   
The crux is we can have Multiple classes in one .cs file; and we can define one Main method in each class.   
  
& while doing compliation we can spcify the compiler to choose the Main methods from the specific class .   
  
for ef see the code below 

using System;

class Test

{

public static void Main()

{

Console.WriteLine("Test");

}

}

class Demo

{

public static void Main()

{

Console.WriteLine("Demo");

}

}

We have got two class which we can save in single .cs file say Hello.cs   
  
while doing compliation we can say 

csc Hello.cs /main:Demo --> In order to choose Main from the Demo class

and

csc Hello.cs /main:Test --> In order to choose Main from the Test class

Happy coding..

[Which of these terms defines the hiding of an object's details from the other program?](http://www.dotnetfunda.com/interviews/show/4070/which-of-these-terms-defines-the-hiding-of-an-object39s-details-from-t" \o )

NOTE: This is objective type question, Please click question title for correct answer.

[If the Function has same parameter but different return type (int and float), Is it a overloading?](http://www.dotnetfunda.com/interviews/show/4261/if-the-function-has-same-parameter-but-different-return-type-int-and-f" \o "If the Function has same parameter but different return type (int and float), Is it a overloading?" \t "_blank)

NOTE: This is objective type question, Please click question title for correct answer.

[How does Composition mechanism works ?](http://www.dotnetfunda.com/interviews/show/4479/how-does-composition-mechanism-works" \o "How does Composition mechanism works ?" \t "_blank)

This mechanism helps to simplify a complex problem into an easier problem.It generally makes different classes and objects to communicate with each other and thus making the problem solved. It communicates with the problem by making different classes and objects to send a message to each other.

[What is the advantage of parametric polymorphism ?](http://www.dotnetfunda.com/interviews/show/4480/what-is-the-advantage-of-parametric-polymorphism" \o "What is the advantage of parametric polymorphism ?" \t "_blank)

Generally in Parametric polymorphism the code is written without following any specification for the type of data present so this particular code can be used any number of times. Hence code re-usability is achieved.

[When We will create architecture and show one application connect with different type of database like SQL or Oracle. Which option is suitable for create architecture?](http://www.dotnetfunda.com/interviews/show/4497/when-we-will-create-architecture-and-show-one-application-connect-with" \o )

NOTE: This is objective type question, Please click question title for correct answer.

[What is difference in between abstrct classes and interfaces ?](http://www.dotnetfunda.com/interviews/show/4537/what-is-difference-in-between-abstrct-classes-and-interfaces" \o "What is difference in between abstrct classes and interfaces ?" \t "_blank)

An interface offers an alternative to an abstract class for creating contract among classes and their client. The main difference in between abstract class and interface are given bellow   
1. Abstract classes can have concrete methods while interfaces have no methods implemented.   
  
2.Interface do not come in inheriting chain,while abstract classes come in inheritance .

[Overloading is Static Polymorphism and Overriding is Dynamic Polymorphism ? True or False ?](http://www.dotnetfunda.com/interviews/show/4588/overloading-is-static-polymorphism-and-overriding-is-dynamic-polymorph" \o )

NOTE: This is objective type question, Please click question title for correct answer.

[Which of the following is not a part of OOPs?](http://www.dotnetfunda.com/interviews/show/4964/which-of-the-following-is-not-a-part-of-oops" \o "Which of the following is not a part of OOPs?" \t "_blank)

NOTE: This is objective type question, Please click question title for correct answer.

[What is the default access modifier of a class?](http://www.dotnetfunda.com/interviews/show/5309/what-is-the-default-access-modifier-of-a-class" \o "What is the default access modifier of a class?" \t "_blank)

The default access modifier for a class is internal if it's defined within the same namespace. It is private if it's defined within another class.   
  
It can declare members (methods etc) with following access modifiers:   
public   
internal   
private   
protected internal

[Can a constructors that is declared within a base class, inherited by subclasses ? Yes or No](http://www.dotnetfunda.com/interviews/show/5365/can-a-constructors-that-is-declared-within-a-base-class-inherited-by-s" \o )

NOTE: This is objective type question, Please click question title for correct answer.

[What is Abstract Class and interfaces?](http://www.dotnetfunda.com/interviews/show/6451/what-is-abstract-class-and-interfaces" \o "What is Abstract Class and interfaces?" \t "_blank)

When we talk about abstract classes we are **defining characteristics of an object type** , specifying what an object is.   
But in the case of an interface we **define a capability and we bond to provide that capability** , we are talking about establishing a contract about what the object can do.

[What output will be come from the code: private void ShowMessage() { try { MessageBox.Show("Try block."); return; } catch { MessageBox.Show("Execption."); } finally { MessageBox.Show("Finally."); } }](http://www.dotnetfunda.com/interviews/show/6471/what-output-will-be-come-from-the-code-private-void-showmessage-try-me" \o )

NOTE: This is objective type question, Please click question title for correct answer.

[How to Create object For Inner Class? or how to access inner class methods?](http://www.dotnetfunda.com/interviews/show/6530/how-to-create-object-for-inner-class-or-how-to-access-inner-class-meth" \o "How to Create object For Inner Class? or how to access inner class methods?" \t "_blank)

class A   
{   
public string DisplayName()   
{   
  
return "This is Class A";   
}   
  
  
public class B   
{   
public string DisplayName()   
{   
  
return "This is Class B";   
}   
}   
}   
  
  
-------------------------------------------------------   
  
var obja=new A.B();   
string txt = obja.DisplayName();   
Response.Write(txt);

[Difference between Encapsulation and Data hiding?](http://www.dotnetfunda.com/interviews/show/6552/difference-between-encapsulation-and-data-hiding" \o "Difference between Encapsulation and Data hiding?" \t "_blank)

Data Encapsulation is nothing but grouping up of related members (variables and methods) into a single unit called class   
  
Data Hiding is nothing but restricting outside access of a class members using access modifiers such as private , protected and internal etc.,

[When to use Interface?](http://www.dotnetfunda.com/interviews/show/6746/when-to-use-interface" \o "When to use Interface?" \t "_blank)

**Interface General Explanation**   
  
we can implement multiple inheritance with help of Interface and it will contain signature of methods or events no implementation will be there.   
  
**When to Use it**   
  
if we want to Implement same signature in different class then we can use it.   
consider a company where there are staffs and supporting staffs. All will be given salary but staff salary calculation is different from supporting staff salary calculation.   
  
here is how it is implemented 

public interface ISalary

{

decimal CalculateSalary();

}

public class Staff : ISalary

{

public decimal CalculateSalary()

{

//Staff salary calcuation logic

}

}

public class SupportingStaffs :ISalary

{

public decimal CalculateSalary()

{

//Supporting Staff salary calcuation logic

}

}

[Function Overloading is also called as?](http://www.dotnetfunda.com/interviews/show/7437/function-overloading-is-also-called-as" \o "Function Overloading is also called as?" \t "_blank)

NOTE: This is objective type question, Please click question title for correct answer.

[Function Overriding is also called as?](http://www.dotnetfunda.com/interviews/show/7438/function-overriding-is-also-called-as" \o "Function Overriding is also called as?" \t "_blank)

NOTE: This is objective type question, Please click question title for correct answer.

[Same Function Name but Different Signature or Parameter is called as?](http://www.dotnetfunda.com/interviews/show/7439/same-function-name-but-different-signature-or-parameter-is-called-as" \o "Same Function Name but Different Signature or Parameter is called as?" \t "_blank)

NOTE: This is objective type question, Please click question title for correct answer.

[Can we define Method Overloading in different classes?](http://www.dotnetfunda.com/interviews/show/7440/can-we-define-method-overloading-in-different-classes" \o "Can we define Method Overloading in different classes?" \t "_blank)

No,we can not define Method Overloading in different classes.It must be on the same class.   
As we know that Function Overloading is defined in the same class.   
  
We can understand it by an example:-   
  
Below function are an example of Overloaded Constructor,which are defined on the same class.   
  
public class Person   
{   
public Person()   
{   
}   
  
public Person(int person\_id)   
{   
}   
}   
  
If we define same function in different classes,so they will not be called as function overloading.

[Can Function Overriding be defined in Same Class?](http://www.dotnetfunda.com/interviews/show/7441/can-function-overriding-be-defined-in-same-class" \o "Can Function Overriding be defined in Same Class?" \t "_blank)

No,Function Overriding can only be defined in different class because it treated as Base and Derived or parent-child relationship.   
  
We can understand it by an example:- 

public class Base\_Class

{

public Base\_Class()

{

}

public virtual void display()

{

Console.WriteLine("base class");

}

}

public class Derive\_Class:Base\_Class

{

public Derive\_Class()

{

}

public override void display()

{

Console.WriteLine("derive class");

}

}

Note:- The Override keyword is used in Function Overriding to override methods.

[Below C# function is of which type? public static Calculate operator +(Calculate c1,Calculate c2) { }](http://www.dotnetfunda.com/interviews/show/7442/below-csharp-function-is-of-which-type-public-static-calculate-operato" \o )

NOTE: This is objective type question, Please click question title for correct answer.

[Does Function Overloading depend on Return Type?](http://www.dotnetfunda.com/interviews/show/7444/does-function-overloading-depend-on-return-type" \o "Does Function Overloading depend on Return Type?" \t "_blank)

No,It does not depend on Return Type.Because if return type is different and function name as well as parameter is also same.   
Then it will give compile time error.   
  
Below example will give us clear picture:- 

public void display(int id)

{

}

public int display(int id)

{

}

Once we compile above example then it will give compile-time error as   
  
Type '\_Default' already defines a member called 'display' with the same parameter types   
  
Here \_Default is a Page Name.

[What are the points which fulfill Function Overloading?](http://www.dotnetfunda.com/interviews/show/7445/what-are-the-points-which-fulfill-function-overloading" \o "What are the points which fulfill Function Overloading?" \t "_blank)

Following are the key points which fulfill Function Overloading?   
  
1). Number of parameters.   
2). Order of parameters.   
3). Data-type of parameters.   
  
We can understand this by an example:- 

public void display(int id,string name)

{

}

public void display(string name,int id)

{

}

public void display(string name,string value,int id)

{

}

public void display(string id,string name)

{

}

[What will happen,if we compile below function? private double sum(int a, int b) { } private double sum(int i, int j) { } Note:- In above both function return type is same as Double.](http://www.dotnetfunda.com/interviews/show/7446/what-will-happenif-we-compile-below-function-private-double-sumint-a-i" \o )

If we compile above function,then it will throw an error saying that:- 

Type '\_Default' already defines a member called 'sum' with the same parameter types.

Here \_Default is a page name.

[What will happen,if we compile below function:- private double sum(int a, int b) { } private int sum(int i, int j) { } Note:- In above function return type is different. 1st function return type is double and, 2nd function return type is int](http://www.dotnetfunda.com/interviews/show/7447/what-will-happenif-we-compile-below-function-private-double-sumint-a-i" \o )

It will throw an error saying that:-   
  
"Type '\_Default' already defines a member called 'sum' with the same parameter types"   
  
Here \_Default is a page name.

[What will happen,if we compile below function? private double sum(int a, int b) { } private int sum(int i,int j,int k) { } Note:- In above both function return type is different but no of parameters are also different.](http://www.dotnetfunda.com/interviews/show/7448/what-will-happenif-we-compile-below-function-private-double-sumint-a-i" \o )

It will not compile,will give us an error saying that:- 

"Type '\_Default' already defines a member called 'sum' with the same parameter types"

Here \_Default is a page name.   
  
We can not say that it's a Function Overloading.Because it does not depend upon return type.

[What is the best example of Function Overloading?](http://www.dotnetfunda.com/interviews/show/7449/what-is-the-best-example-of-function-overloading" \o "What is the best example of Function Overloading?" \t "_blank)

Overloaded Constructor is the best example of Function Overloading.   
  
For ex:-   
  
In VB.Net 

Public Class Person

Sub New()

End Sub

Sub New(ByVal person\_id As Integer)

End Sub

End Class

In C# 

public class Person

{

public Person()

{

}

public Person(int person\_id)

{

}

}

[What will happen,if we compile below code public interface intf\_person { public int save\_data(Person obj); public DataSet load\_data(); }](http://www.dotnetfunda.com/interviews/show/7452/what-will-happenif-we-compile-below-code-public-interface-intfperson-p" \o )

It will not compile.As we can see,we have taken 1 interface intf\_person, inside interface,we defined 2 method with public access modifiers.   
  
If we compile,then it will give compile error as   
  
"The modifier 'public' is not valid for this item"   
Meaning that we can not write public access modifiers inside any interface.Because all the methods inside Interfaces are Public By-default.So we do not have to write Public keyword inside Interface.

[What is final class related to OOP's](http://www.dotnetfunda.com/interviews/show/8087/what-is-final-class-related-to-oop39s" \o "What is final class related to OOP's" \t "_blank)

A class that is defined as final class can not be inherited further. All Methods of a final class are inherently final and must not be declared as final in the class definition.   
In java there is keyword “final”,but In c# there is no keyword like “final” but the same thing is achieved by keyword “sealed”.   
  
namespace OOPS\_Concept   
{   
sealed class SealedClassDemo   
{   
public void test1()   
{   
Console.WriteLine("Method Test1");   
}   
}   
  
class childclass : SealedClassDemo   
{   
}   
}   
Above code will generate an error saying that class cannot be inherited.   
  
In C#, Methods cannot be “sealed” directly. Methods of only derived class can be made sealed with keyword sealed and override.   
  
namespace OOPS\_Concept   
{   
class SealedMethodDemo   
{   
public virtual void Method1()   
{   
Console.Write("Base class Method1");   
}   
}   
  
class ChildClass : SealedMethodDemo   
{   
public sealed override void Method1()   
{   
Console.Write("Derived class Method1");   
}   
}   
}

[How can you define polymorphism?](http://www.dotnetfunda.com/interviews/show/8233/how-can-you-define-polymorphism" \o "How can you define polymorphism?" \t "_blank)

The word polymorphism is meant for different forms.   
It means the ability of an entity to exhibit different forms at runtime, also allows methods and properties in multiple classes to have the same signature with different behavoiur.

[Explain Encapsulation?](http://www.dotnetfunda.com/interviews/show/8235/explain-encapsulation" \o "Explain Encapsulation?" \t "_blank)

It is an approach to implement abstraction.   
Abstraction is the process of hiding complexity or unwanted details. Encapsulation helps to achieve data hiding through which data security is achieved.

[Explain the concept Data Hiding?](http://www.dotnetfunda.com/interviews/show/8236/explain-the-concept-data-hiding" \o "Explain the concept Data Hiding?" \t "_blank)

A class contains private data members which are not accessible to the users of other class.   
these data members can be modified only internally by the class.   
Class contains public functions through which user can interact with these data which implies that direct manuplation of data is not possible by the user.

[How to declare an array?](http://www.dotnetfunda.com/interviews/show/8237/how-to-declare-an-array" \o "How to declare an array?" \t "_blank)

A array can be created by using the 'new' keyword.   
**Syntax:**   
arrayRefVar = new dataType[arraySize];   
It creates an array using new dataType[arraySize].   
It assigns the reference of the newly created array to the variable arrayRefVar.   
  
Declaring an array variable, creating an array, and assigning the reference of the array to the variable can be combined in one statement, as shown below:   
  
dataType[] arrayRefVar = new dataType[arraySize];   
 **Alternatively way to create arrays:**   
dataType[] arrayRefVar = {value0, value1, ..., valuek};   
  
The array elements are accessed through the index.

[what is the use of private constructor?](http://www.dotnetfunda.com/interviews/show/8925/what-is-the-use-of-private-constructor" \o "what is the use of private constructor?" \t "_blank)

When we declare a class constructor as private , we can not do 2 things:-   
  
-- We can not create a object of the class from out side of the class.   
-- We can not inherit the class.   
  
Private constructor is used for controlling the object generation which Is singleton design pattern implementation.. 

public class Singleton

{

private static Singleton instance = new Singleton();

//Private constructor, which does not allow object creation

//from outside the classs..

private Singleton() { }

public static Singleton GetInstance

{

get

{

return instance;

}

}

}

[Can an Abstract class have a constructor?](http://www.dotnetfunda.com/interviews/show/9075/can-an-abstract-class-have-a-constructor" \o "Can an Abstract class have a constructor?" \t "_blank)

**Yes**,an **Abstract Class** can have a constructor.In general,a class constructor is used to initialize fields or variables.An abstract class constructor is used to initialize fields of the abstract class.We would provide a constructor for an abstract class if we want to initialize certain fields of the abstract class before the instantiation of a child-class takes place. An abstract class constructor can also be used to execute code that is relevant for every child class.This prevents duplicate code.

[We cannot create an instance of an abstract class.So,what is the use of a constructor in an abstract class?](http://www.dotnetfunda.com/interviews/show/9076/we-cannot-create-an-instance-of-an-abstract-classsowhat-is-the-use-of" \o "We cannot create an instance of an abstract class.So,what is the use of a constructor in an abstract class?" \t "_blank)

Though we cannot create an instance of an abstract class,we can create instances of the classes that are derived from the abstract class.So,when an instance of derived class is created,the parent abstract class constructor is automatically called.   
  
**Note:**Abstract classes can not be directly instantiated.The abstract class constructor gets executed through a derived class.So,it is a good practice to use protected access modifier with abstract class constructor.Using public does not make sense.

[Which of the class we can inherit?](http://www.dotnetfunda.com/interviews/show/9082/which-of-the-class-we-can-inherit" \o "Which of the class we can inherit?" \t "_blank)

NOTE: This is objective type question, Please click question title for correct answer.

[Which class prevents inheritance?](http://www.dotnetfunda.com/interviews/show/9083/which-class-prevents-inheritance" \o "Which class prevents inheritance?" \t "_blank)

NOTE: This is objective type question, Please click question title for correct answer.

[Which way is used to convert Value-Type to Reference-Type?](http://www.dotnetfunda.com/interviews/show/9092/which-way-is-used-to-convert-value-type-to-reference-type" \o "Which way is used to convert Value-Type to Reference-Type?" \t "_blank)

NOTE: This is objective type question, Please click question title for correct answer.

[Which way is used to convert Reference-Type to Value-Type?](http://www.dotnetfunda.com/interviews/show/9093/which-way-is-used-to-convert-reference-type-to-value-type" \o "Which way is used to convert Reference-Type to Value-Type?" \t "_blank)

NOTE: This is objective type question, Please click question title for correct answer.

[Which of the following statement(s) is true concerning Static Classes?](http://www.dotnetfunda.com/interviews/show/9418/which-of-the-following-statements-is-true-concerning-static-classes" \o "Which of the following statement(s) is true concerning Static Classes?" \t "_blank)

NOTE: This is objective type question, Please click question title for correct answer.

[What is the equivalent of static class in VB.Net?](http://www.dotnetfunda.com/interviews/show/9419/what-is-the-equivalent-of-static-class-in-vbnet" \o "What is the equivalent of static class in VB.Net?" \t "_blank)

NOTE: This is objective type question, Please click question title for correct answer.

[Which statement is true about Sealed Classes?](http://www.dotnetfunda.com/interviews/show/9420/which-statement-is-true-about-sealed-classes" \o "Which statement is true about Sealed Classes?" \t "_blank)

NOTE: This is objective type question, Please click question title for correct answer.

[What is the syntax of Sealed Class in C#?](http://www.dotnetfunda.com/interviews/show/9421/what-is-the-syntax-of-sealed-class-in-csharp" \o "What is the syntax of Sealed Class in C#?" \t "_blank)

Sealed class is declared with **Sealed keyword**.   
**Syntax:-** 

sealed class credit

{

public int Add(int a,int b)

{

return a + b;

}

}

As sealed class do not need to be derived means if we declare any class as Sealed class,then no body can enforce to inherit.If we take an example of Credit card,then it's information need not be shared by any one.So we can make such class as sealed.

[What is the equivalent of Sealed Class in VB.Net?](http://www.dotnetfunda.com/interviews/show/9422/what-is-the-equivalent-of-sealed-class-in-vbnet" \o "What is the equivalent of Sealed Class in VB.Net?" \t "_blank)

**NotInheritable keyword** is an equivalent of **Sealed Class** .   
**For Ex:-** 

Public NotInheritable Class Credit

Private Sub New()

'Constructor Initialization

End Sub

End Class

**For Example:-** 

NotInheritable Class S1

'do your code

End Class

Class S2

Inherits S1

**When derive a class from S1 then will get below error:-   
S2 cannot onherit from class 'S1' because 'S1' is declared 'NotInheritable'**

[What do we mean by partial class in OOP?](http://www.dotnetfunda.com/interviews/show/9423/what-do-we-mean-by-partial-class-in-oop" \o "What do we mean by partial class in OOP?" \t "_blank)

A **partial class** is a Class and is defined in 2 or more classes that are split the file into multiple files.And developers can use these files to work separately and later it is compiled to make a single to.   
  
**For Example:-** 

public partial class Employee

{

}

[Which statement is correct about Partial Classes?](http://www.dotnetfunda.com/interviews/show/9424/which-statement-is-correct-about-partial-classes" \o "Which statement is correct about Partial Classes?" \t "_blank)

NOTE: This is objective type question, Please click question title for correct answer.

[What are the advantages of partial classes?](http://www.dotnetfunda.com/interviews/show/9425/what-are-the-advantages-of-partial-classes" \o "What are the advantages of partial classes?" \t "_blank)

**Following are the advantages of Partial classes:-**   
**1).** We can separate User Interface part code and business logic code so that it is easy to read and understand by anybody.   
**2).** More than one developers can work at a time.   
**3).** As Partial classes can be separated then it can again be reunited of compilation.

[How can we identify Partial Class in C#?](http://www.dotnetfunda.com/interviews/show/9426/how-can-we-identify-partial-class-in-csharp" \o "How can we identify Partial Class in C#?" \t "_blank)

When we create any application in Dot Net and add any new web form then 2 files are generated which is named as **aspx.cs/aspx.vb** and **aspx.designer.cs/vb**.These two files have the same class with the **partial keyword** .The **.aspx.cs/vb**class is used for writing business logic code where **aspx.designer.cs/vb** file is used for having asp.net controls.

[Which statement is/are correct about Interface?](http://www.dotnetfunda.com/interviews/show/9539/which-statement-isare-correct-about-interface" \o "Which statement is/are correct about Interface?" \t "_blank)

NOTE: This is objective type question, Please click question title for correct answer.

[Which statement is correct about Abstract classes?](http://www.dotnetfunda.com/interviews/show/9540/which-statement-is-correct-about-abstract-classes" \o "Which statement is correct about Abstract classes?" \t "_blank)

NOTE: This is objective type question, Please click question title for correct answer.

[Which statement is correct about Method Overloading?](http://www.dotnetfunda.com/interviews/show/9541/which-statement-is-correct-about-method-overloading" \o "Which statement is correct about Method Overloading?" \t "_blank)

NOTE: This is objective type question, Please click question title for correct answer.

[Which statement is correct about Method Overriding?](http://www.dotnetfunda.com/interviews/show/9542/which-statement-is-correct-about-method-overriding" \o "Which statement is correct about Method Overriding?" \t "_blank)

NOTE: This is objective type question, Please click question title for correct answer.

[What are the rules about Function Overloading?](http://www.dotnetfunda.com/interviews/show/9564/what-are-the-rules-about-function-overloading" \o "What are the rules about Function Overloading?" \t "_blank)

NOTE: This is objective type question, Please click question title for correct answer.

[What are the rules about Function Overriding?](http://www.dotnetfunda.com/interviews/show/9565/what-are-the-rules-about-function-overriding" \o "What are the rules about Function Overriding?" \t "_blank)

NOTE: This is objective type question, Please click question title for correct answer.

[Which statement is correct about Operator Overloading?](http://www.dotnetfunda.com/interviews/show/9569/which-statement-is-correct-about-operator-overloading" \o "Which statement is correct about Operator Overloading?" \t "_blank)

NOTE: This is objective type question, Please click question title for correct answer.

[An interface has got four Interface methods (m1,m2,m3,m4) and implement these interface methods as follows. Implement m1() & m2() in base class Implement m3() & m4() in derived class](http://www.dotnetfunda.com/interviews/show/9647/an-interface-has-got-four-interface-methods-m1m2m3m4-and-implement-the" \o )

**Declare an interface class** 

interface myInterface

{

void m1();

void m2();

void m3();

void m4();

}

**Declare Base class which inherits an Interface Class** 

public class mybaseClass:myInterface

{

public void m1()

{

Console.WriteLine("m1");

}

public void m2()

{

Console.WriteLine("m2");

}

public virtual void m3()

{ }

public virtual void m4()

{ }

}

**Declare a Derived class which Inherits Base class & Interface Class**

public class derivedClass : mybaseClass, myInterface

{

public override void m3()

{

Console.WriteLine("m3");

}

public override void m4()

{

Console.WriteLine("m4");

}

}

class Program

{

static void Main(string[] args)

{

derivedClass dc = new derivedClass();

dc.m1();

dc.m2();

dc.m3();

dc.m4();

Console.ReadKey();

}

}

[An two interface has got same method names, how can you access differently.](http://www.dotnetfunda.com/interviews/show/9648/an-two-interface-has-got-same-method-names-how-can-you-access-differen" \o "An two interface has got same method names, how can you access differently." \t "_blank)

**Declare two interfaces with same method names** 

interface myInterface1

{

void m1();

}

interface myInterface2

{

void m1();

}

**Implement the Interface method in Base Class** 

public class myClass : myInterface1, myInterface2

{

public void m1()

{

Console.WriteLine("m1 accessed");

}

}

**Create an instance of base class & references of interface classes** 

class Program

{

static void Main(string[] args)

{

myClass mc = new myClass();

myInterface1 mi1 = (myInterface1)mc;

myInterface2 mi2 = (myInterface2)mc;

mc.m1();

mi1.m1();

mi2.m1();

Console.ReadKey();

}

}

[Which statements are correct about Abstract classes in Dot Net?](http://www.dotnetfunda.com/interviews/show/9688/which-statements-are-correct-about-abstract-classes-in-dot-net" \o "Which statements are correct about Abstract classes in Dot Net?" \t "_blank)

NOTE: This is objective type question, Please click question title for correct answer.

[What are the different ways a method can be Overloaded in C#?](http://www.dotnetfunda.com/interviews/show/9744/what-are-the-different-ways-a-method-can-be-overloaded-in-csharp" \o "What are the different ways a method can be Overloaded in C#?" \t "_blank)

NOTE: This is objective type question, Please click question title for correct answer.

[Which class we can use when its member or method is directly called using class name?](http://www.dotnetfunda.com/interviews/show/9945/which-class-we-can-use-when-its-member-or-method-is-directly-called-us" \o "Which class we can use when its member or method is directly called using class name?" \t "_blank)

NOTE: This is objective type question, Please click question title for correct answer.

[Which class we can use when we do not have to create a constructor?](http://www.dotnetfunda.com/interviews/show/9946/which-class-we-can-use-when-we-do-not-have-to-create-a-constructor" \o "Which class we can use when we do not have to create a constructor?" \t "_blank)

NOTE: This is objective type question, Please click question title for correct answer.

[Which class we can make an object but can not inherit?](http://www.dotnetfunda.com/interviews/show/9948/which-class-we-can-make-an-object-but-can-not-inherit" \o "Which class we can make an object but can not inherit?" \t "_blank)

NOTE: This is objective type question, Please click question title for correct answer.

[Which class is called to be inherit?](http://www.dotnetfunda.com/interviews/show/9949/which-class-is-called-to-be-inherit" \o "Which class is called to be inherit?" \t "_blank)

NOTE: This is objective type question, Please click question title for correct answer.

[What is Static Constructor?](http://www.dotnetfunda.com/interviews/show/10764/what-is-static-constructor" \o "What is Static Constructor?" \t "_blank)

When constructors are used to set the value of a type's data at the time of construction, if we want the value of such static data is to be preserved regardless of how many objects of the type are created, we have to define the constructor with static keyword.  
  
Example: 

Class SimplestaticConstr

{

Static SimplestaticConstr()

{

//code inside this constructor

}

}

[What are the different categories of Inheritance?](http://www.dotnetfunda.com/interviews/show/10765/what-are-the-different-categories-of-inheritance" \o "What are the different categories of Inheritance?" \t "_blank)

Inheritance in OOP is of four types :   
  
**Single inheritance** : Contains one base class and one derived class.   
Example : 

public class Base

{

}

public class Derived: Base

{

}

**Hierarchical inheritance** : Contains one base class and multiple derived classes of the same base class.   
Example: 

public class Base

{

}

public class Derived1: Base

{

}

public class Derived2: Base

{

}

**Multilevel inheritance** : Contains a class derived from a derived class.   
Example: 

public class Base

{

}

public class Derived1: Base

{

}

public class Derived2: Derived1

{

}

**Multiple inheritance** : Contains several base classes and a derived class.   
  
Example: 

public class Base1

{

}

public class Base2

{

}

public class Derived1: Base1,Base2

{

}

[What is sealed class?](http://www.dotnetfunda.com/interviews/show/10766/what-is-sealed-class" \o "What is sealed class?" \t "_blank)

Sealed classes are those classes which can not be inherited and thus any sealed class member can not be derived in any other class. A sealed class cannot also be an abstract class.   
  
Simple Sealed Class. 

sealed class SealedClass

{

}

Simple Example Programming: 

using System;

class ClassSeal

{

static void Main(string[] args)

{

SealedClass sealedCls = new SealedClass(); //Sealed class declaration

int total = sealedCls.Multiple(14, 53);

Console.WriteLine("Total = " + total.ToString());

}

}

// Sealed class Definition

sealed class SealedClass

{

public int Multiple(int x, int y)

{

return x \* y;

}

}