1. **What is .Net framework?**

.NET is network enabled technology which provides the facility to develop, run and deploy the .net based applications. It also provide the interoperability between different platforms.

1. **Different Versions of .Net framework?**
2. **What is latest .Net framework and Vs available?**

4.7.1

1. **What is OOPS? Important OOPS concepts?**

Object Oriented Programming Structure. It is a set of rules.

1. **What is Class and how to declare the class?**

Class is an object

1. **What is object and how to declare the Object?**

Object is basically an instance of class. It allocated the memory to access the members of class.

Ex : ClassName ObjectName = new ClassName();

1. **What is Access modifier in .Net?**

Public, Private, Internal, Protected Internal, Protected

**Lesson 2**

1. **What are naming convention for class in .net?**

Only alpha numeric c and ‘\_’ allowed. Class should not start with number.

1. **Can we create class name with numbers?**

Yes. But Number should not be first letter

1. **Can we use special character to create the class?**

No only ‘\_’ allowed

1. **By default class follows which access modifier?**

Internal

1. **Types of Access modifiers in C#**

|  |  |
| --- | --- |
| Public | Accessible in any project |
| Private | Members can be accessed in same class only |
| Internal | Accessible in same project, same library, same assembly only |
| Protected | Accessible in child/derived class only |
| Protected Internal | Works like protected and internal |

**Lesson 3**

1. **Can we declare class as private/protected/protected internal?**

No

1. **By default class members are, which access modifier?**

**private**

1. **Difference between Public, private and Internal access modifier?**

**Lesson 4**

1. **Difference between protected and protected internal?**

**Lesson 5**

1. **Default access modifier for class members/variables are**

Private

**Lesson 6**

1. **Is it necessary to have Main method in console application?**

Yes

1. **What error you get if main method not available in console application compile time or runtime error?**

Compile time error you get

1. **What is entry point in console application?**

Main method

1. **Is it necessary to have parameter in Main method?**

It is not mandatory

1. **Can we have Two main methods?**

No

1. **Can we move the main method to any other class in the solution?**

Yes

**Lesson 7**

1. **What is inheritance?**

Inheritance allows us to define a class in terms of another class which makes it easier to create and maintain an application. This also provides an opportunity to reuse the code functionality and speeds up implementation time.

1. **What is base/parent and child/derived class.**
2. **How to reuse inherited members in child class?**

You can call the base class members directly ex: GetBaseClassMember() or base. GetBaseClassMember()

1. **Does c# supports multiple inheritance?**

No

1. **How to achieve multiple inheritance?**

This can be achievable using Interface inheritance

**Lesson 8**

1. **Why c# does not support multiple inheritance?**

Ambiguity issue. When Two parent classes has same method child class confused which method it is using.

1. **Is circular inheritance possible ? A:B, and B:A**

No it is not possible. You can not inherit the child class to base class

1. **How do you prevent a class from being inherited?**

Sealed class are not inheritable.

1. **What is Sealed keyword?**

To prevent the class being inherited.

1. **Can we create the object of Sealed class?**

Yes we can create Instance object of sealed class but we can not inherit the sealed class.

1. **Can derive class have public modifier when there are no modifiers specified on the base class?**

No We can not inherit non public base classes.

1. **Can we mark methods as Sealed?**

No we can not make sealed methods. Reason we can not override.

**Lesson 9**

1. **what do you mean by upcasting and downcasting ?**

Upcasting : assigning a derived class object to a base class. This is implicit. BaseClass b = new BaseClass();

Downcasting : assigning base class object to derived class. This is explicit and it throws runtime error.

**BaseClass b = new BaseClass() ?** // Will Work

**BaseClass b = new ChildClass() ?** // This will work (UpCasting)

**ChildClass c = new ChildClass() ? ?** // Will Work

**ChildClass c = new BaseClass() ? ?** // Will not Work (DownCasting)

**Lesson 10**

1. **What is interface?**

Interface is skeleton of a class. It has only the definition of properties and methods but no implementation.

1. **By default interface is?**

Interface is default Internal. Interface must be public or internal. And all the methods inside the interface must be public only.

1. **Can we have access modifier in interface?**

We cannot give any Access modifier in Interface. All methods in interface are public only.

1. **Can we have variables in interface?**

No. Interface cannot have any variables.

1. **Can we create instance of interface?**

No, We can not create instance of Interface.

1. **Can we create constructor in interface?**

No, Interface cannot have Constructor. Reason it has no implementation.

1. **Can we declare properties in interface?**

Yes, Properties can be declare in Interface. But same method names will give conflicts..

1. **Can interface inherit another interface?**

Yes, It can inherit from another interface.

1. **Can interface inherit class?**

No

1. **How to achieve multiple inheritance through interface?**

ClassName : Interface1,Interface2

1. **If 2 interface has same method then how to implement the method?**

Explicit Implementation like InterfaceName.MemberName Ex: void InterFaceName.MethodName

{

}

1. **What is implicit implementation and when to use in interface?**

Directly implementing interface members with out using the interface name. when Class using one Interface inheritance.

1. **What is explicit implementation and when to use?**

Using InterfaceName.MemberName to implement the common name member of multilple interface inheritance.

1. **Is it compulsory to implement all methods of interface?**

Yes.

1. **How to call interface implemented methods?**

Call directly

1. **How to call explicit implemented methods of interface?**

InterfaceName.MethodName

**Lesson 11**

1. **What is Constructor? Use of constructor?**

Constructor is special method of a class which will invoke automatically whenever instance or object of class is crated. Constructor should match the class name and does not have any return types. Main use of constructor is object initialization and memory allocation of variables. Every class must have one default constructor.

1. **Different types of constructor?**
2. Default constructor
3. Parameterized constructor
4. Copy Constructor
5. Static Constructor
6. Private Constructor
7. **What is Default constructor?**

With out having any parameters called default constructor**.**

1. **What is parameterized constructor?**

A constructor with at least one parameter is called parameterized constructor.

1. **Use of parameterized constructor?**

It Initialize any variables in the class before the object been created.

1. **What is constructor overloading?**

Constructor with different parameters are called constructor overload.

1. **Is it mandatory to have default and parameterized constructor in class?**

Not necessary to have both when default constructor not exists we can not create object of the class with any parameters.

1. **If we have only parameterized constructor, can we create the object of class?**

Yes

1. **What is copy constructor?**

A parameterized constructor that contains a parameter of same class type is called as copy constructor.

1. **Purpose of copy constructor?**

Main purpose of copy constructors is to initialize new instance to the value of an existing instance.

1. **What is private constructor?**

A constructor with private access modifier is called private constructor private emp(){};

1. **Use/purpose of private constructor?**

Is it generally used in classes that contains static members only. It is also used in Singleton Design patterns. Reason when class has all members are static there is no need to create the object of the class we can call directly those members.

1. **Can we have default and private constructor both in class?**

No

1. **Can we create the object of class if we have only private constructor in class?**

No

1. **Can we inherit the class if we have only private constructor in class?**

No

1. **Can we have parameters in private constructor?**

Yes

1. **What is static constructor?**

A Constructor with static key called static constructor.

1. **Can we have access modifier in static constructor?**

No

1. **Can we pass parameters in static constructor?**

No

1. **How to call static constructor or when static constructor is called?**

When create object of the class static constructor will called like default constructor.

1. **How many times we can call static constructor?**

Static constructor will called only first time when object has created. Later the default constructor will called.

1. **When we create instance of class, which constructor is called first?**

When class has both static and default constructors. When object created for first time static constructor will called later default constructor will be called.

1. **Use/Purpose of static constructor?**

The static construct will be useful to initialize the static members of the class when object been created.

1. **Can we have static constructor only?**

Yes

1. **When we create instance of child class which constructor is first called, child or Parent?**

Base Class Constructor called first then Child Class Constructor.

1. **How can we call one constructor from another in the same class ?**

:this() we can call default constructor from another constructor in same class

Public SomeConstructor(string a) : this(){}

1. **Can we call base class parameterized constructor from child class constructor?**

Using :this(params)

// Base Class constructor call first when Base and Child has default constructors

// To call child class default constructor first use :this()

// To call child class parameter constructor first use :this(parameters)

// To call child class default constructor first use : base()

// you can not call child class parameter constructor frist using base(param)

1. **How to call explicitly base class constructor from child class?**

Using :Base() on base class constructor will call the child class constructor first

1. **if we have only parameterized constructor in base class, can we create object of child class?**
2. **Default access modifier/specifier for constructor ?**

Public

**Lesson 12**

1. **What is destructor?**

Destructor runs before when object free from memory. This method will start from ‘~’ letter. Developer has no control when to call. The CLR garbage control will make this call.

1. **Can we have access modifier in destructor?**

No

1. **Can we have parameters in destructor?**

No

1. **How many destructor we can have in one class?**

One

1. **Can we define destructor in struct data type?**

No

1. **Do we have any control when destructor will be called?**

No. Garbage collector has control

1. **Do we always need to implement the destructor?**

No

1. **When we define destructor, which method of garbage collector is called?**

Garbage collector us Dispose and Finalize methods to dispose the object. Default it use finalize method.

1. **How to call destructor explicitly?**

Using GC.Collect method.

1. **What is polymorphism?**

One name with many forms. There are two types of polymorphism

1. **Types of polymorphism?**

Overloading : Having different methods with same name and having different parameters.

Overriding : it is new implementation of base class method.

1. **What is method overloading? Example of method Overloading?**

Having same method name with different parameters.

1. **When we should use method overloading?**

We using method overloading Inheritance

1. **What are different names used for method overloading?**

Early Binding, Static, Compile Time

**Lesson 13**

1. **What is method overriding?**

An override method that provides a new implementation of a member that is inherited from a base class. The overridden base method must have same signature of the override method.

1. **How to achieve method overriding?**

Base class declare member as virtual in derived class use override keyword

1. **Is it mandatory to implement virtual method of base class in child class?**

No

1. **Can we override non-virtual method?**

No

1. **Can we use the virtual modifier with the static, abstract, private or override modifiers.**

No

1. **Can we use virtual for internal, protected and protected internal?**

Yes

1. **What are different names/terms used for overriding?**

Late Binding, Compile Time

1. **If parent class virtual method and child have same method without override keyword, Will code be compiled?**

Yes, It will work but Framework recommend to use new keywork

Public new void SameAsBaseClassMethod(){}

1. **In above case which error/warning will be shown?**

Compile time warning.

1. **What is “new” keyword in method overriding?:**

method hiding

1. **When to use “new” keyword in method overriding?**

In method definition ’ new’ keyword will hide the base class method implication in child class. Note : Base Class method name is same as Child Class and Base Class method declare as Virtual.

1. **What’s the difference between new and override keyword?**

New keyword will write its own implementation of base class virtual method. Override will do same as new. But main difference is new key will completely hide the base class method. Which mean derived class new method will be treated as individual method. It has no relation with base class method.

1. **Can we allow a class to be inherited but prevent method to be overridden?**

Method can not be declare as sealed. But you can prevent a virtual member of base class not to overridden to child to child class.

Base Class

Public virtual void BaseMethod(){}

Child Class 1

Public sealed override void BaseMethod(){}

Child Class 2 it’s a base class is Child Class 1

Public override void BaseMethod(){} // you can not call this will throw error.

1. **What is Params keyword?**

Using Params you can specify a method parameter that takes a variable number of arguments.

1. **How to achieve method overloading through Params?**

Public void somemethod(params int[]){}

1. **Order of params parameters?:**

it must be placed at last in method

1. **What is abstract class?**

Abstract class can have abstract methods and non abstract methods it behaves like interface as well as class.

1. **What is the syntax of abstract class?**

Public abstract class abstractcalss

{

Public abstract void test(){}

Public void test1(){}

}

1. **What is the abstract method?**

Abstract method only have signature no implementation.

1. **Can we declare abstract method as private? :**

**NO**

1. **Can we create instance of abstract class? :**

**No**

1. **Can abstract class inherit another class? :**

**Yes**

1. **Can abstract class inherit interface? :**

**Yes**

1. **Can abstract class have constructor? :**

**Yes**

1. **Can we declare Abstract class as static?**

**No**

1. **Can we declare Static methods in abstract class?**

**yes**

1. **Difference between Abstract class and interface?(Most imp)**