1) What is C#?

C# is a simple, modern, general purpose programming language. It is an object oriented programming language developed by Microsoft. It is a safe and managed language that is compiled by .NET framework to generate Microsoft intermediate language (machine code).

2) What is the reason behind the invention of C#?

C# is designed for Common Language Infrastructure (CLI). It contains the executable code and runtime environment that makes the users able to use various high-level languages on different computer platforms and architectures.

3) What are the main reasons to use C# language?

These are top reasons to use C# language:

Easy to learn

General purpose and object oriented programming language

Component oriented

Structured language

Can be compiled on variety of computer platforms

Produces efficient programs

Part of .net framework

4) What is the difference between public, static and void?

You can access public declared variables anywhere in the application.

Static declared variables are globally accessible without creating an instance of the class.

Void is a type modifier that specifies that the method doesn't return any value.

5) What are constructors in C#?

A constructor is a member function in the class and has the same name as its class. Whenever the object class is created, the constructor is automatically invoked. It constructs the value of data members while initializing the class.

6) What are the different types of constructors in C#?

Basically, there are five types of constructors:

Static constructor

Private constructor

Copy constructor

Default constructor

Parameterized constructor

7) What is static constructor?

Static constructor is used to initialize static data members as soon as the class is referenced first time.

8) What is method overloading in C#?

Method overloading is mechanism to create multiple methods with the same name and unique signature in the same class. When you go for compilation, the compiler uses overload resolution to determine the specific method to be invoked.

9) Is overriding of a function possible in the same class?

No

10) What is array?

Array is a set of related instances either value or reference types.

There are three types of array supported by C#:

Single Dimensional Array: It contains a single row. It is also known as vector array.

Multi Dimensional Array: It is rectangular and contains rows and columns.

Jagged Array: It also contains rows and columns but it has an irregular shape.

11) What is ArrayList?

ArrayList is a dynamic array. You can add and remove the elements from an ArrayList at runtime. In the ArrayList, elements are not automatically sorted.

12) What is a collection?

A collection works as a container for instances of other classes. All classes implement ICollection interface.

13) What is an interface?

Interface is an abstract class that has only public abstract method. These methods only have declaration not the definition. These abstract methods must be implemented in the inherited classes.

14) What is the lock statement in C#?

Lock statement is used to ensure that one thread doesn?t enter a critical section of code while another thread is in the critical section. If another thread attempts to enter a locked code it will wait, block, until the object is released.

15) What is serialization?

If you want to transport an object through network then you have to convert the object into a stream of bytes. The process of converting an object into a stream of bytes is called serialization.

16) How to declare a property in a class?

int m\_PersonID = 0;

public int PersonID

{

get { return m\_PersonID; }

set { m\_PersonID = value; }

}

17) What is the difference between early binding and late binding in C#?

Early binding and late binding are the concept of polymorphism. There are two types of polymorphism in C#.

Compile Time Polymorphism: It is also known as early binding.

Run Time Polymorphism: It is also known as late binding or method overriding or dynamic polymorphism.

18) Which are the access modifiers available in C#?

Following are the access modifiers generally used for accessibility:

Public: If you define an attribute or method as public, it can be accessed from any code of the project.

Private: A private defined attribute or method can be accessed by any code within the containing class only.

Protected: If you define the method or attribute as protected it can be accessed by any method in the inherited classes and any method within the same class.

Internal: If you define an attribute or a method as internal, it is restricted to classes within the current position assembly.

Protected internal: If you define an attribute or method as protected internal, access is restricted to classes within the current project assembly or types derived from the containing class.

19) What is the difference between abstract class and interface in C#?

Abstract class can have abstract and concrete methods whereas interface has only abstract methods.

20) What is the difference between dispose() and finalize() methods in C#?

The dispose() method is explicitly called by user to free unmanaged resources such as files, database connections etc whereas finalize() method is implicitly called by garbage collector to free unmanaged resources like files, database connections etc.

The dispose() method belongs to IDisposable interface whereas finalize() method belongs the Object class.

21) What is the difference between method overloading and method overriding in C#?

Method parameters must be different in method overloading whereas it must be same in method overriding.

Inheritance is not required in method overloading, it occurs within the same class. But inheritance is required in method overriding.

22) What is object pool in .Net?

Object pool is a container of ready to use objects. It reduces the overhead of creating new object.

23) What is delegate in C#?

A delegate in C# is an object that holds the reference to a method. It is like function pointer in C++.

24) What is Hashtable?

A Hashtable is a collection of key/value pairs. It contains values based on the key.

25) What is Reflection?

Reflection allows us to get metadata and assemblies of an object at runtime.

26) What is Garbage Collection?

Garbage Collection is a process of releasing memory automatically occupied by objects which are no longer accessible

**Difference between ref and out Keyword-**

[ref] means that the parameter has a value on it before going into the function.

[out] means that the parameter has no official value before going into the function.

**Static and instance Class members-**

Static members are invoked using class name while instance members are invoked using object.

Example-

class Test

{

int x;

static int y;

void F() {

x = 1; // Ok, same as this.x = 1

y = 1; // Ok, same as Test.y = 1

}

static void G() {

x = 1; // Error, cannot access this.x

y = 1; // Ok, same as Test.y = 1

}

static void Main() {

Test t = new Test();

t.x = 1; // Ok

t.y = 1; // Error, cannot access static member through instance

Test.x = 1; // Error, cannot access instance member through type

Test.y = 1; // Ok

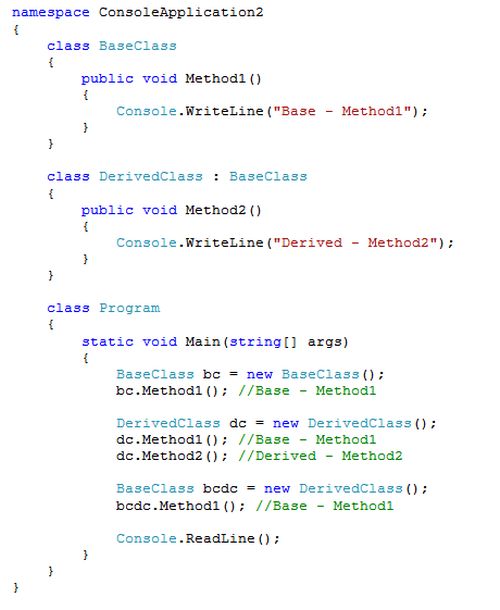
}

}

**Method overriding vs method hiding**

For hiding the base class method from derived class simply declare the derived class method with the new keyword.  
Whereas in C#, for overriding the base class method in a derived class, you need to declare the base class method as virtual and the derived class method as overriden.

New is reference-type specific, overriding is object-type specific.







**Why Properties or What is the need for encapsulation –**

In Properties, we use predefined Set or Get method.

Marking the class field public & exposing is risky, as you will not have control what gets assigned & returned.

To understand this clearly with an example lets take a student class who have an ID, pass mark, name. Now in this example some problem with a public field

The ID should not be -ve.

Name can not be set to null

A pass mark should be read-only.

If student name is missing No Name should be returned.

To remove this problem We use Get and set method.

// A simple example

public class student

{

public int ID;

public int passmark;

public string name;

public class programme

{

    public static void main()

    {

       student s1 = new student();

       s1.ID = -101; // here ID can't be -ve

       s1.Name = null ; // here Name can't be null

    }

}

Now we take an example of get and set method

public class student

{

    private int \_ID;

    private int \_passmark;

    private string\_name ;

    // for id property

   public void SetID(int ID)

   {

       if(ID<=0)

       {

         throw new exception("student ID should be greater then 0");

       }

       this.\_ID = ID;

    }

    public int getID()

    {

       return\_ID;

     }

   }

   public class programme

   {

       public static void main()

       {

         student s1 = new student ();

         s1.SetID(101);

      }

      // Like this we also can use for Name property

      public void SetName(string Name)

      {

        if(string.IsNullOrEmpty(Name))

        {

          throw new exeception("name can not be null");

        }

        this.\_Name = Name;

     }

     public string GetName()

     {

        if( string.IsNullOrEmpty(This.Name))

        {

          return "No Name";

       }

       else

      {

        return this.\_name;

      }

      // Like this we also can use for Passmark property

      public int Getpassmark()

      {

        return this.\_passmark;

      }

}

Struct vs class

