Question-1: What is Delegate?

Ans: Delegate-

1. Delegate is a type which is use to hold the information of the method.
2. Delegate is a type safe function pointer which is use to invoke the methods.
3. Delegate is a reference pointer to method.

Rules to declare Delegate:

1. Delegate can declare inside the class or outside the class.
2. Datatype of delegate and datatype of method must be same.
3. Parameters of (delegate and method) both must be same.
4. We can create object for the delegate and pass the method name as parameter for delegate object.

Steps to work with Delegate:

1. **Create a delegate-**
   Syntax- public delegate return_type delegate_name(parameter);
   Example- public delegate void My_Delegate (int x, int y);

   Note- Here My_Delegate is delegate_name & void is return_type.

2. **Create a method for delegate-**
   Syntax-
   ```
   public returnname methodname()
   {
   }
   ```
   Example-
   ```
   public void show ()
   {
   }
   ```
   Note- Here void is returntype and show is methodname.

3. **Create an object for delegate and pass methodname as parameter-**
   Syntax- delegatename object_name=new delegatename();
   Example- MyDeleagte D1= new MyDelegate();
   Note: Here MyDelegate is delegatename and D1 is objectname.

4. **Call the delegate object-**
   Whenever we invoke the delegate then delegate will invoke the method-
   Syntax- delegateobjectname();

Question-2: What is Difference between method overloading and delegate?

Ans: Differences between method overloading and delegate-

Method Overloading:

- ✓ It means- It is a process of defining multiple methods with same name but with different parameter.

Syntax-

```
class {
        public returntype methodname1(int x)
                {
                  }
        public returntype methodname1(int x, int y)
          {
              }
         }
```

Example-

```
class {
        public returntype Show (int x)
                {
                  }
        public returntype Show (int x, int y)
          {
              }
         }
```

Note- Here Show is method names which name is same in both but in first show method have only one parameter while in second method have two parameters , it's diff.

Deleagte-

- ✓ It Means, it is process of defining multiple methods with different methodname but with same parameter.

Syntax-

```
Public delegate returntype delagatename();
 Class {
        Public returntype methodname1()
             {

              }
         Public returntype methodneme2()
            {

             }
        }
```

Example-

```
Public delegate void MyDelegate();
 Class {
        Public void Print1(string name)
             {

              }
         Public void Print2(string name)
            {

             }
        }
```

Note- Here Print1 and Print 2 is different method names but in first method have only one parameter and also second method have one parameters, it's same.

Question-3: What are Anonymous Types and Anonymous Method?

Ans: Anonymous Types:

- ✓ Anonymous types means- Unknown Datatypes.
- ✓ Anonymous types are also called as implicitly typed local variables.
- ✓ Anonymous types must declare with var keyword.

Example-

```
var a=10;
var b=20;
var c= a+b;
var d= abc;
```

- ✓ Here var is frequently used in LINQ queries and foreach loop.
- ✓ Var must always declare as local variable.

Anonymous Method:

- ✓ A method without name is called as anonymous method.
- ✓ Anonymous method is used to simplify the code.
- ✓ Anonymous method is used to add a block of code to the delegate reference.

Example-

```
using System;
public delegate void MyDelagate(int x, int y);
class A
    {
      Static void Main()
        {
            MyDelegate obj= new delegate(int x,int y)
          {
            Console.WriteLine(x+y);
            };
        Obj(6,5);
       }
     }
```

Question-4: What is Collection?

Ans: Collection:

- A collection is a data structure that hold a set of object in specific manner.
- Collection is used to implement data structure in dotnet.
- Data structure is used to store data and manipulate data.
- Collection is a container objects which is used to store group of objects.
- A collection is a group of individual object represents a single unit.

What is need of storing group of objects in collection?

- To perform different types of operations on (group of objects) like-
  - **Insertion**
  - **Deletion**
  - **Updating**
  - **Searching**
  - **Sorting**
- If we want to pass the (group of objects) as parameter for a method then we use collection.
- If we want a method to return multiple objects then declare collection as return type of the method.

Collection:

- **List**
  - ✓ ArrayList
  - ✓ LinkedList
  - ✓ SortedList
  - ✓ List<>
- **Set**
  - ✓ HashSet
- **Map**
  - ✓ Dictionary
  - ✓ KeyValuePair
- **Tuple**
- **Queue**

Question-5: Define difference between Array and ArrayList?

Ans:

S.NO	Array	ArrayList
1.	Array is used to store homogeneous values	ArrayList is used to Heterogeneous values.
2.	The Size of Array is fixed.	The size of ArrayList is not fixed.
3.	In Array- we can't increase or decrease the size of Array depending on the Requirement.	In ArrayList- we can increase or decrease the size of ArrayList depending on the Requirement.
4.	We can't insert or remove the value from Array at a specific position.	We can insert or remove the value from ArrayList at a specific position.
5.	In Array – Readymade method support is not available.	In ArrayList – Readymade method support is available.
6.	In Array- Manually done searching and sorting operation.	In ArrayList- Predefine Method supports searching and sorting operation.
7.	Array is available under System namespace.	ArrayList is available under System.Collections namespace.

Note: Homogeneous- Objects of a single type**, Heterogeneous –** objects of a multiple type.

Question-6: Define difference between String and StringBuilder?

Ans:

S.NO	String	StringBuilder
1.	String is immutable which means value will not change.	String is mutable which means value will change.
2.	String is a class available under System namespace.	StringBuilder is available under System. Text namespace.
3.	String will always allocate a new memory whenever we perform concatenation operation.	StringBuilder will modify the existing memory whenever we append a string.
4.	Performance wise string is slow because every time, it will create new instance.	Performance wise string is fast because every time, it uses same instance of object to perform any action.

Question-7: What is Partial Class?

Ans: Partial Class: Declaring multiple classes with same name is called partial class at compile time. All the partial classes will become as single class.

Question-8: What is Assembly?

Ans: Assembly is the compiled format of any dotnet program which may be .dll or .exe. Assembly is collection of namespaces.

Question-9: What is Managed and Unmanaged Code?

Ans:

Managed Code-

- Manage code is the code that is executed directly by the CLR instead of the OS.
- Code compiler, first compiles the managed code to intermediate language (IL) code also called as MSIL code.
- This code does not depend on machine configuration and can be executed on different machine.

UnManaged Code-

- The code, which is developed outside of dotnet framework is known as unmanaged code.
- Unmanaged code compiles straight to machine code and directly executed by operating System.

Question-10: What is Garbage Collector?

Ans: A Garbage Collector is integral part of dotnet framework which will take care about automatic memory management.

Question-11: Explain Boxing and Unboxing?

Ans: Boxing- Boxing is a process of converting value type datatype to reference type datatype.

Unboxing- Unboxing is a process of converting reference type datatype to valuetype datatype.

Question-12: What is ArrayList?

Ans: ArrayList:

- ArrayList Class is a resizable array.
- ArrayList will maintain the data in index format.
- ArrayList will allow duplicate objects.
- ArrayList will allow null values.

Question-13: What is difference between Structure and Class?

Ans:

Difference between Structure and Class-

S.NO.	Structure	Class
1.	It must declare with "struct" keyword.	It must declare with "Class" Keyword
2.	Structure is a value type datatype.	Class is a reference type datatype.
3.	Memory is allocated on stack.	Memory is allocated on heap.
4.	It's recommended to store small amount of data.	It's recommended to store large amount of data.
5.	Structures are inherits from System.ValueType.	Classes are inherits from System.ObjectType.
6.	"new" keyword is optional to create object.	''new '' keyword is mandatory to create class.
7.	It doesn't support default constructor and destructor.	It supports default constructor and destructor.
8.	It doesn't support static constructor.	It supports static constructor.
9.	Structure cannot inherit to other type.	Class can inherit to other type.
10.	It cannot be declared as abstract.	It must be declared as static.

Question-14: What is Hash Table in C#.Net?

Ans: Hash Table:

- ✓ It will maintain data in "Key" and "Value" in pair format.
- ✓ Key must be duplicate.
- ✓ It displays the output in random manner.
- ✓ It will maintain the data in "DictionaryEntry" Format.
- ✓ This DictionaryEntry Format is predefined structure.

Syntax-

```
Struct DictionaryEntry
  {
     Public object Key {get; set;}
      Public object Key {get; set;}
   }
```

Question-15: What is Extension Method? Explain with Example.

Ans: Extension Method-

- It is a concept of adding new methods to existing class without applying Inheritance.
- While working with extension methods-
  - ✓ No need to inherit the original class.
  - ✓ No need to modify the original class.

Rules of Extension Method:

1. It should be declared in static class.
2. Method parameter should be class name with "this" keyword.

   Syntax-

   ```
   static returntype methodname (this classname objects)
           {
           }
   ```

   Example-

   ```
   static void show (this A args ) {

           }
   ```

3. It can be called by using objectname. Generally – we use the extension method in LINQ Query.

Question-16: What is difference between .dll and .exe file?

Ans: Difference between .dll and .exe file-

Sno.	.dll	.exe
1.	It is an extension for a dynamic link library.	It is an extension for executable files.
2.	.dll file van be reused by other application.	.exe file can't reused by other application.
3.	.dll does not have any entry point.	.exe file defines an entry point.
4.	.dll file depend on other .	While .exe file runs independently.
5.	.dll file share the same process and memory space of the calling application.	While .exe creates separates process and memory space.

Question-17: What is Abstract class?

Ans: Abstract Class:

- ✓ It is a class which have both abstract and non-abstract method.
- ✓ We can't create an instance for an abstract class.
- ✓ Its implementation must be provided in derived class to use it.

Syntax-

```
abstract class A
    {
    }
class B: A
    {
    }
```

Question-18: What is Abstract Method?

Ans: Abstract Method:

- ✓ A method without method of body is called abstract method.
- ✓ It is necessary to provide its implementation in derived class.
- ✓ It must be declared with "abstract" keyword.

Syntax-

```
abstract returntype methodname (list of parameters)
```

Question-19: What is An Interface?

Ans: Interface:

- ✓ Interface is a type.
- ✓ It consists of public abstract method and public static final variable.
- ✓ It is used as a contract or agreement between itself and its implemented class.

Syntax-

```
Interface InterfaceName{
    Variables;
    Methods;
}
```

Question-20: What is down casting and up casting?

Ans:

Down Casting- It means super class reference assigned to sub class reference.

Assigned to

Super class reference ----------→ Sub class reference

Up Casting- It means sub class reference assigned to super class reference.

Assigned to

Sub class reference ----------→ Super class reference

Question-21: What is Generics?

Ans: Generics-

- ✓ Generics are used to "Introduce to deal with type safe objects".
- ✓ It makes the code stable by detecting the bugs at compile time.

Advantages:

- It is general datatype.
- It's avoided unnecessary typecasting.
- It's avoided overloading.
- It's declared by place holder and type parameter.

Question-21: What is difference between Generics and Collections?

Ans: Differences between Generics and Collections-

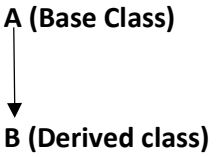
Sno	Generics	Collections
1.	When we perform same type operation then we use Generics.	When we perform any type operation then we use Collections.
2.	Generics does not support Typecasting.	Collections supports Typecasting.
3.	It does not support arithmetic operations.	It supports arithmetic operations.

Question-22: What is Inheritance? Define it's types.

Ans: Inheritance- It is a mechanism of establishing the relationship between the classes.

There are five types-

I. **Single Inheritance-** Creating a derived class(child) by using a single base class(parent) is called as Single Inheritance.



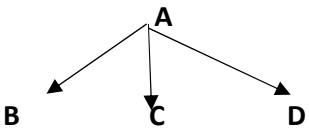
Ex- Class A { }

Class B: A { }

II. **Multi-Level Inheritance:** Creating a derived class by using another derived class is called as Multi-Level Inheritance.

Ex- class A{ }
class B: A{}
class C: B{}

III. **Hierarchical Inheritance:** Creating multiple derived class by using single base class is called as "Hierarchical Inheritance".



Ex- class A{}
class B: A: class C: A: class D: A {{}{}}

IV. **Multiple Inheritance:** Creating a derived class by using multiple base class is called as "Multiple Inheritance".

Ex- class A,B
{
}
class C: A,B { }

V. **Hybrid Inheritance:** It is a combination of multilevel and Hierarchical level Inheritance.

Question-23: Why C#.Net does not support multiple inheritance?

Ans: If same method exists in both classes so ambiguity problem will occur so C#.Net does not support multiple inheritance. It can achieve by using Interface.

Question-24: What is constructor? Define its type.

Ans: Constructor: Constructor is a default method which is used to initialize the value for instance variables at the time of creating object.

There are three types in c#.net-

- Default Constructor
- Parameterized Constructor
- Copy Constructor

Default Constructor- It is used to initialize the default value for instance variable.

Syntax- class A

```
{
  Public A(){}
    }
```

Parameterized Constructor- A constructor is parameterized constructor when it accepts a specific number of parameters.

Copy Constructor- A copy constructor is a number function that initializes an object using another object of same class.

Question-25: Which Scenario we use abstract method?

Ans: When we use an abstract class then we use a abstract method. Because When we inherits the derived class by base class means abstract class, derived class must implements the abstract method.

Question-26: Why we use "abstract" keyword in abstract class?

Ans: Abstract keyword enables you to create classes and class members that are incomplete and must be implemented in derived class.

Question-27:What is Exception Handling?

Ans: Exception Handling- Exception Handling is a mechanism of handling runtime errors.

Errors are two types-

- Runtime Error
- Compile Time Error

Runtime Error- The error that will occur at the time of execution of the program is called as Runtime error.

Compile time Error- The error that will occur at the time of compilation of the program is called compile time error.

Question-28: How to Handles the Exceptions?

Ans: We can Handles the exceptions in 3 ways-

- ✓ By using logical implementation
- ✓ By using try catch implementation
- ✓ By using application exception

Question-29: What is try-catch -finally?

Ans:

try- It is a block of code in which exception occur.

catch- It is used to display error messages.

Finally- the code that we create inside finally block will gets executed even if exception occurs.

Question-30: What are Access modifiers?

Ans: Access modifiers- it is used to specify the- "Scope of accessibility of a member of a class or type of the class itself."

Why to use access modifiers- For implementation of encapsulation.

Question-31: What is Sealed class and Sealed Method in c#.Net?

Ans:

Sealed class-

- It is used to restrict the users from inheriting the class.
- A class can be sealed by using the “sealed” keyword.
- Sealed keyword tells the compiler that the class is sealed so can’t be executed.
- No class can be derived from a sealed class.

Syntax-

```
sealed class class_name
    {
        Datamembers-------
         Methods----------
     }
```

Sealed method-

- A method can be sealed and in that case the method can not be overridden.
- However, a method can be sealed in classes, in which they have been inherited.
- If you want to declare a method as sealed, then it has to be declared as vitual in its base class.

Example-

```
using System;
    Sealed class SealedClass
        {
            public int Add (int a, int b)
                {
                    return a+b;
                        }
                            }
    Class Program
            {
            static void Main(string args [])
{
            SealedClass slc=new SealedClass();
             Int total=slc.Add(6,4);
              Console.WriteLine("Total ="+total.ToString());
    }
                    }
```

Question-32: What is CLR in c#.Net? Define its components also.

Ans:

CLR- (common language runtime):

- CLR is basic and virtual machine component of the .net framework.
- It is a runtime environment in the .net framework that runs the codes and helps in making the development programs.
- It is responsible for managing the execution of .net programs regardless of any .net programming langaugae.

Main components of CLR-

- **CLS (common language specification)**
- **CTS (common type system)**
- **Garbage Collector**
- **MSIL (Microsoft intermediate language)**
- **JIT (Just in Time compiler)**

CLS (Common Language Specification)-

- It defines a subset of common type system.
- It provides the execution support to other programming languages in .net framework.

CTS (Common Type System)-

- It provides common datatypes for all .net supportable languages.
- At compile time language datatype will convert into CTS types.

These are 2 types-

a) **Value types-** Stored--> value--->directly--> compile time
b) **Reference types-** Contains--> memory address of value allots ---> runtime

Garbage Collector-

It is used to provide the automatic memory management feature.

MSIL (Microsoft intermediate language)-

Whenever we compile any .net supportive language program then the language compiles will generate .exe file or .dll file which internally consists of MSIL code.

JIT (Just in Time compiler)-

It is responsible for converting the CIL (common intermediate language) into machine code or native code.

Types of JIT-

- **Pre-JIT compiler-**
  It compiles the complete source code into native code in a single compilation cycle.

- **Econo-JIT compiler-**
  It compiles only those methods that are called at runtime.
  However, these compiled methods are removed when they are not required.

- **Normal JIT or Standard JIT-**
  It compiles only those methods that are called at runtime.
  These methods are compiled the first time they are called and then they are stored in cache.
  When the same methods are called again, the compiled code from cache is used for execution.

Question-33: Define HashSet and Dictionary in c#.net.

Ans:

HashSet-

- It is an unordered collection of unique elements.
- It supports implementation of sets and uses the hash table for storage.
- It is a generic type collection and define under System.Collections.Generics namespace.

Points to remember-

- ✓ Elements must be unique.
- ✓ Duplicate values /elements are not allowed.
- ✓ Support only same type of element.
- ✓ When we add a new element then hash set sizes automatically increased.

Dictionary-

- It is used to store group of objects in the form of key and value pair formats.
- It can not allow the duplication of keys.
- It allows duplication of values only.

In other word-

It is a collection of keys and values where keys are like word and values are like definition. In Dictionary-we uses Tkey and Tvalue.
Tkey denotes- Type of Key and Tvalue denoted- Type of value.

Question-33: What are the differences between Constant and Read-only.

Ans: Differences between Constant and Read-only-

Sno.	Constant	Read-only
1.	const fields has to be initialized while declaration only	while readonly fields can be initialized at declaration or in the constructor.
2.	const variables can declared in methods	while readonly fields cannot be declared in methods.
3.	const fields cannot be used with static modifier	while readonly fields can be used with static modifier.
4.	A const field is a compile-time constant	the readonly field can be used for run time constants.