C# (oop concepts)

1) Inheritance (Types of Inheritance)
2) Polymorphism (Types of Polymorphism)
3) Abstraction
4) Encapsulation.

Q# Different types of Inheritance:

1. Single Inheritance
2. Multiple Inheritance
3. Multilevel Inheritance
4. Hesarical Inheritance

Q# What is Polymosphsims and what and its types.

Polymosphsim is the ability of a variable a object os function to take on Mutiple forms.

Two types of Polymosphsim:

1. Compile Time (Overloading)
2. Run Time (Overriding)

- In which we can create multiple methods of the (same name in the same class) and method work in diffesent ways.

* Compile time meaning when you will build the project then you see some built errors. Because the .Net compiler has caught them before running the program.

But some errors you will see like (object reference not set to an object) at run time because compiler is not able to catch these errors before running. These are run time errors.

* Method overloading can be done in 3 ways.

```
Public class Methodoverloading
{
Public int add (int a, int b)
{
   return a+b;
}
Public int add (int a, int b, int c)
{
   return a+b+c;
}
```

No. of Parameters are different

Q.# What is the difference b/w Overloading and Overriding?

Method overloading is a kind of compile time polymorphism, in which we can create multiple methods of the (same name in the same class) and all methods work in different ways.

Overriding:

Method overriding is creating a method in the (DERIVED) class with the (same NAME) & (signature) as a method in the base class.

```
Class Baseclass
{
  Public virtual void show()
  C.w(" Base class ");
}

class Desived class : Baseclass
{
  Public override void show()
  C.w(" Desived class ");
}
```

same uncalled name but one is in base class and another is in derived class.

class Progam.C

Baseclass obj
obj.show()
↓
This will
because

→ Overrid
base cl
keyword

→ Method
Inhe
class.

→ Meth
(Ba
poss

Class Program.cs,

Baseclass obj = new Derivedclass();
obj.show();

↓

This will call Derivedclass show() method because of overriding.

→ Overriding uses (virtual) keyword for base class method and (override) keyword for derived class method.

→ Method overloading does not need Inheritance. It is possible in same class.

→ Method overriding needs inheritance (Baseclass / Derived class). It is not possible in same class.

Q) What is the difference b/w Abstract class and an Interface:

Abstract class contains both (Declaration and Deffination) of methods.

```
Public abstract class Car
{
    // Method Declaration
    Public abstract void Engin();
    // Method Defined
    Public void Dashboard()
    {
        C.W("Dashboard");
    }
}
```

→ Abstract classs and Inteface can Only used for inheritance not for object creation.

Add (...) { (... ( ) ... }

Interface :

Interfa... of ...

Interf...
{
// m...
// de...
void ...
void ...
}

. Abstra... constan... meth...

. Inte... me...

. Abs... mult... nu...

Interface :

Interface contains (only Declaration) of methods.

Interface car
{
// method having only
// declaration not definations

Void Engine ( ); → only method declaration is allowed.
Void Dashboard ( );

}

Abstract class can contain, fields, constants, constructor, static members, methods.

Interface can contain Undefined method only nothing else.

Abstract class does not support multiple inheritance. Interface support multiple inheritance.