1. .net framework
2. CLR
3. CTS
4. JIT
5. MSIL
6. ASSEMBLY
7. MODULE
8. Metadata
9. Components of an assembly
10. Compilation and execution .NET applications
11. Assembly versioning(solves multi-version comploment deployment issue)
12. CLS(common language specification)
13. Solution Folder

-solution (.sln)

-project1 folder

-.csproj

-file1.cs

-file2.cs

-project2 folder

-.csproj

-file1.cs

-file2.cs

===========================C# language===================================

Types of variables:-

1. Value type:

Four princiles of OOPS

1)Encapsulation:

Putting related things together . we achieve by defining all the related members like fields,properties and methos inside the class.

2)Abstraction:

It is about identifying the relevant data or information of an object w.r.t domain. It is also hiding irrelevant data from the object.

3)inheritance:

This a kind-of relation among th eobjects which avoids recoding something which is already present

Constructor and destructor

a)Constructor:

* is a special method which is used for doing any initial task automatically when the object is created.
* It is called only once in the life time of the object
* Its name must be same as of class name
* Can be overloaded
* Can be public,private or static

b)Destructor

* it is also called automatically when object is destroyed
* used for resource clearing statements
* cant be overloaded
* no access specifier or modifier
* name is same as class name preceded with tilde(~) character

c)constructor chaining

* is used during inheritance where data is to be chained and pass from derived to base class constructor.
* Eg if 1 mango is needed by me and 2 by mom then I will take 3 mangoes and pass remaining to mom

**When to use private and static constructor:**

Private ctor is used to to avoid creation object of the class bt the cusumers

Static ctor is used to initialize static data.

Class employee{

Int ecode ;

String ename;

Int salary;

Static string company;

Public employee(int ec,string en,int sal)

{

This.ecode=ec;

This.ename=en;

This.salary=salary;

}

**Static Employee()**

**{**

**Company=”GEP”;**

**}**

}

Employee e1=new Employee(101,””)

**===========================Polymorphism===========================**

Is achieve using OVERRIDING of methods from base to derived

Overriding can be

1. Mandatory at derived class ----------🡪when method implementations depends on derived objects.
2. All the methods are based on derived class implementation ,We use here interface
3. Only few methods are based on derived class implementation, rest are commonly given at base level. We use here abstract methods in a abstract base class.
4. Optional overriding at derived class ->when there is a default implementations os the method is provided by base class and optionally derived classes may override based on their need. This is implemented using virtual methods.

**Note : In dynamic polymorphism reference is given to the type of object and not the content.**

**====================Virtual Methods**

1. Virtual methods gives permission to optionally override a method at derived class.
2. We use override keyword at derived level.
3. Interface and abstract class cant be instantiated.
4. Multiple interface can be implemented in a class but only one abstract class can be inherited.

======================Sealed class

1. To avoid a class from getting inherited , we can declare it as sealed.
2. We can have sealed methods or sealed class
3. Sealed methods are not meant for over

Method overloading

Method overriding

Method Hiding

======================Access specifier

1. **Private** :these members are only accessible inside its class
2. **Protected :these members can be accessed upon derived classes but not outside**
3. **Public :unlimited access, anywhere it is accessible inside and outside the class and even outside the current project**
4. **Internal: it is like public inside the project but private for outside the project i.e. it is not exposed to external assemblies**
5. **Protected internal: it is like public inside the current project and becomes protected outside the**

===================partial class methods

Partial keyword gives us facility to define a class at multiple places and even in multiple files provided name space is same

Usage:

To categorize and group the related methods in different places of thr same class definition.

==================Error handling in js========================

Types of errors

1. **Syntax error**

Occurs when we use undefined keywords in the program or improper use of grammar of the language. Developer has to rectify. It is reported during compilation itself. So it is called compile time error

1. **Logical error:**

Occurs when the desired output is not there for inputs.it is run time errors and developers needs to find the error and change the logic.

In visual studio debugging options we can use to find logical error:

* Break points(F9)
* Watch windows
* Immediate windows -----------analyzing the what if-else expression
* Local windows

Line-BY-Line Debugging-🡪F10—step over

Step-into->F11-> to enter the child method in debugging mode

Step-out----🡪shift+F11-🡪 no further debugging and executes normally

**3)Exception:**

* These are the errors due to some external situation and program behaves abnormal and terminates sometimes.
* It is a runtime errors
* When an exception occurs , CLR wraps the error details like line number ,methods,class,module,reason of the errors etc called stack trace into an exception object and throws it out of the application.
* Application needs to handle it using try/catch/finally else it terminates
* Every exception has a base class system. Exception
* We can have multiple catch blocks per try block
* Only one finally block per try block

===============================Custom or User-defined Exception=========================

Steps:

1. Define the exception class for the situation.
2. Throw the exception when it is logically needed.
3. Handle the exception once thrown -----------throw new<<CustomerException Class>>

Inner Exception:When an exception is being handled then there may be the chance another exception

We should carry the main exception and inner exception along so that the main error should not be lost.