Introduction to .NET

* Microsoft .NET is a platform that is used to develop the computer software and application
* Developed by Microsoft corporation in 2002.
* Latest version is ".NET Framework 4.8", released in 18 April 2019.
* Used to develop console, windows, web, mobile, web services based application.
* .NET supports multiple languages like as Visual C++, VB.NET, Visual C#, Visual J++, Python, Pascal, Perl and many more.....

History of C#

* Developed by Microsoft in 2000.
* Based on Java and C++ but has many additional extensions.
* Java and C# both are being updated to keep up with each other.
* Cross-development with visual basic, visual C++, and many more .NET languages

Why to use C#

* Pure object oriented language.
* Power of C with ease of Microsoft visual Basic
* Easy to learn for everybody
* Much cleaner than C++

* Provides more support for web development
* More powerful like java.
* Latest version is C#10, that is released in 8th November 2022.

C# overview

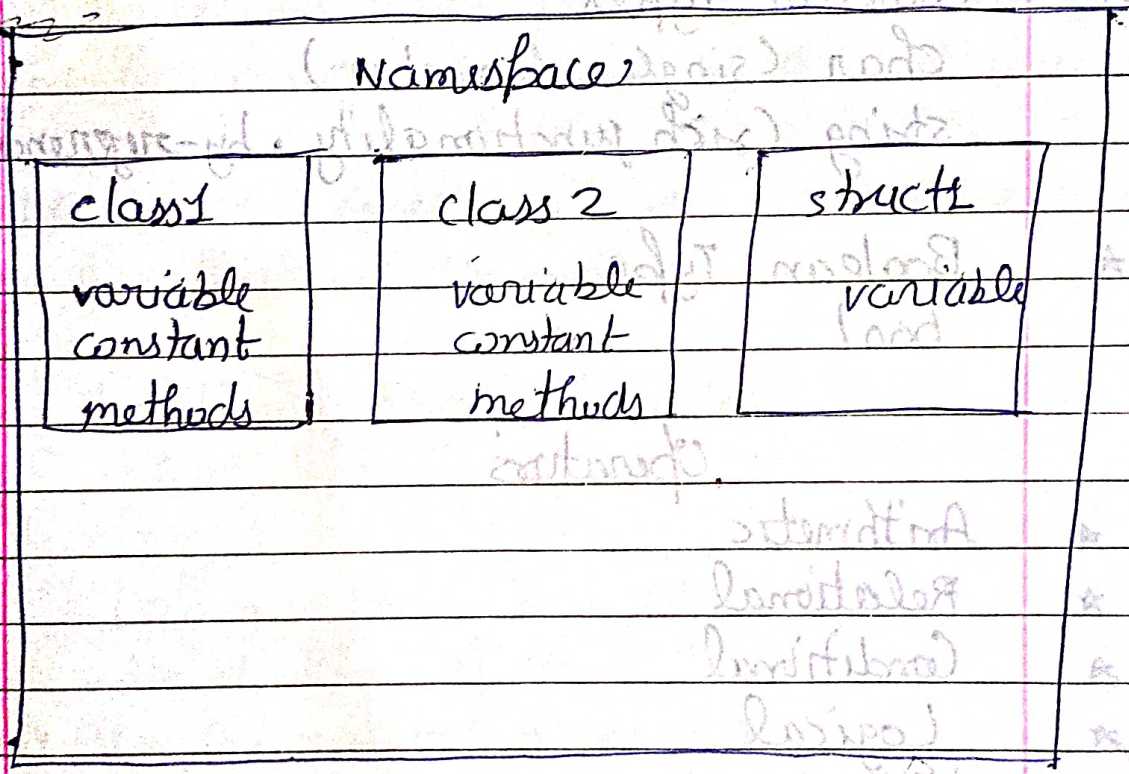
* Object Oriented
* Everything belongs to a class
* Complete C# program:

```
using System;
namespace Console Test
{
    class Class1
    {
        static void Main(string[] args)
        {
        }
    }
}
```

Directive =
Alias =
Nested =
Station =

C# program structure

* Namespaces — used to declare a scope that contains a set of related object. Contain types and other namespaces.
* Type declarations -
  Classes, structs, interfaces, enums, and delegates.
* Members -
  Constants, fields, methods, properties, indexers, events, operators, constructors, destructors



Data Types

- **Integer Types:**
  byte, sbyte (8 bit or 1 byte), short (16 bit) 2 byte
  int, uint (32 bit), long, ulong (64 bit)
  4 byte 8 byte

- **Floating point types:**
  float - (precision of 7 digits)
  double - (precision of 15-16 digits)

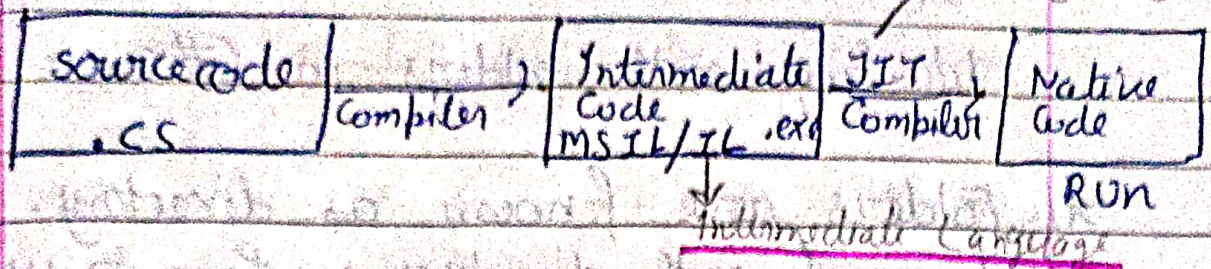
- **Exact Numeric type**
  decimal (28 significant digits)

- **Character Types**
  char (single character)
  string (rich functionality, by-reference type)

- **Boolean Type**
  bool.

Operators

- Arithmetic
- Relational
- Conditional
- Logical
- Bitwise
- Assignment
- Increment/Decrement etc



The Console Class

* Provides methods for input and output
* Input -
  Read(...) - read a single character
  ReadLine(...) - reads a single line of character
* Output -
  Write(...) - prints the specified argument on the console
  WriteLine(...) - prints specified data to the console and moves to the next line.

Note- The Console class is located under System Namespace.

Conditional Statement in C#

Just like C/C++

Looping statement in C#

Almost like C/C++

Note- One new loop is foreach

foreach loop is an item based loop. It is used to get all items one by one from a collection. It doesn't work on the basis of index. It works on the basis of item.

Syntax- foreach (datatype varname in collectionname)
{
// statements. .. Console.WriteLine(varname);
}

Working with folders in C#:-

*1- Folders are known as directory.
*2- To work with directory, we have to use "System.IO" namespace.
*3- We can create, delete and manage directory by using C#.

*4- Function to check for Existance of a directory:-

bool Directory.Exists(Fullpath_of_directory);

Ex- bool b = Directory.Exists("e:/csharp");

*5- Function to create a new directory:-

Directory.Create(Fullpath_of_directory);

Ex- Directory.CreateDirectory("e:/MyKSS");

*6- Function to remove a directory:-

Directory.Delete(Fullpath_of_dir);

Ex- Directory.Delete("d:/c sharp");

How to start a process or an application using C#:-

*1- To work with process we have to use "System.Diagnostics" namespace.
*2- Process class is used to work with process.
*3- Start() function of process class is used to start a process.
* Syntax- Process.Start("name of exe file of process");
Ex- Process.Start("notepad.exe");

Array in C#

*1- Collection of similar type of data items is known as an Array.
*2- First index of array is zero and last index is (size-1).
*3- We can find the size of array in C# by using "length" property.
*4- Syntax for array declaration:-
datatype[] arrayname = new datatype[size];
Ex- int[] nums = new int[10];

Some Important function of Array -

"Array" class provides features to work with array.

① Sort() -
This function is used to sort the items of an array in ascending order.
Syntax - Array.Sort(Array Name);

② Reverse() -
This function is used to reverse the sequences of items in an array.
Syntax - Array.Reverse(ArrayName);

③ IndexOf() -
This function is used to find the index of an item in an array. If item not found then it will return -1. This function used linear Searching algorithm
Syntax - Array.IndexOf(Arrayname, Searchvalue);

④ BinarySearch() -
It is used to find an item in an array by using binary search algorithm. To search items using this function array must be in sorted ~~array~~ order.

If item found then this function will return index of that item and if item not found then this function will return -1.

Syntax-

Array.BinarySearch(arrayname, searchvalue);

⑤ Copy()-

It is used to copy the items of an array into another array.

Syntax-

Array.Copy(SourceArray, DestArray, Length);

Note- Length is used to specify no. of items to copy.

2-D Array

2-D array is used to store the very large amount of data.

2-D array is used to store and manage data in matrix format.

Syntax- Datatype [,] = new datatype [rows, columns]

String in C#-

* String is a datatype in C#.
* String is a collection of characters.

Syntax to declare a string variable-

String VariableName;

Ex- string name;

Note- Length of a string is known by using "length" property

Some important functions of string

① ToUpper() Ex- StringVariable.ToUpper();
② ToLower()
③ IndexOf() - used to know the first occurance index of a character in a string.

Syntax-
int stringvariable.IndexOf(string/char);

④ LastIndexOf() - used to know the last occurance index of a character in a string.

Syntax-
int stringVariable.LastIndexOf(string/char);

⑤ Concat() - used to concat two strings
Syntax= StringVariable.Concat(str2);

⑥ SubString() - used to get part of a string i.e to get a sub string from a string

Syntax-
string str_var.SubString(StartIndex);
note- This will extract substring from start index to end of string.

Syntax: String str_var.SubString(StartIndex, ~~EndIndex~~ length);

⑦ Split() - used to convert a string into an array
It devides a string into various words.
Syntax -

string [] stringVariable.Split(DevideForm);

Working with Date and time in C#

We can work with date time values in C#. for that C# provides a "DateTime" class
DateTime class is located under "System" namespace.

Syntax to create an object of DateTime Class

DateTime objname = new DateTime();
Ex- DateTime d = new DateTime();

Syntax to create an object of specific DateTime

DateTime ObjName = new DateTime(int year, int month, int day);
Ex- DateTime d = new DateTime(2015, 1, 5);

We can get current date time by using a static property "Now" as shown below -
DateTime.Now;
class

dd	Date
MM	Month
yy	For 2 digit year
yyyy	For 4 digit year
HH	Hour in 24 format
hh	" 12 "
mm	minutes
ss	seconds
tt	For AM/PM

For specific format:-

```
DateTime dt = DateTime.Now;
string sdt = dt.ToString("dd/MM/yyyy hh:mm:ss tt");
```

UDF in C#

User defined function are created by programmers. It is a primary task during creation of software.

Syntax:-

```
<Access Specifier> ReturnType FunctionName (List of parameters)
{
    //statements;
}
```

Note- Default access specifier is "internal" as -

```
void add(int x, int y)
{
    int z = x+y;
    Console.WriteLine("The sum is: "+z);
}
```

Object Oriented Programming

- Object oriented programming is a very famous and modern programming technique
- It is a methodology that is used to write computer programs by using three concept of object
- OOP uses the concept of object to create programs
- OOP is very close to our daily life. It tries to implement human behaviour in programs
- There are various OOP languages (OOP's)
  Ex- C#, Java, VB.NET, PHP, Python, C++ etc.
- Below are the main concepts in OOP's -
  class
  object
  Data Abstraction
  Data Encapsulation
  Inheritance
  Polymorphism
  Message Passing.

Note- If we use above concepts in our program then our program will be object oriented.

That concepts are also known as pillars/ basics/ features/ concepts of OOP's

class

* It is baginning of OOP's.
* A class can contains various variables and various functions.
* In OOP's; ~~data~~ variables are known as "data members" and functions are known as "data methods"
* A class is a collection of data members and data methods in a single unit.
* class works like a container.
* 'class' keyword is used to create a class
* By default members and methods of class are ~~"internal"~~. By default ~~class~~ "private".

Syntax -

```
<access_specifier> class <class_name>
{
    //data members
    // data methods

}
```

Object

* Object is a copy of class
* Object is used to initialize memory for a class. Object creation means loading a copy of class into memory
* Object is used to access and use the features of a class during runtime of program.
* object is a virtual copy of class
* Object is a run-time entity that is used to access the feature of a class

Syntax to create an ~~class~~ object

```
class_name objectname = new ClassName();
```

Ex: student s = new student();

Constructor

It is also a member function of class.
Constructor is a special function of a class.
It is used to initialize data member of class
Constructor has below special features -

* Its name must be same as class name
* It can't return a value. Even no need to write void.
* Constructor are created in public scope
* No need to call a constructor function. because they executes automatically when we create the object of that class.

* Constructor can have parameters.

Syntax-

```
public <class_name> (Parameters)
{
    //statements
    //statements
}
```

Types of Constructor:

1. Default Constructor- A constructor without any parameter is known as Default constructor.
2. Parameterized Constructor- A constructor having some parameters is known as parameterized constructor.

Access Specifier

1. private- Private members and methods are accessible only within the block where they are declared. It is most Secure (restricted) access specifier.
2. internal- Internal members and methods are accessible in whole current assembly

It is namespace level access specifier. Internal members are not accessible outside the current assembly.

3 protected - Protected members and methods are accessible in current block and inside child class.
Protected members are used in case of inheritance.

4. public - public is least secure access specifier.
public members and methods are accessible from whole project.

Data Encapsulation

* Wrapping of data members and data methods in a single unit is known as Data Encapsulation.
* It provides features of portability.
* "class" is used to implement/achieve data Encapsulation.

Inheritance

It is one of the most famous feature of OOP's Inheritance provides the feature of re-usability of code.

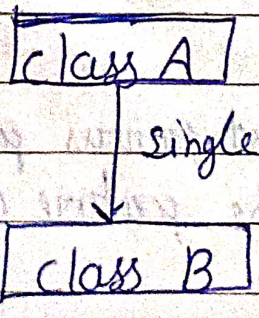
* To access/use the features of a class inside another class is known as Inheritance.
* The class that provides feature (being inherit) is known as parent/super/base/main class.
* The class that uses the features (that inherits another class) is known as child/sub/derived class.
* ":" symbol is used to inherit a class in C#.

Syntax to inherit a class:-

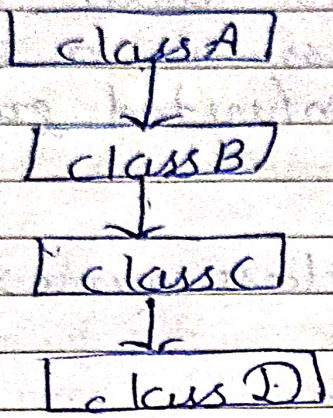
```
class <Sub_class_name> : <Super_class_name>
{
    // data members
    // data methods

}
```

Types of inheritance in C#

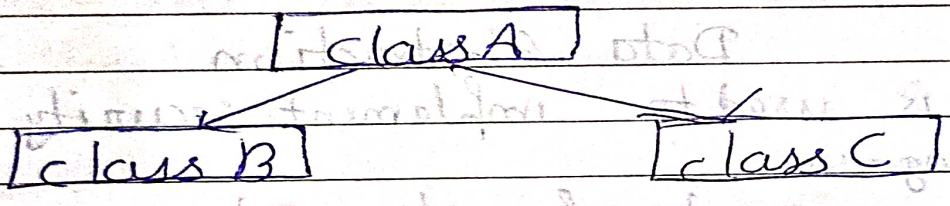
1- Single Inheritance



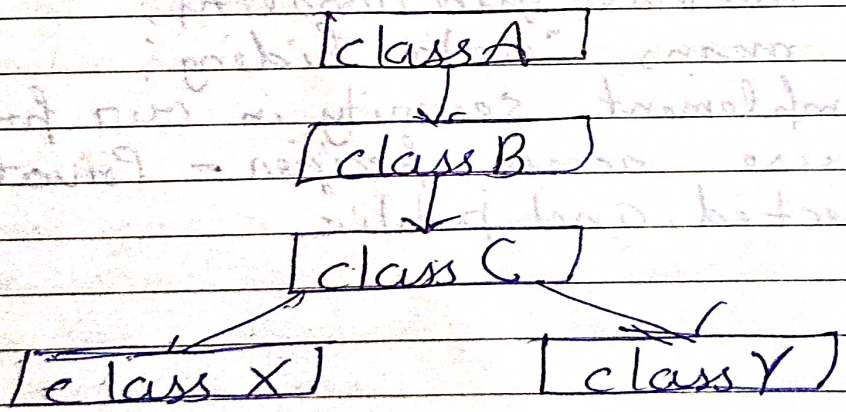
2) Multi-level Inheritance



3) Hierarchical Inheritance



4) Hybrid Inheritance:-



Note- Multiple inheritance is not supported in C# / java.

sealed Keyword

'sealed' keyword is used to lock a class. sealed class is a special class that can't be inherited from a child class

Syntax -

```
sealed class <class_name>
{
    // data members
    // data methods
}
```

Data Abstraction

It is used to implement security in our program.

It means to provide only required information to end user without providing details of internal functionality.

It means "data hiding".

To implement security in our program we can use access specifier - Private, Internal, protected, and public

Poly morphism

* It is a combination of two words - poly and morphism.
* Here, poly means many and morphism means form. Hence Polymorphism is 'one thing in many form'.
* It makes easy to remember function names.
It is of two types -

① Compile Time Polymorphism / Early Binding
⇒ Function overloading
② Run - Time Polymorphism / Late Binding
⇒ Function overriding

Function Overloading

* To define various function with same name having different-2 signature (parameters) is known as function overloading.
* In C#, we can define many functions with same function name having different-2 parameters in same scope. This concept is known as function overloading. It makes easy to remember function names.

Date
Page

* We can overload a function into below basis -

① On the basis of number of parameters -
Ex- int Add (int num1, int num2)
int Add (int num1, int num2, int num3);

② On the basis of types of parameters -
Ex- int Add (int n1, int n2);
float Add (float n1, float n2);

Function Overriding

* To define multiple function having same signatures & same name is known as function overriding
* We can redefine a function of a base class inside child class. This concept is known as function overiding
* For function overiding function must have in different - 2 scope (parent class & child class).
* Function overiding defines new definition for a function of base class inside child class.
* To redefine a function of base class we must have declare that function as 'virtual' function. For that we

Have to use 'virtual' keyword in function declaration of base class.

* To override / redefine a function of base class we have to use 'override' keyword in function declaration of child class.

Delegate in C#

* Delegate is a function pointer in C#
* Delegates are used to create call-back functions and events.
* Delegates stores the address of functions.
* It is necessary to have same signature of delegate and function.

Syntax to create a delegate -

<access specifier> delegate returntype ^delegate_name (Parameters);

Ex- public delegate int mydel(int x);

Syntax to assign address of a function to a delegate -

<delegate_name> Obj_name = new <delegate name> (Function name);

Ex- mydel md = new model(area);

Syntax to call a function by using delegate

<delegate_object>(parameters_for_function);

Message Passing

In OOP's object of same class or object of various class can interact with each other. They can transfer/assign values of properties to each other. This concept is known as Message Passing

Random class in C#

- Random is a class in C# that is used to generate random values.
- This class is located under 'System' namespace.
- Random class has a 'Next()' function that is used to get random number of a range.

Syntax -

```
Random r = new Random();
int r.Next(MinValue, MaxValue);
```

C# generic

In C# generic means not specified to a particular data type. C# allows you to define generic classes, interfaces, abstract classes, fields, methods etc. It can be declared by specifying a type parameter in angle brackets after a type name, e.g. TypeName<T> where T is a type parameter.

Ex- class DataStore<T>
{ public T Data {get; set;}
}

Ex- class Raj<T, K>
{ public T key {get; set;}
public K hi {get; set;}

* Instantiating generic class

Ex- Datastore<String> store = new DataStore<>();

Conversion from one datatype to another

(1) double d = 173.33;
int i = (int) d;

(2) Using built-in functions-
- ToBoolean()
- ToByte()
- ToChar()
- ToDateTime()

- ToDecimal()
- ToDouble()
- ToInt16()
- ToInt32()
- ToInt64()
- ToSbyte()
- ToString()

Ex- int i = 5;
string s = i.ToString();

Note- These all are methods of Convert class.

Ex- int i = 5;
String s = Convert.ToString(i);

Note- You can use these methods by two ways using Convert class and using <var name>.ToKDataType