-> namespace is a container where multiple class available and other namespace available

-> c# use pascalcase like :- FloatValue;

-> process

Code - compiler - cil - CLR (JIT) - machine code

-> Console.WriteLine("write & next line");

Console.Write("without next line");

-> placeholder syntax

string name="karan";

Console.WriteLine("i am : {0}, name);

Console.ReadLine(); // read input

/\* Multi line comments \*/

-> two type of type casting

1. Implicit casting

Nani dol nu Pani moti dol ma samay

Int x=5; (auto casting)

float y=x;

2. Explicit casting

int x=(int) 5.67F;

3. Using method

float y=Convert.ToInt32(3.55);

OR int.Parse(3.55);

Like :- ToFloat,ToDouble ..

-> int.MAXVALUE = it give max value available.

int.MINVALUE = it give min value

Like :- float,byte etc.

-> precision =(.) Pela and (.) Pachhi thayne how many number able to write

float :- 7, double :- 15 , decimal :- 27

decimal d=5.35m;

-> escape sequence = \n,\t,\"

Verbatim literal :- string s=@"kar\n"

Print the escape sequence

Expression= oparand + operator

2+2=4 oparand =2;

-> DateTime dt = DateTime.Now;

Console.WriteLine("{0 :hh:mm:ss tt},dt);

DateTime.MinValue // 1/1/2001 12:00:00PM

DareTime.MaxValue // 12/31/9999 11:59:59 PM

-> multidimensional array

int [ , ] arr=new int[3,4];

Console.Write(arr[2,4]);

arr.GetLength(0) = 3

arr.GetLength(1)= 4 based on initialisation

arr.Rank = 2 it give dimensions

-> goto statement is valid in c#

-> method and functions are set of instructions but method is associated with object

-> Named argument :-

public static void karan(string name,int age){// argument}

Call method :-

karan(age : 24,name : "jay");

-> for each(int item in arr){

Console.WriteLine(item);

}

-> struct – value type – stack memory

Class – reference type – heap memory

-> PassByReference for primitive data

Call :- karan( ref value);

Declare :- public void karan(ref int a){}

// Alternative of ref is out keyword for pass int data or other without define the value

-> var is a value type in c#

Also use for local define not pass and return in function.

var a=25; a="ka"; //not valid

.GetType(); for check the data type

-> dynamic is decided at run time

Not mandatory to define the value

dynamic a=25; a="ka"; // valid

Also pass into function as a parameter and return.

-> by default access modifier is private.

-> static constructor created using static keyword.

static constructor use to initialize static variable.

It call only once no matter how many objects are created.

It doesn't have parameters and access modifier

-> we can not create a obj of static class

Create only one child class

Not inherite the static class

-> destructor initialize

~ class\_name(){ }

-> Properties

private int \_age;

public int age

{

set { //condition

this.\_age=value;

}

get{ //condition

return this. \_age;

}

}

In main :- set-> obj.age=56; //get -> obj.age

-> operator overloading ( code )

-> sealed class ( code )

We can not create a child class for sealed

sealed class karan{

}

-> sealed override method ( code )

-> indexer is same concept of encapsulation but it use for which value we can set and get in list or array .( code )

-> delegates is a third party use for authentication represent other person.

It hold method's reference in an object.

It also called function pointer .

It is reference type

It requires same signature as method like :- number of parameter and return type

Use for better performance & method call run time.

-> single cast delegate :- it refer only one method.

-> multi cast delegate :- it refer more than one method. Also use += and -= in obj.

-> multiple delegate :- more than one delegate use in single program.

-> annonymous method :- using delegate keywords we create this method.

Like :- mydelegate obj = delegate (int a){

// Statement. };

-> lemda func :- it same like annonymous method also represents this method.

For making lemda removes the unuse keywords .like :- delegate and data type.

Like :- mydelegate obj = (a) => {

// Statement. };

-> lemda operator :- =>

-> abstraction :- call private method using another public method.

-> abstract class :- use abstract keyword

Not create object of abstract class.

Same functionality use many class using inheritance.

It inherit one class and multiple interface.

When we implement abstract method in child class then use override keyword.

-> interface :- we can not use access modifier in any method in interface .

not create default method.

It inherit not a class and multiple interface.

-> Generic method :-

In which method is allow all data type in parameters it is called generic method.

(Code)

It is reuse the code, type safety and Better performance.

-> Generic class

In which class allowed all the data type is called generic class.( Code )

-> ArrayList - it is a non - generic data structure . Also multiple data type value can store in one ArrayList.

Declaration :-

ArrayList arr= new ArrayList ();

arr.Add(20);

arr.Add("karan"); // also store

double,char etc

arr.Insert(2,45); insert in position

arr[2]=45; same but overlap value

arr.Remove(45); remove value

arr.RemoveAt(2); remove 2nd value

-> Hashtable - non generic data structure

-> Stack - non generic - that's why it allows to store multiple data type element.

Also store null and duplicate value. (code)

->Queue - non generic - that's why it allows to store multiple data type element.

Also store null and duplicate value. (code)

-> List<> - generic data structure

That specify the which data type we store.

Store duplicate value

Store null value for reference type

Like :- string. ( Code )

-> Dictionary < , > - generic data structure

That specify the both key and value data type. Key must be unique.

Key can't be null but value can be.

( Code).

-> string - it is immutable.

When we like : concat the string with another so it create new copy in heap memory.

This problem solved using stringBuilder.

-> finally block always execute.

When exception is available then the contant of finally block is not show , so for showing the contant debugging the code.

-> for writing multiple catch block need

Most specific to most general.

-> way of exception with throw keyword

Default throw Default catch

Default throw Our catch

Our throw Default catch

Our throw to Our catch (code)

-> partial class

It is use for a specific features.

Use partial keyword.

File name is different for all partial class and same namespace and same type.

We can create partial struct and interface.

Multiple devloper work on defferent features of single class using partial class.

Ex :-

public partial class karan

{ // code. }

public partial class karan

{ // Code. }

public class main {

public static void Main(string [] args){

karan obj = new karan();

}

} // (Code)

-> string builder

It is a dynamic object that allows to expand the character in string.

It not create a new object in memory. It expands the memory space.

Default capacity is 16 character. (Code)

-> extension method

Inject the additional method without modifying, deriving and recomping the original class , structure, interface.

Process :-

Main class

Create static class for static extension method. In this method pass the reference of main class in parameters like this class\_name obj .

Create another class with main function and call extension method.

In namespace we set reference a second Main class in property.

And run the second main class…