#VAR

->Why C# is a strongly typed language?

=>It means you must declare a variable type before you can use it

# What is var?

->The "var" keyword is used to declare a var type variable.

->The var type variable can be used to store a simple .Net data type, a complex type, an anonymous type or a user defined type.

# Restrictions

=>var can only be used when a local variable is declared and initialized in the same statement.

=>The variable can not be initialized as null.

=> var can not be used on feilds at class scope.

# When to use var

->Use of "var" is not recommended everywhere.

->The var was created to handle declarations when the type is not known such as generic types.

-> Don't use var for simple lacal variable types that are known to you.

->Use of var when you are not sure what type of data will be

stored in a variable.

#CONSTANTS

> A variable whose value can not be changed during the execution the program is called a constant variable.

>>Two types

1) Compile time (const)

2) Runtime (readonly)

#READONLY

-> In C# you are allowed to declare a feild using readonly modifier.

-> It indicates that the assignment to the feilds is only the part of the declaration or in a constructor to the same class.

-> Such type of feild can be assigned of reassigned multiple times only at the declaration or in a constructor.

#Important Points about READONLY

1)It's not just to assign a value at the time of declaration, we can assign the value for readonly throught the constructor.

2)Readonly allows, readonly constant as well as non readonly constant variable into the expression.

int readonly x = 50;

int y = 30;

int readonly z = x+y;

3)Readonly can be declared only at the class level not inside the method.

#CONSTANT

-> Constants are feilds whose values are set at compile time and can never be changed.

-> Use constants to provide meaningful names instead of numeric literals.

#Important Points about CONST

1) It must be assigned a value at the time of declaration.

2) CONST only allows constant variables into the expression.

int const x 58;

int const y = 30;

int const z X+y;

intiia = 10;

int const sum = a+y;//Compile time error

3) const can be declared at the class level as well as inside the method.

#STRUCTURE

-> Structure is a value type.

-> A collection of variables of different types under a single

unit.

# Class VS Structure

1) In C# Class is a reference type & Structure is a value type.

2) Class can contain default constructor or destructor but structure will contain only constructor that have parameter

3) We can implement inheritance using Class but structure won't support inheritance.

4) Unlike class, structure can be instantiated with or without using a new operator.

#STRING

-> In C#, you can use string as array of characters.

-> More common practice is to use of string keyword to declare a

string type variable.

-> The string keyword is a part of the System.String class.

#Creating a string

1. By assigning a string literal to a string variable.

2. By using String class constructor

3. By using the string concatenation operator

4. By retrieving a property or calling a method that returns a string

5. By calling a formatting method to convert a value or an object to its string representation.

#Creating a string

1. By assigning a string literal to a string variable.

string name = "Ahmed";

2. By using String class constructor

String name = new String("Ahmed") ;

3. By using the string concatenation operator

string name = "Cecilia"+" "+"Hansen";

4. By retrieving a property or calling a method that returns a string.

5. By calling a formatting method to convert a value or an object to its string representation.

#ENUM

-> In C# enum(or enumeration type) is used to assign constant names to a group of numeric integer values.

-> It makes constant value more readable.

-> It is type.

-> It's main use to assign the names or string values to integral constant, that makes program easy to read and maintain.

# INHERITANCE

-> Inheritance is an important pillar of OOP.

-> It is the mechanism in C# by which one class is allowed to inherit the features(feilds and methods) of another class.

-Important Terminology

1) Super Class

-> The class whose features are inherited is known as super/base/parent class.

2) Sub Class

-> The class that inherits the other class is known as sub/derived/child/ extended class.

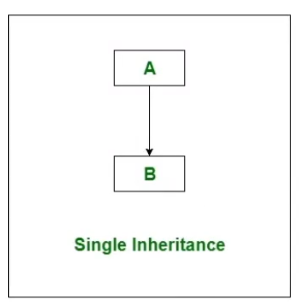
3) Reusability

-> When we want to create a new class and there is already a class that includes some of the code that we want, we can derive our new class from the existing class.|

#Types of Inheritance

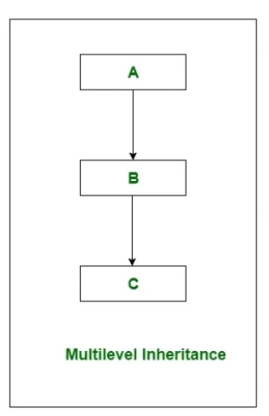
1. Single Inheritance

-> When one class is extended by only one class, it is called Single level Inheritance.



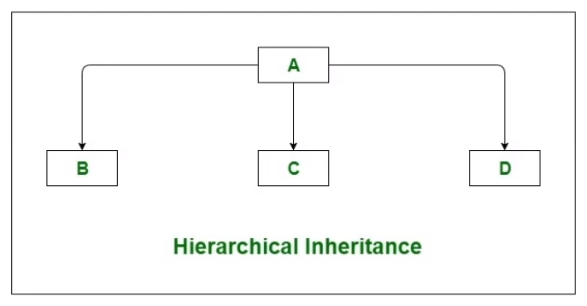
2. Multilevel Inheritance

-> A class which is extended by a class and that class is extended by another class forming a chain inheritance is called multilevel Inheritance



3. Hierarchical Inheritance

-> A class which is inherited by many subclasses is known as hierarchical Inheritance.



4. Multiple Inheritance

-> In multiple Inheritance, a subclass inherits from more than one immediate superclass.

-> C# does not support multiple inheritance.

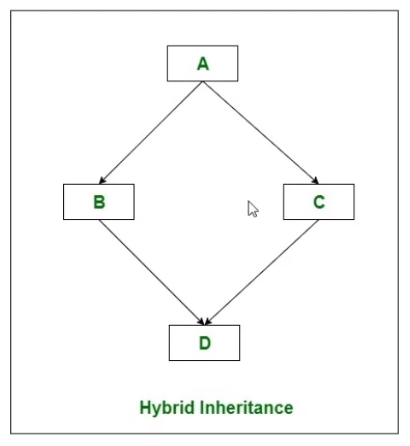
-> In C#, we can achieve multiple inheritance only through interface.

5. Hybrid Inheritance

-> A hybrid inheritance is a combination of single and multiple inheritance or it is a mix of two or more types of inheritance.

-> C# does not support hybrid inheritance.

-> In C#, we can achieve hybrid inheritance only through interface.



CONSTRUCTOR CHAINING

-> In C#, Constructor Chaining is an approach to invoke constructor from another constructor.

-> To achieve Constructor Chaining we need to use "this" keyword after our constructor definition.