Final Term

Abstraction

Abstraction is the process of defining a class by providing the necessary and essential details of an obj to the outside world and hiding the unnecessary things.

Simply → It means

We need to display what is necessary and compulsory and need to hide the unnecessary things to the outside world

In C# we can hide the members of a class by using private access modifiers.

Real time example

→ Car

The things a driver should know are as follows.

a) Name of the Car
b) Color of the Car
c) Gear
d) ~~Brick~~ Break

The things which should be hidden to a car driver are as follows.

a) Engine
b) Sinencer.

Abstraction VS Encapsulation

Abstraction	Encapsulation
→ Used to hide unwanted data and show only required properties and method	→ It is used to bind data members and member functions into a single unit to prevent outsiders to access it directly.
→ It hide inner implenentations	→ It hides data members and member functions

How to active Abstraction

a) Abstract class (0 - 100% Abstraction achive করা যায়)

b) Interface (100% direct "কি" করি)

Abstract Class & Abstract Methods

Abstract class is a restricted class that can not be used to create obj. (to access it, it must be inherited from another class)

Abstract Method can only be used in abstract class and it does not have a body. The body provides by the derived class.

Imp - points

i) Abstract class is declared with the help of "abstract" keyword.

ii) You are not allowed to create obj of the abstract class.

যেহেতু obj create করা যায় না
সেহেতু new operator use " " "

iii) You can not use the abstract class directly with a new operator.

iv) Class that contains "abstract" keyword with some of the methods (not all abstract method) is known as ~~Abst~~ Abstract Base class.

যদি সব method abstract keyword use করে then Pure Abstract Base Class

v) Class that contains "abstract" keyword with all of its methods is known as pure Abstract Base.

vi) You are not allowed to declare the abstract method outside of the abstract class.

vii) You are not allowed to declare the abstract class as Sealed Class.

Interface

It is similar to an abstract class but all of its methods are abstract by default. It means, it can't have non-abstract method.

→

Imp points

→ We use "interface" keyword to define an interface.

→ Like abstract classes, interface can not be used to create obj.

→ Interface methods do not have a body -
The body is provided by the "implemented" class.
↓
derived class.

→ On implementation of an interface, you must override all of its method.

→ Interface can contain properties and method but not fields/variable.

→ Interface member are be defaut abstract and public.

→ Interface can not contain a Constructor.
(because in can't be used to create any obj)

Why and When to use interface

i) To achive security.

ii) C# does not support "multiple inheritance". It can be achived through interface.

inherit করা যাবে না

Sealed

Sealed Class → A sealed class in a class that does not allow (inheritance) which means the sealed class will restrict user to inherit a class.

→ A Sealed class is defined by using a "sealed" keyword that notifies the compilar that the class is sealed so that it can not be extended and there in no chance of inheriting a specific class.

sealed method কে override করা যায় না

Sealed Method → We can also use the "sealed" keyword on a methods that overrides a virtual methods in a base class to allow other classes to derive from base class but to prevent them from overriding specific virtual method.

Parent class

Base

Base keyword is used to access fields, constructors and the methods of base class. You can not use it inside the static method.

Generics

→ Generic means the general form, not specific.

→ Generic is not specific to a particular data type.

→ C# allows to define generic on (C# এ কোন concept এ apply করা যায়)

- → classes,
- → interfaces
- → abstract classes
- → fields
- → methods
- → static methods
- → properties
- → events
- → delegates
- → operators

এই topic এর উপরে (কিভাবে apply করতে পারি)

Using the type parameter and without the specific data type.

#Type parameter :- এটা কাকে generic কে specify করে

A generic type is declared by specifying a type parameter in an angle bracket after a type name.

ex :- TypeName<T>

T → type parameter or placeholder

TypeName হবে আমার যে কোন একটি field এর কোন C# keyword/ T এর জায়গায় যে কোন word হতে পারে.

Generic Class

→ The generic class can be defined by putting the <T> sign after the class name.

→ It isn't mandatory to put the "T" word in the generic type definition. You can use any word in the TestClass<> class declaration.

→ Example :- public class TestClass <T> { }

imp

→ The System.Collection.Generic namespace also define ar number of classes.

এরকম built in class গুলো থাকে
/ " / generic " " "

built in / Generic Classes :

1) Collection <T>
2) Dictionary <T>
3) List <T>
4) Queue <T>
5) Stack <T>

Adv of Generics

→ Reusability :- You can use single generic type definition for multiple purposes in the same code without any alterations.

ex :- you can create a generic method to add two number.

→ Type safety :-

↓

Generic data type provide better type safety, especially in the case of collections.

~~Performance~~

When using generic you need to define the type of obj to be passed to a collection. This helps compiler to ensure that only those obj types that are defined in definition can be passed to the collection.

(Compiler কে help করে ensure করতে যে type pass হচ্ছে তা same থাকবে)

→ Performance :- Generic types provides better performance as compared to normal system types because they reduce the need for boxing, unboxing and type casting of variable or obj.

Ref & out keyword

[ref] → keyword is used to pass arguments to method as reference type and is used to state that the parameter passed cannot be modified by the method.

→ The parameter must be initialized first before it is passed to ref.

→ It is not required to assign or initialize the value of a parameter before returning to the calling method.

[out] → keyword is used to pass arguments to method as reference type and is used to state that the parameter passed must be modified by the method.

→ It is not compulsory to initialized parameter or argument before it is passed to an out.

→ A called method is required to assign or ~~initialzi~~. initialize a value of a parameter before returning to the calling method.

.ref & out

→ Both ref and out are treated differently at run time and they are treated the same at compile time.

→ Properties are not variables, so it can not be passed as an out or ref parameter.

#C# | obj Class → Base class of all class

→ The obj class is the base class for all the classes in .NET FrameWork.

→ It is present in the System namespace.

→ EveryClass in C# is directly or indirectly derived from the obj class.

→ If a class does not extend any other class then it is the direct child class of obj class. and if extends other class then it is an indirectly derived.

Collection

what is collecting collection.

↳ Collections are similar to arrays. it provides more flexible way of working with a group of obj.

→ Collection classes are specialized classes for data storage and retrival.

→ Collecting types implement the following common functionality.

1) Adding and inserting items to a collection.

2) Removing items from a collection.

3) Finding, sorting, searching items.

4) Replacing items.

5) Copy collection and items.

6) Capacity and Count properties. to find the capacity of the collection and number of items in the collection.

Work with Collections

→ There are 3 ways to work with collections which are -

i) System . Collections . Generic class .

ii) System . Collections . Concurrent class .

iii) System . Collections class .

Types of Collections .

→ .Net supports two types of collections

1) Generic Collections

- A generic collections is a class that provides type safety without having to derive from a base collection type and implement type - specific numbers .
- The Generic Collection classes are in the System . Collection . Generic ~~class~~ .namespace .

~~ii) Non - Generic Collections~~

- A Generic Collections is strongly typed .
- The Generic Collections stored elements internally in arrays of their actual types .

ii) Non - Generic Collections

- A Non- Generic Collections is a specialized class for data storage and retrival that provides support for stacks, queues, list and hash table.
- The Non - Generic Collection Classes are in the System . Collections .
- A Non - Generic Collections is not strongly typed.
- The Non - Generic Collections store elements internally in obj arrays so it can store any type of data.

Build in Classes

	Non Generic	Generic
i)	ArrayList	List
ii)	HashTable	Dictionary
iii)	SortedList	SortedList
iv)	Stack	Stack
v)	Queue	Queue

Collection & Description

a) **ArrayList** :- The ArrayList collection is similar to the Array data type in C#

<Biggest diff is the dynamic Nature of the array list collection.>

b) **Stack** :- The stack is special case collection which represents a last in first out (LIFO) concept

c) **Queue** :- The queue is special case collection which represents a first in first out (FIFO) concept

d) HashTable :- Special collection that is used to store key-value items.

e) SortList :- The SortList is a collection which stores key-value pairs in the ascending order of key by default.

f) Bit Array :- Is an array of data structure which stores bits