**Basic Definitions**

Data encapsulation is a mechanism of bundling the data, and the functions that use them and data abstraction is a mechanism of exposing only the interfaces or relevant parts and hiding the implementation details or complexity from the user.

**Object**

Real world objects shares 2 main characteristics, state and behavior. Human have state (name, age) and behavior (running, sleeping). Car have state (current speed, current gear) and state (applying brake, changing gear). Software objects are conceptually similar to real-world objects: they too consist of state and related behavior. An object stores its state in fields and exposes its behavior through methods.

**Class**

Class is a “template” / “blueprint” that is used to create objects. Basically, a class will consists of field, static field, method, static method and constructor. Field is used to hold the state of the class (eg: name of Student object). Method is used to represent the behavior of the class (eg: how a Student object going to stand-up). Constructor is used to create a new Instance of the Class.

**Instance**

An **instance** is a single and unique unit of a class. An instance, in object-oriented programming (OOP), is a specific realization of any object. An object may be varied in a number of ways. Each realized variation of that object is an instance. The creation of a realized instance is called instantiation.

Each time a program runs, it is an instance of that program. In languages that create objects from classes, an object is an instantiation of a class. That is, it is a member of a given class that has specified values rather than variables. In a non-programming context, you could think of "dog" as a class and your particular dog as an instance of that class.

**Constructor**

In class-based object-oriented programming, a constructor (abbreviation: ctor) in a class is a special type of subroutine called to create an object. It prepares the new object for use, often accepting arguments that the constructor uses to set required member variables.

A constructor resembles an instance method, but it differs from a method in that it has no explicit return type, it is not implicitly inherited and it usually has different rules for scope modifiers.

**VSTS/TFS:**

VSTS and Team Foundation Server (TFS) both provide an integrated, collaborative environment that supports Git, continuous integration, and Agile tools for planning and tracking work. VSTS is the cloud offering that provides a scalable, reliable, and globally available hosted service.