**Different between Class, Abstract class and Interface**

Classes, abstract classes, and interfaces are all fundamental concepts in object-oriented programming and are used to define the structure and behavior of objects in a software system.

Classes are the basic building blocks of object-oriented programming and are used to define objects and their behavior. A class contains member variables, which define the state of an object, and member functions, which define the behavior of an object. Classes can be instantiated to create objects, which are instances of the class.

Abstract classes are similar to classes, but they cannot be instantiated directly. Instead, they must be subclassed, and the subclasses must implement the abstract methods defined in the abstract class. Abstract classes are used as a base class to provide common functionality to subclasses.

Interfaces, on the other hand, are used to define a set of contracts or agreements that must be followed by any class that implements the interface. An interface defines the methods and properties that a class must implement, but does not provide implementation for these methods. Instead, the implementation is left up to the classes that implement the interface. Interfaces provide a way to define a blueprint for a class without specifying its implementation.

Classes, abstract classes, and interfaces are different in terms of their functionality and the role they play in object-oriented programming. Classes are used to define objects and their behavior, abstract classes are used as a base class to provide common functionality to subclasses, and interfaces are used to define a blueprint for a class without specifying its implementation.