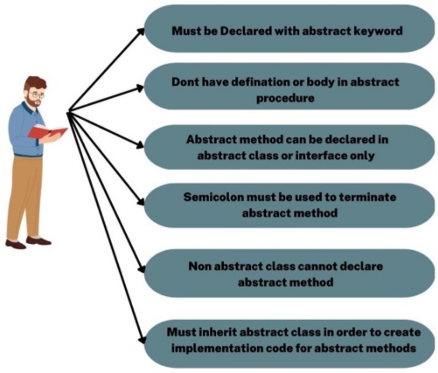
ABSTRACT CLASS

Introduction to Abstract Class

Java abstract class is a class that can not be instantiated by itself, it needs to be subclassed by another class to use its properties. An abstract class is declared using the “abstract” keyword in its class definition.
Abstract classes are a key component of OOP in Java, allowing you to define incomplete classes that other classes can extend.


Must be Declared with abstract keyword
Dont have defination or body in abstract procedure
Abstract method can be declared in abstract class or interface only
Semicolon must be used to terminate abstract method
Non abstract class cannot declare abstract method
Must inherit abstract class in order to create implementation code for abstract methods

In Java, the following some important observations about abstract classes are as follows:

- An instance of an abstract class can not be created.
- Constructors are allowed. Abstract Constructor can’t be created.
- We can have an abstract class without any abstract method.
- There can be a **final method** in abstract class but any abstract method in class(abstract class) can not be declared as final or in simpler terms final method can not be abstract itself as it will yield an error: “Illegal combination of modifiers: abstract and final”.
- We can define static methods in an abstract class. But we can’t create abstract static methods.
- We can use the **abstract keyword** for declaring ***top-level classes (Outer class) as well as inner classes*** as abstract. Abstract Class cannot be **‘final’**.
- If a **class** contains at least **one abstract method** then compulsory should declare a class as abstract.
- If the **Child class** is unable to provide implementation to all abstract methods of the **Parent class** then we should declare that **Child class as abstract** so that the next level Child class should provide implementation to the remaining abstract method.

Properties of Abstract Class

Let us elaborate on these observations and do justify them with help of clean java programs as follows.

Observation 1
In Java, just like in C++ an instance of an abstract class cannot be created, we can have references to abstract class type though.

Observation 2
Like C++, an **abstract class** can contain **constructors** in Java. And a constructor of an abstract class is called when an instance of an inherited class is created.

Observation 3
In Java, we can have ***an abstract class without any abstract method***. This allows us to ***create classes that cannot be instantiated but can only be inherited***.

Observation 4
Abstract classes can also have ***final*** *methods* (methods that cannot be overridden).

Observation 5
For any abstract java class we are not allowed to create an object i.e., for an abstract class instantiation is not possible.

Observation 6
Similar to the interface ***we can define static methods in an abstract class*** that ***can be called independently without an object.***

Observation 7
We can use the **abstract keyword** for declaring top-level classes (Outer class) as well as inner classes as abstract.

Observation 8
If a **class contains at least one abstract method** then **compulsory that we should declare the class as abstract** otherwise we will get a compile-time error. If a class contains at least one abstract method then, implementation is not complete for that class, and hence it is not recommended to create an object so in order to restrict object creation for such partial classes we use **abstract keyword.**

Observation 9
If the **Child** class is unable to provide implementation to all abstract methods of the Parent class then we should declare that Child class as abstract so that the next level Child class should provide implementation to the remaining abstract method.