Abstraction

1. What is Abstraction?

Abstraction is the process of hiding certain details and showing only essential information to the user.

For example, let's look at a real-life example of a man driving a car. As far as he knows, pressing the accelerator increases the speed of the car, or applying the brakes stops it. However, he doesn't know how the speed actually increases when pressing the accelerator. He doesn't know anything about the car's inner mechanisms or how the accelerator, brakes, etc., work.

2. Can we make the `abstract` methods `static` in Java?

In Java, if we make the

`abstract` methods `static`, they will become part of the class, and we can directly call them which is unnecessary. Calling an undefined method is not necessary, therefore it is not allowed.

3. What is the difference between abstract class and concrete (normal) class?

The differences between abstract class and concrete class are:

- We cannot create an object of an abstract class. Only objects of its non-abstract (or concrete) subclasses can be created.
- An abstract class can have zero or more abstract methods, but abstract methods are not allowed in a non-abstract class (concrete class).

4. What are the differences between `final` and `abstract` methods?

Differences between

final and abstract methods are as follows:

Final Method	Abstract Method
A final method cannot be overridden	An abstract method must be overridden
A final method has a body	An abstract method does not have a body
A final method can be invoked	An abstract method cannot be invoked directly

5. What is an Abstract method in Java? When it is used?

A method that is declared with an

abstract modifier and has no implementation (means no body) is called as an abstract method in Java.

It does not contain any body. It simply has a method declaration followed by a semicolon(

;).

An abstract method can be used,

- when the same method has to perform different tasks depending on the object calling it.
- when it needs to be overridden in its non-abstract subclasses.

6. Can we define an abstract method inside non-abstract class (concrete class)?

No, we cannot define an abstract method in a non-abstract class.

7. Is it compulsory for a class which is declared as abstract to have at least one abstract method?

No, an abstract class may or may not have abstract methods.

8. We can't instantiate an abstract class. Then why constructors are allowed in abstract class?

It is because we can’t create objects of abstract classes but we can create objects of their subclasses. From the subclass constructor, there will be an implicit call to the superclass constructor. That’s why abstract classes should have constructors.

Even if we don't write a constructor for our abstract class, the compiler will keep the default constructor.

9. Can we create an abstract attribute?

No, we cannot create an abstract attribute. The

abstract keyword is used to declare methods that must be implemented by subclasses, but attributes cannot be overridden, as they represent a state that is stored in memory and is not meant to be changed by subclasses.

Therefore, declaring an abstract attribute would be meaningless and don't make sense in the context of object-oriented programming.

10. What will happen if we do not override all abstract methods in subclass?

Java compiler will generate a compile-time error. We have to override all abstract methods in the subclass.