Interface VS Abstract Class

Interface	Abstract Class
• No need to instantiate • Public and final vars • No constructors • Can extend multiple interfaces • Good for unrelated and generic classes • Only define functionality, not implement it • Acts like a template, so classes must override and redefine the abstract methods if they “implement” the interface. This is because these are not previously setup in the body of an interface function. • Only contains abstract methods that are all public. • Public class and members by default but can be changed to private in Java 9. • In Java 8, we can use private methods and variables by encapsulating it inside a class.	• No need to instantiate/but can have constructors • Public or private methods but not necessarily final vars • Can extend only one abstract class But can implement multiple interfaces Good for closely related classes. • At least 1 abstract class. • Can provide implementation of interface but not vice-versa.

General note: Static in java means can be accessible from anywhere. Its kind of like a singleton where only one instance exists so no need to declare it. So if modify it, it modifies the one instance available.