O03 Interfaces

Interfaces
Abstract classes and methods

Interfaces

An `interface` can be thought of as a contract that a class can satisfy.

- Uses `interface` keyword rather than `class`
- Cannot be instantiated (can't be created with `new`)
- Can contain (all implicitly `public`):
  - *Abstract methods* (method declaration without a body)
  - *Default methods* (using `default` modifier)
  - Static methods (using `static` modifier)
  - Constants (implicitly `static final`)
- Classes implement interfaces via `implements` keyword
  - A class which implements an interface must provide the specified functionality.

Interfaces as Types

An interface can be used as a type

- A variable declared with an interface type can hold a reference to a object of any class that implements that interface.

Abstract Classes and Methods

The `abstract` keyword in a class declaration states that the class is abstract, and therefore cannot be instantiated (its subclasses may be, if they are not abstract).

The `abstract` keyword in a method declaration states that the method declaration is abstract; the implementation must be provided by a subclass (like abstract methods in an interface, but applied selectively and explicitly).