protected: for attributes, methods and constructors, making them accessible in the same package and subclasses.

static: for methods and attributes. Static methods/attributes can be accessed without creating an object of a class.

final: for classes, attributes and methods, which makes them non-changeable (impossible to inherit or override).

abstract: -An abstract class is a restricted class that cannot be used to create objects (to access it, it must be inherited from another class). -An abstract method can only be used in an abstract class, and it does not have a body. the body is provided by the subclass (inherited from).

synchronized: A piece of logic marked with synchronized becomes a synchronized block, allowing only one thread to execute at any given time. - tells every thread to go update their value from main memory when they enter the block, and flush the result back to main memory when they exit the block.

volatile: tells all threads to do read and write operations on main memory only.

native: only for methods, indicate that the method is implemented in native code using JNI (Java Native Interface)

strictfp: (strict floating-point) ensuring the same result on every platform while performing operations in the floating-point variable

transient: Serialization is the process of converting an object into a byte stream, and deserialization is the opposite of it. variable as transient, then that variable is not serialized. for derived fields, any non-serializable references

assert: can remove the if and throw statement with a single assert statement. Java assertions use the assert keyword, there are no libraries needed or packages to import for backward compatibility, the JVM disables assertion validation by default. They must be explicitly enabled using either the -enableassertions command line argument, or its shorthand -ea

super: refers to superclass (parent) objects. call superclass methods, and to access the superclass constructor. eliminate the confusion between superclasses and subclasses that have methods with the same name. * Inheritance and Polymorphism.

this: The ‘this’ keyword refers to the current object in a method or constructor.
The most common use of the this keyword is to eliminate the confusion between class attributes and parameters with the same name (because a class attribute is shadowed by a method or constructor parameter) Invoke current class constructor -Invoke current class method -Return the current class object -Pass an argument in the method call -Pass an argument in the constructor call