**Enumerations (Keyword enum)**: it is a special type in Java that has list of constants which define fixed set of related values.

E.g.: DaysOfWeek, enum PaymentMenthod {Credit card, Debit card, Paypal}.

Note: enumerations can only hold a value that was declared in the list, other values are not allowed. We can say that enumeration gives us the way to explicitly specify the only values that are data type can legally have and are usually used to define a set of values that represent a specific collection.

We can define methods, instance variables and constructors in our enumerations. The constant created in the enumerator are the objects of enumeration type, so if we create a constructor in the enumerator, the constructor is called when each enumeration constant is created.so each enumeration constant has its own copy of instance variable defined by the enumeration,

* Enum has a list of constants which are implicitly defined as public static final. The type of these constants is same as the type of enumeration they are defined, i.e., these constants are self-typed.

A screenshot of a computer program

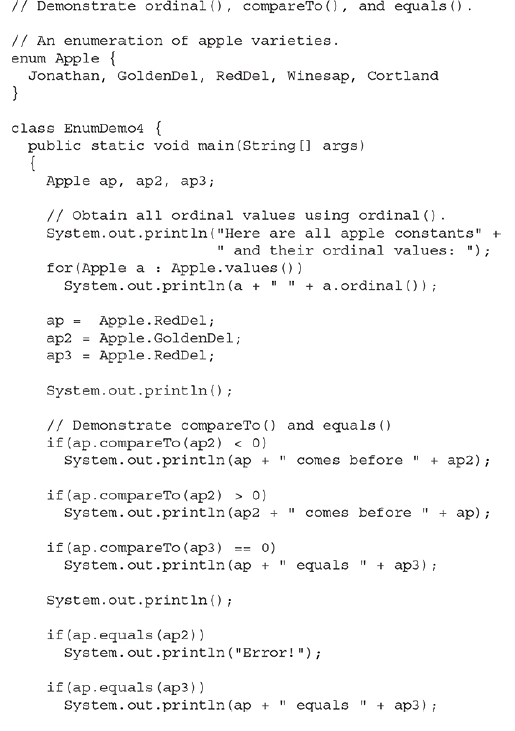
Description automatically generated

* Here we have defined an enum payment method, with the three constants (credit card, debit card, PayPal).
* Each constant is implicitly declared as public static final member of PaymentMethod.
* The type of the constants is the type of enumeration in which they are declared which is PaymentMethod in this case.
* Even though enumerations are a class type we do not instantiate an enum with **new** keyword, because enums are immutable and have predefined set of constant instances defined within the enum declaration itself.
* We define them same as primitive types (example PaymentMethod paymentMethod)
* Enums work well with switch statements compared to integer and string constants.

values() vs valuesOf()

* The values () method returns an array containing all the constants declared in the enum declaration.
* The valueOf() method returns the enum constant with specific name.

All enumerations automatically inherit one superclass java,lang.enum.

* It also inherits several methods that are available for use by all enumerations.
* ordinal( ) method - it returns the position in its enum declaration where initial constant is assigned as 0
* compareTo( ) - based on the ordinal value it compares 1 enum constant with another
* equals( ) - equals( ) method can compare enumeration constant with any other object however the method will return only true if the two objects contains the same value and they are from the same enumeration. Simply having ordinal values in common will not cause equals( ) to return true if the two constants are from different enumerations.
*  A white screen with black text

  Description automatically generated